Simple Mobile Input Manual

Version 1.0.1

Versions **1**

Basic Setup **2**
- Prefab 2
- Joystick 3
  - Configuration 3
  - Bind event 4
- Action Button 6
  - Configuration 6
  - Bind events 7

Joystick Input UI Script **10**
- Inspector 10
  - Identifier 10
  - Components 10
  - Visual 10
  - Adjustments 11
  - Positioning 12
  - Events 12
  - Gizmos 13
  - Core Components 15
- Events binding 16
  - Unity Event 16
  - Static Event 16

Action Input UI Script **17**
- Inspector 17
  - Identifier 17
  - Components 17
  - Visual 17
  - Positioning 18
  - Events 18
  - Gizmos 19
  - Core Components 20
- Events binding 21
  - Unity Event 21
  - Static Event 21

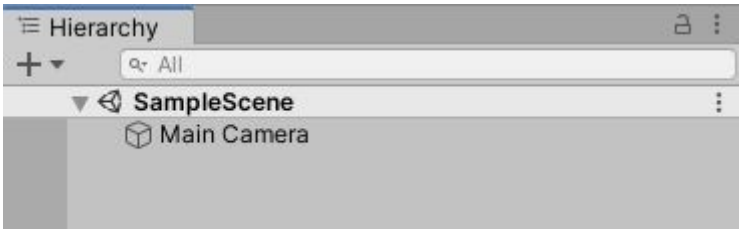
Versions

Version	Changes
1.0.0	Initial release
1.0.1	Bug fixes and expose more methods.

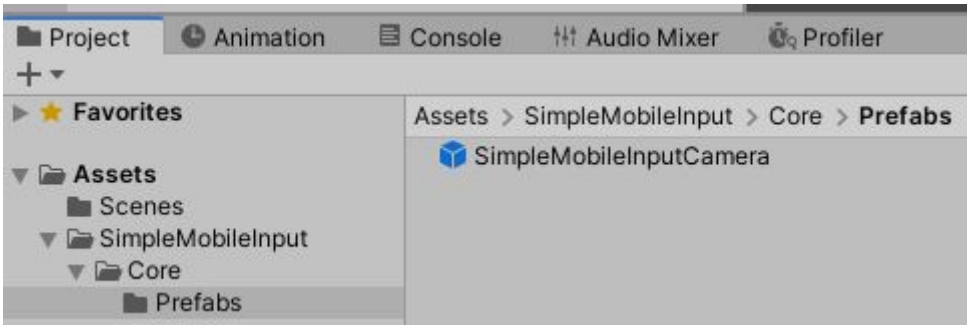
Basic Setup

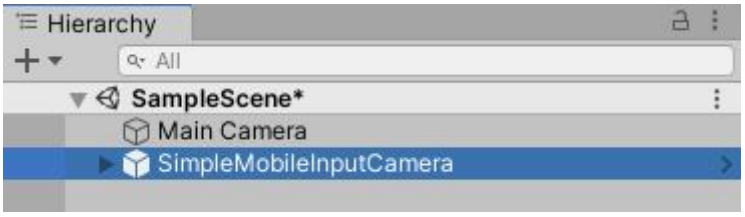
Prefab

With an empty scene.



Drag the prefab **SimpleMobileInputCamera** in the Assets/SimpleMobileInput/Core/Prefabs into the scene.



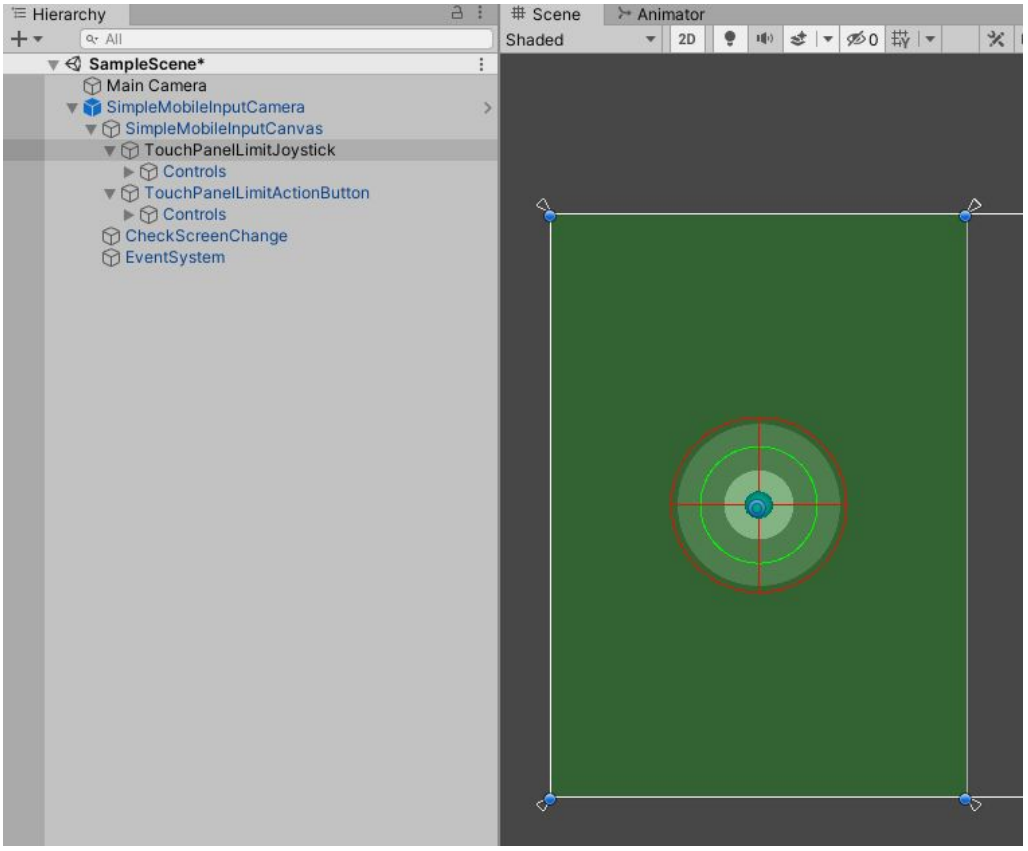


Important, all components on the camera need to be on the UI layer to be seen.

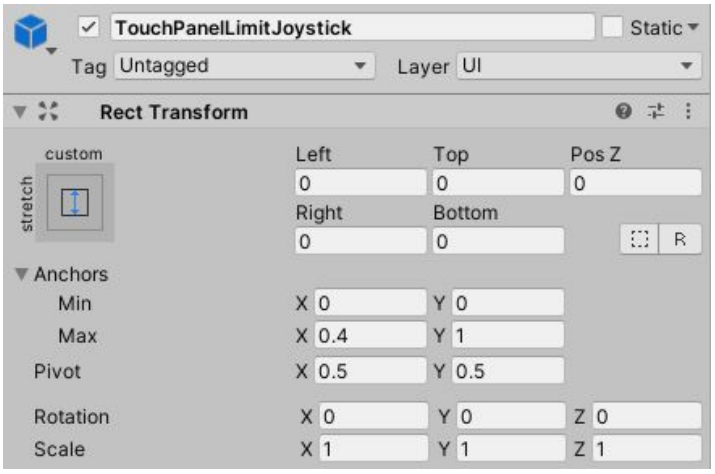
Joystick

Configuration

Set the configuration on the joystick panel.



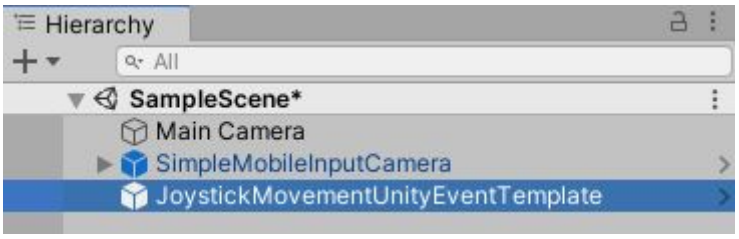
Begin with the size of the panel, this will determine the area that can be touched. In this context, 40% of the screen on the left side.



Bind event

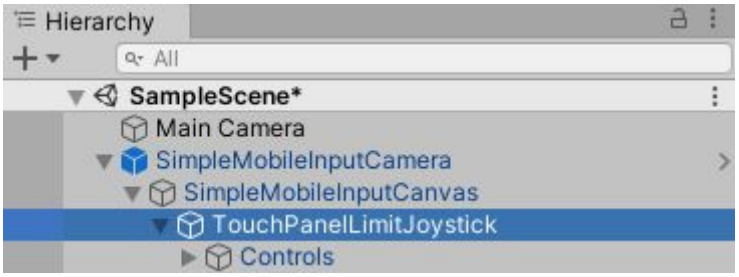
Bind event with a gameobject that contains a script with your movement.

In this context, we will use the prefab **JoystickMovementUnityEventTemplate** in Assets/SimpleMobileInput/Core/Prefabs as an example.

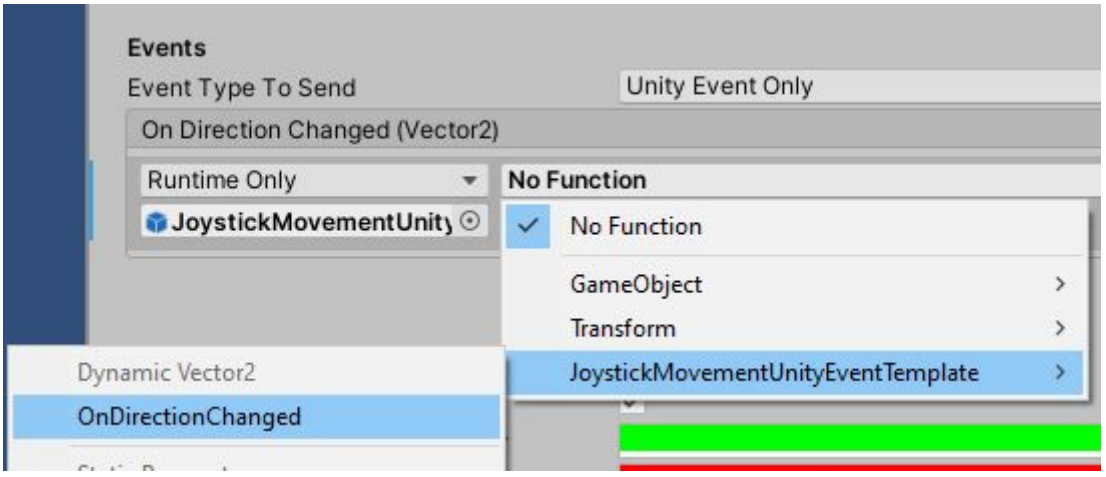


Drag the **JoystickMovementUnityEventTemplate** in unity event of the JoystickInputUI inspector.

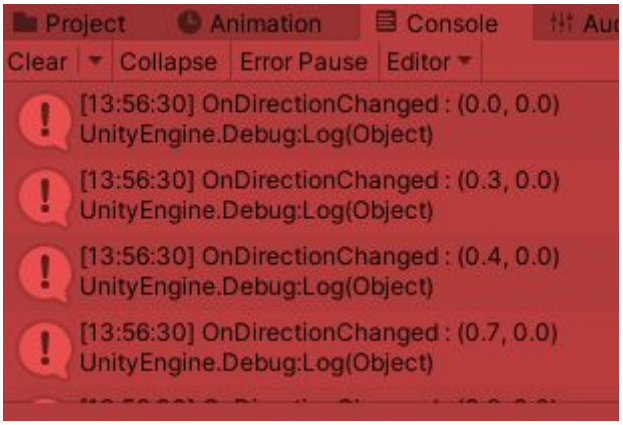
- Select the TouchPanelLimitJoystick



- Drag the gameobject **JoystickMovementUnityEventTemplate** in the **JoystickInputUI** inspector.
- Choose the function **OnDirectionChanged**.



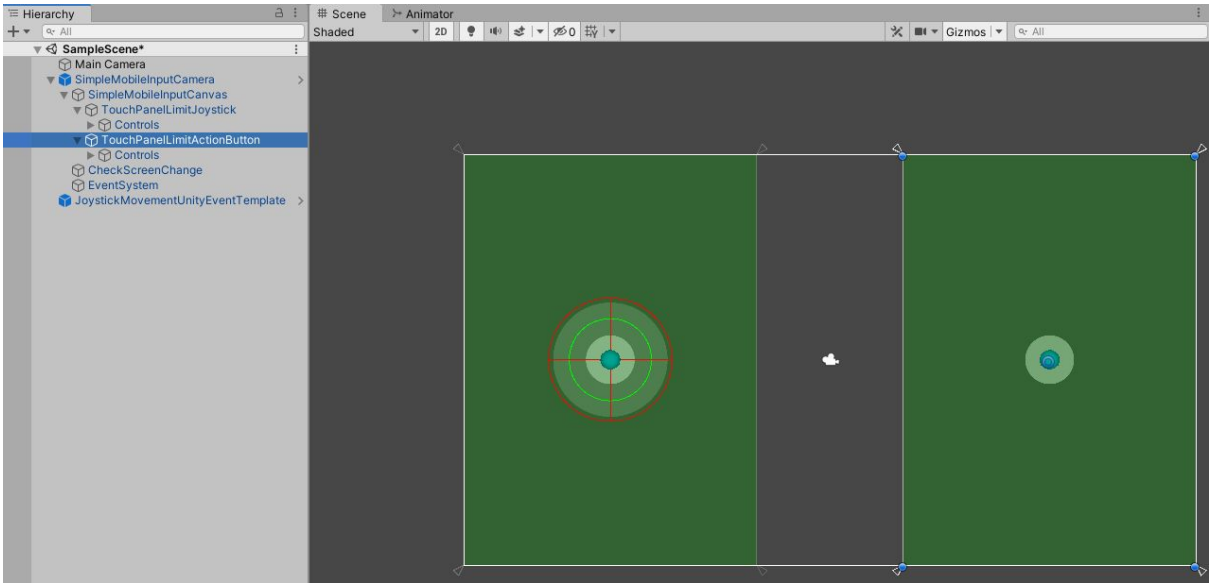
Play in the editor and you will see the log of the direction taken by the joystick.



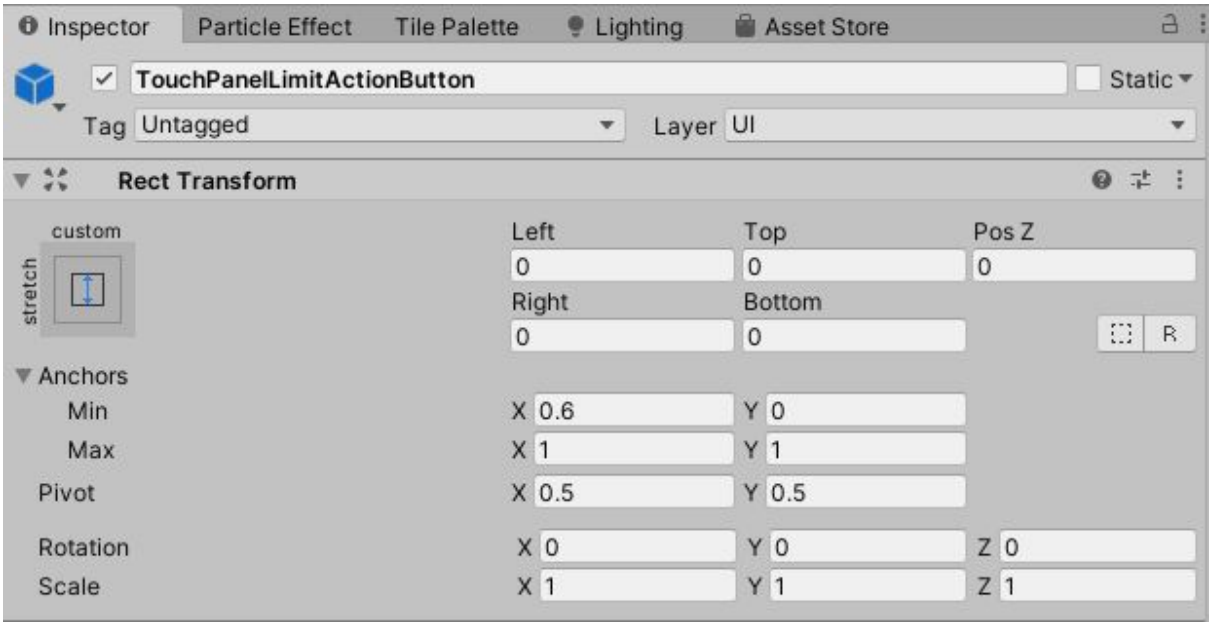
Action Button

Configuration

Set the configuration on the action button panel.



Begin with the size of the panel, this will determine the area that can be touched.
In this context, 40% of the screen on the right side.



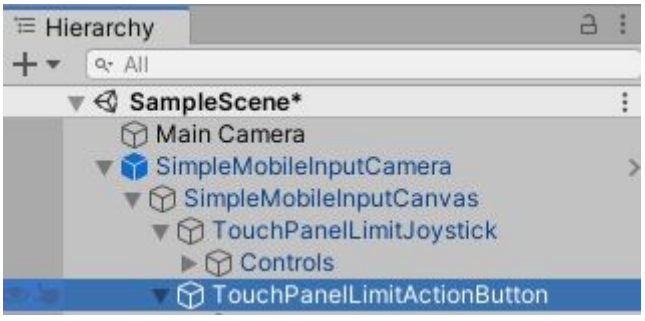
Bind events

Bind event with a gameobject that contains a script with your action.

In this context, we will use the prefab **ActionButtonUnityEventTemplate** in Assets/SimpleMobileInput/Core/Prefabs as an example.

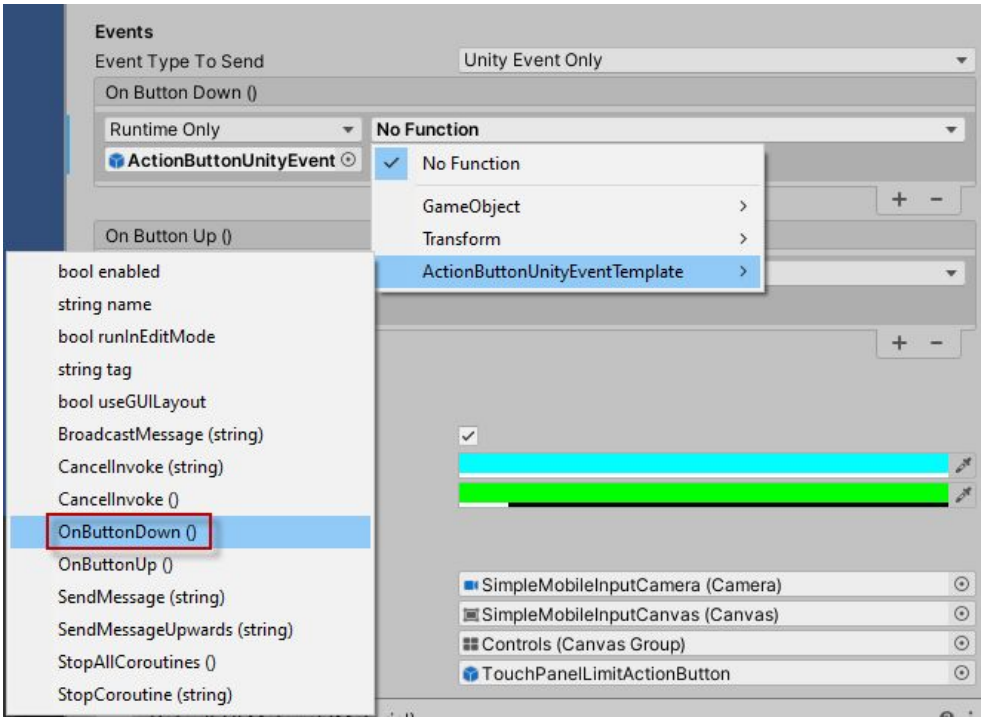
Drag the **ActionButtonUnityEventTemplate** in unity event of the ActionInputUI inspector.

- Select the TouchPanelLimitAction

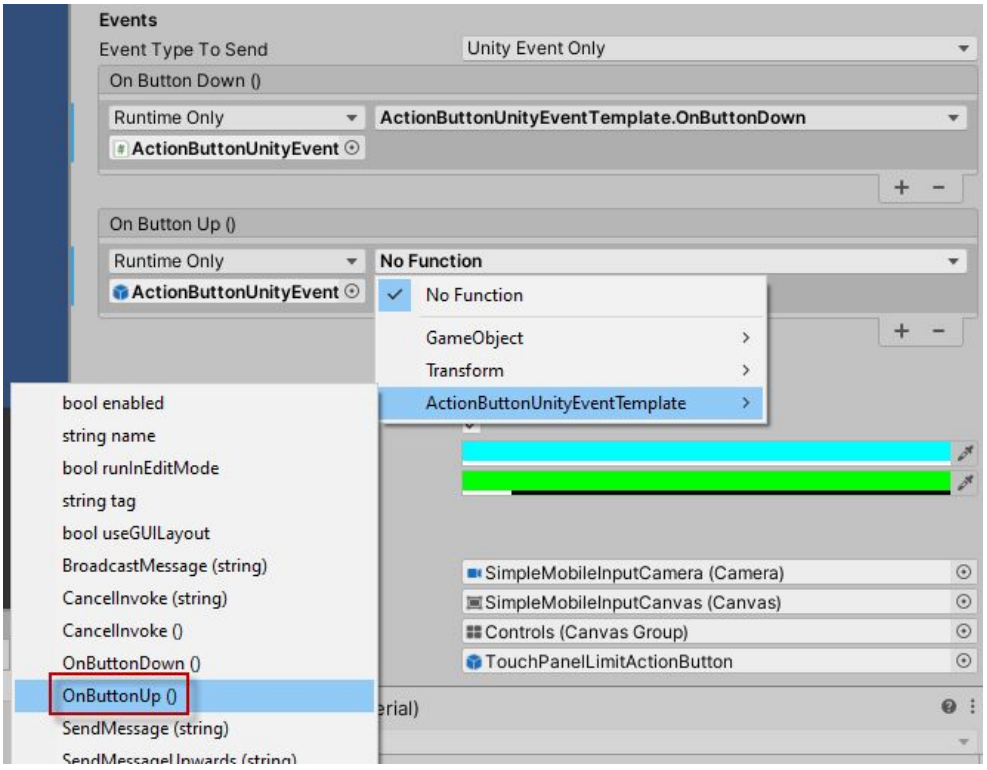


- Drag the gameobject **ActionButtonUnityEventTemplate** in the **ActionInputUI** inspector.

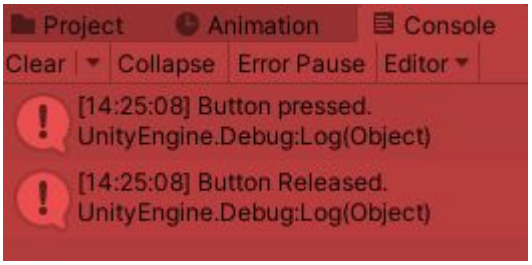
- Choose the function **OnButtonDown** for the OnButtonDown event.



- Choose the function **OnButtonUp** for the OnButtonUp event.



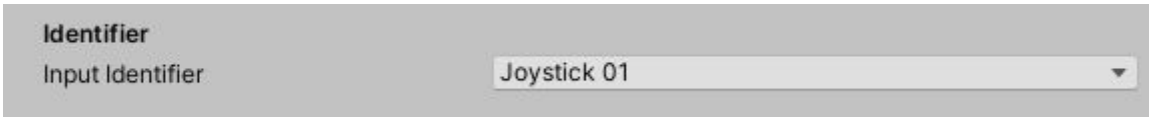
Play in the editor and you will see the log of the direction taken by the joystick.



Joystick Input UI Script

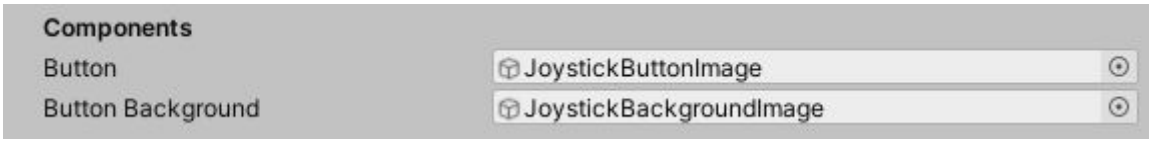
Inspector

Identifier



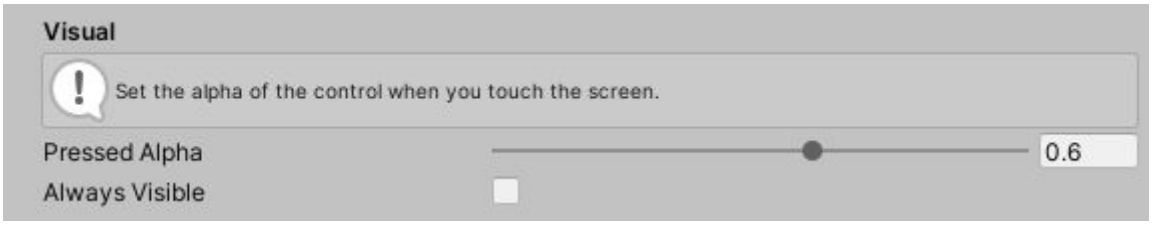
Property	Description
Input Identifier	Identifier used with the static event to be able to know exactly which control sends them.

Components



Properties	Descriptions
Button	Gameobject that represents the center of the joystick.
Button Background	Gameobject that represents the background of the joystick.

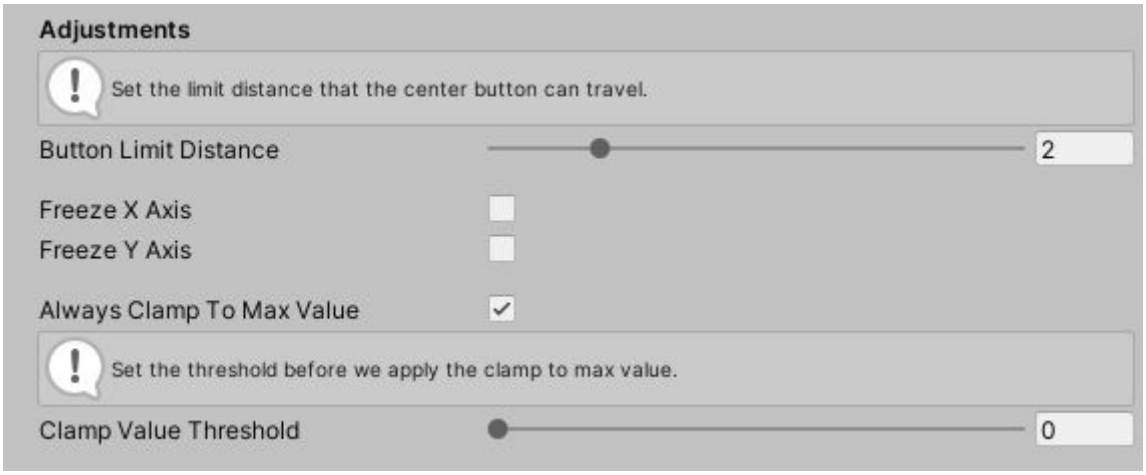
Visual



Properties	Descriptions
Pressed Alpha	Determine the alpha when we touch the control. Can be useful to give feedback to

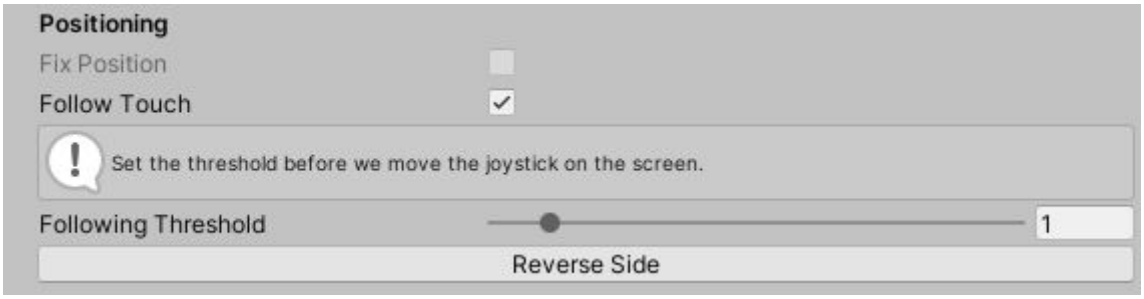
	the user.
Always Visible	Determine if we want the joystick always visible on screen.

Adjustments



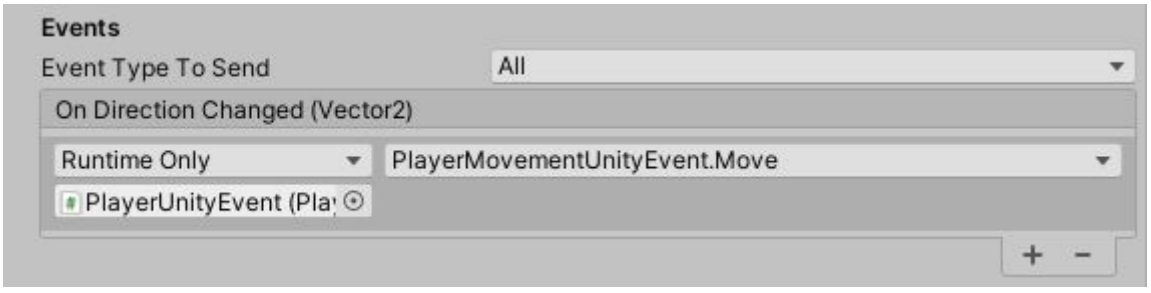
Properties	Descriptions
Button Limit Distance	Determine the limit that the center joystick button can travel.
Freeze X Axis	Determine if we want to freeze the X axis of the joystick. The control will therefore only be horizontal.
Freeze Y Axis	Determine if we want to freeze the Y axis of the joystick. The control will therefore only be vertical.
Always Clamp To Max Value	Determine if the sent value will always be limited to the maximum value. If enabled, the joystick will not have sensitivity.
Clamp Value Threshold	Determine the threshold we want to apply the clamp to max value.

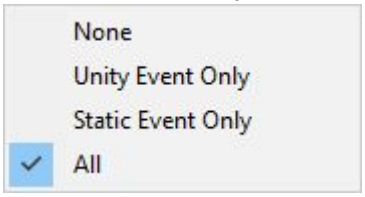
Positioning



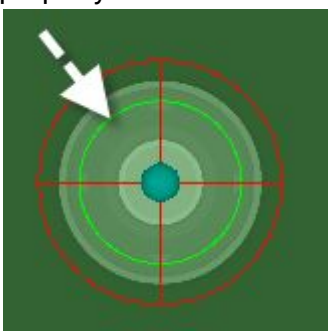
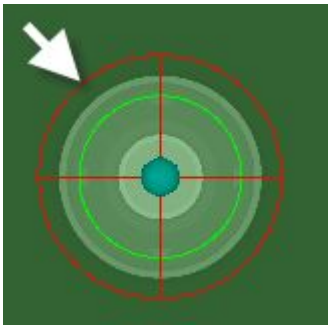
Properties	Descriptions
Fix Position	Determine if the joystick's position is fixed. It will keep the origin position.
Follow Touch	Determine if the joystick can follow the touch position of the finger. Can be useful for not having to re-adjust the position of the finger on screen when playing a game.
Following Threshold	Determine the distance threshold before the repositioning of the joystick on screen.
Reverse Side	Reverse the side of the control on the x axis. Can be used to make a lefty mode.

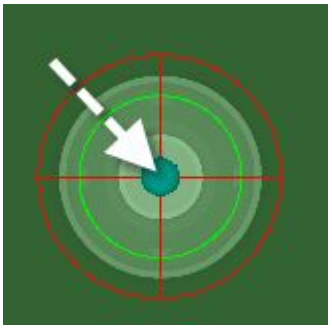
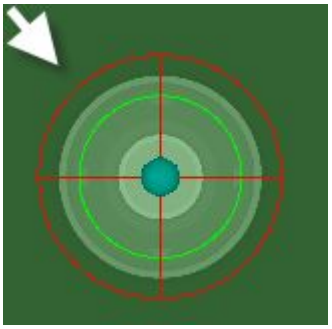
Events



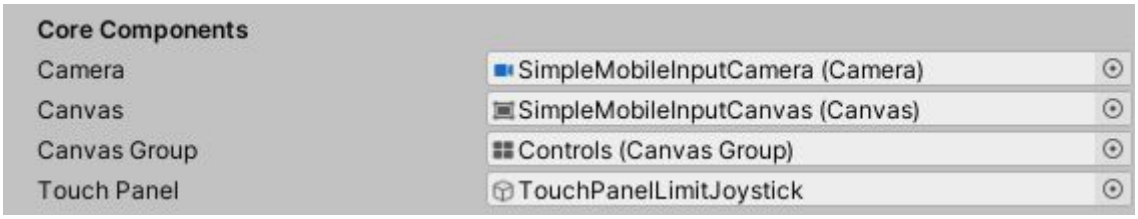
Properties	Descriptions
Event Type To Send	Determine which type of events to be sent. 
On Direction Changed	Unity event used to send the direction of the joystick.

Gizmos

Properties	Descriptions
Show Gizmos	Determine if the gizmos are visible in the editor.
Joystick Button Limit Gizmo Color	Represent the **Button Limit Distance** property. 
Repositioning Threshold Gizmo Color	Represent the **Following Threshold** property. 
Target Position Gizmo Color	Represent the target of the touch. (The current position of the finger)

	
Touch Zone Limit Gizmo Color	Represent the area that can be touched. 

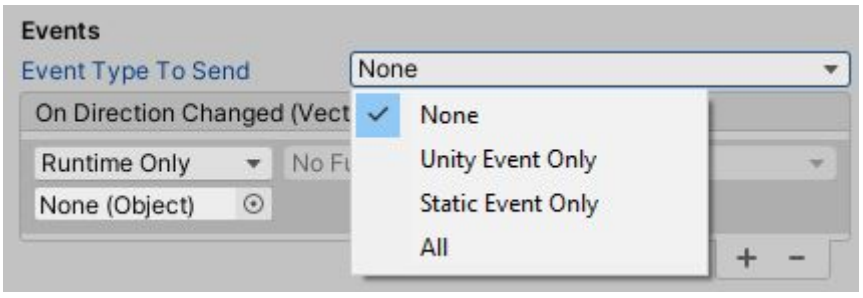
Core Components



Properties	Descriptions
Camera	Camera used for the UI.
Canvas	Canvas used for the UI.
Canvas Group	Canvas group used for the UI and used to fix the alpha.
Touch Panel	Panel used to determine the area that can be touched.

Events binding

There are two types of events you can use. **Unity Event** and **Static Event**.
You can select which one to send.



Unity Event

You will need to select **Unity Event Only** or **All** in the **Event Type To Send** in the **JoystickInputUI**'s inspector.

Drag your gameobject with a movement script and select the function.
The function will need to have the signature :

FunctionName(Vector2 direction)

Script reference: ***JoystickMovementUnityEventTemplate.cs***

Static Event

You will need to select **Static Event Only** or **All** in the **Event Type To Send** in the **JoystickInputUI**'s inspector.

Your script will also need to have the Input Identifier.

```
[SerializeField]
private JoystickInputIdentifier _joystickInputIdentifier = JoystickInputIdentifier.None;
```

The function will need to have the signature :

- FunctionName(JoystickInputIdentifier joystickInputIdentifier, Vector2 direction)

Bind the event in the OnEnable of your script.

```
JoystickInputUI.OnDirectionChanged += OnDirectionChanged;
```

Unbind the event in the OnDisable of your script for not having memory leak.

```
JoystickInputUI.OnDirectionChanged -= OnDirectionChanged;
```

Script reference: ***JoystickMovementStaticEventTemplate.cs***

Action Input UI Script

Inspector

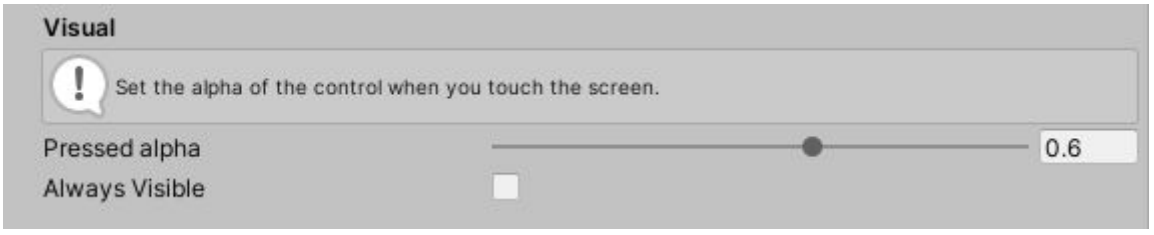
Identifier

Property	Description
Input Identifier	Identifier used with the static event to be able to know exactly which control sends them.

Components

Property	Description
Button	Gameobject that represents the action button.

Visual

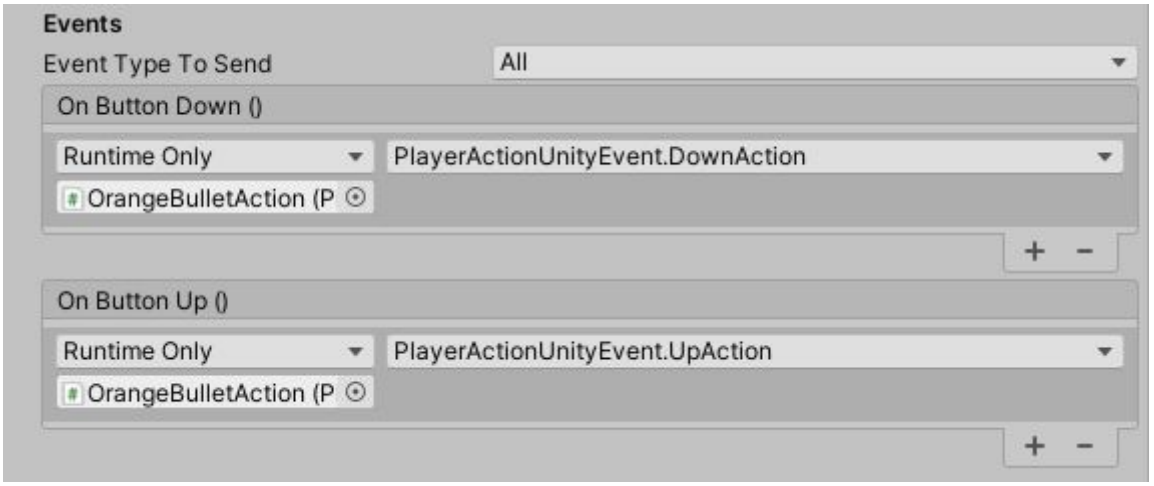


Properties	Descriptions
Pressed Alpha	Determine the alpha when we touch the control. Can be useful to give feedback to the user.
Always Visible	Determine if we want the action button always visible on screen.

Positioning

Properties	Descriptions
Fix Position	Determine if the action button position is fixed. It will keep the origin position.
Reverse Side	Reverse the side of the control on the x axis. Can be used to make a lefty mode.

Events



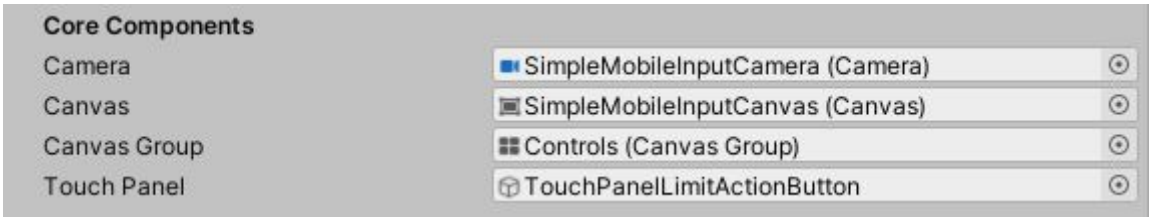
Properties	Descriptions
Event Type To Send	Determine which type of events to be sent. None Unity Event Only Static Event Only ✓ All
On Button Down	Unity event used when the action button is touched.

On Button Up	Unity event used when the action button is released.

Gizmos

Properties	Descriptions
Show Gizmos	Determine if the gizmos are visible in the editor.
Target Position Gizmo Color	Represent the target of the touch. (The current position of the finger) 
Touch Zone Limit Gizmo Color	Represent the area that can be touched. 

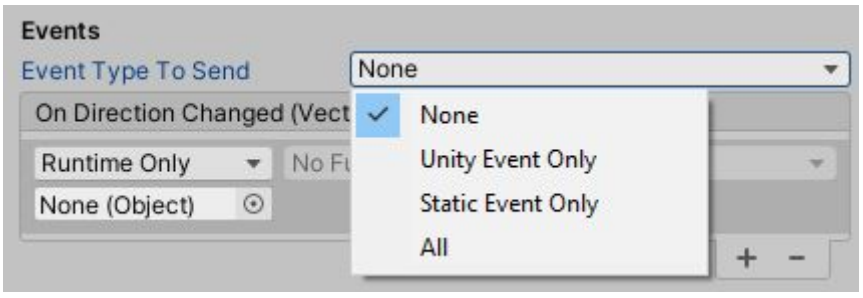
Core Components



Properties	Descriptions
Camera	Camera used for the UI.
Canvas	Canvas used for the UI.
Canvas Group	Canvas group used for the UI and used to fix the alpha.
Touch Panel	Panel used to determine the area that can be touched.

Events binding

There are two types of events you can use. **Unity Event** and **Static Event**.
You can select which one to send.



Unity Event

You will need to select **Unity Event Only** or **All** in the **Event Type To Send** in the **ActionInputUI**'s inspector.

Drag your gameobject with an action script and select the function.
The function will need to have the signature : FunctionName()

Script reference: ***ActionButtonUnityEventTemplate.cs***

Static Event

You will need to select **Static Event Only** or **All** in the **Event Type To Send** in the **JoystickInputUI**'s inspector.

Your script will also need to have the Input Identifier.

```
[SerializeField]
 private ActionInputIdentifier _actionInputIdentifier = ActionInputIdentifier.None;
```

The function will need to have the signature :

FunctionName(*ActionInputIdentifier actionInputIdentifier*)

Bind the event in the OnEnable of your script.

```
ActionInputUI.OnActionDown += OnButtonDown;
ActionInputUI.OnActionUp += OnButtonUp;
```

Unbind the event in the OnDisable of your script for not having memory leak.

```
ActionInputUI.OnActionDown -= OnButtonDown;
ActionInputUI.OnActionUp -= OnButtonUp;
```

Script reference: ***ActionButtonStaticEventTemplate.cs***