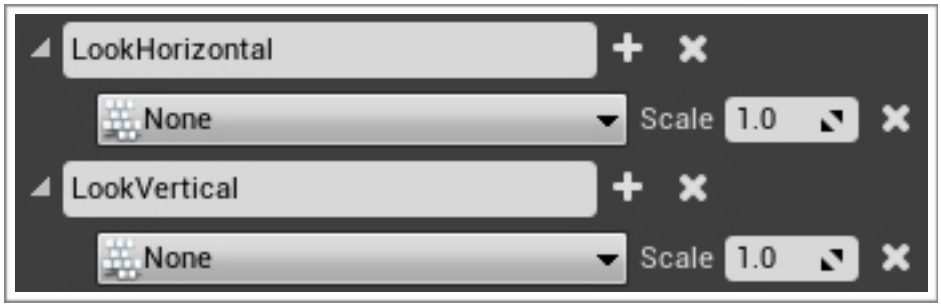
欢迎继续我们的学习。
接下来我们需要创建映射，让玩家可以四处张望。

创建Look的映射

打开Project Settings，创建两个Axis Mappings，分别命名为LookHorizontal和LookVertical。



将LookHorizontal对应的键值更改为Mouse X。

这样，当我们向右方移动鼠标的时候，映射就会输出一个正值，反而则输出一个负值。
接下来将LookVertical对应的键值更改为Mouse Y。

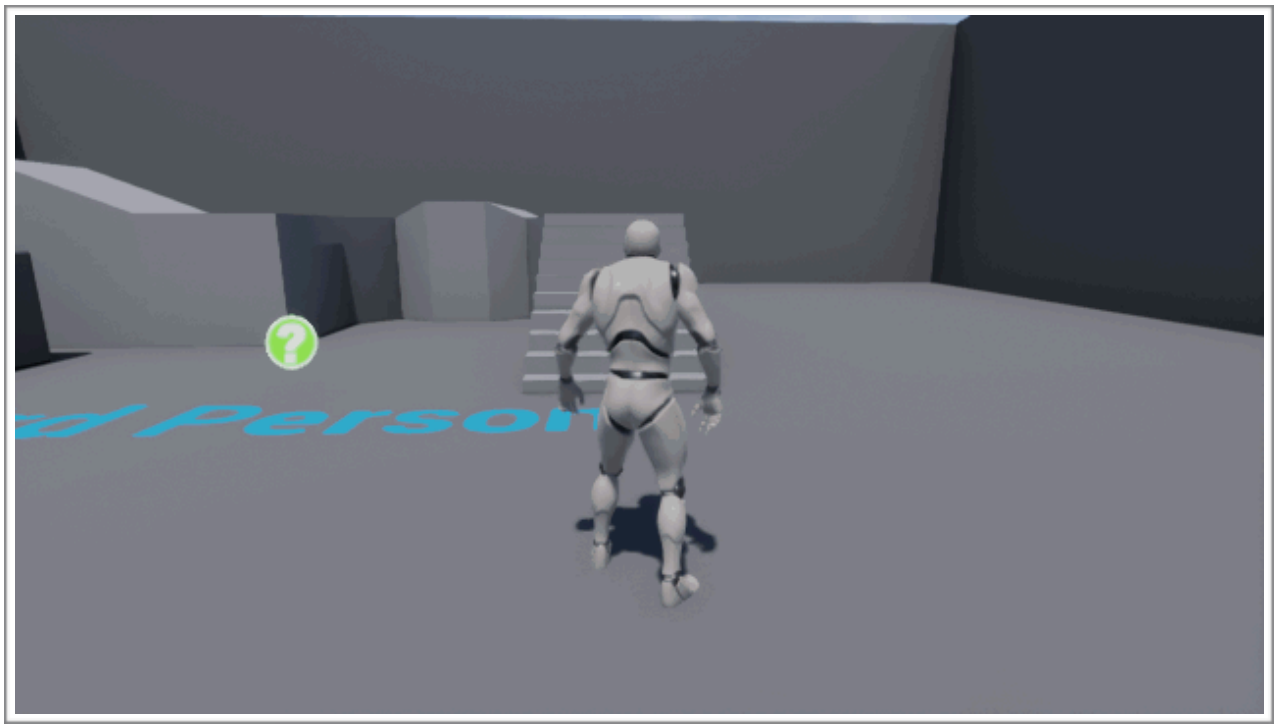
这样，当我们向上方移动鼠标的时候，映射就会输出一个正值，反而则输出一个负值。
接下来，我们需要创建让角色向四周观察的逻辑。

实现Look的逻辑

如果Pawn角色没有Camera组件，UE4会自动创建一个。默认情况下，该Camera会使用controller的rotation信息。

注意：如果想了解关于controller的更多信息，可以复习下AI系列的课程。

尽管controller是非物理的，它们仍然有自己的rotation。这就意味着我们可以让Pawn角色和摄像机面向不同的方向。例如，在第三人称游戏中，角色和摄像机通常不会面向同一个方向。

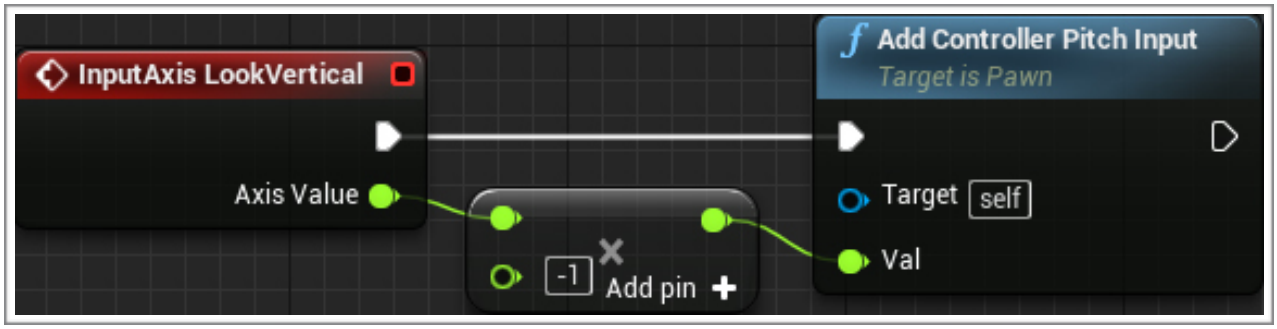


为了在第一人称游戏中旋转摄像机，我们只需要更改controller的rotation信息。
打开BP_Player，然后创建一个LookHorizontal事件。

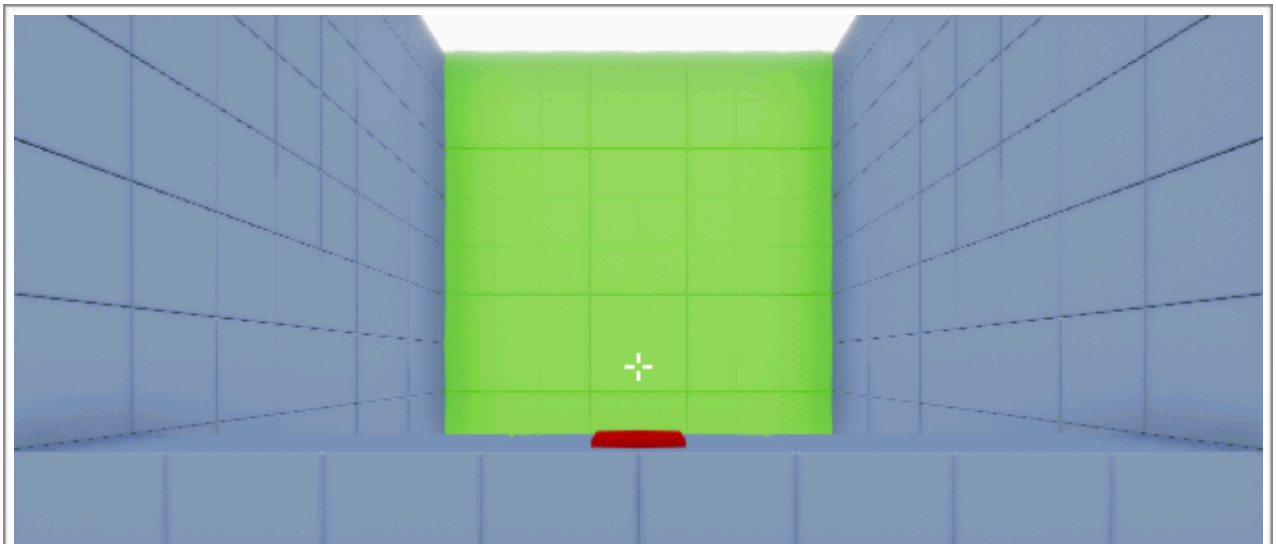
为了让摄像机可以向左或者右方旋转，我们需要设置controller的yaw。
在视图中创建一个Add Controller Yaw Input节点，然后使用下面的方式来连接节点：

现在，当我们在水平方向移动鼠标的时候，controller将会向左侧或者右侧旋转。而因为摄像机使用了controller的rotation信息，所以它也会向左右侧旋转。
接下来对LookVertical重复类似的过程，使用Add Controller Yaw Input节点替代Add Controller Pitch Input节点。

如果现在立即测试游戏，会发现垂直方向的查看是反向的。这就意味着当我们向上移动鼠标的时候，摄像机会向下看。
如果你是个强迫症患者，那么显然需要让Axis Value乘以-1。这样就会将Axis Value转换为所需的数据。



点击工具栏上的Compile按钮，然后点击Play按钮，移动鼠标来四处观察游戏场景。



好了，现在基本的运动和观察控制都已经设置完毕，接下来就可以来点好玩的了。
这一课的内容就到这里了，我们下一课再见~

讨论群-笨猫学编程QQ群：
375143733

答疑论坛：
http://www.vr910.com/forum.php?mod=forumdisplay&fid=52

知乎专栏：

https://zhuanlan.zhihu.com/kidscoding

新浪博客:
http://blog.sina.com.cn/eseedo

Github:
https://github.com/eseedo

个人网站：
http://icode.ai/