输入

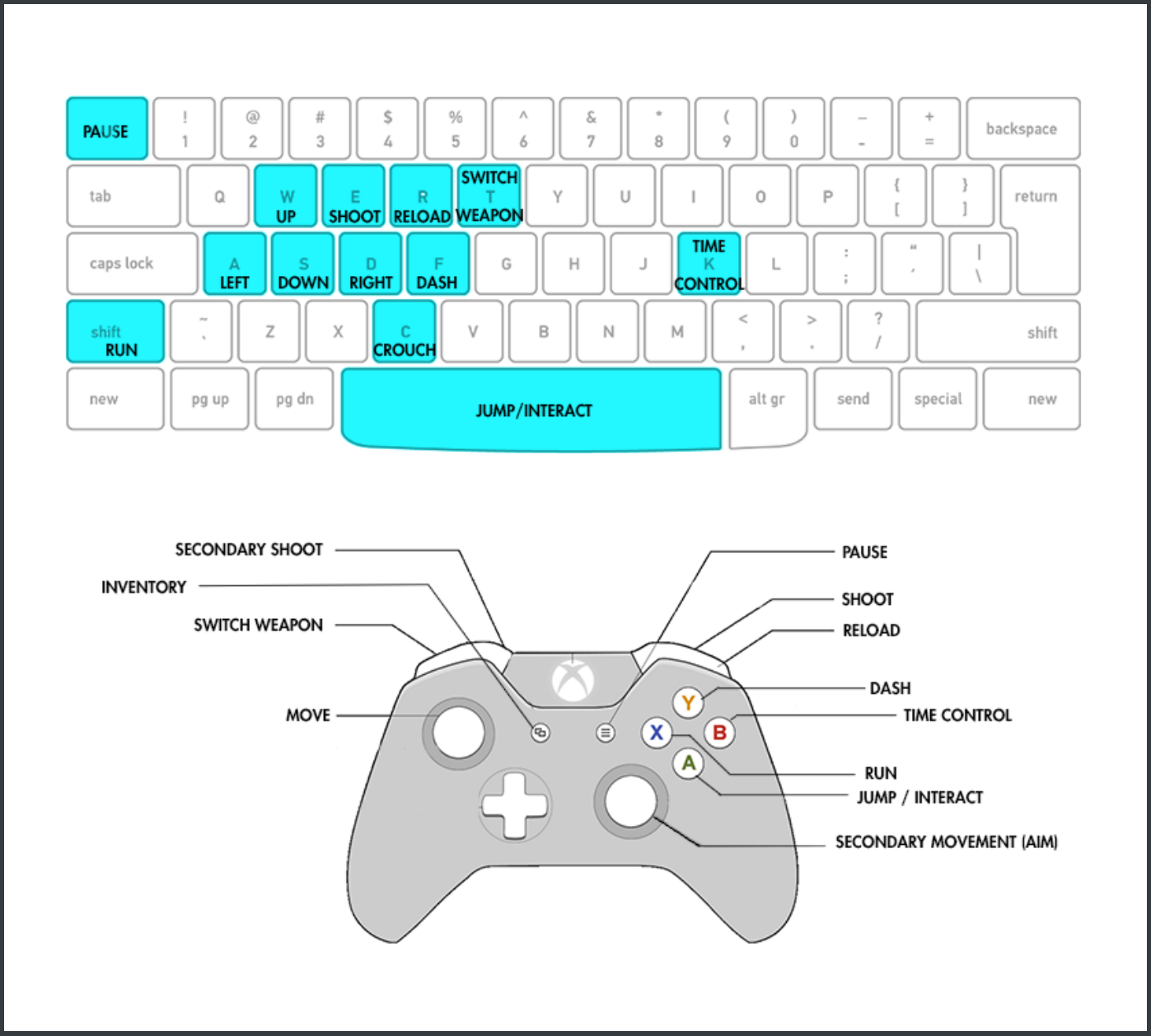
此页面说明如何在游戏中设置输入。

介绍

输入是每个游戏的核心。自顶向下引擎也不例外。它目前支持移动控件（iOS、Android...）、键盘、游戏手柄（为 windows xbox pad 设置，但可以随意重新绑定其他控制器的键）和鼠标。

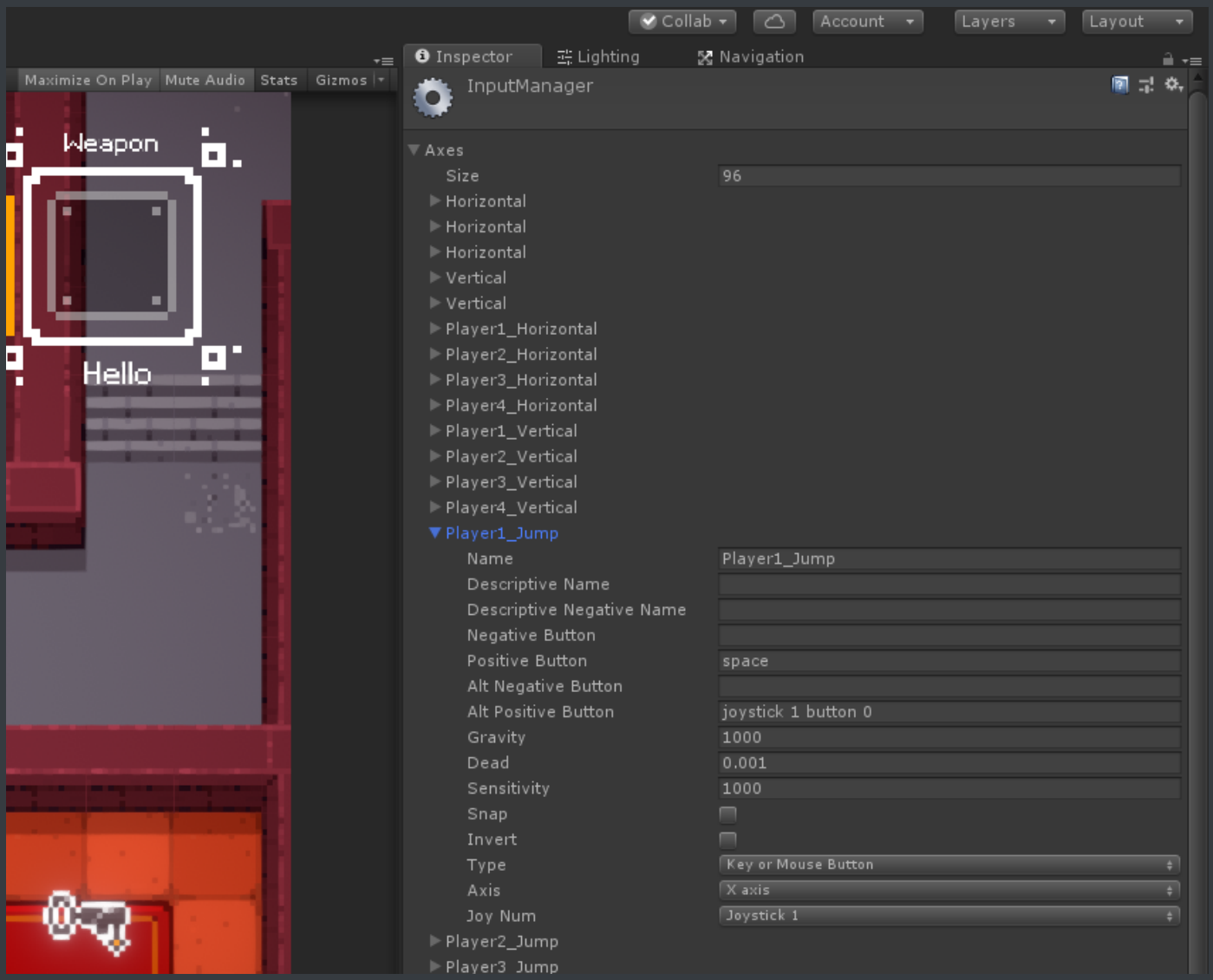
默认输入

默认情况下，已创建键盘布局以支持 4 个同时播放器。现在这对于单人游戏来说可能不太舒服，请随意重新映射按钮以满足您的需要（请参阅以下部分）。无论如何，这里是玩家 1 的默认布局（点击图片展开）（查看其他玩家绑定的输入设置）：



Player 1 的 PC 布局的默认键盘和 xbox pad

输入设置 – 如何更改键绑定?



这是很多轴

对于键盘和游戏手柄（以及较小程度上的鼠标），键绑定是**通过本机 Input settings**定义的，就像在任何好的 Unity 项目中一样。您可以通过顶部菜单编辑 > 项目设置 > 输入访问这些。这应该打开一个相当大的"轴"列表。如果你以前从未见过这个面板，不妨先看看官方文档。

现在这个列表很长。该列表已填充以支持 4 个播放器（您可以根据需要添加更多），并建议一键配置。随意**编辑其中的每一个**以根据您的喜好重新绑定键。

输入管理器和 GUI

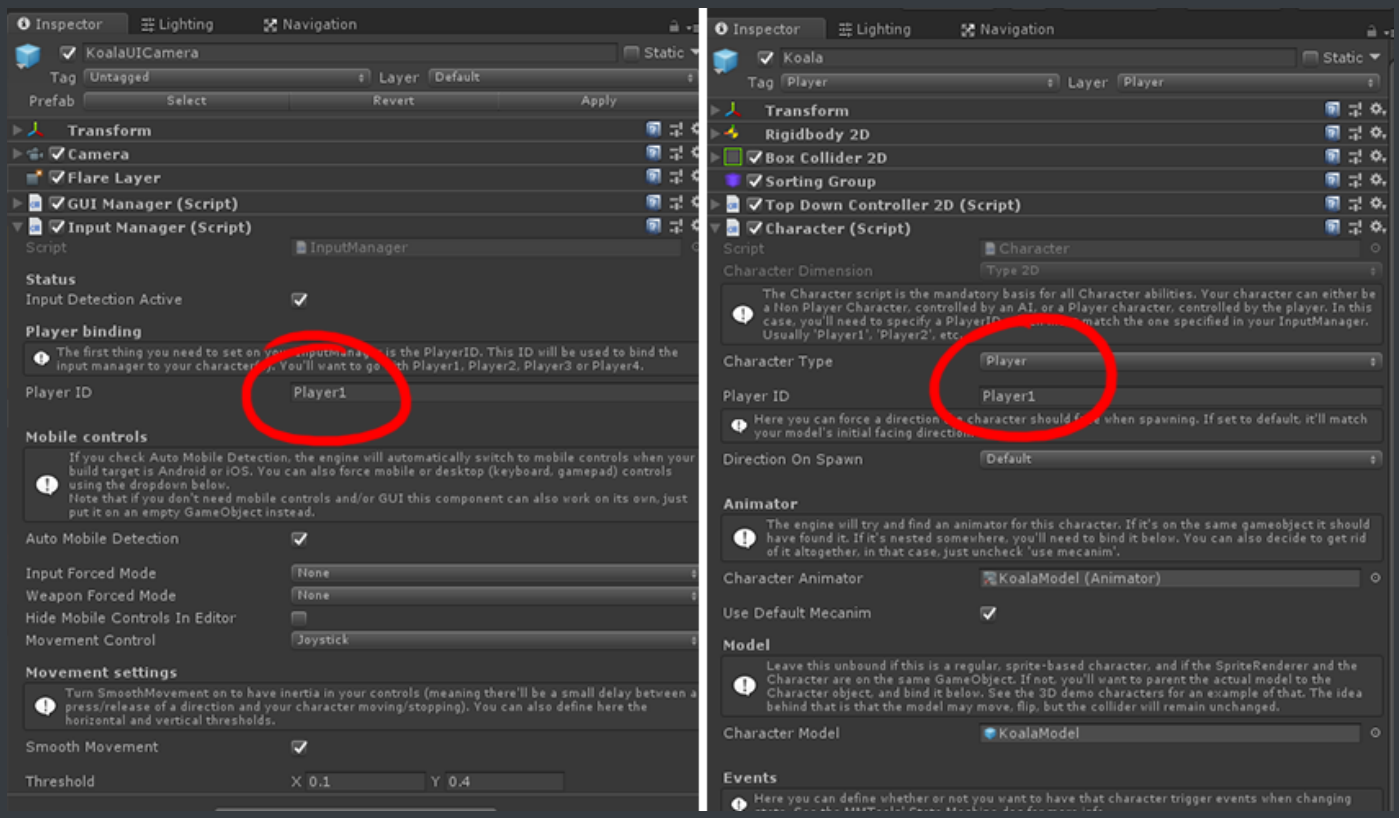
要在屏幕上移动您的角色，您通常使用**input**，无论是通过触摸屏幕还是按下机械键。为了处理输入检测并将其传递给您的角色，TopDown 引擎使用输入管理器。自上而下引擎的大多数关卡中都有一个 UI 相机。这个 UI 相机预制件也嵌套在其中，一堆移动控件，用于在移动设备上播放时控制播放器。出于这个原因，为了使绑定更容易，您将在 UICamera 预制件的顶层

找到 InputManager。如果您愿意，您可以决定将其放置在自己的游戏对象上。

该**输入管理**是一个相当长的类。如果你看一看，你会发现它实际上**并不太复杂**，只是非常冗长。它有很多方法可以绑定到移动控件，并为每个注册的按钮管理一个小型状态机。

无论您是要使用移动检测还是强制移动控制，您都可以从其检查器中指定**PlayerID**（有关详细信息，请参阅下一节）。您还可以决定是喜欢虚拟箭头/dpad 还是虚拟操纵杆。最后，您可以打开或关闭平滑移动（应该立即按下还是快速按下左键？），以及水平和垂直阈值。

玩家ID



您的 InputManager 的 PlayerID（右）必须与您要控制的角色上的那个相匹配

这里有一件事情是**非常重要的**，了解是**PlayerID**概念。引擎能够将输入从 InputManager 发送到场景中的可玩角色（具有类型设置为 Player 的 Character 组件的对象）。每个 InputManager 都有一个 PlayerID 属性，它将把它的信息发送**给所有与它自己的 PlayerID 属性匹配的角色**。所以有几种方法可以工作：

- 最常见的方法是简单地**让引擎处理这个**。将 InputManager 添加到场景中，将其 PlayerID 设置为"Player1"。然后在关卡管理器中，当您添加角色时，选中"**自动属性玩家 ID** "复选框，它会自动为可玩角色提供"玩家 1"ID。如果您处于多人游戏级别，请使

用"Player2"、"Player3"和"Player4"。您当然可以添加自己的，但这些是默认的。
- 您还可以在 prefab**上**设置 PlayerID 。例如，您可以有一个 Dog 角色，其 PlayerID 是"JoeTheDog"（为什么不呢？）。如果您向场景中添加 InputManager 并希望它以该 Dog 角色为目标，请将其自己的 PlayerID 参数设置为"JoeTheDog"。按播放键，您应该能够控制您的狗角色。如果由于某种原因您不想使用关卡管理器并希望立即在场景中移动预制件，那么该方法非常有用。

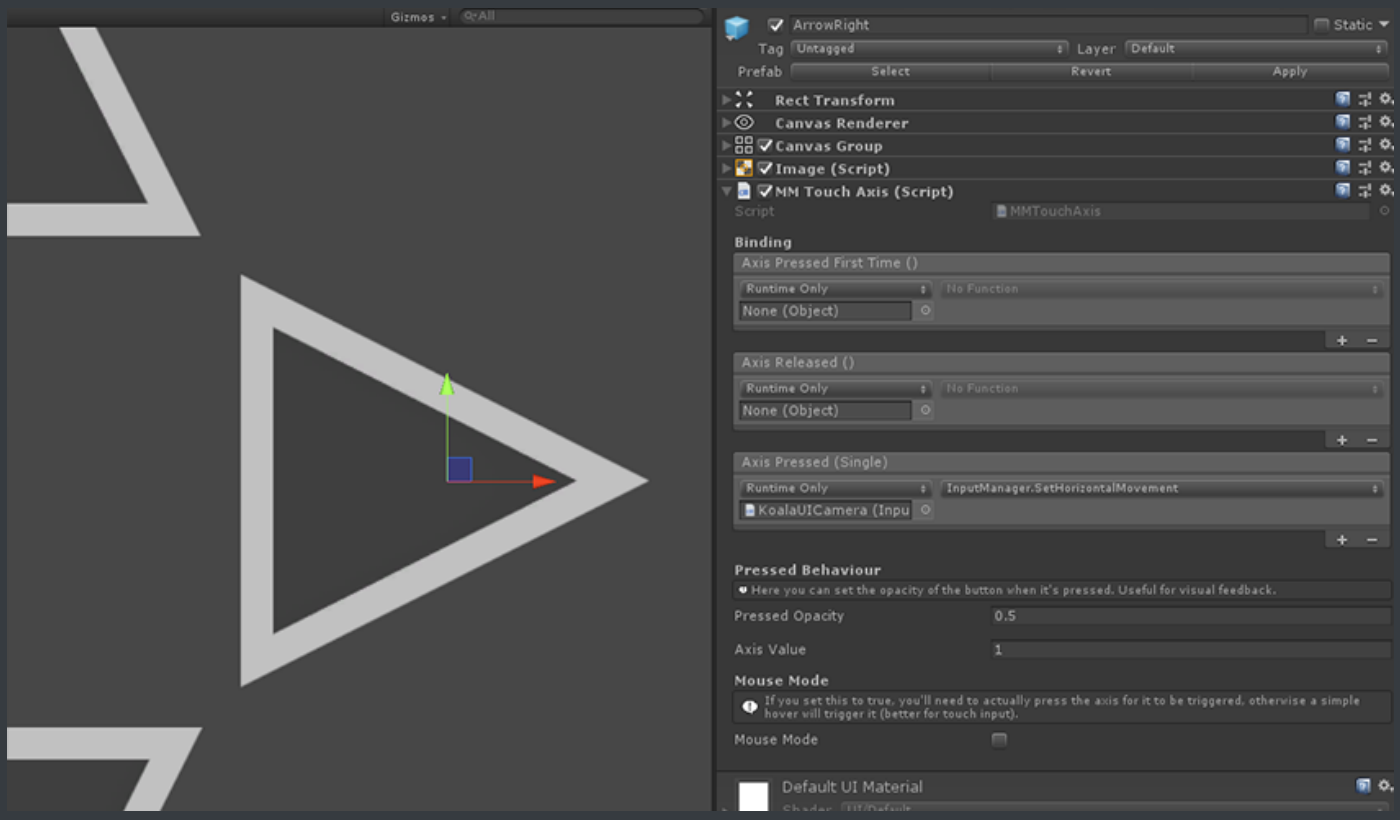
我**在调整角色时经常**使用第二种方法。假设我想更改角色的跳跃行为。我将复制我角色的预制件，将该副本放入场景中，将其 PlayerID 设置为 Player1。然后在 LevelManager 中，我将原始预制件作为玩家预制件。我按下播放键，我现在用相同的输入控制两个角色。然后我可以更改其中一个的跳转设置（或者甚至添加一个新的 Jump 脚本来代替旧的），并**非常快速**地**测试两个**，看看我的更改是否改善了游戏玩法。

平台检测如何工作?

从 InputManager 的检查器中，您可以打开或关闭**自动移动检测**。

如果您展开 UICamera 预制件，您会看到它包含相当多的东西，尤其是 Arrows、Joystick 和 Buttons 画布组。**默认情况下禁用**它们以允许更简单的级别版本，但您可以根据需要打开它们，或者调整它们。然后将它们绑定到 GUIManager 组件（以及其他内容）。移动检测的工作方式非常简单：如果您的目标是移动平台（iOS 或 Android），它会在您按下播放键时显示控件。如果你的目标是另一个平台，它会**隐藏和禁用它们**。您还可以从检查器强制使用一种模式或另一种模式。

虚拟箭头、操纵杆和按钮



触摸轴（箭头）检查器

该资产配备了一个功能齐全的箭头方向键、一个操纵杆和一些按钮，当然**您可以重新混合所有这些**，添加按钮，添加操纵杆等以匹配您自己的游戏玩法。您将在**MMControlsTestScene**演示场景中找到引擎中包含的所有现成的移动输入构建块的演示。

移动箭头非常简单，你只需要一个 Rect+CanvasGroup 对象上的 MMTouchAxis 组件。然后在其检查器中，您需要设置轴值（通常为左/下为 -1，右/上为 1）。然后您需要将它绑定到您的 InputManager，只需在 AxisPressed 事件上。为此，只需将您的 InputManager（在大多数情况下是场景中的 UICamera 预制件）拖到"仅运行时"下方的小框上，然后选择适当的方法（SetVerticalMovement、SetHorizontalMovement 或它们的辅助方法）。

移动操纵杆更容易设置。只需将 MMTouchJoystick 组件添加到 Rect+CanvasGroup 对象，定义启用哪个轴（例如，您可能需要仅水平摇杆）、最大范围（旋钮可以从其底部移动多远），然后绑定您的输入经理给它。请注意，您还需要指定一个目标相机（通常是您的 UICamera）。

可重新定位的操纵杆：与常规操纵杆的**外观**相似，当您抬起拇指并将其放回屏幕时，这些操纵杆会在更大的区域内重新定位。

按钮也以相同的方式工作，但对于它们，您将能够指定三个不同的事件：按钮按下（第一次按下时）、按钮按下（只要此按钮保持按下状态，每一帧）和按钮up（当它被释放时）。这将允许您在角色能力中（例如）在按钮被释放时调用某些方法。如果您不使用所有事件，则不必绑定所有事件。

滑动区域：让您在其表面上拖动拇指，并触发滑动事件，包括角度、方向、长度等。

自定义移动输入

使用上面的类，您应该能够以任何您想要的方式自定义您的移动控件。例如，如果您想在 Loft3D 演示场景中添加第二个操纵杆，并让它控制可以在该场景中拾取的霰弹枪，您需要执行以下简单步骤：

- 在项目视图中选择**ShotgunWithAmmo**预制件，将其瞄准控制设置为 PrimaryThenSecondaryMovement
- 选择您的**UICamera**，将**InputForcedMode**设置为 Mobile （以便您可以在编辑器中对其进行测试）
- 复制UICamera 画布下的**操纵杆**节点，将其向右移动一点
- 在其操纵杆旋钮上，在**MMTouchJoystick**组件上，将 Binding 更改为 InputManager.SetSecondaryMovement（**vector2**版本）
- 大功告成，按**播放**键，玩得开心！

禁用输入

如果您想禁用输入，您可以通过两种主要方式进行：

- 在 InputManager 级别，通过将 InputDetectionActive 设置为 false（通过脚本或其检查器）。这将防止通过 InputManager 读取所有输入。
- 在能力级别，通过将 AbilityPermitted 设置为 false

触感不错

自上而下引擎包括 More Mountains 的另一个资产：Nice Touch。**如果您已经拥有 TopDown 引擎，请不要购买它！** 我创建了 Nice Touch 来提供**简单快速的输入解决方案**。它处理键盘、游戏手柄、鼠标和触摸输入。还有很多其他的输入解决方案。Nice Touch/MMControls 旨在提供更简单的替代方案，更快地设置，而无需进行不必要的设置。随意不使用这些并用您自己

的替换它们。

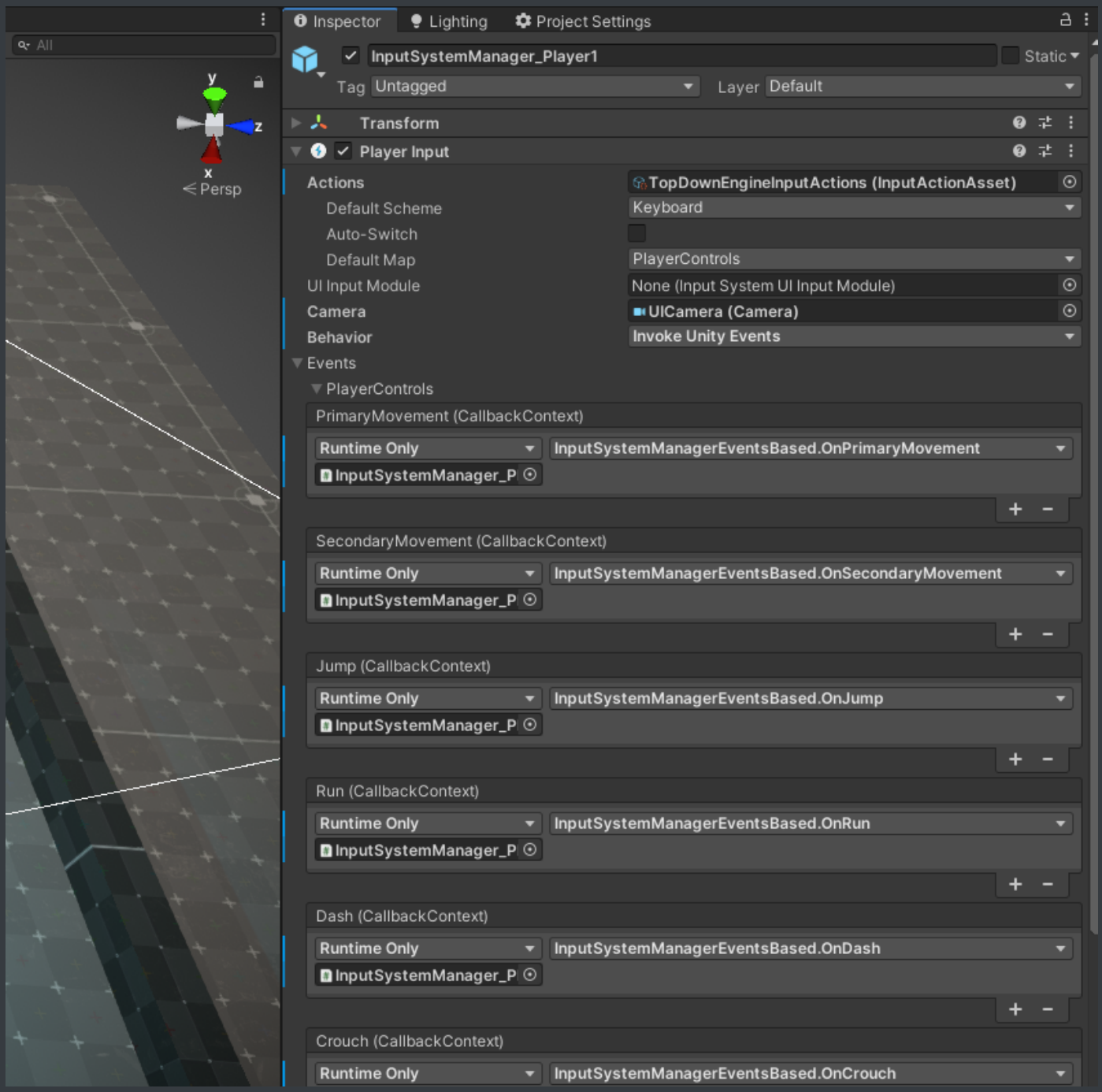
您将在 MMTools/MMControls 中找到这些脚本。同一个文件夹还包括一个测试场景：MMControlsTestScene。您需要在游戏中保留该文件夹。

使用 Unity 的"新"输入系统

Unity 在3 年前发布了一个**新的输入系统**，尽管仍未完全记录或稳定，但最近它被标记为"已验证"。如果尽管使用它**存在**所有相关**风险**，但您仍想在生产中使用它，TopDown 引擎提供了两个正在使用的示例。您可以在 TopDownEngine/Common/ScriptsInputSystem 下找到它们，当然它们需要您**安装和设置**Input System 包。

您会注意到 InputManager 被**InputSystemManager**或**InputSystemManagerEventsBased替换**，具体取决于您查看的演示（事件版本最适合多人游戏）。在这两种情况下，您都会在相机装备下找到 InputSystemManager。两者都依赖于另一个类 TopDownEngineInputActions，它只是"新"系统的常规自动生成输出。除了该系统的特定 API 调用以及该类的使用之外，没有任何特定的内容，它的工作方式与常规版本相同。建议您对该系统有很好的了解才能选择此选项，与该系统的其余部分一样，它不如经典选项直观。

安装 InputSystem 包后，Unity 可能会丢失一些引用，您可能需要手动设置绑定。这应该只发生在 MinimalScene3D_InputSystem_Multiplayer 场景中，另一个应该是安全的。如果发生这种情况，您的 InputSystemManager_Player1 和 InputSystemManager_Player2 的检查器应该如下所示：



MinimalScene3D_InputSystem_Multiplayer 场景中的 InputSystemManager_Player1 检查器