UE4 GAMEABILITY

2018/6/28

刘连友

1. 目标

这个文档只是记录我在翻阅UE4里GamePlayAbilitySystem时的一些想法。我想让不知道这个系统的人，在看完这个文档后能有一个更加清楚直白的了解这个功能的实现方式。但是这不表示可以不用看代码了，这个文档只是想让大家看代码的时候能够更加顺畅，花费的时间能更加少。

1. 概述

GamePlayAbilitySystem是一个系统的工具，我们可以把它当作游戏里任何角色状态改变的触发器，比如人物从IDEL –> Move -> Jump -> Swim -> Couch…, 血量的改变，法力值的改变，Buff/Debuff 的改变 等等都可以通过这个系统来实现，这里的Ability不能只是解释成技能，更应该说是能力。就像一个石头如果有人所拥有的一切能力，比如有走路的能力，有说话的能力，有吃饭的能力，有思考的能力等等，那我们就可以认为这个石头就是人。我想UE4里的GamePlayAbilitySystem的内在思想（哲学）就是这个。

在具体实现技能的Pipeline，GamePlayAbilitySystem的开发者将这个系统分为3部分：

1. 类似Manager的类：

* UGamePlayAbilityComponent：

它更像一个非单例的一个Manager, 每一个Pawn或者说Character都有只能拥有一个Component,这个规则更像是一个口头规则，因为代码里没有强制规定，而且一个Pawn或者Character有多个UGamePlayAbilityComponent也没有关系。

1. 主要功能部分：

* GamePlayAbility：

这个主要解决是我能做什么

* GamePlayEffect：

这个主要解决我做的事情最终的效果和结果怎么样

这个功能又主要分为2个部分：

* GamePlayCue

这个主要是解决这个技能的效果。别人是怎么知道我用了技能的

* Aggerator:

这个是为了解决我这个技能才生的实际数值效果的方式和结果

1. 辅助功能部分：

* GamePlayTask:
* FPredictionKey:
* GamePlayAbilitySpac， GamePlayEffectSpac:
* AbilitySystemGlobals:

这个是用来存储所有的GameAbilitySystem的全局设定。

* 各种Handle:
* GamePlayAbilitySpecHandle：
* GamePlayEffectSpecHandle:

1. 这些Handle的作用是用来标记需要网络同步的对象。
2. 用这些Handle可以更加方便容易的检索到目标对象。

* NetExecutedPolicy, InstancingPolicy, ReplicatedPolicy

这些策略是用于Ability的生成与执行的

在GamePlayAbilitySystem 还有一个比较隐晦的设计思路：所有的能力可以分为两种类型：可以实例的，不可以实例的。

不可以实例的是指有些能力比如跳，游泳，这些能力是不许要实例。

可以实例的是指那些比如释放一个火球术，这个就需要一个火球的实例在世界中。

需要修改的地方：

1. 因为所有的GameAbility, GameTask, GameEffect 都是继承与UObject的，UE4里面的固有网络通讯都是通过Actor来实现，但是Actor非常庞大，如果有所有的GameAbility都是继承与Actor，那有太多的东西冗余，并且非常需要消耗带宽，所以他就做了一个小小Trick, 通过Handle 来定位需要同步的对象，通过FReplicationKey来同步必要的状态。在UE4里面的也是可以对UObject 来进行网络同步的，但是这个只能用来UObject只能从UPACKAGE读取出来的。而且有些Ability不一定非要详细的同步，所以通过这个方式可以最小化带宽。

TIPS:

* CD0:在UE4里全称是Class Default Object.其实就是有基本值的Class,这里的基本值是指 通过Config文件初始化的对象。可以理解为T::StaticClass();