辅助系统

虚幻四高级程序开发专业

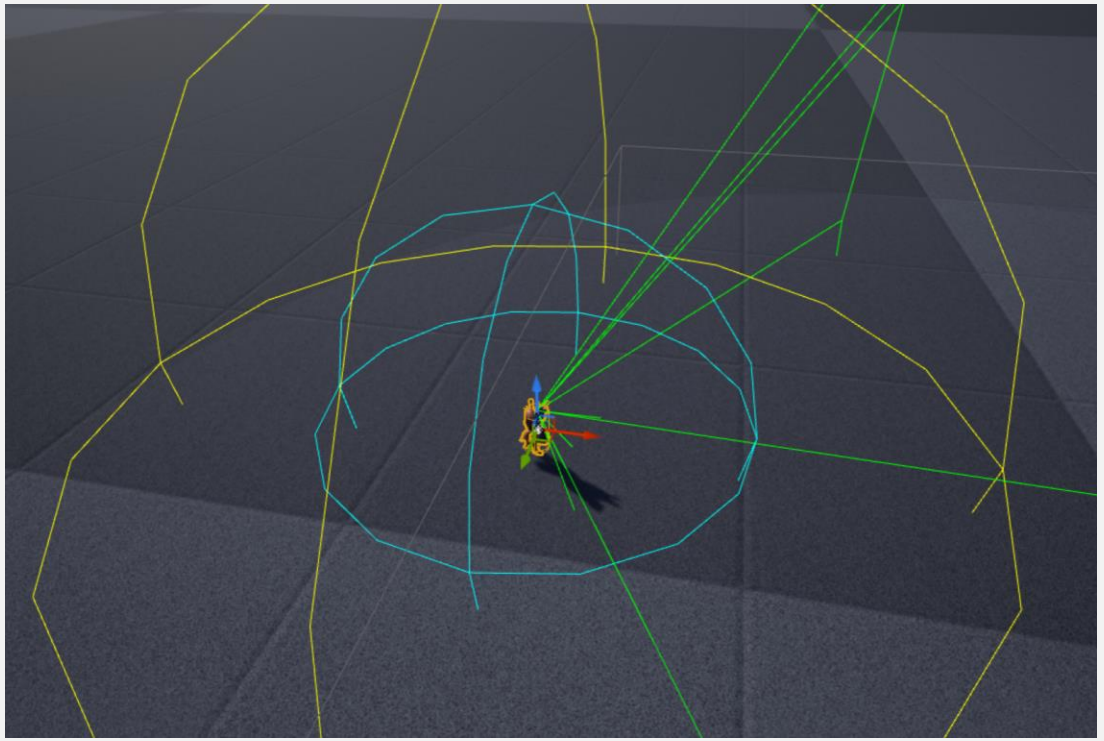
01

视觉听觉组件

视觉听觉

在虚幻中，为玩家提供了视觉和听觉组件，他可以为玩家提供视觉信息获取和听觉信息获取（获取结果为位置和发生对象），玩家可以根据获取信息进行逻辑操作，以裁定执行的动作。

注意：当目标被看到，就不会检查目标是否发出噪音，也就是目标站在视觉观察区，制造噪音将不被听到



UPawnSensingComponent

UPawnSensingComponent（旧版本）组件为我们提供了听觉反馈和视觉反馈，如果你希望获取到视觉和听觉能力，可以添加此组件。

如果你（Pawn）希望获得此能力，就添加组件即可。添加完成后需要绑定通知事件，可以获得到通知结果。

绑定函数原型

```
UFUNCTION()
void OnSeePlayer(APawn* Pawn);
UFUNCTION()
void OnHearNoise(APawn* PawnInstigator, const FVector& Location, float Volume);
```

绑定

```
SensingComp->OnSeePawn.AddDynamic(this, &AEnemyCharacter::OnSeePlayer);
SensingComp->OnHearNoise.AddDynamic(this, &AEnemyCharacter::OnHearNoise);
```

设置

参数设置可以参照下面的设置代码

```
SensingComp = CreateDefaultSubobject<UPawnSensingComponent>(TEXT("SensingComp"));
//视距
SensingComp->SightRadius = 2000;
//视力角度
SensingComp->SetPeripheralVisionAngle(60.0f);
//忽略遮挡可以听到声音的距离
SensingComp->HearingThreshold = 600;
//在没有遮挡的情况下听到的声音最大距离
SensingComp->LOSHearingThreshold = 1200;

SensingComp->OnSeePawn.AddDynamic(this, &AEnemyCharacter::OnSeePlayer);
SensingComp->OnHearNoise.AddDynamic(this, &AEnemyCharacter::OnHearNoise);
```

噪音发射器

当角色挂载了感知组件，默认玩家只要进入视觉范围就会被观察到。但是噪音就无法通过简单的构建完成，我们需要在发出噪音的角色身上添加噪音发射组件UPawnNoiseEmitterComponent，然后在角色身上调用MakeNoise函数，即可发出噪音。

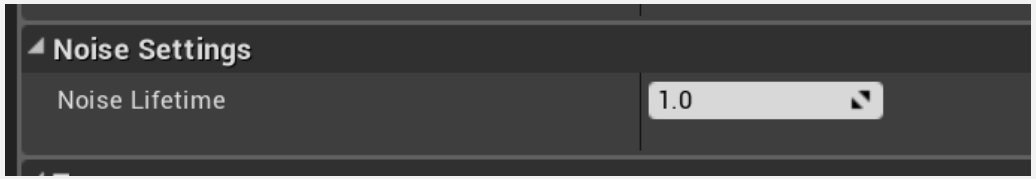
注意：噪音的大小会影响传递的距离，噪音为1则在最远处也可以传递，0.5则距离减半。MakeNoise函数在Actor身上，只要是A类均可调用

MakeNoise中的最大范围参数可以增加声音传播范围，但仅适用于AIPerception

听到2次?

当我们调用MakeNoise时，却获得了两次声音的传递结果，这是为什么呢？因为在噪声发射组件中，默认噪声存在的周期时1秒（噪声组件细节面板），而听觉组件的检查周期是0.5秒，所以导致噪声会被听到两次。

噪声组件



感知组件

02

AIMessage

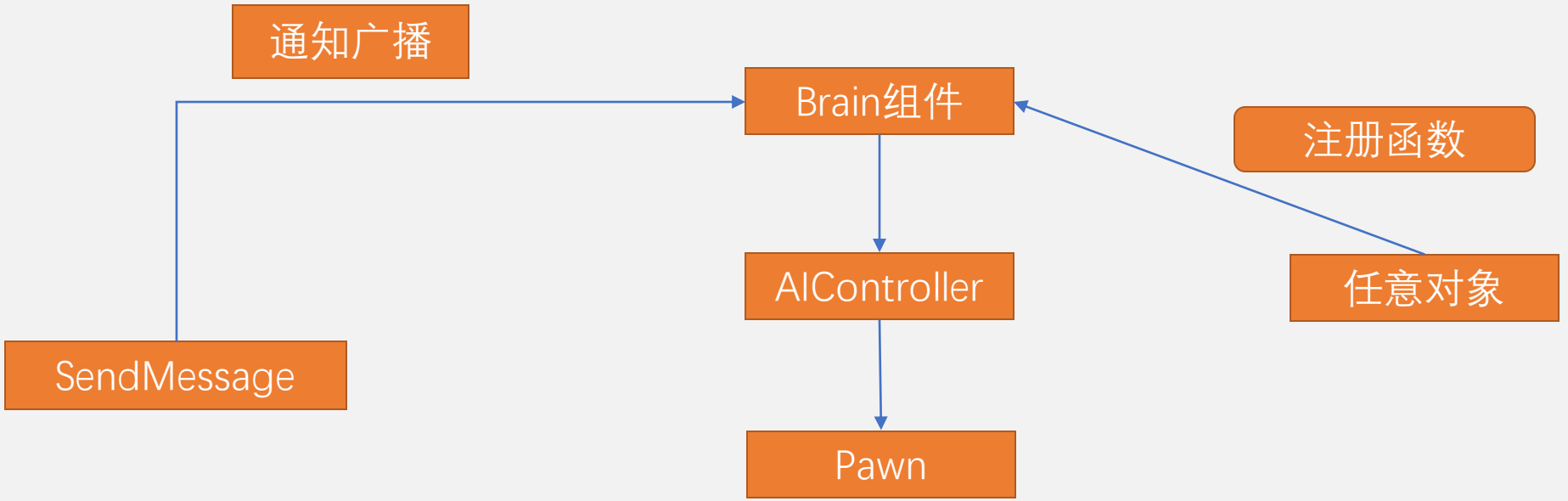
AIMessage

AIMessage允许我们任何时候通过发送指令的方式，向C++中（目前蓝图不支持）发送一条指令。如果希望接收指令，则需要讲回调函数绑定到角色（带有AiController并且添加了BrainComponent组件）身上。然后通过SendAIMessage方式即可向绑定函数发送消息。

注意：发送消息和绑定函数均需要提供MessageName，用来裁定发送消息时传递给哪个函数。

优势：更灵活的通过代理的方式讲动作进行定制广播，接收消息的人不受到任何限制，但是只有对象持有BrainCompoent组件方可使用。

AIMessage



操作步骤

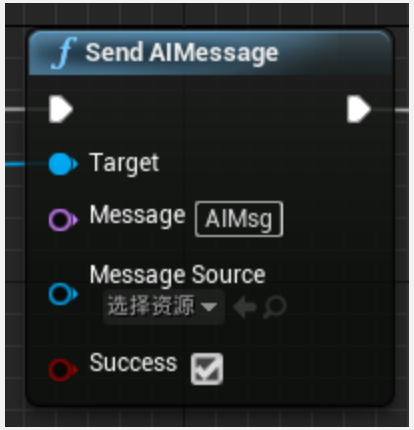
1.构建回调函数（原型如下），在需要接收信息的对象身上添加

```
void ReceiveAiMessage(class UBrainComponent* obj, const struct FAIMessage& msg);
```

2.注册接收函数（实例代码如下），由于是测试代码，所以将回调通知写到了怪物Pawn身上，所以第一个参数传了this。注意消息名称是接收消息的重要依据。**并且需要有效持有返回值Handle（返回对象是共享指针对象，如不有效持有，添加进去的绑定将会在调用结束后被删除，无法响应结果）**

```
ObHandle = FAIMessageObserver::Create(this, TEXT("AIMsg"), FOnAIMessage::CreateUObject(this, &AEnemyCharacter::ReceiveAiMessage));
```

3.调用发送消息，切记名称必须存有注册



03

感知系统

感知系统

在AI设计中，感知对于AI来说非常重要！AI的所有动作产生均来自于外界数据的变化，例如玩家的进入区域，离开区域，发出噪声等。在虚幻中，为我们提供了一套强大的感知系统，可以帮助我们构建更加强大的数据获取方式，供行为树使用！

感知一般分为感知者和感知源，感知的类型分为伤害，视觉，听觉，预知，团队，触碰。

Perception AI System

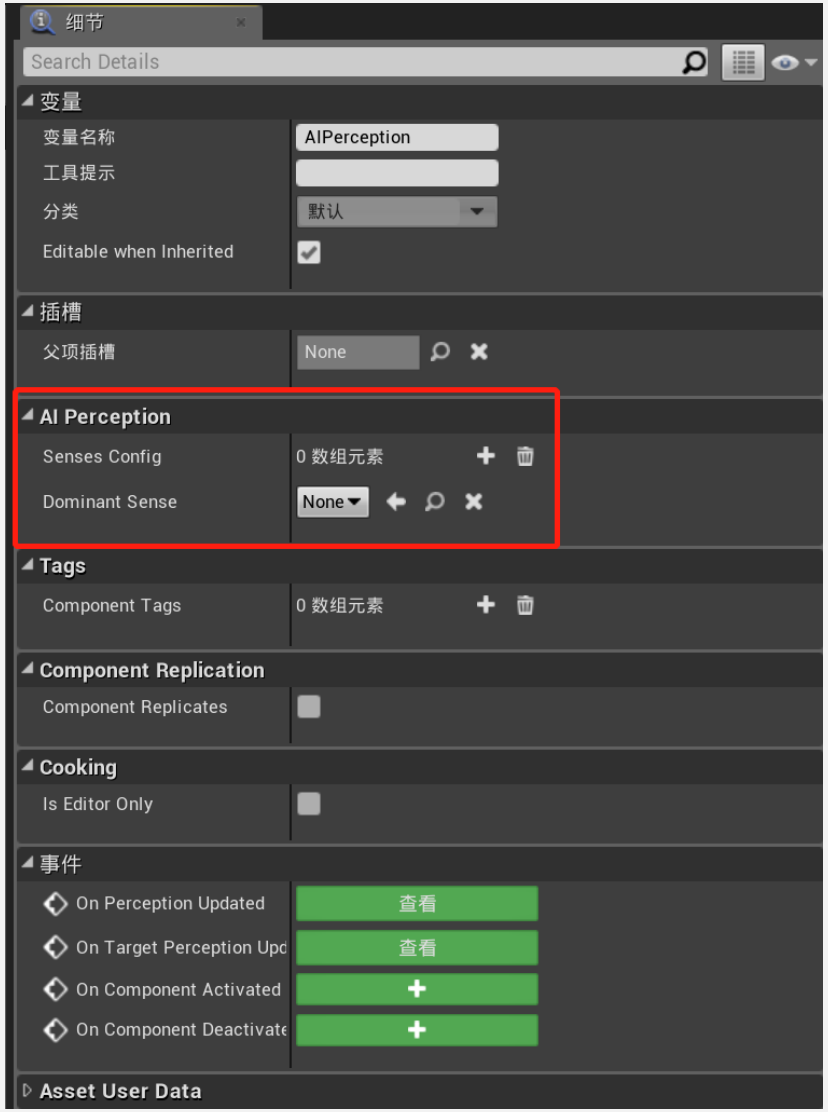
感知系统由两个重要的组件构成

AIPerception组件：主要用来接收感知结果。需要将组件附加到**AIController**上。

AIPerceptionStimuliSource组件：附加到希望**被侦测的角色身上（Pawn）**，被当作感知源。感知组件会关注感知源，而忽略不具备感知源的对象。

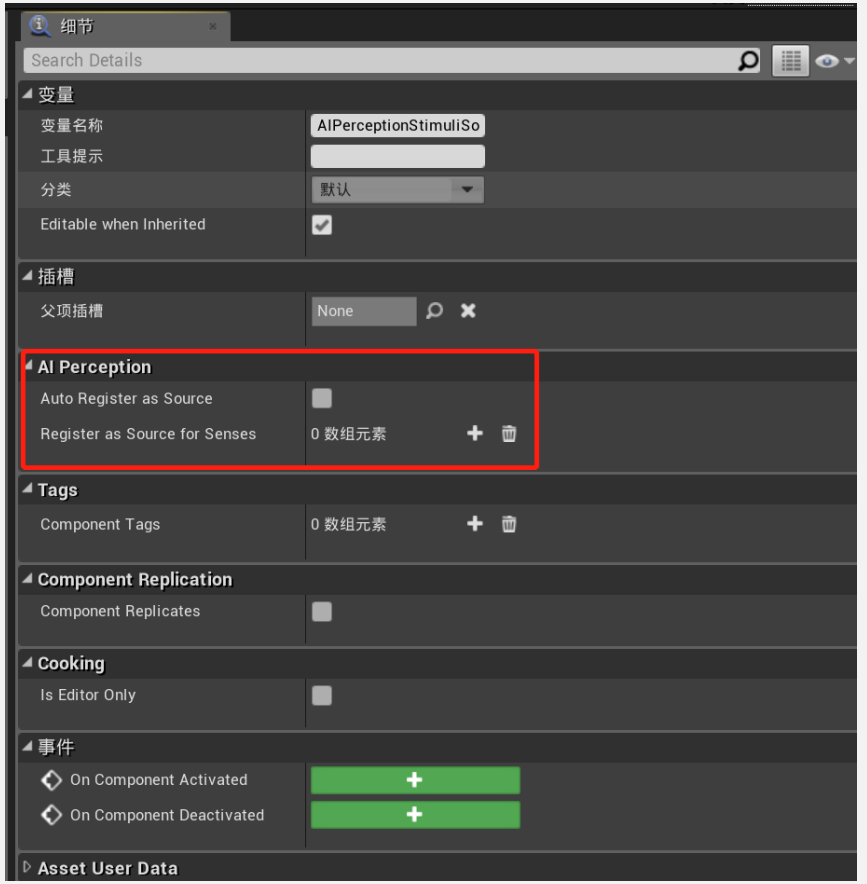
AIPerception

感知组件必须要添加感知配置信息，配置信息决定了感知组件需要关注的数据内容。



AIPerceptionStimuliSource

感知源组件，是用来发射感知信息的，需要注册感知源类型。添加后勾选自动注册即可



AI系统调试

运行时，**选中AI角色**，按键盘的“ ’”键，即可查看AI调试信息。

小键盘可以调整显示信息

按键0：显示导航网格数据

按键1：显示AI调试信息

按键2：显示行为树信息

按键3：显示EQS信息

按键4：显示AI Perception信息

04

视觉

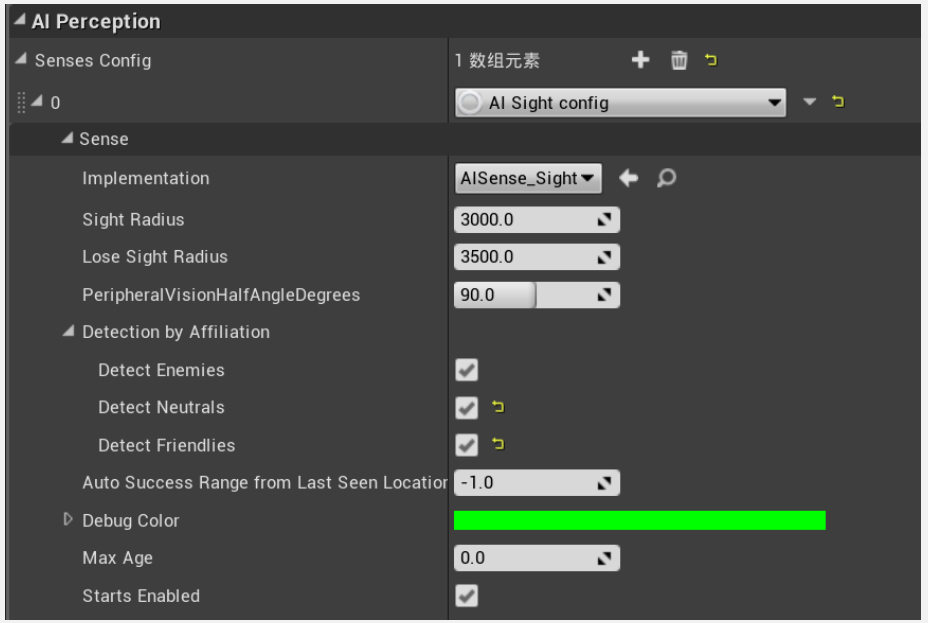
操作

注意：所有的Character都会被默认注册为视觉源，均可以被视觉感知发现。

需要在AIController中添加AIPerception组件，并添加视觉感知。

需要注意：**在进行感知探测时默认只会探测敌人，而所有的角色都没有注册过阵营，默认是中立阵营，不能被感知，所以我们需要勾选全阵营探测**

Auto Success Range... 此参数用来管理当探测到角色后，角色移动偏差多少后再次探测。参数越大，检测精度越低。例如玩家离开了视觉区域，而由于检测距离过大，而无法被检测



注册阵营

阵营是通过ID进行判定。一共可以设定255个阵营ID。如果希望角色具有阵营特性，需要继承接口（**在角色对象类继承**）IGenericTeamAgentInterface，通过SetGenericTeamId函数设置阵营ID，通过GetGenericTeamId即可获得阵营ID

```
virtual ETeamAttitude::Type GetTeamAttitudeTowards(const AActor& Other) const override;
```

比较代码如下

GetTeamAttitudeTowards，用来判定碰到的角色是否是同一个阵营（**如需重写规则，则需要在AIController中重写此函数，完成判定。AIController继承了IGenericTeamAgentInterface接口**）。

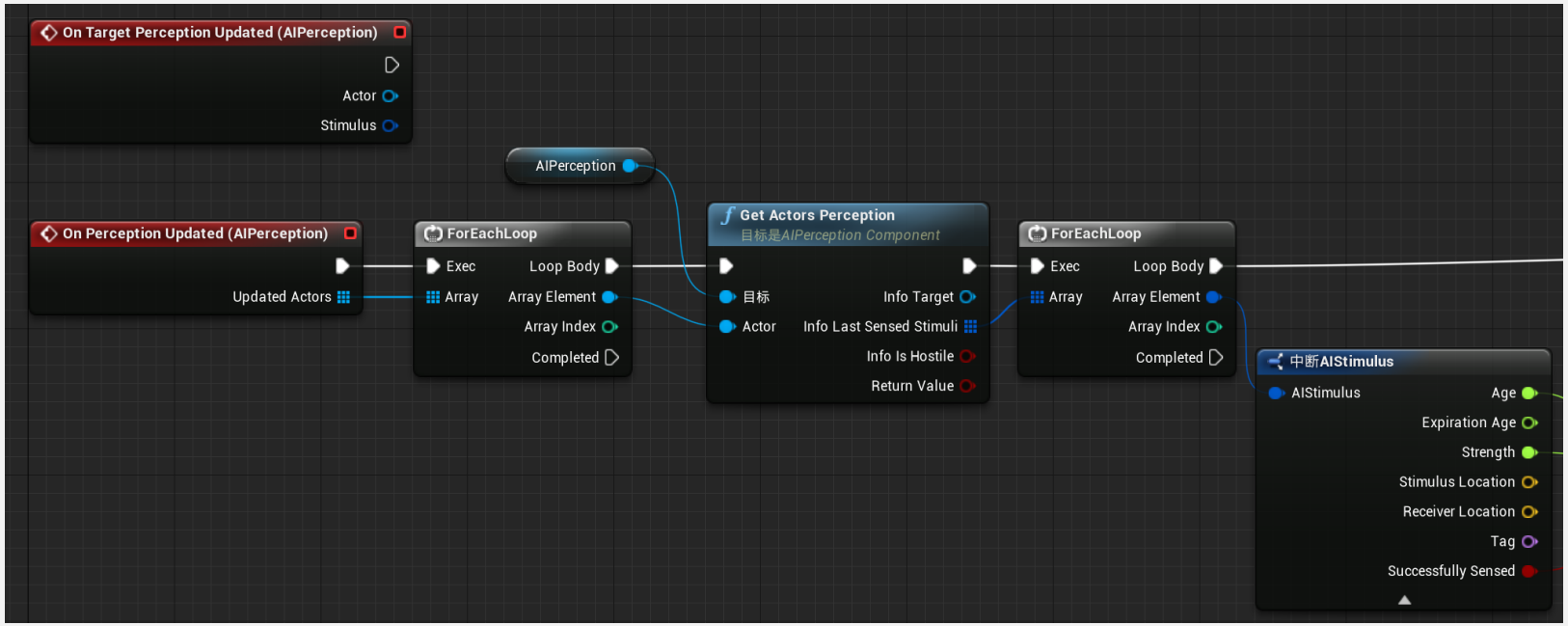
```
ETeamAttitude::Type AEnemyAIController::GetTeamAttitudeTowards(const AActor& Other) const
{
    const IGenericTeamAgentInterface* OtherTeamAgent = Cast<const IGenericTeamAgentInterface>(&Other);
    if (!OtherTeamAgent)
    {
        return ETeamAttitude::Hostile;
    }
    OtherTeamAgent->GetGenericTeamId().GetId();//即可获得目标队伍的ID（整形）
    GetGenericTeamId().GetId();//获取的是自己的队伍ID（整形）
    //根据需求设计逻辑

    return ETeamAttitude::Hostile;
}
```

响应事件

AIPerception组件事件响应：On Perception Updated，On Target Perception Updated。区别，前者返回的当前感知系统所有更新（多目标），后者返回一个目标目标更新。更新中会带有发生源数据

发生源反馈是数组形式（Info Last Senced Stimuli），数组对应感知组件（AIController）中感知配置信息数组中的侦测项。**例如，配置中第一侦测视觉，则数组第一位为视觉反馈，以此类推。**

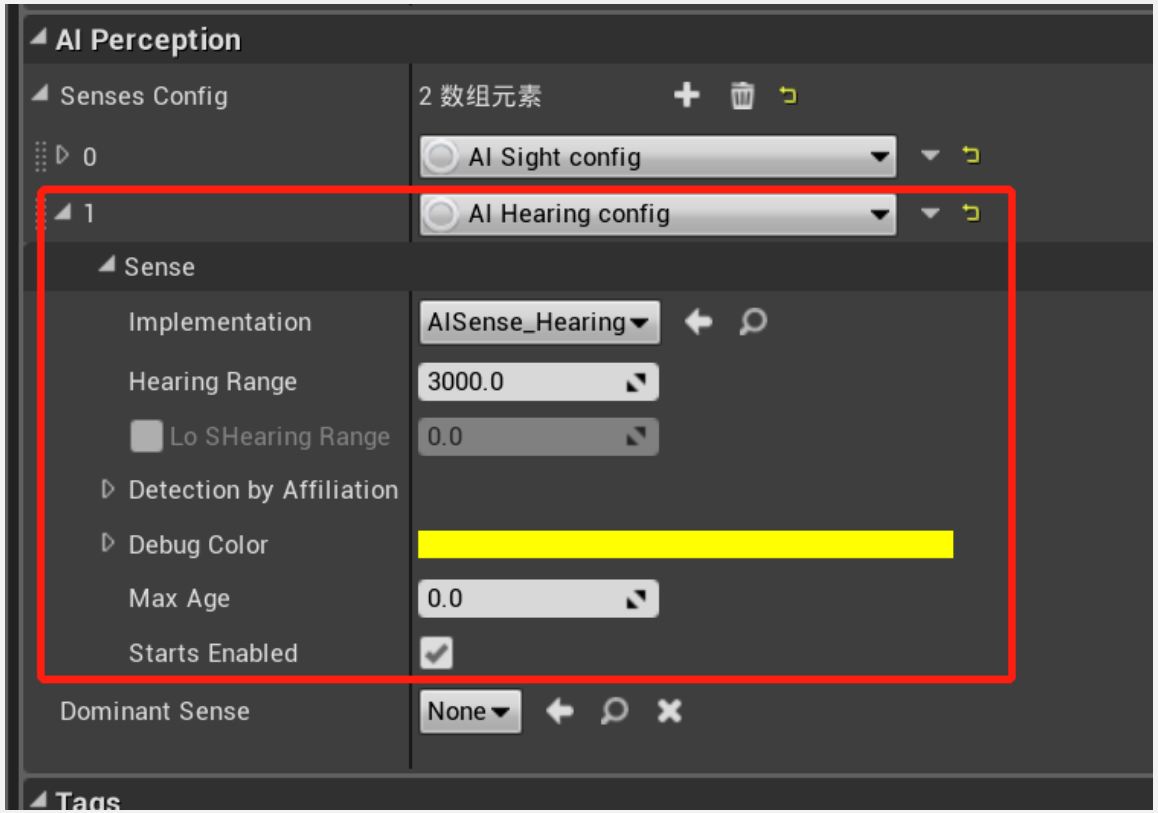


05

听觉

操作

在AIPerception组件中添加听觉配置



发出噪音

调用节点Report Noise Event即可发出噪音。注意Instigator必须是由声音感知源。默认Character会被注册声音感知源

感谢观看