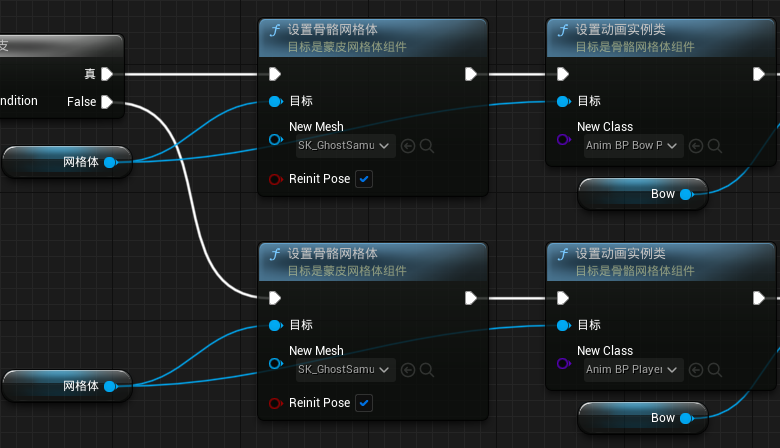
# PlayerPart:

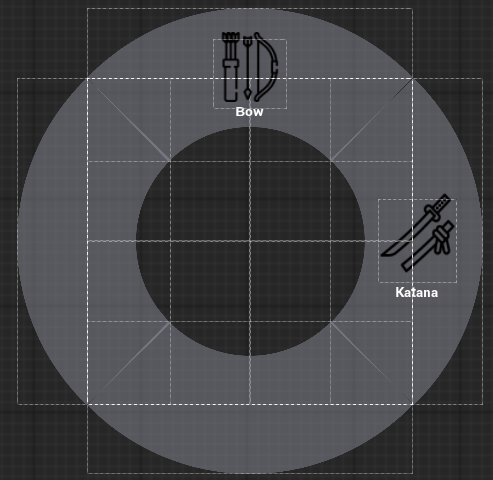
## 玩家骨架网格体/动画蓝图切换：

主体切换节点为：

Set Skeletal Mesh & Set Anim Instance Class

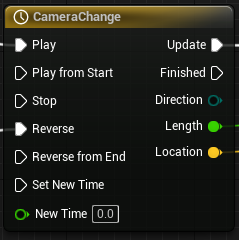


切换方式为创建Widget选择，通过查找鼠标指针位置FLR来确认角度，从而选择设定的目标，目标设定完成后将bool值传递至创建控件的玩家蓝图中，调用分支事件( If )来确认切换目标。



## 摄像机移动：

使用TimeLine来实现较为平滑的相机移动

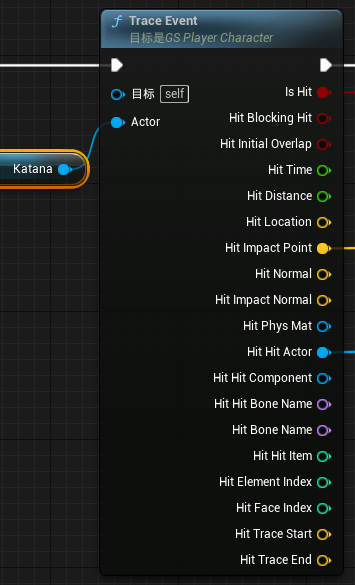


创建曲线设定为所需的移动速度，移动位置等，可以很方便的通过Press Release输入来实现Play和Reverse操作，适用于瞄准等需要一定时间的缓慢值变化的操作使用。

当然，通过插值计算的方式同样可以达成值的线性变化等操作。

## 攻击事件：

这里使用胶囊体通道检测来确认玩家的伤害应用



（此处为C++中定义的Function）

原理为在玩家实现攻击的动画时，触发AnimNotify来开启Trace事件（绑定TimerHandle），使用定时器Tick来达成攻击中的多次判定，判定结果返回至事件中进行后续处理。

由于胶囊体的判定有多次，所以此处需要使用if或DoOnce等方式使得攻击命中后的事件仅实现一次。

## 目标锁定：

### 目标选择：

使用针对Object进行多球体追踪（Multi Sphere Trace For Objects）来进行范围目标确认



Object类型可以使用自定义的碰撞类型（例：Player、Enemy等等）自定义可以防止与其他场景内物体发生碰撞。

通过追踪检测获取到检测物体数组后，使用数组循环来确认各个Actor与自身的距离，取得其中距离最近的单位返回

### 目标切换：

同样进行一次追踪检测，但是对于检测到的Actor数组进行Index判定，将最后输出的Actor转为除上次检测Actor的其他Actor。

锁定的传递：

进行事件时可以增设Interface事件将LockOnWidget消息传递至锁定的目标Actor上，从而使得目标Actor执行Widget的显示或隐藏事件。

## 处决操作：

首先需要对于玩家与敌方单位的距离进行判断，使用LinerTrace函数获取敌方单位是否位于距离范围内。



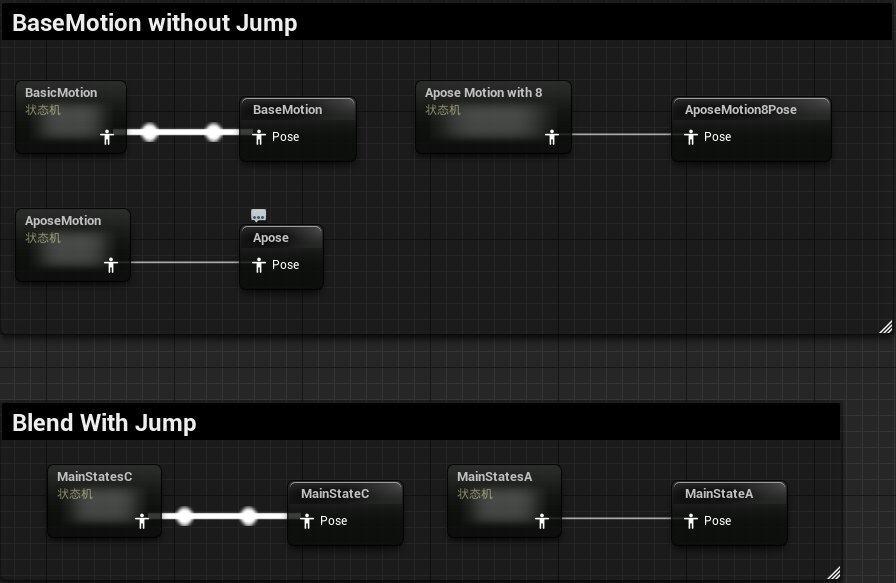
（自定义线性检测）

bool IsHit = false;  
FHitResult Hit;  
Player = UGameplayStatics::GetPlayerCharacter(GetWorld(),0);  
FVector start = Player->GetActorLocation();  
FVector end = Player->GetActorForwardVector() \* 100.f+Player->GetActorLocation();  
IsHit = UKismetSystemLibrary::LineTraceSingle(  
 GetWorld(),  
 start,  
 end,  
 UEngineTypes::ConvertToTraceType(ECC\_Camera),  
 false,  
 ActorsToIgnore,  
 EDrawDebugTrace::None,  
 Hit,  
 true,  
 FLinearColor::Red,  
 FLinearColor::Green,  
 5.0f  
 );  
 return Hit.GetActor();

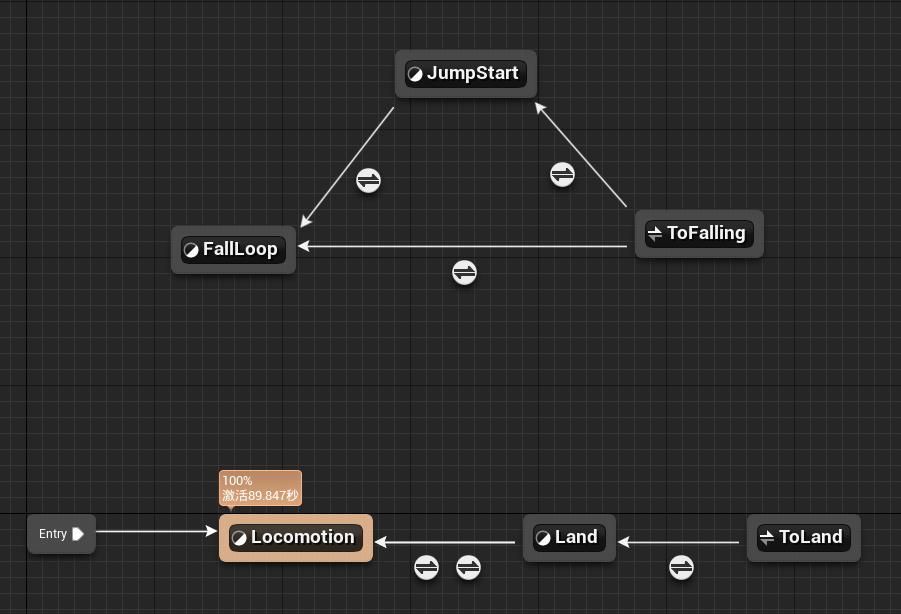
获得检测结果后使用dot对玩家Actor与敌方Actor的向前向量进行计算，若结果近相等于1，则将可以进行处决的消息通过Interface传递至敌方Actor，玩家开始播放处决动画，敌方actor获取消息开始处决事件，同样开始播放被处决动画，若想要提高处决位置的准确性，可以先通过动画确认具体的玩家位置，而后在世界进行时将玩家移动至指定位置。

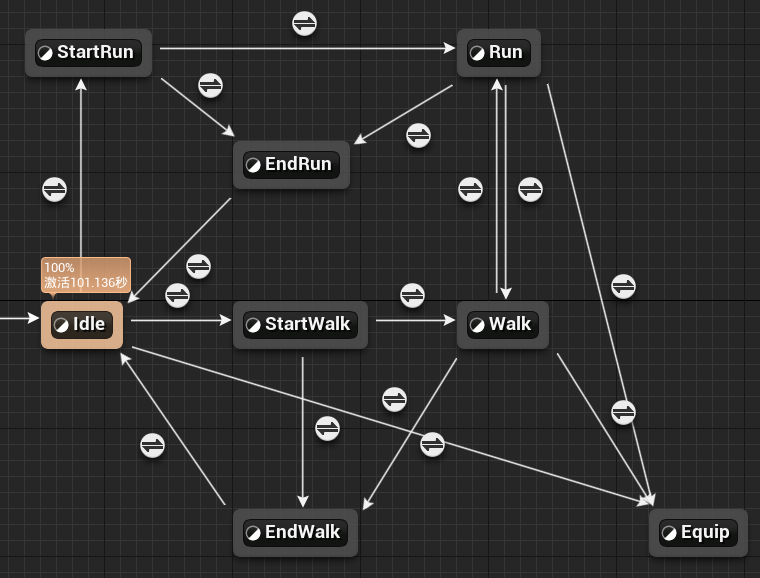
## 动画蓝图：

基本状态机：

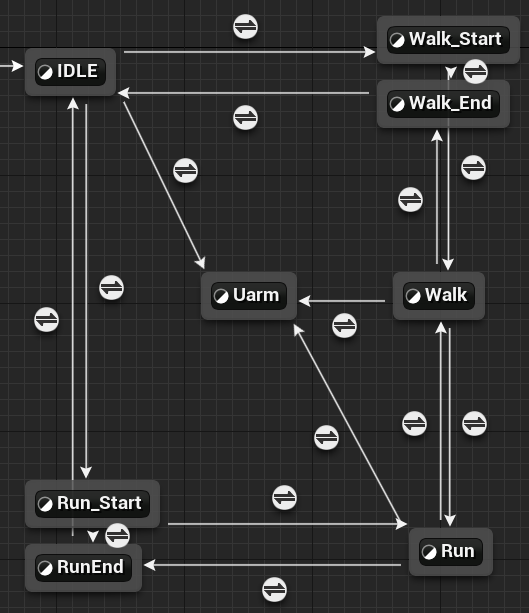


分为基本动作（非战斗）以及Apose动作（战斗状态），而后再混入跳跃状态





通常状态

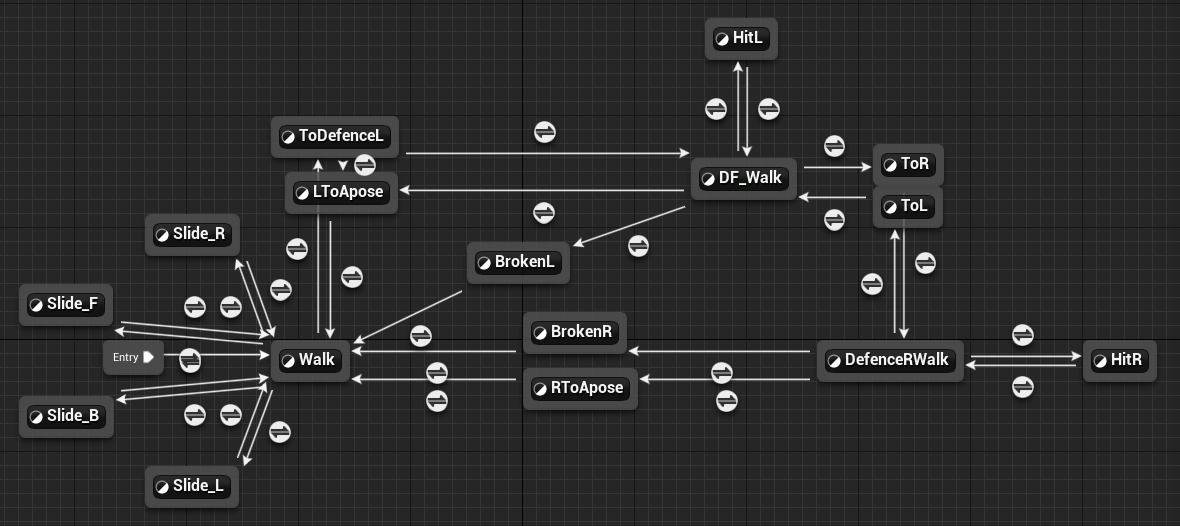


战斗状态

状态机切换时候使用整型混合姿势



在Unarm和Equip状态动画中添加AnimNotify来控制Index为0或1，混入时间清零保证状态切换不存在过渡。



战斗状态下且锁定敌方单位

锁定后通过bool变量再次进行一次状态机切换，将玩家的动作方式通过BlendSpace转变为8方向基本行动，此处除行动外加入翻滚格挡等动画状态。

注意：

善用动画通知来达成动画的状态切换

状态机的动画与动画蒙太奇的混合需要通过插槽来实现，由于蒙太奇的动画在任意状态下都可以启动，所以需要根据动画蓝图中的状态对于蒙太奇播放进行限制（例如：在翻滚进行时无法调用蒙太奇下的攻击动画，攻击进行时无法进行翻滚动作等操作），此类操作都可以通过AnimNotify来进行一定程度的控制。

当使用到MovementComponent进行速度方向等变量的获取时，可能得到的值并不适用于状态机之间的变化中，所以需要额外设定轴值变量（键盘直接输入的轴事件），其值仅为1或0，通过此值可以更加快速的对于方向等值进行简单的判断，状态机变化中的过渡规则取值方式有很多，可以多尝试选择适合当前动画/设计方案的规则。