需求:

你是如何实现敌人AI

设计：

使用FSM有限状态机，每个状态表示一个游戏角色的状态，每个转换表示从一个状态到另一个状态的条件。

实现：

FSM需要有几个固定脚本，

Transition为状态机转换的条件，

StateID为每个状态机的状态唯一标识，

FSMState为状态基类，其中有状态机的StateID，还有转换条件(用字典存储，Transition为键，StateID为值)，还需要指定状态机所属的状态机管理,方便在子类脚本调用切换状态，

还有添加删除条件，返回想要转换的状态的方法，以及DOBeforeEntering（进入状态之前），DOAfterEntering（离开状态之前），Act（行为），Reason（状态检测）方法

FsmSystem为状态机管理，其中就有当前执行的状态机和添加删除,PerformTransition(切换当前状态的方法)等方法

应用：

假设敌人AI有以下几种状态:

1. Patrol：巡逻
2. Chase：追击
3. Attack：攻击
4. Flee：逃跑

条件

1. LostPlayer：玩家离开感知范围
2. SeePlayer：玩家进入感知范围
3. AtkPlayer：玩家进入攻击范围
4. FleePlayer：血量低于阈值，逃离玩家

首先我们初始化一个状态机系统，用来管理我们所有状态的添加、移除、切换等，再定义每个状态类，每个类都继承与FSMState状态基类，并实现DOBeforeEntering()，DOAfterEntering()，

Act()，Reason()方法，再去添加我们状态间的转换条件，例如:

1.Patrol -> Chase: SeePlayer(玩家进入感知范围)条件满足时，从Patrol状态切换到Chase状态。

2.Chase -> Patrol : LostPlayer(玩家离开感知范围)条件满足时，从Chase状态切换到Patrol状态。

3.Chase -> Attack: AtkPlayer(玩家进入攻击范围)条件满足时，从Chase 状态切换到Attack状态。

4.Attack -> Chase: SeePlayer(玩家离开攻击范围)条件满足时，从Attack状态切换到Chase 状态。

5.Chase/Patrol/Attack -> Flee: FleePlayer(血量低于阈值,逃离)条件满足时，则切换到Flee状态

游戏开始时，将状态机当前状态设为Patrol，然后在游戏的每一帧中，状态机管理器都会检测当前状态并执行Act和Reason方法，当满足转换条件时，去调用状态基管理器中的PerformTransition()方法切换到新状态，当切换状态时，先调用旧状态的DOAfterEntering()方法，再去调用DOBeforeEntering()方法对状态进行初始化和清理

使用FSM状态基碰到过哪些问题？

常见来说的话就是状态数量太多，导致状态难以管理和理解，这块设计时最好考虑清楚，去平衡我们状态数量和状态转移的复杂性，设计时尽可能的简洁，避免创建不必要的状态和过于复杂的状态转换逻辑