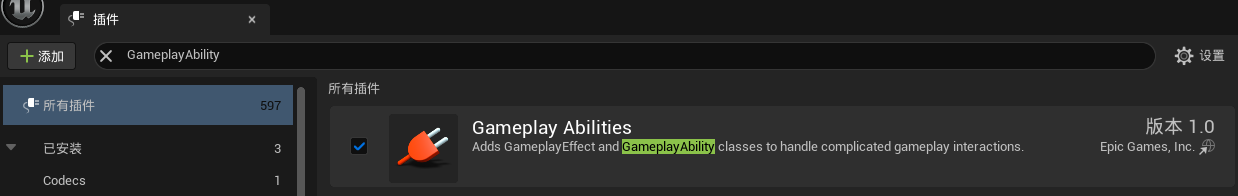
参考文档：

https://www.bilibili.com/video/BV1Jx4y1Z7ig?spm\_id\_from=333.788.videopod.episodes&vd\_source=79fbe818ff96aae3677e123f0374bd91&p=4

首先创建一个空白C++项目，项目名称为CnGas2

创建好后在插件中搜索GameplayAbility



勾选后要求重启以启用。

重启后，在VS中，找到项目文件夹的Source/<项目名>/<项目名>.Build.cs，打开进行编辑。

在这个位置写上一些东西

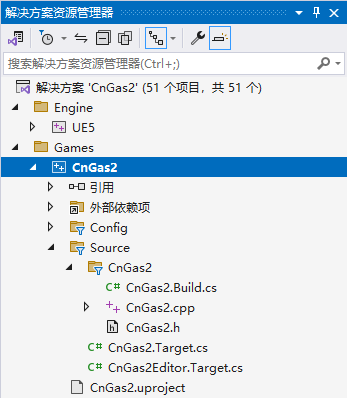


写上

"GameplayAbilities","GameplayTags" ,"GameplayTasks"



然后重新生成一下。右键下图选中的，选择生成。

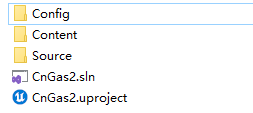
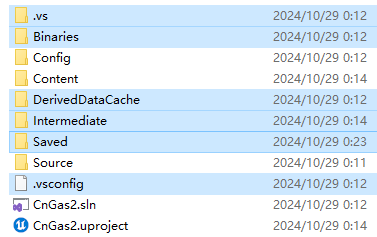


生成时要关掉引擎，不然会报错

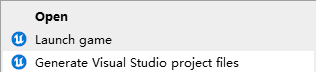


如果报错的话，要从Windows文件管理器中手动生成。

首先要先删除如图选择的东西



然后右键uproject，选择Generate Visual Studio Project Files



构建成功之后，打开uproject。会问你是否要构建，选择是。在打开引擎之后就可以把引擎关掉。

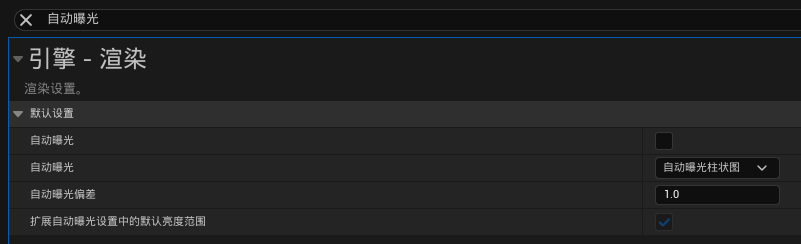
然后把VS打开，用VS来打开引擎。打开VS后，重新生成。

然后重新打开引擎。（目前没有区分是用uproject打开还是用sln打开）

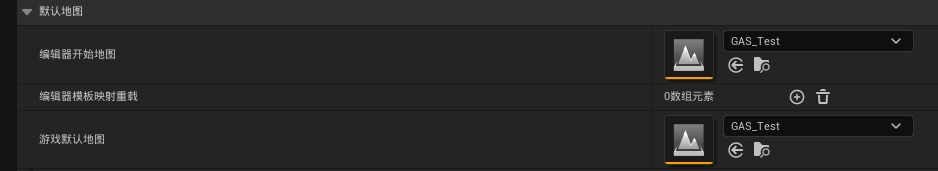
之后我们就可以创建一个新的关卡了。我们选择创建Basic关卡，并放置在创建的Map文件夹下。命名为GAS\_Test

然后进行一些项目设置。

关闭项目的自动曝光



替换项目的地图加载。



然后将Aura的资源导入到引擎中。



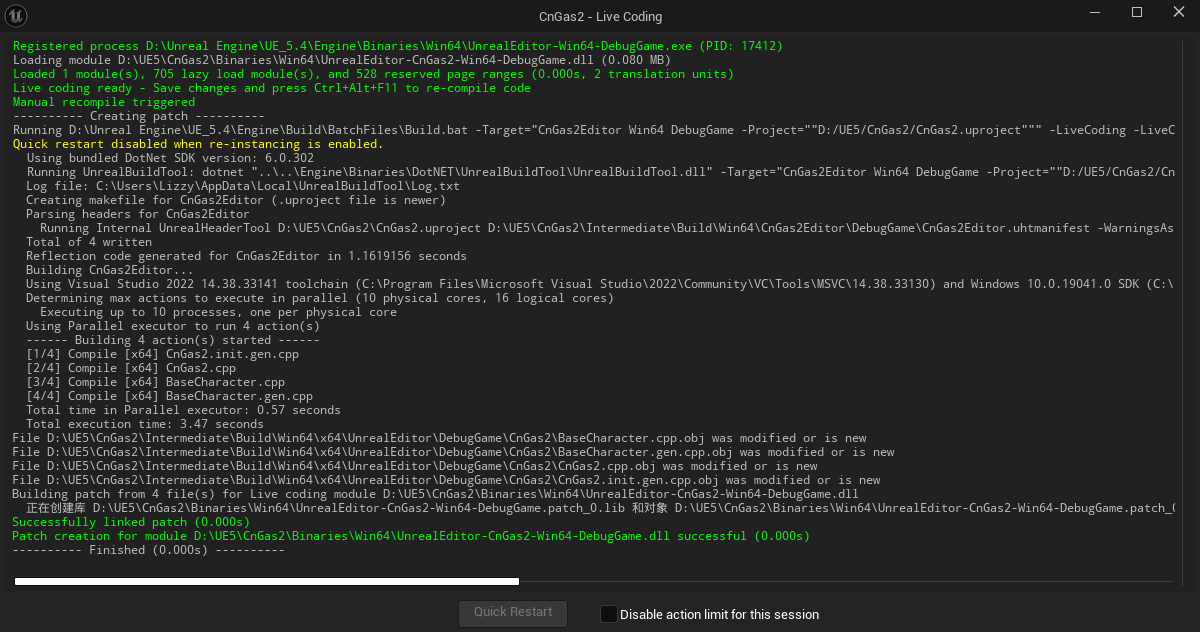
打开项目本地文件，将Aura资源直接放到Content文件夹内。

然后在虚幻商城中找到“虚幻争霸：心菲”，添加到项目

创建一个cpp基类，创建Character类，public，命名为BaseCharacter

因为现在我们是通过vs启动的ue，所以创建的BaseCharacter.cpp会直接在VS中打开。

UE也会出现下面的界面，来表示实时编译。





点全部重新加载

这时在内容浏览器中可以发现All下面有一个新的C++类的文件夹。

我们打开内容文件夹，创建GaneAsset文件夹，在里面实现资产

在GameAsset文件夹中创建文件夹，命名为Character

然后在C++类文件夹中找到刚创建的BaseCharacter，右键选择“创建基于它的蓝图类”



命名为BP\_BaseCharacter，放在Character文件夹下。

基于他创建两个子类，一个是BP\_Player，一个是BP\_Enemy

接下来打开BP\_Player，选择心扉的骨骼网格体。

接下来实现游戏模式等Gameplay

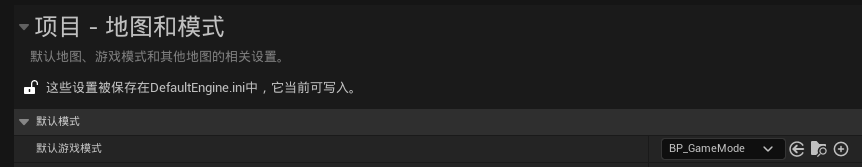
首先创建Gameplay文件夹，在里面创建GameModeBase类蓝图，命名为BP\_GameMode

Gamemode是用来管理游戏获胜规则的一个东西，在服务器端有，客户端没有。

创建BP\_PlayerController

将PlayerController赋值给GameMode，将GameMode赋值给世界场景设置

然后在项目设置中给默认游戏模式也改一下。



将地图中的PlayerStart删除，在场景中创建BP\_Player实例化对象，也就是创建一个心扉

然后可以给BP\_Player加摄像机和弹簧臂。

接下来实现角色与相机控制

相机的两种模式：第一人称、第三人称

移动和跳跃动画的制作

1.角色移动  
先不使用增强输入。在项目设置中使用按键映射。



这就是第三人称模板的。

在心扉的蓝图中将相关的节点复制到BP\_Player中

这个项目不用手柄控制，可以直接把相关节点删了。

它的移动方式是通过鼠标旋转，wasd控制鼠标方向上的前后左右移动。

移动节点族就是复制自心菲的蓝图，按键映射就是第三人称模板的映射。

进入游戏后会发现角色的袖子会由于物理碰撞变得很乱。简单修复一下，直接删除即可。

3.两种相机模式和夹角限制  
接下来修改设置，让角色的移动方式成为第三人称的方式

给BP\_Player的

CharacterMovement的**将旋转朝向运动**勾选（OrientRotationToMovement）

SpringArm的**使用Pawn控制旋转**勾选（UsePawnControlRotation）

根组件的**使用控制器旋转Yaw**取消勾选（UseControllerRotationYaw）  
完成上述三个，就基本完成了第三人称的实现。这也是经典的第三人称的控制方式。

这个设置是控制是以摄像机为中心还是以人为中心。显然，第三人称是以摄像机为中心，第一人称是以人为中心。

但是有的时候，比如释放某种技能的时候，是不能转体的，需要朝向一个特定的方向。这就需要两种相机模式和夹角限制。

夹角限制就是弹簧臂的角度的限制。

通过**获取玩家摄像机管理器**函数，对玩家摄像机管理器的PitchView进行设置最小值和最大值即可。  


然后实现锁定玩家摄像机

创建函数命名为LockCamera，设置形参NeedLock

想要搜索节点，根据前面给出的那三个组件的相关内容的英文名即可搜索到。