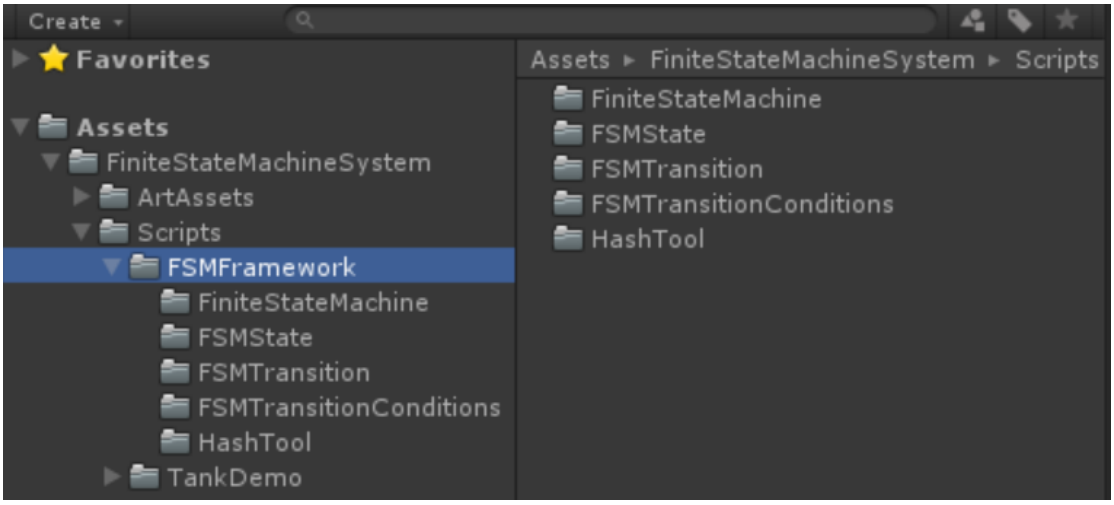
The introduction of Finite State Machine System

1. Purpose

This article illustrates this Finite State Machine System,to tell you how to use it properly.

2. The Finite State Machine Framework overview



(1) FSMState：

This is a abstract class, and it represents the state concept in the Finite State Machine System.

It has a list which stores some transitions,in these transitions, each transition contain the next state that this state may switch to and some conditions that determine whether this state could switch to the next state or not in the next frame.

When you want to create a concrete state of your own, then write a state class derived from it,override it virtual function if necessary:

OnEnter(): This method is used to initialize variables or something, after the Finite State Machine changes to this state

OnUpdate(): Every action, movement or communication the AI does should be placed here

OnExit(): This method is used to make anything necessary, as reseting variables before the Finite State Machine changes to another one.

(2) FSMEntry：

This state class is the first state when the finite state machine system working and you better not to change it.

(3) FSMExit：

This state class is the exit state when the finite state machine system enter to it.Once the FSM system enter this state,the finite state machine system can not switch to other state,represent the finite state machine is stop running.

(4) **FSMTransition：** This transition class link one state to another state called the next state,and it hold the next state reference and a list of condition which determine whether to change to next state or not.

Remember,in a transition all the condition must be match,then it will return true to notify the Finite State Machine to switch current state to the next state,otherwise,it will always return false,to notify the the Finite State Machine stay in current state.

(5) **IFSMTransitionCondition:** This is the interface of all the condition class,if you want to create your own condition,then write your own condition class derived from it and implement this interface function.

(6) **FSMDefaultTransitionCondition:** This is the default condition,the CheckCondition function always return the true value.you should not change it.

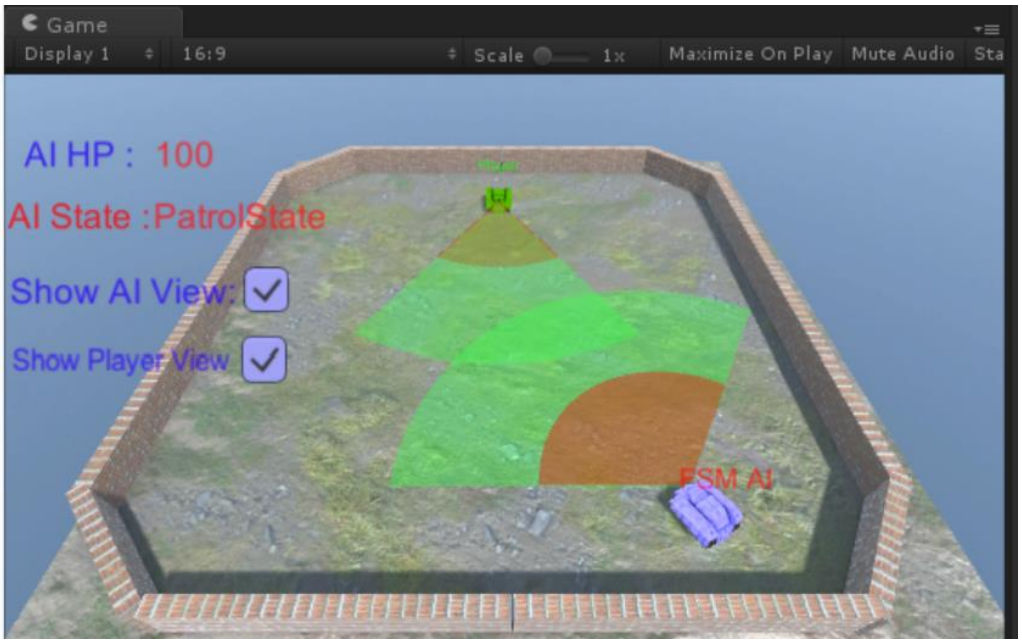
(7) **HashTool:** This is a tool class,*the StringToHash(string stringValue)* function will convert the string value to a hash string.

The reason why we convert the string to hash,because this can be improve the performance .In the Finite State Machine System inernal,searching the specific state given by its name,in fact, is searching by state name's hash value.

(8) **FiniteStateMachine:** This is the Finite State Machine class that each AI or GameObject in your game must have in order to use this framework. It stores the AI's States in a Dictionary, has methods to add and delete a state. The most important is that it has the function named *CreateFSMStateToAnotherFSMStateTransition(string fsmStateName, string nextFSMStateName, IFSMTransitionCondition[] fsmTransitionConditionArray)* ,it used to create link between one state to another.

After you add some state to this finite state machine and create the link among them,you should Use SetDefaultState() function to set the first start running state,and use SetDefaultState() to set the end state.Finally, invoke the *OnInitialize()* function in your script,and put the *OnUpdate()* function to your script's Update() or FixedUpdate() function.

3. Implement Finite State Machine used for AI in tank game demo



The logic of AI tank is simple,if it has not find a player in its sight view(green color),it will follow the patrol waypoints one by one in patrol state.once it find a player tank in its sight view,it will chase the player.If the player in the attack range(red color part) of it,the AI tank will stop chasing, aiming to the player and start to attack.if the life value of AI tank is reduce to zero,it will die.

(1) First,let's take a look at the Controller class,it has the ability to draw the unit sight view no matter in edtor or in run time

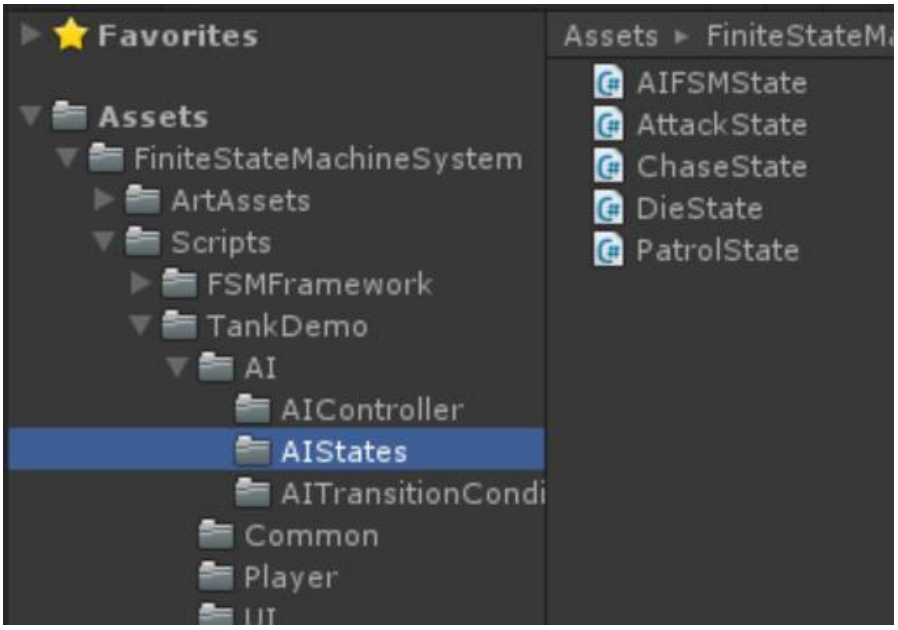
```
/// <summary>
/// This is the unit base class,and it can draw sight view in runtime.
/// </summary>
2 个引用
public class Controller : MonoBehaviour
{
    //the sight distance of the unit
    public float sightRange = 15.0f;
    //the attack distance of th unit
    public float attackRange = 7.5f;

    //the field of view of the unit
    public float fieldOfView = 60.0f;

    //this is the position where we start draw unit view
    public Transform drawViewRoot;
```

So,we create the AIController class derived from the Controller Class. We use the AIController class as the "Black Board" which means almost all the variable are public and the all states in finite state machine can access them.Later,we will initialize the finite state machine, create some states and use transition to link states,etc.

(2) Second,we create the base state class named AIFSMState,it derived from the FSMState,

```
public class AIFSMState : FSMState
{
    protected AIController m_AIController;

    4 个引用
    public AIFSMState(string fsmStateName, AIController aiController)
        : base(fsmStateName)
    {
        this.m_AIController = aiController;
    }
}
```

This is the base state of all the other state of AI Tank,it create the relation between it and the AIController instance.



Then,we create some states which AI tank logic must have,they are the patrol state,the chase state,the attack state and the die state.These states all derived from the AIFSMState Class.

(3) Next we create some necessary condition classes,these conditions determine if one state can switch to another state. For example, there is a transition link the patrol state to the chase state,in its condition list has a condition named *PatrolToChaseCondition*,this condition check if there have a player tank in the sight view of this AI tank, if one player tank in the sight view of the AI Tank,then AI tank select this player tank as chase target,and return true to notify the finite state machine to switch patrol state to the chase state.otherwise it will always return false to notify the finite state machine keep the AI tank stay in patrol state.

(4) Finally In the AI Controller script, we begin to initialize the Finite State Machine system,create some necessary states and add them to the Finite State Machine,then create the transitions among them.Next we set the default and the end state.In the end,invoke the finite state machine OnInitialize() function.

```
private void FSMInitialize()
{
    //create the finite state machine
    FiniteStateMachine = new FiniteStateMachine();

    //create states and add them to the finite state machine we just created
    FiniteStateMachine.AddState(new PatrolState("PatrolState", this));
    FiniteStateMachine.AddState(new ChaseState("ChaseState", this));
    FiniteStateMachine.AddState(new AttackState("AttackState", this));
    FiniteStateMachine.AddState(new DieState("DieState", this));

    //according to the transition conditions, creating the transtion between states.
    FiniteStateMachine.CreateFSMStateToAnotherFSMStateTransition("PatrolState", "ChaseState", new IFSMTransitionCondition[1] { new PatrolToChaseCondition(this) });
    FiniteStateMachine.CreateFSMStateToAnotherFSMStateTransition("PatrolState", "AttackState", new IFSMTransitionCondition[1] { new PatrolToAttackCondition(this) });
    FiniteStateMachine AIController.FiniteStateMachine
    ition("ChaseState", "AttackState", new IFSMTransitionCondition[1] { new ChaseToAttackCondition(this) });
    ition("ChaseState", "PatrolState", new IFSMTransitionCondition[1] { new ChaseToPatrolCondition(this) });

    FiniteStateMachine.CreateFSMStateToAnotherFSMStateTransition("AttackState", "ChaseState", new IFSMTransitionCondition[1] { new AttackToChaseCondition(this) });
    FiniteStateMachine.CreateFSMStateToAnotherFSMStateTransition("AttackState", "PatrolState", new IFSMTransitionCondition[1] { new AttackToPatrolCondition(this) });

    FiniteStateMachine.CreateAnyFSMStateToFSMStateTransition("DieState", new IFSMTransitionCondition[1] { new AnyToDieCondition(this) });

    //set default state
    FiniteStateMachine.SetDefaultState("PatrolState");
    //set the end state
    FiniteStateMachine.SetEndState("DieState", new IFSMTransitionCondition[1] { new DieToExitCondition(this) });

    //initialize the fsm before we running it
    FiniteStateMachine.OnInitialize();
}
```

(5) Put the FSMInitialize() function to the AI Controller's Awake() function:

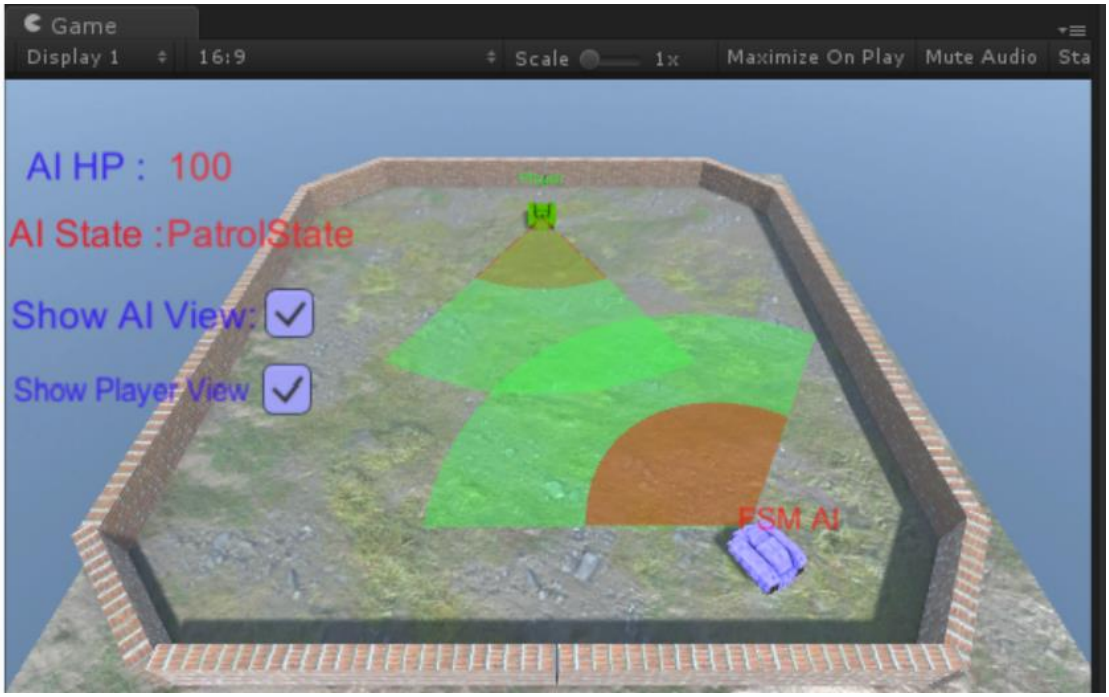
```
protected override void Awake()
{
    base.Awake();

    this.NavMeshAgent = GetComponent<NavMeshAgent>();

    this.Animator = GetComponent<Animator>();

    this.TrackRender = aiTankTrack.GetComponent<Renderer>();

    this.MainBodyRender = aiTankMainBody.GetComponent<Renderer>();

    this.NozzleRender = aiTankNozzle.GetComponent<Renderer>();

    this.ParticleSystemArray = explosionEffect.GetComponentsInChildren<ParticleSystem>();

    //Initialize FSM process
    FSMInitialize();

    m_FSMUpdateTimer = Time.time + updateInveral;

    LifeSystem = GetComponent<LifeSystem>();
}
```

(6) Put the the finite state machine's OnUpdate() function to the AI Controller's Update() function:

```
0 个引用
void Update()
{
    //The frequency we update the FSM
    if (Time.time > m_FSMUpdateTimer)
    {
        m_FSMUpdateTimer = Time.time + updateInveral;

        //find all tanks which tag are "PlayerTank"
        PlayerTanksInScene = GameObject.FindGameObjectsWithTag("PlayerTank");

        //Update FSM
        FiniteStateMachine.OnUpdate();
    }

    // when tank is moving,offset tank track texture
    AnimatingTankTrack();

    //show this tank current life value
    UIManager.GetInstance().UpdateAIHP_AIState_DisplayInfo(LifeSystem.CurrentLifeValue.ToString(), F
}
```

(7) Now,we have finished the AI tank's finite state machine.

4. Running the tank demo scene



(1) As the player tank, you use 'W' and 'S' Key to make it move forward and backward,use 'A' and 'D' Key to make it rotate left or right,use the space key to make player tank openfire

(2) The *AP HP* UI text display the AI tanks life value,and the *AI State* UI text display the AI tank's current state.*Show AI view* toggle will determine whether the AI tank draw the sight view(in green color) and attack view(in red color) when game is running.

5. Use this Finite State Machine Framework to create your own AI

You can write your own AI system by using this finite state machine framework,and you can the refer to the coding style of the finite state machine system in the tank demo,when you writing yours.

6. Contact

If you have technical question, please contact me by email. My email address is 18311310080@163.com