**事件系统**

使用事件系统可以根据输入（即键盘、鼠标、触摸或自定义输入）将事件发送到应用程序中的对象。事件系统包含一些共同协作以发送事件的组件。

**概述**

将事件系统组件添加到游戏对象时，应该会注意到该组件未公开太多功能，这是因为事件系统本身设计为事件系统模块之间通信的管理器和协调器。

事件系统的主要作用如下：

* 管理视为选中状态的游戏对象
* 管理正在使用的输入模块
* 管理射线投射（如果需要）
* 根据需要更新所有输入模块

**输入模块**

输入模块包含定义事件系统行为方式的主要逻辑，可用于：

* 处理输入
* 管理事件状态
* 将事件发送到场景对象。

事件系统中一次只能有一个输入模块处于激活状态，这些输入模块必须是与事件系统组件位于同一游戏对象上的组件。

如果希望编写自定义输入模块，建议发送 Unity 中现有 UI 组件支持的事件，但是也可以扩展事件和编写自定义的事件（详见消息系统文档）。

**射线投射器**

射线投射器用于确定指针在哪个对象上方。输入模块通常使用场景中配置的射线投射器来计算指针设备位于哪个对象上方。

默认情况下提供了 3 种射线投射器：

* 图形射线投射器 (Graphic Raycaster) - 用于 UI 元素
* 2D 物理射线投射器 (Physics 2D Raycaster) - 用于 2D 物理元素
* 物理射线投射器 (Physics Raycaster) - 用于 3D 物理元素

如果在场景中配置了 2D/3D 射线投射器，很容易让非 UI 元素接收来自输入模块的消息。只需附加一个脚本来实现其中一个事件接口即可。

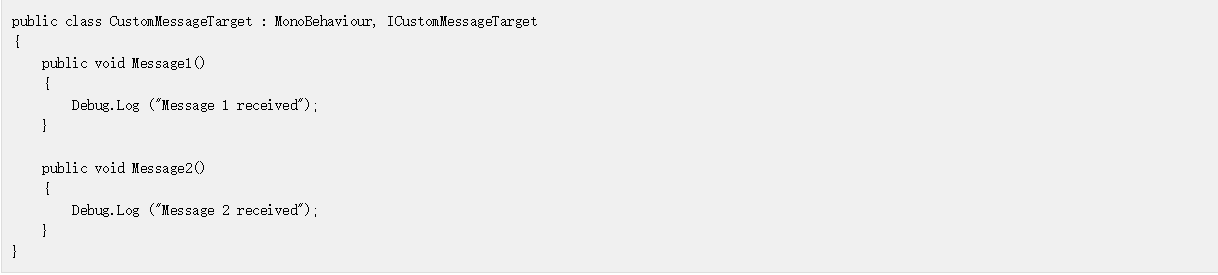
# 消息系统

新的 UI 系统使用一种消息系统来取代 SendMessage。该系统是纯 C# 系统，旨在解决 SendMessage 存在的一些问题。该系统使用可在 MonoBehaviour 上实现的自定义接口来指示组件能够从消息系统接收回调。调用时会指定目标游戏对象；该调用将在游戏对象的所有（实现了指定接口以据此发出该调用的）组件上发出。借助消息系统可传递自定义用户数据，并可指定事件应在游戏对象层级视图中传播的距离：应该仅为指定的游戏对象执行，还是应该在子对象和父对象上也执行。除此之外，消息框架还提供 helper 函数来搜索和查找实现了给定消息接口的游戏对象。

消息系统是通用型系统，不仅可用于 UI 系统，还可用于一般游戏代码。添加自定义消息事件相对简单，借助 UI 系统用于所有事件处理的相同框架即可实现。

## 如何定义自定义消息？

如果希望定义自定义消息，此过程相对简单。在 UnityEngine.EventSystems 命名空间中，有一个名为“IEventSystemHandler”的基本接口。从此接口扩展的任何内容都可以视为通过消息系统接收事件的目标。

一旦定义了此接口，即可由 MonoBehaviour 实现。此接口实现后，定义了在针对此 MonoBehaviour 游戏对象发出指定消息时将会执行的函数。

现在有了可接收消息的脚本之后，我们需要发出消息。通常，此消息用于响应发生的某个松散耦合事件。例如，在 UI 系统中，我们为 PointerEnter 和 PointerExit 等事件发出事件，还有为了响应用户在应用程序中的输入而发生的各种其他事件。

要发送消息，可使用一个静态 helper 类来执行此操作。在参数方面，需要消息的目标对象、一些特定于用户的数据以及一个映射到所需目标消息接口中特定函数的仿函数 (functor)。

ExecuteEvents.Execute<ICustomMessageTarget>(target, null, (x,y)=>x.Message1());

此代码将在游戏对象目标上实现了 ICustomMessageTarget 接口的所有组件上执行 Message1 函数。ExecuteEvents 类的脚本文档中介绍了执行函数的其他形式，例如在子对象或父对象中执行。

# 输入模块

输入模块可用于配置和自定义事件系统的主要逻辑。系统提供了两个开箱即用的输入模块，一个用于独立平台，另一个用于触控输入。每个模块都会按照您在给定配置上的预期接收和分发事件。

输入模块是执行事件系统“业务逻辑”的位置。启用事件系统后，它会查看所连接的输入模块，并将更新处理传递给具体模块。

输入模块可根据您希望支持的输入系统进行扩展或修改。输入系统的目的是将特定于硬件的输入（例如触摸、游戏杆、鼠标、运动控制器）映射到通过消息系统发送的事件。

内置的输入模块可支持常见的游戏配置，如触控输入、控制器输入、键盘输入和鼠标输入。如果在 MonoBehaviour 上实现特定接口，这些输入模块会向应用程序中的控件发送各种事件。所有 UI 组件都实现了对具体组件有意义的接口。

**支持的事件**

事件系统支持许多事件，并可在用户编写的自定义输入模块中进一步自定义它们。

独立输入模块和触摸输入模块支持的事件由接口提供，通过实现该接口即可在 MonoBehaviour 上实现这些事件。如果配置了有效的事件系统，则会在正确的时间调用事件。

* IPointerEnterHandler - OnPointerEnter - 当指针进入对象时调用
* IPointerExitHandler - OnPointerExit - 当指针退出对象时调用
* IPointerDownHandler - OnPointerDown - 在对象上按下指针时调用
* IPointerUpHandler - OnPointerUp - 松开指针时调用（在指针正在点击的游戏对象上调用）
* IPointerClickHandler - OnPointerClick - 在同一对象上按下再松开指针时调用
* IInitializePotentialDragHandler - OnInitializePotentialDrag - 在找到拖动目标时调用，可用于初始化值
* IBeginDragHandler - OnBeginDrag - 即将开始拖动时在拖动对象上调用
* IDragHandler - OnDrag - 发生拖动时在拖动对象上调用
* IEndDragHandler - OnEndDrag - 拖动完成时在拖动对象上调用
* IDropHandler - OnDrop - 在拖动目标对象上调用
* IScrollHandler - OnScroll - 当鼠标滚轮滚动时调用
* IUpdateSelectedHandler - OnUpdateSelected - 每次勾选时在选定对象上调用
* ISelectHandler - OnSelect - 当对象成为选定对象时调用
* IDeselectHandler - OnDeselect - 取消选择选定对象时调用
* IMoveHandler - OnMove - 发生移动事件（上、下、左、右等）时调用
* ISubmitHandler - OnSubmit - 按下 Submit 按钮时调用
* ICancelHandler - OnCancel - 按下 Cancel 按钮时调用

**射线投射器**

事件系统需要一种方法来检测当前输入事件需要发送到的位置，而此方法由射线投射器 (Raycaster) 提供。给定屏幕空间位置的情况下，射线投射器将收集所有潜在目标，确定它们是否在给定位置下，然后返回最接近屏幕的对象。提供了几种类型的射线投射器：

* 图形射线投射器 (Graphic Raycaster) - 用于 UI 元素，位于画布上，并在画布中搜索
* 2D 物理射线投射器 (Physics 2D Raycaster) - 用于 2D 物理元素
* 物理射线投射器 (Physics Raycaster) - 用于 3D 物理元素

当场景中存在并启用了射线投射器时，只要从输入模块发出查询，事件系统就会使用该射线投射器。

如果使用多个射线投射器，那么这些射线投射器都会进行针对性投射，并且结果将根据与元素的距离进行排序。

# 事件系统管理器

该子系统负责控制构成事件的所有其他元素。该系统会协调哪个输入模块当前处于激活状态，哪个游戏对象当前被视为“已选中”，以及许多其他高级事件系统概念。

每次“更新”时，事件系统都会收到调用、查看其输入模块并确定应该将哪个输入模块用于此活动。然后，系统会将处理委托给模块。

## 属性

| **属性：** | **功能：** |
| --- | --- |
| **First Selected** | 首先选择的游戏对象。 |
| **Send Navigation Events** | EventSystem 是否应允许导航事件（移动/提交/取消）。 |
| **Drag Threshold** | 拖拽操作的容限区域（以像素为单位）。 |

在 Properies 表下面是“Add Default Input Modules”按钮。

# 图形射线投射器

图形射线投射器用于对画布进行射线投射。射线投射器查看画布上的所有图形，并确定是否有任何图形被射中。

可将图形射线投射器配置为忽略背面图形以及被其前面的 2D 或 3D 对象阻挡。如果希望将此元素的处理强制为射线投射的前面或后面，也可应用手动优先级。

## 属性

| **属性：** | **功能：** |
| --- | --- |
| **Ignore Reversed Graphics** | 是否应考虑背对射线投射器的图形？ |
| **Blocked Objects** | 将阻挡图形射线投射的对象的类型。 |
| **Blocking Mask** | 将阻挡图形射线投射的对象的类型。 |

# 物理射线投射器

射线投射器对场景中的 3D 对象进行射线投射。因此可将消息发送到实现事件接口的 3D 物理对象。

## 属性

| **属性：** | **功能：** |
| --- | --- |
| **Depth** | 获取配置的摄像机的深度。 |
| **Event Camera** | 获取用于此模块的摄像机。 |
| **Event Mask** | 摄像机遮罩与 eventMask 的逻辑和。 |
| **Final Event Mask** | 摄像机遮罩与 eventMask 的逻辑和。 |

# 2D 物理射线投射器

2D 射线投射器对场景中的 2D 对象进行射线投射。因此可将消息发送到实现事件接口的 2D 物理对象。此情况下需要使用摄像机游戏对象，并会将 2D 物理射线投射器添加到摄像机游戏对象（如果未将 3D 物理射线投射器添加到摄像机游戏对象）。

有关射线投射器的更多信息，请参阅[射线投射器](file:///E:\UnityDocumentation\Manual\Raycasters.html)。

## 属性

| **属性：** | **功能：** |
| --- | --- |
| **Event Camera** | 用于为此射线投射器生成射线的摄像机。 |
| **Priority** | 该投射器相对于其他投射器的优先级。 |
| **Sort Order Priority** | 基于排序顺序的射线投射器优先级。 |
| **Render Order Priority** | 基于渲染顺序的射线投射器优先级。 |

**独立输入模块**

根据设计，该模块与控制器/鼠标输入具有相同的功能。响应输入时会发送按钮按压、拖拽以及类似事件。

当鼠标/输入设备移动时，该模块将指针事件发送到组件，并使用[图形射线投射器 (Graphics Raycaster)](file:///E:\\UnityDocumentation\\ScriptReference\\UI.GraphicRaycaster.html)和[物理射线投射器 (Physics Raycaster)](file:///E:\\UnityDocumentation\\ScriptReference\\EventSystems.PhysicsRaycaster.html) 来计算给定指针设备当前指向的元素。您可以配置这些射线投射器来检测或忽略场景的某些部分，从而满足您的要求。

该模块会发送 Move 事件和 Submit/Cancel 事件来响应通过输入管理器跟踪的输入。对于键盘和控制器输入均是如此。可在模块的检视面板中配置跟踪的轴和键。

**属性**

| ***属性：*** | ***功能：*** |
| --- | --- |
| **Horizontal Axis** | 为水平轴按钮输入所需的管理器名称。 |
| **Vertical Axis** | 为垂直轴输入所需的管理器名称。 |
| **Submit Button** | 为 Submit 按钮输入所需的管理器名称。 |
| **Cancel Button** | 为 Cancel 按钮输入所需的管理器名称。 |
| **Input Actions Per Second** | 每秒允许的键盘/控制器输入数量。 |
| **Repeat Delay** | 每秒输入操作重复率生效前的延迟秒数。 |
| **Force Module Active** | 勾选此复选框可强制该\_\_独立输入模块 (Standalone Input Module)\_\_ 处于活动状态。 |

**详细信息**

该模块：

* 使用垂直/水平轴进行键盘和控制器导航
* 使用 Submit/Cancel 按钮发送提交和取消事件
* 在事件之间有一个超时值仅允许每秒的最大事件数。

该模块的流程如下

* 如果输入了输入管理器中的有效轴，则向所选对象发送 Move 事件
* 如果按下了 Submit 或 Cancel 按钮，则向所选对象发送 Submit 或 Cancel 事件
* 处理鼠标输入
  + 如果这是新的按压操作
    - 发送 PointerEnter 事件（向上发送到层级视图中可对其进行处理的每个对象）
    - 发送 PointerPress 事件
    - 缓存拖动处理程序（层级视图中可对其进行处理的第一个元素）
    - 将 BeginDrag 事件发送到拖动处理程序
    - 在事件系统中将“Pressed”对象设置为 Selected
  + 如果这是持续按压操作
    - 处理移动
    - 将 DragEvent 发送到缓存的拖动处理程序
    - 如果触摸在对象之间移动，则处理 PointerEnter 和 PointerExit 事件
  + 如果这是释放操作
    - 将 PointerUp 事件发送到收到 PointerPress 的对象
    - 如果当前悬停对象与 PointerPress 对象相同，则发送 PointerClick 事件
    - 如果缓存了拖动处理程序，则发送 Drop 事件
    - 将 EndDrag 事件发送到缓存的拖动处理程序
  + 处理滚轮事件

**触摸输入模块**

**注意：触摸输入模块 (TouchInputModule) 已弃用。现在触摸输入的处理在 [StandaloneInputModule](file:///E:\\UnityDocumentation\\Manual\\script-StandaloneInputModule.html) 中进行。**

该模块设计用于触摸设备，可发送触摸和拖动操作的指针事件来响应用户输入。该模块支持多点触控。

该模块使用场景配置的射线投射器来计算当前正在触摸的元素。每次当前触摸操作都将相应发出射线投射。

**属性**

| ***属性：*** | ***功能：*** |
| --- | --- |
| **Force Module Active** | 强制激活此模块。 |

**详细信息**

该模块的流程如下：

* 对于每个触摸事件
  + 如果这是新的按压操作
    - 发送 PointerEnter 事件（向上发送到层级视图中可对其进行处理的每个对象）
    - 发送 PointerPress 事件
    - 缓存拖动处理程序（层级视图中可对其进行处理的第一个元素）
    - 将 BeginDrag 事件发送到拖动处理程序
    - 在事件系统中将“Pressed”对象设置为 Selected
  + 如果这是持续按压操作
    - 处理移动
    - 将 DragEvent 发送到缓存的拖动处理程序
    - 如果触摸在对象之间移动，则处理 PointerEnter 和 PointerExit 事件
  + 如果这是释放操作
    - 将 PointerUp 事件发送到收到 PointerPress 的对象
    - 如果当前悬停对象与 PointerPress 对象相同，则发送 PointerClick 事件
    - 如果缓存了拖动处理程序，则发送 Drop 事件
    - 将 EndDrag 事件发送到缓存的拖动处理程序

# 事件触发器

事件触发器从事件系统接收事件，并为每个事件调用已注册的函数。

事件触发器可用于指定希望为每个事件系统事件调用的函数。您可以为单个事件分配多个函数，每当事件触发器收到该事件时，将调用这些函数。

请注意，将事件触发器组件附加到游戏对象将使该对象拦截所有事件，并且不会从此对象开始发生事件冒泡！

## 事件

通过单击 Add New Event Type 按钮，可有选择地将每个[支持的事件](file:///E:\UnityDocumentation\Manual\SupportedEvents.html)包含在事件触发器中。