增强输入系统

这个文档会讨论关于增强输入系统实现蓄力、点摁、组合等功能

创建空白的第三人称模板。我们给角色模板的输入映射上下文添加一个额外的输入操作，来实现攻击功能。默认的bool类型就可以。添加的时候，需要用到手柄，并且将扳机键设置为攻击键。

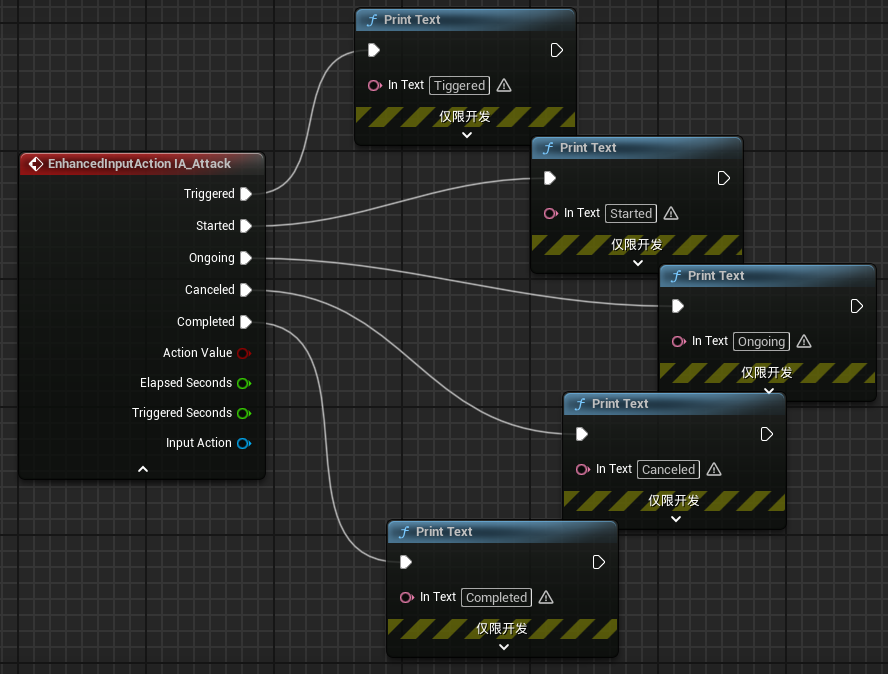
创建子弹蓝图，来简单实现攻击发射子弹





我们重点要关注输入映射上下文中的触发器。

先创建如图的节点族，来方便我们测试。Print的颜色可以设置成不同的颜色。



在输入映射上下文中，当：

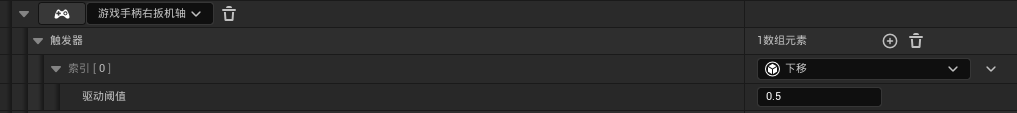
## 没有触发器：

Started、Triggered、Completed能够显示。每帧一个Triggered。

手柄和键盘一样。

## 触发器为“下移”：

将下移理解为下压，因为它类似于压力板的一种反馈机制。而手边上的压力板，就是扳机键。



索引中的驱动阈值，可以理解为当压力达到该值的时候才能触发Started和Triggered和Completed。这一点在扳机键这种压力触发器上能够体现。

对于bool型的输入操作，下移触发器在没有压力触发器的设备（如键鼠）上并不能体现驱动索引的效果。

## 触发器为“已按下”或“已松开”

已按下是输入超出启动阈值的时候触发发射一次。Triggered只能触发一次。

阈值不能设置为0，否则无法触发。一般设置为0.1到1之间。

已松开是超出启动阈值的时候触发返回。假设驱动阈值为0.5，则在摁下摁到0.5之后，会Tick触发Ongoing。因为现在扳机键已经大于0.5了，所以一定会有松到0.5的情况。当松到0.5时，会触发一次Triggered且仅触发一次，并触发一次Completed。

已松开可以理解为：触发器开关在驱动阈值处，当触发时，启动Started和不断触发Ongoing，当结束时，触发一次Triggered和Completed。

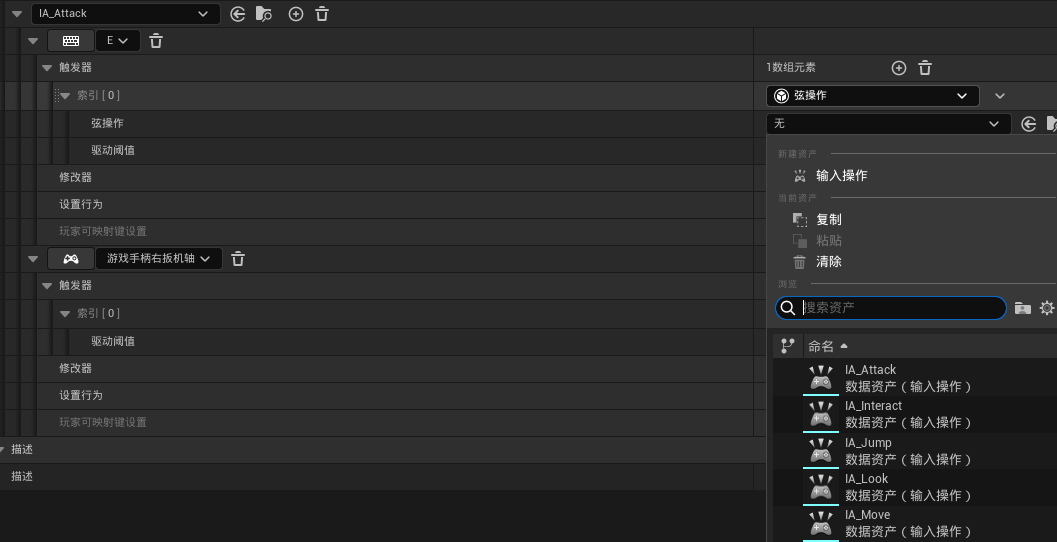
一般也是用于扳机键的。如果用于键盘，驱动值直接就是1。也就是说，如果驱动阈值在0.1到1之间，可以直接触发。

## 触发器为“弦操作”：

这一个用E键或者鼠标左键来做。

应用必须触发的弦操作，以便该触发器的操作进行触发。

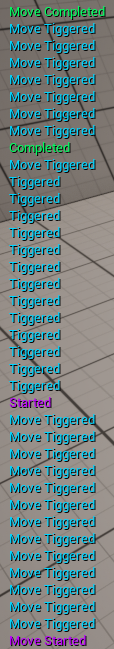
我们需要将弦操作的索引展开，会发现他能捆绑一个IA



意思是，只有当选择的IA触发的时候，该IA才能触发。比如只有在进行移动的时候才能攻击。

当触发时，先触发一次Started，然后按帧触发Tiggered，最后触发Completed。

且当本IA触发时，选择的IA并不能触发。当本IA停止触发时，会在Completed之前触发一次选择的IA。一般这一次可以忽略不计，但是如果有bug需要将它修理。



上述的假定是当移动的时候才能攻击。一般来说，如果我们不需要移动时才能攻击，而是摁住某个按键时才能触发另一个按键，我们只需要新建一个IA即可。

（举例：Ctrl C组合按键）

驱动阈值和压力板概念一样，这里不做过多阐述。

## 触发器为“脉冲”

输入被启动时间隔性的进行发射触发。

超出限制时，就会触发Completed。在释放时，触发Canceled。

自我描述：

* 以间隔为周期，在按键时间内，每个周期触发多次Ongoing和一次Triggered。先Ongoing再Triggered。开始时触发仅用于在Started后面加一帧Triggered，不代表每个周期先Triggered。
* 限制次数指的是Trigger的最多触发次数。触发限制为0则为无限制，或者理解为限制次数无穷大。
* 在触发限制次数内，如果松开按键则触发Canceled。
* 无论是否开始时触发，Triggered的次数一定可以达到触发限制的次数。
* 如果达到限制次数，最后面两帧一定是Triggered和Completed
* 无论任何情况，如果你真的**不幸**在Triggered这一帧松开按键，Triggered后面跟的一定是Completed。



当触发限制为0时

在摁住E键时，先触发一次Started，然后在摁住E键的一段长时间内，每个周期（周期即为间隔）触发一次Triggered和多次Ongoing。在松开E键的瞬间，触发Canceled。当不勾选开始时触发时，只有第一个周期的Triggered不触发。

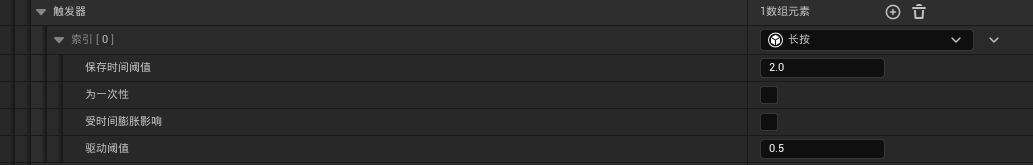
当触发限制不为0时，比如为5时（触发限制是一个整数），就会在Triggered触发5次后Completed。最后一次Trigger直接跟Completed

没有达到触发限制次数时，松开按键和无限制一样，会触发Canceled。

如果达到触发Completed的条件，无论是否勾选开始时触发，触发次数一定等于触发限制，最后一次Triggered后面一定紧跟Completed

警告，无论是否达到Completed的条件，无论是否有触发限制，如果在Triggered这一帧松开按键，也会触发Completed。也就是说，即使没有触发限制，也能触发Completed。经测试，对于12600kf，当间隔为0.0165时，Triggered的触发频率就已经几乎等于Ongoing的触发频率了，仅仅是Triggered频率略高于Ongoing。这时更能测试出上面的警告的情况。

## 触发器为“长按”



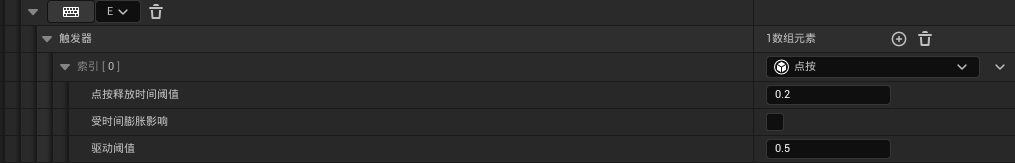
当为长按时，先触发Started，然后在时间阈值内触发多次Ongoing。在达到时间阈值后开始触发Triggered，并在Triggered触发结束之后触发Completed。

如果在时间阈值期间松开按键，则触发Canceled。

如果没有勾选“为一次性”，则可以按帧且无时间上限的多次触发Triggered，并在松开按键的瞬间触发Completed。

如果勾选“为一次性”，则仅触发一次Triggered，且在触发完这一次后触发Completed。触发完Completed之后，无论是否松开按键都不会再触发。

## 触发器为“点按”（时间内需要松开）



必须在点按释放时间阈值秒内启动再释放输入以触发。

首先关注“点按释放时间阈值”，

如果我们一直摁住该按键不松开，则先触发Started，然后按帧触发Ongoing，并在达到时间之后触发Canceled。

如果在时间阈值内松开，则在松开之后触发一次Triggered并触发Completed。在松开之前，也就是触发Trigger之前，依然会按帧触发Ongoing。

关键区别就在于时间阈值内是否松开。如果没松开则触发Canceled，松开则触发一次Triggered和Completed。没松开不会触发Completed。

是否收时间膨胀影响，可以在蓝图中搜索时间膨胀，会有全局时间膨胀和Actor时间膨胀两种。这里只简单提及，不做阐述。

## 触发器为“长按和松开”



当长按按键时，先触发Started，然后按帧多次触发Ongoing。

如果在时间阈值内松开按键，则触发Canceled；如果在时间阈值外松开按键，则触发一次Triggered和一次Completed。

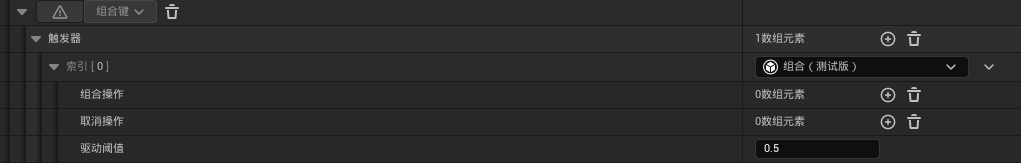
“点按”和“长按和松开”是比较对称的。

“长按”和“长按和松开”的区别在于，达到时间阈值之后该干什么。如果是长按，达到时间阈值之后应该接着摁，来多次Triggered；如果是“长按和松开”，达到时间阈值后应该松开，来一次Triggered。

## 触发器为“组合（测试版）”

组合，顾名思义，就是通过组合按键来完成一个操作。

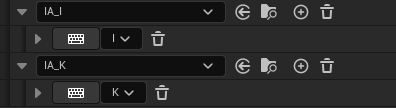
当选择组合之后，我们的按键就已经被禁用了。现在即使删掉该触发器，也并不会记录原来的按键。所以删除组合触发器时记得设置按键。



索引中有两个需要我们关注：“组合操作”和“取消操作”。他们都是数组。我们需要给这两个数组添加内容，每个数组中内容索引从0开始。

为了完成下面学习操作，我们需要创建两个IA，一个表示I键，一个表示K键。（必须要）将两个输入操作添加到输入映射上下文中，没有实际用处，仅用于按键触发。





先添加一个组合操作，在这个组合操作对象的索引中有“组合步操作”“组合步骤完成状态”“按键的时间”。我们给组合步操作赋值为IA\_I。

关注组合步骤完成状态。他对应的就是输入映射的5个状态



我们可以加两个组合操作，都用已触发来举例。



在默认情况下（IA\_IK中没有任何触发器和修改器），当摁下I键时，I键会一直Triggered，也就是已触发。在I键Triggered期间，不会接受K键的输入。只有当I键松开后（Completed后）才能接受K键的输入。如果在IA\_I的按键的时间内实现了IA\_K的Triggered，则完成了该IA\_Attack的Triggered，并完成Completed。

* 在摁下**组合操作**的第一个索引（这里是IA\_I键）时，IA\_Attack会触发Started和Ongoing，Ongoing会最多触发到**完成状态**失败（松开IA\_I键）瞬间时刻再经过“**按键的时间**”的这一时刻。
* 在**完成状态**失败后，开始计时，在**按键的时间**内：  
  --如果该索引的**完成状态**又成功实现，则重新计时。  
  --如果**按键的时间**内没有达到**组合操作**中下一个索引的要求（这里是IA\_K键实现Triggered），则触发Canceled。  
  --如果**按键的时间**内达到**组合操作**中下一个索引的要求，则检测再下一个索引条件（如果有的话）。  
  --如果是最后一个索引（如上述案例的IA\_K）且最后一个索引**完成状态**，则IA\_Attack触发一次Triggered并触发一次Completed。

具体能不能判断触发下一索引的条件，取决于该索引的**完成状态**是否成功实现后失败，且失败时间是否小于等于**按键的时间**。如果失败了且在**按键的时间**内成功实现下一个索引的完成状态，则继续递归，直到所有索引条件都完成，然后触发Triggered和Completed，否则触发Canceled。

为了正常使用组合按键，以上述案例举例，先摁I键，直到松开该按键后的0.5秒内按K键，这样才能正常触发（Triggered）IA\_Attack。

## 案例：英雄联盟技能升级组合按键

组合触发器的多按键事件一般需要松开前面一个按键，才能实现。也就是说，按照上面的案例，是不能实现类似lol中Ctrl Q两个按键一起按给技能升级这样的效果。为了实现这样的效果，可以用弦操作来完成。当然，这种操作也可以用组合来完成，不过有点麻烦。

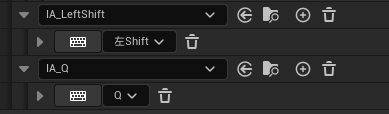
下面来用组合实现这个效果。

首先复习一下英雄联盟技能升级的按键要求：

摁下Shift后不能松开，在按着Shift期间，无论按多长时间Q，只能触发一次技能升级，且按下Q就立刻升级。

首先创建两个IA，实现IA\_LeftShift和IA\_Q，将它们加入到输入映射上下文中。





创建IA\_SkillQUpgrade，加入到输入映射上下文中。添加触发器为组合。

接下来，我们希望摁住Shift的同时摁下Q，即可触发IA\_SkillQUpgrade。

摁住Shift，期间要既能成功也能失败

（因为要成功维持时间统计，失败用于接收Q键输入，且成功频率远小于失败频率）。

能实现这个要求的只有脉冲。所以给IA\_LeftShift加一个脉冲。

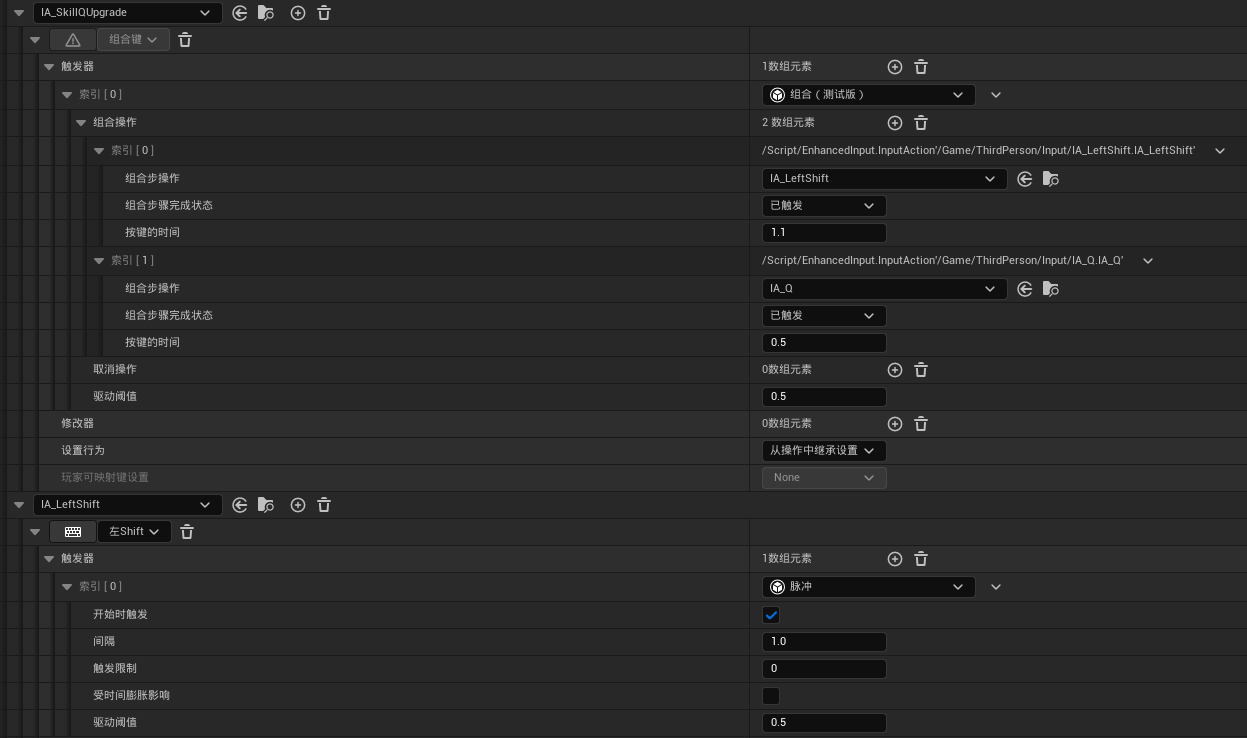
而对于Q键，我们要求他在摁下即可触发且只触发一次。（两个问题，一个问题是只触发一次，对于Started理论上可以实现，但是不能。另一个问题是，当最后一个成功时，应该直接触发IA\_Skill的Triggered，所以前面说的应该可以不用实现，但是实际体验又不行。草测两个问题都是因为组合目前是测试版，有bug）

我们希望成功概率远小于失败概率，就要考虑脉冲的间隔。因为脉冲的Ongoing的次数是按照帧率来实现的，所以当脉冲的**间隔**比较大的时候，Ongoing的触发概率远大于Triggered。而在脉冲那里提到，对于12600kf，当间隔为0.0165时，Triggered的触发频率几乎等于Ongoing的触发频率，当时间过短的时候，会增大处理器的压力。所以我们选择脉冲的间隔比较大，将它设为组合触发器的**组合操作**的**完成状态**为成功。所以组合操作中的IA\_LeftShift的完成状态选择“正在进行”，即Triggered。（注意，这里组合步骤完成状态可以选择多个，但我们这个案例只给LeftShift用Triggered）

**LeftShift按键的时间**应当比IA\_LeftShift的脉冲间隔略长。

Q的按键的时间应当尽可能的短，来尽快的触发Triggered。

因为组合还处于测试版，还有一些bug，所以理论上实现即可。



这样能实现，但依然有概率不触发。首先是理论上的概率，概率值为脉冲时间间隔内Ongoing次数的倒数。但是实际上，不触发的几率高于理论概率。猜测可能是因为测试版。

所以下面给出弦操作的解决方案。

弦操作就很简单了。也不会有失败的概率。



技能升级的实际效果应当是在IA\_SkillUpgrade的Started上实现，这样就能摁下立刻触发且只触发一次。

当然也可以在Triggered上实现。但是Triggered会触发很多次，所以需要DoOnce，并在Completed时对DoOnce进行Reset。效果一样，但是游戏帧率低。

如果需要组合按键，

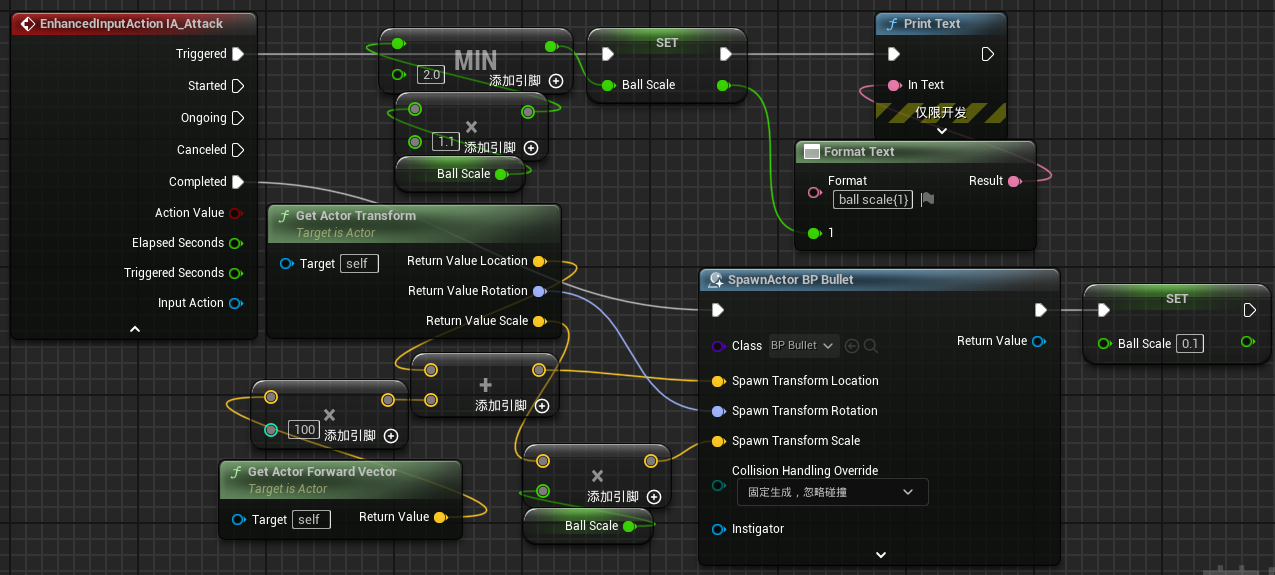
在按下最后一个按键瞬间就触发，则会用到Started

在按下最后一个按键时一直触发，则会用到Triggered

在松开最后一个按键瞬间触发，则会用到Completed

## 案例：蓄力放大球

类似蔚的Q，蓄力可以增加球的大小，大小有上限。



实际上，我们并不需要设置一个变量来保存BallScale。我们可以用ElapsedTime（实际消耗时间）或者TriggeredTime通过某种运算来实现。

