NEW AI SYSTEM DOCUMENTATION:

The New AI System manages all NPC behavior, be it friendly or enemies. This new version should be far more efficient, secure, easy to read, and user friendly for any non coders trying to make simple behaviors.

HOW TO SET UP AN NPC:

Go to: Assets > Scripts > GameSystems > NewAISystem

Put an NPC behavior script on the base object of the NPC in question (ideally the one with the animator on it so it can be set automatically) be sure a NavMeshAgent is on the same object, if not, add one. Create an empty child object called StateMachine. Add the StateMachine behavior to that object. Set the animationController component for the NPCBehavoir, as the animator for the NPC, and the Machine component as the StateMachine you just created.

All the AI states that create the behavior will be created in child objects of the StateMachine objects. These child objects will be empty created objects with only the AIState script inside of them.

CREATING A BASIC FRIENDLY AI:

Tutorial for creating a friendly AI that wanders around doing tasks:

Create the basic NPC described above.

Go to: Assets > Scripts > GameSystems > NewAISystem > Prefabs > FriendlyAIStatePrefabs

Add a NextTask prefab, and at least two FriendlyTask prefabs. Set the FriendlyTask objects name variable to the tasks you want them to perform. For task duration, if you want the task to last a set amount of time, set both X and Y to the number of seconds you want the task to last (can be an int or a float). If you want it to be a variable amount of time, set X to the minimum time, and Y to the maximum time.

In the NextTask object, set the number next to the tasks variable to the number of tasks you have. Expand the drop box, and set each element to the name you gave one of the tasks, and the location you want that task to be performed.

Now the AI works but isn't animated. If the animator that the NPC Behavior is attached to doesn't have a controller, add one. In the controller (not the StateMachine), add a state for movement, and a state for every task to be performed. Add appropriate animations to each state. Add a transition from "Any State" to each animation state. Create a parameter for each

transition named appropriately, and set each transition's conditions to only happen when the corresponding parameter is true.

Now for each state in the State Machine, set AnimParam to the name of the parameter that corresponds with the animation you want the state to perform (NextTask should be the movement parameter). Now the state machine should animate.

AI STATES:

Use the AI state template (the most basic of which being the stateIdle AI state) to create AI states to populate your state machine.

Variables:
Name (string): Name of the state. Set in inspector or in script, whichever fits the specific state machine you are creating.

Machine (StateMachine): Set automatically. Used to make calls to the state machine.

AnimParam (string): Used in creating animation state machines. Will be discussed in creating animations section.

Functions:

StateEnter(): Is called once when the state is transitioned too. Put state initialization behavior in here.

StateExit(): Is called once when the state is transitioned from. Put last minute calls in here. Best used for animations and self contained state logic, not for controlling the AI itself.

StateUpdate(): Called once per frame. Update function for state logic.

StatePhysicsUpdate(): Called once every physics update frame. FixedUpdate function for state logic.

string ExitConditions(): Called once every frame. Checks conditions for the state to transition to the next state. Returns the name of the state it is transitioning too.

STATE MACHINE:

StateMachine behavior manages the AI states and handles any communication with the NPC Behavior.

Variables:

m_defaultState (string): State that the state machine will be in when the scene starts. Not setting this will lead to a non functioning AI.

_behavior (NPCBehavior): There is circular dependency between the State machine and the NPC behavior. Largely they exist for the purpose of abstraction. Do not make calls directly to the NPC behavior from AI States. Best practice is to set this in the inspector, but failure to do so will simply lead to it being set automatically at runtime.

m_currentState (string): Bugtesting variable. Simply used to see current AI state in the inspector. Setting this won't break anything (unless you make something dependent on GetCurrentStatename Which you should not) and it will be reset next state change, but there is no reason to mess with this. Simply use it for bug testing.

Functions:

GetBehavior()/SetBehavior(): Getter and Setter for behavior. Don't use these. Only for NPCBehavior Script to use.

GetStateName(): Returns state name. Use for bug testing if need be. Do not use for dependencies.

Vector3 GetDestination(): Returns destination intended for NavMeshAgent.

SetDestination(Vector3): Sets destination intended for NavMeshAgent. State Machine will tell the NPCBehavior to go to location once set.

StopMovement(): Sets destination to the agent's current destination. The State machine will tell the NPCBehavior to stop moving.

GetLocation(): Gets the current location of the agent (NOT the destination).

SetAnim(string param, bool mode): Tells the NPC Behavior to set animation parameter *param* to boolean value *mode.*

NPC BEHAVIOR:

NPCBehavior is what controls any actions required by an AI. Strictly communicates with the State Machine, and not directly with AI states or any other script.

Other than the one variable mentioned below, nothing in this script should be referenced or modified. If something that would benefit from breaking this rule comes up, first contact Jeremiah.

Variables:

m_machine (StateMachine): State Machine that the behavior references. Best practice to set manually, but if set up properly, will be set automatically.

m_animationController (animator) referenced animator. Best practice to set manually. If set up properly can be set automatically, but it may not be possible to set up properly in some cases, so set manually if you can.