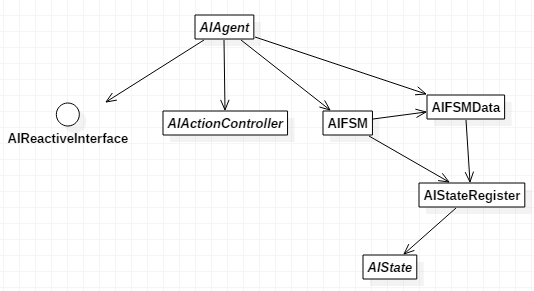
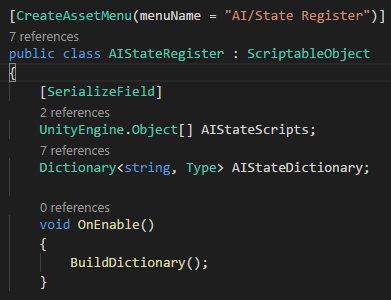
**AI模块技术框架**

1. 核心类



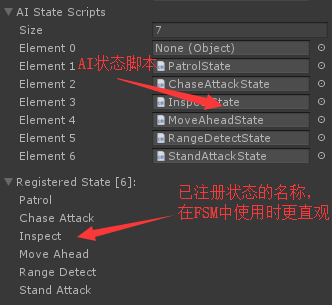
1. AIStateRegister

状态注册机，每一个新增加的AI状态需要在该状态机中注册过才能在FSM中使用。该类为ScriptableObject，将其中数据保存到asset文件中保证全局唯一，当状态注册机中的状态发生改变时，状态机内的引用也会动态更新。其中保存了一个字典，使用时可以简单的使用状态名得到对应的状态类型。



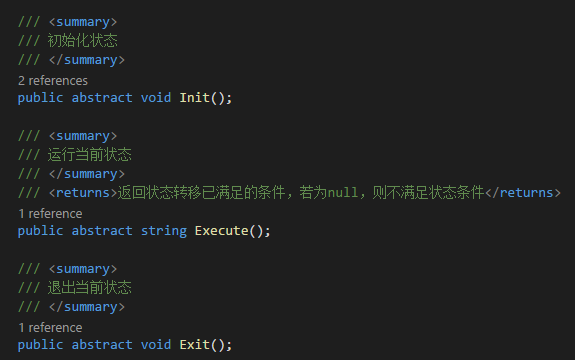
为该类编写了editor，使用时只需要将状态脚本拖入即可。状态脚本需要实现一些公共的接口，若拖入与状态无关的脚本，会判定为无效脚本，已注册状态处不会显示。





2. AIState

AI状态的抽象类，在里面实现了每个状态都需要的初始化操作，然后每个不同的状态需要实现公共的抽象函数来给FSM调用。

3. AIFSMData

ScriptableObject，用于将状态机的状态转移图保存为asset在FSM中确定状态转移。为该类编写editor，使用时通过添加状态，确定每个状态不同返回条件的下一个状态和确定初始状态快速创建状态机。状态都是在状态注册机中注册过的状态，下一个状态和初始状态是该状态机内拥有的状态。



4. AIFSM

控制状态机更新的脚本，通过传入AIFSMData进行初始化和运行其中的状态逻辑。

初始化时动态将状态脚本挂载到gameObject上，所以设置好AIFSMData后，不需要手动将各个状态的脚本拖到对象上。挂载完脚本后，选择初始化状态运行，正式开始状态机逻辑。



状态机更新先运行当前状态，然后判断是否满足该状态的某个条件，如果满足，则从状态转移图中获取下一个状态，退出当前状态，初始化下一个状态。



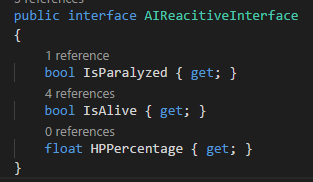
5. AIActionController

抽象类，用于控制AI的动画运行及动画的网络同步，里面定义了许多虚函数，可以在每个状态中进行调用。由于目前网络同步中用到的RPC不支持函数重载，每个需要使用RPC的函数名都必须唯一。



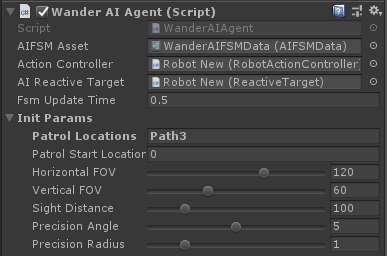
6. AIReacitiveInterface

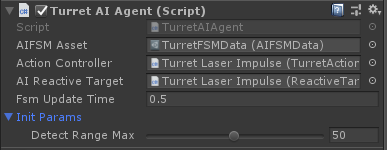
AI受击需要实现的接口，用于获取AI被玩家攻击的一些参数。



7. AIAgent

AI代理的抽象类，用于管理FSM的更新，控制AIActionController调用以及AI对玩家的攻击做出对应行为。不同的代理是设置不同状态机中状态需要获取的参数。可以设置状态机的更新频率，因为里面需要调用RPC，而网络方面不支持每帧都发数据的高频操作。

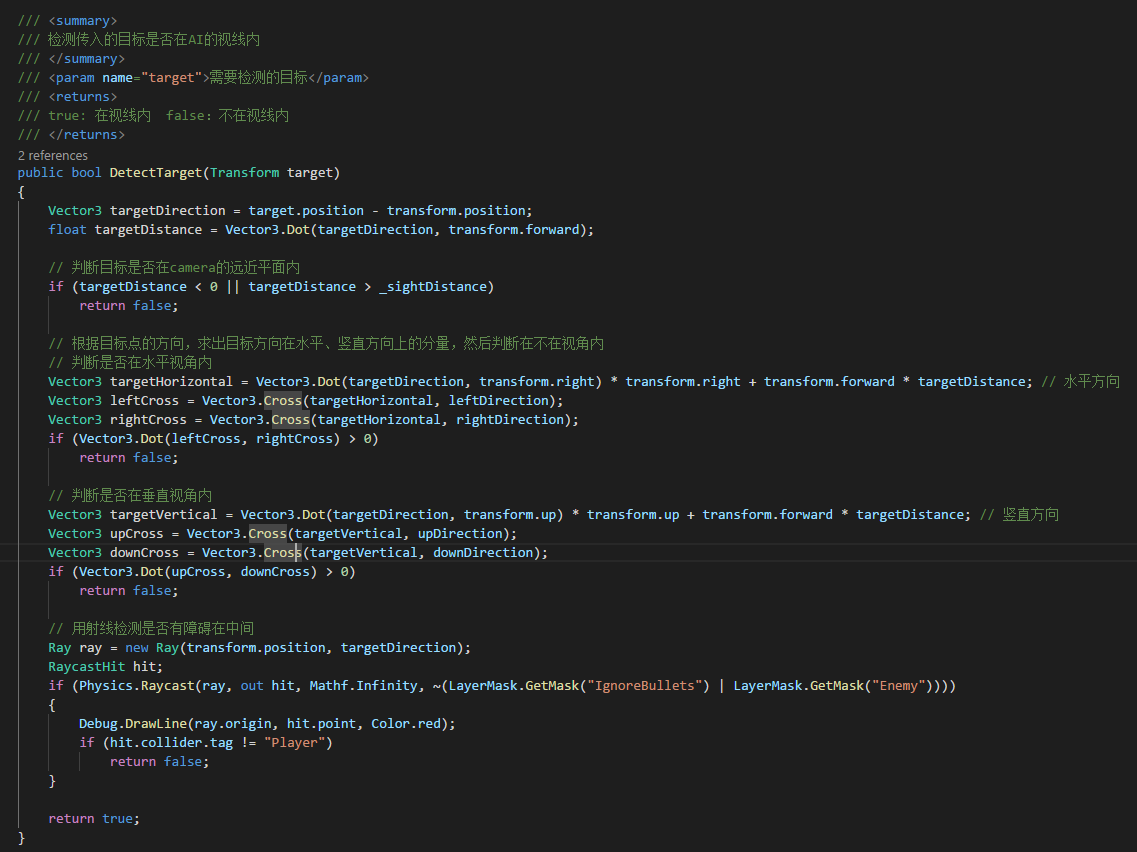




二、工具类

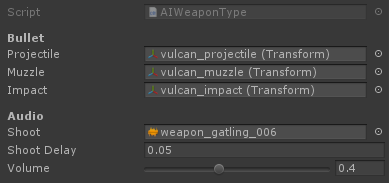
1. AICamera

为了AI能够在场景中看到和玩家一样的物体，为AI写了一个跟玩家透视投影摄像机一样的检测区域，通过设置垂直水平FOV及可视距离来调整摄像机的检测范围。先判断目标是否在远近平面内，然后判断目标是否在垂直水平FOV内，最后判断AI与玩家之间是否有障碍来确认一个玩家是否在视线内。



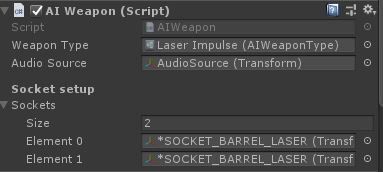
2. AIWeaponType

设置AI的不同武器类型，保存为asset文件



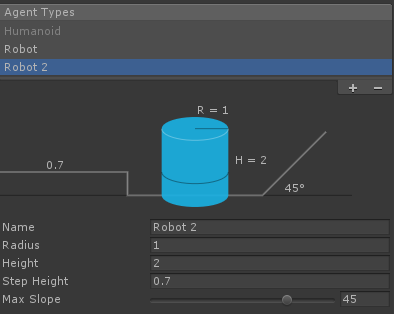
3. AIWeapon

通过设置AIWeaponType确定武器类型，添加枪口位置用于发射子弹。射击是通过定时来实现的，这样即便FSM的更新频率很低，射击的频率也不会受影响。

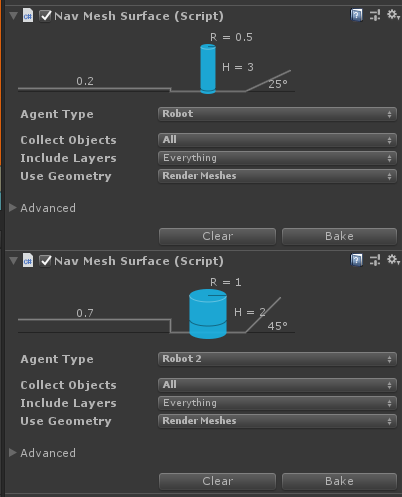


4. Navigation

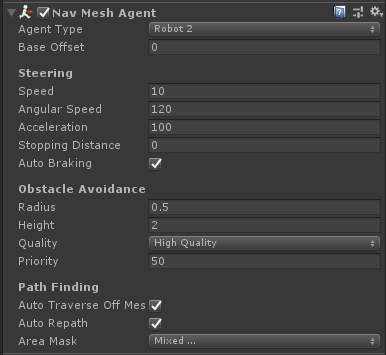
Unity提供的寻路插件，可以设置不同的agent实现对不同大小物体的寻路。



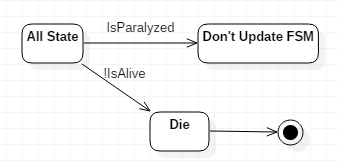
使用前还需在场景中烘焙寻路网格



最后将需要寻路的物体挂上NavMeshAgent即可

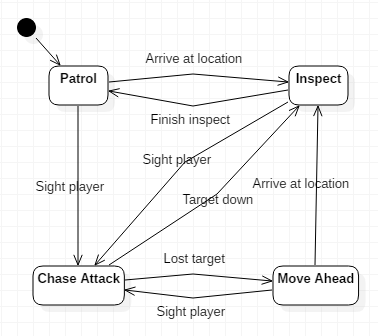


1. 目前拥有的AI状态机



1. WanderAI





2. Turret

