Button Long Press

Unity Asset by Kodo Linija

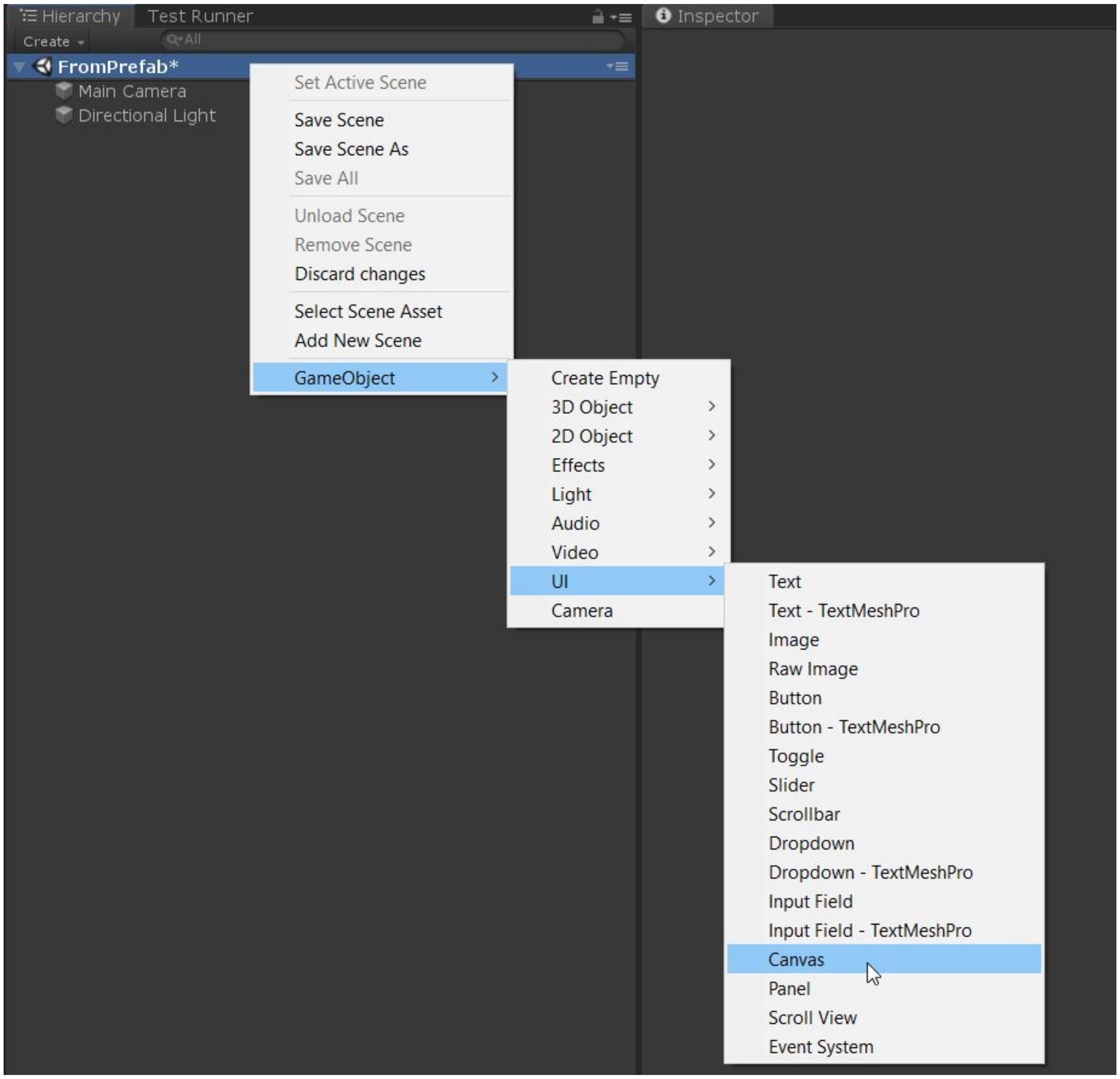
Thank you for purchasing the **Button Long Press** Unity Asset from **Kodo Linija**! This guide should help you get started quickly.

The asset supports mouse, touch, keyboard and gamepad events, on both Legacy and new Input System.

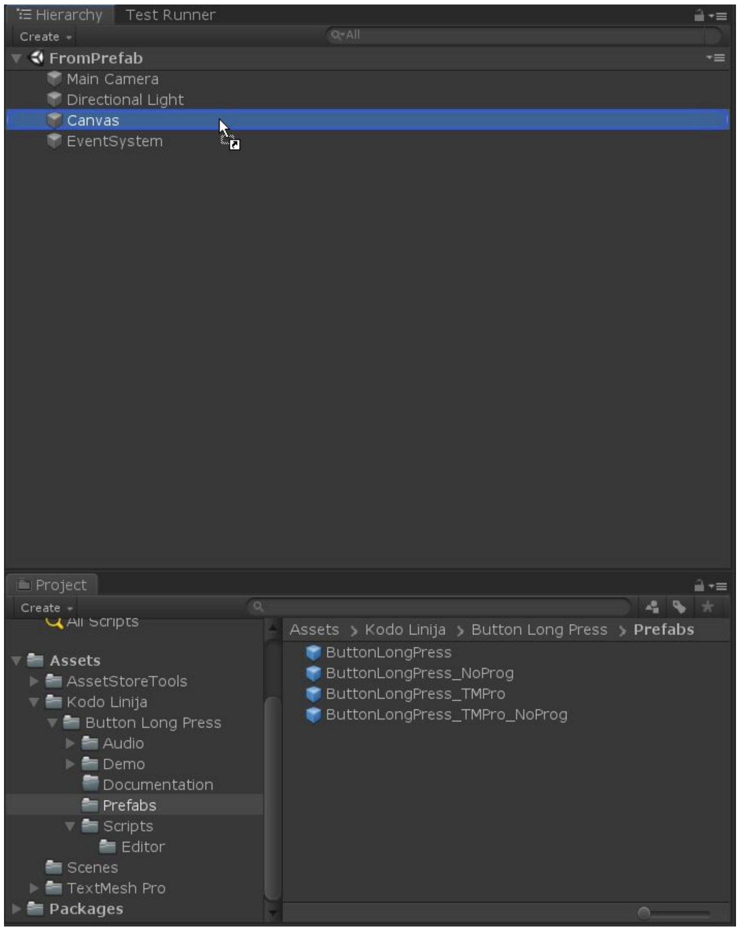
Table of Contents

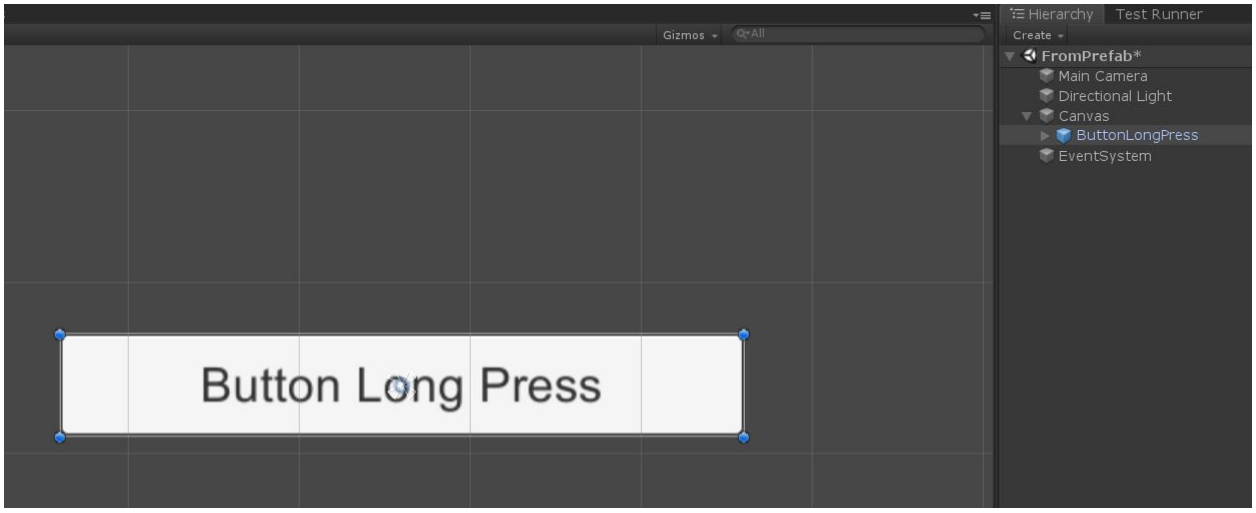
Creating a ButtonLongPress From Prefab

1. Make sure you have **UI Canvas** game object in your scene



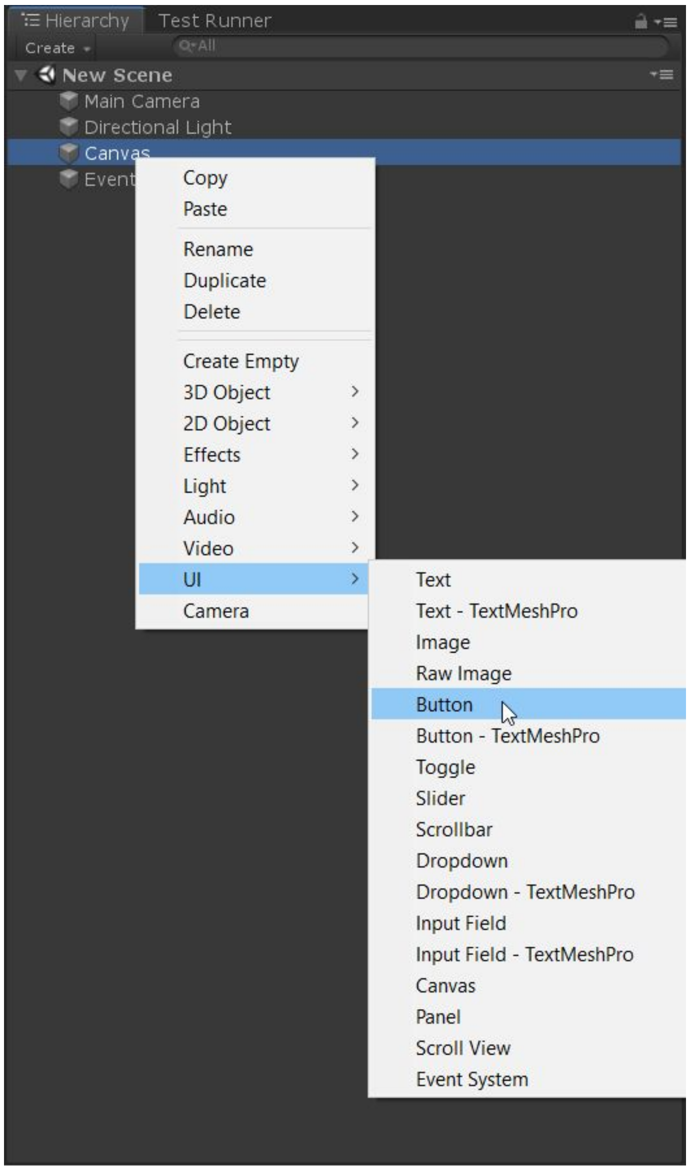
2. Drag a prefab of your choice from **Assets > Kodo Linija > Button Long Press > Prefabs** into your **Canvas**.



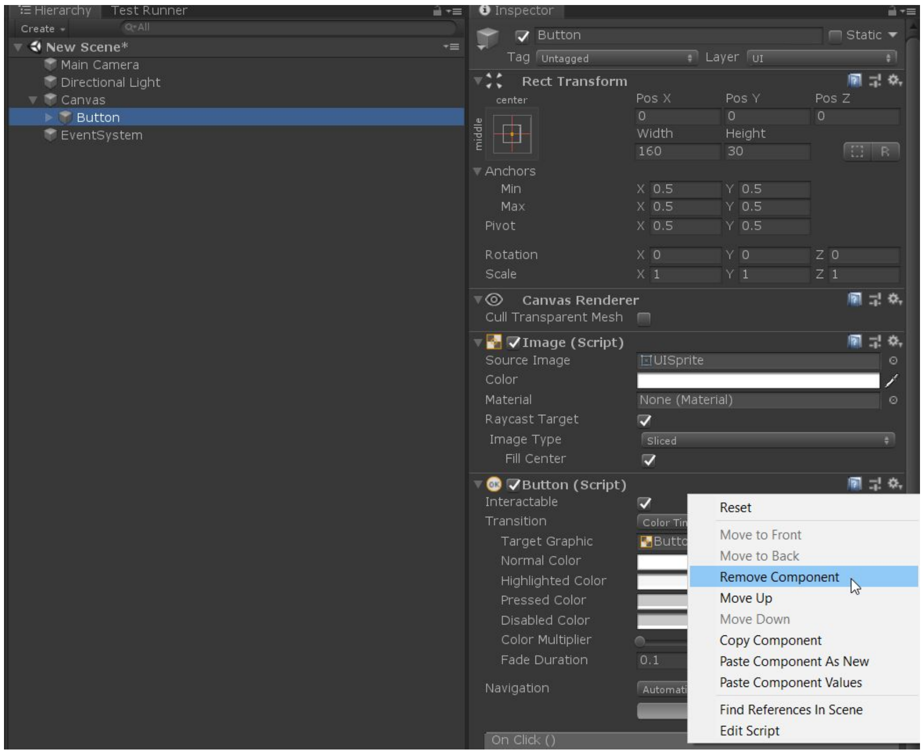


Creating ButtonLongPress From Scratch

1. Create a **UI Button** (or TextMeshPro version). It should be inside your **UI Canvas** object.



2. Remove the **Button** component from the button game object you just created. Note that you will lose your button settings, right now there is no way to copy settings from existing buttons.



Button ✓ Static
Tag Untagged Layer UI

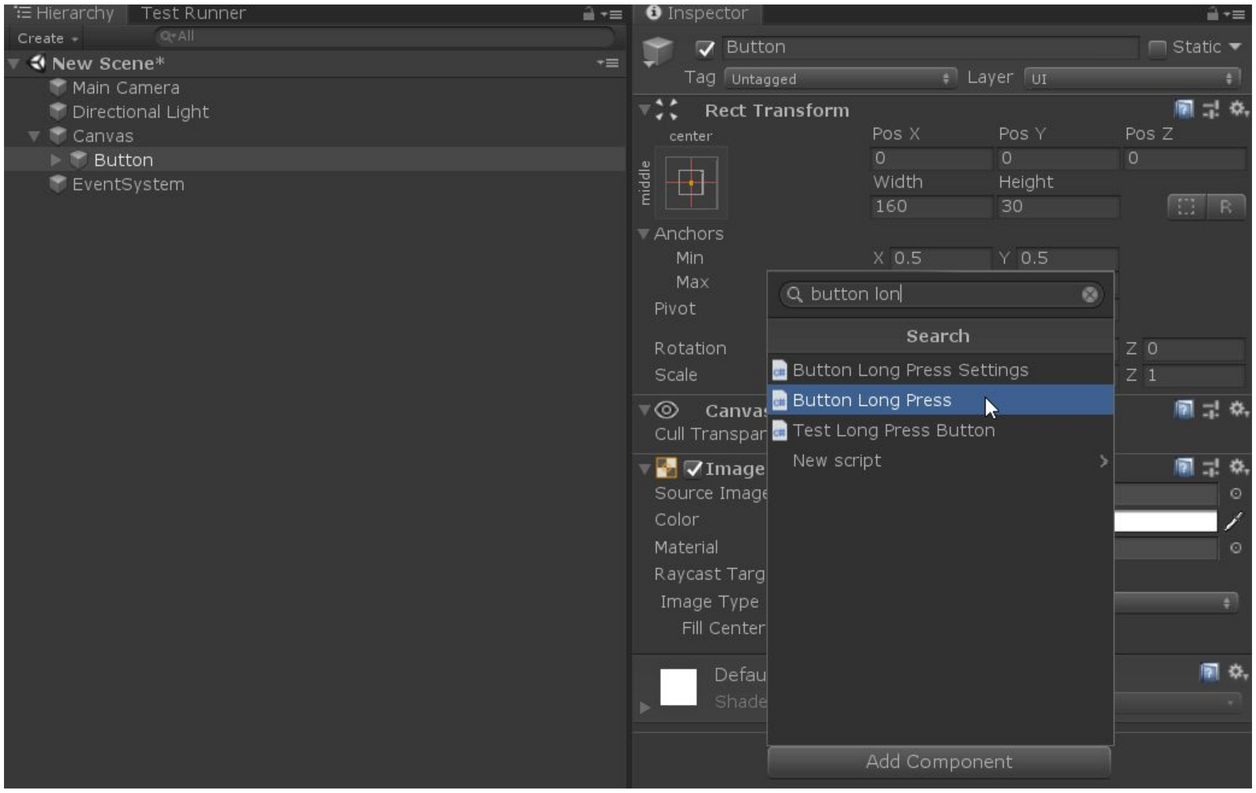
Rect Transform
center middle
Pos X 0 Pos Y 0 Pos Z 0
Width 160 Height 30
Anchors
Min X 0.5 Y 0.5
Max X 0.5 Y 0.5
Pivot X 0.5 Y 0.5
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1

Canvas Renderer
Cull Transparent Mesh

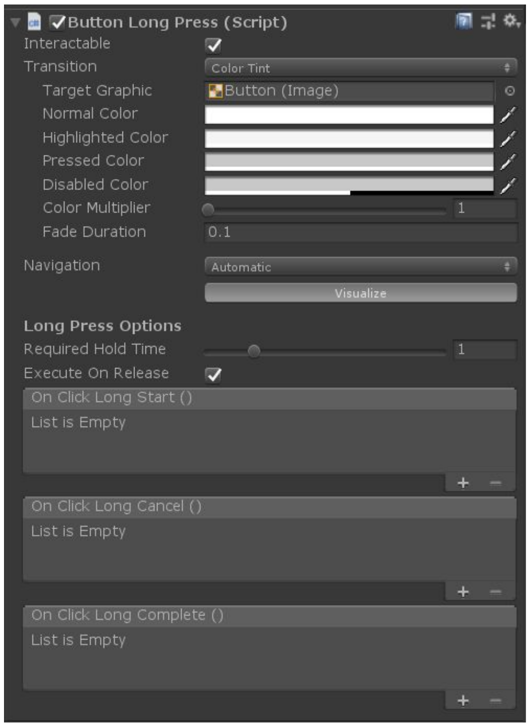
✓ Image (Script)
Source Image UISprite
Color
Material None (Material)
Raycast Target ✓
Image Type Sliced
Fill Center ✓

✓ Button (Script)
Interactable ✓
Transition Color Tin
Target Graphic Butto
Normal Color
Highlighted Color
Pressed Color
Disabled Color
Color Multiplier
Fade Duration 0.1
Navigation Automati
On Click ()

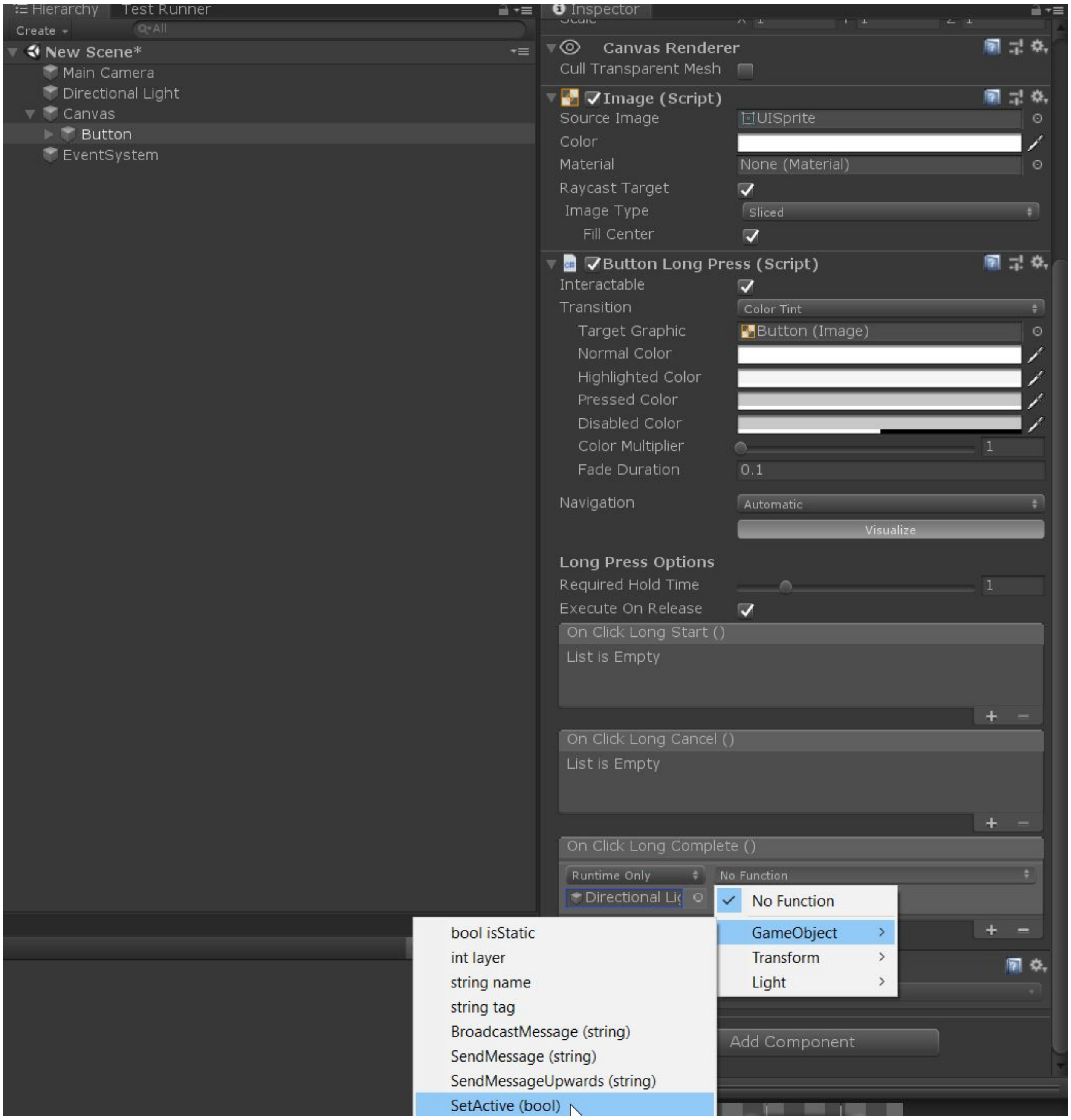
3. Inside the inspector, click **Add Component**, find the **Button Long Press** component, and click on it to add it.



The result should be very similar to what you had in your regular button, but with the **On Click** event block replaced with **Long Press Options**.



4. Set the **Required Hold Time** (in seconds), **Execute On Release** and **On Click Long Complete** event in the inspector to configure the button and set the action. Additionally you can use the **On Click Long Start** and **On Click Long Cancel** events to react to click start and incomplete release.



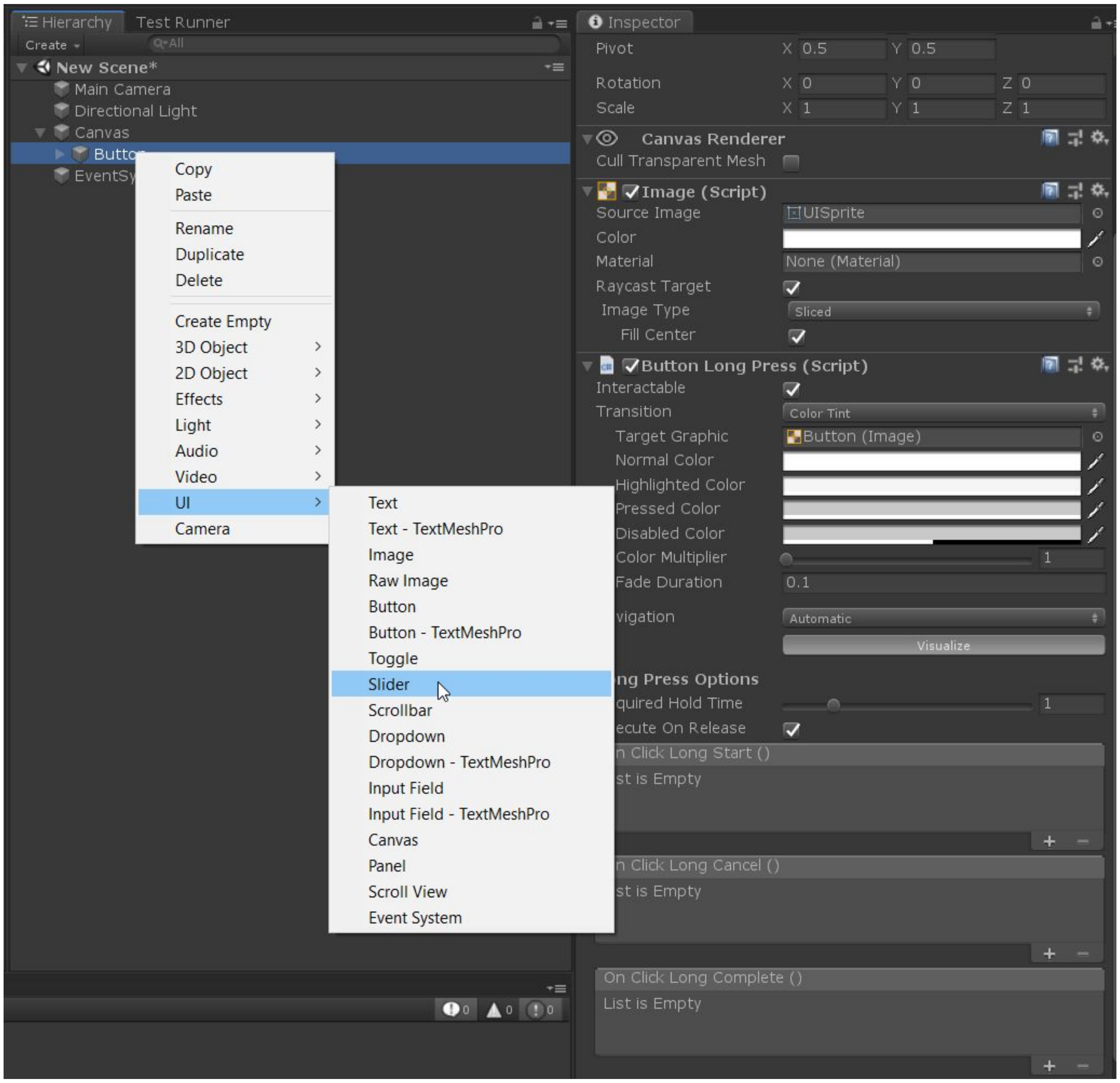
Canvas Renderer
Cull Transparent Mesh ☐

Image (Script)
Source Image: UISprite
Color
Material: None (Material)
Raycast Target ✓
Image Type: Sliced
Fill Center ✓

Button Long Press (Script)
Interactable ✓
Transition: Color Tint
Target Graphic: Button (Image)
Normal Color
Highlighted Color
Pressed Color
Disabled Color
Color Multiplier: 1
Fade Duration: 0.1
Navigation: Automatic

Long Press Options
Required Hold Time: 1
Execute On Release ✓
On Click Long Start ()
List is Empty
On Click Long Cancel ()
List is Empty
On Click Long Complete ()
Runtime Only | No Function
Directional Lig

5. Optionally, you can create a **UI Slider** game object inside the button. **ButtonLongPress** will automatically update that slider's value depending on the click hold progress.



Pivot X 0.5 Y 0.5
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1

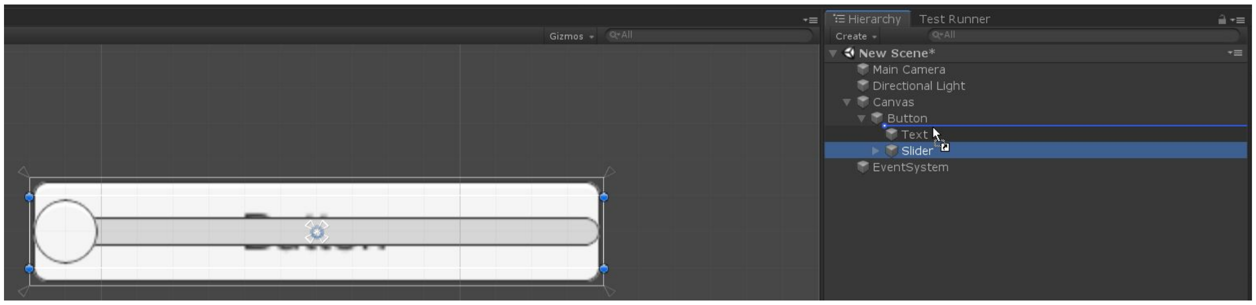
Canvas Renderer
Cull Transparent Mesh

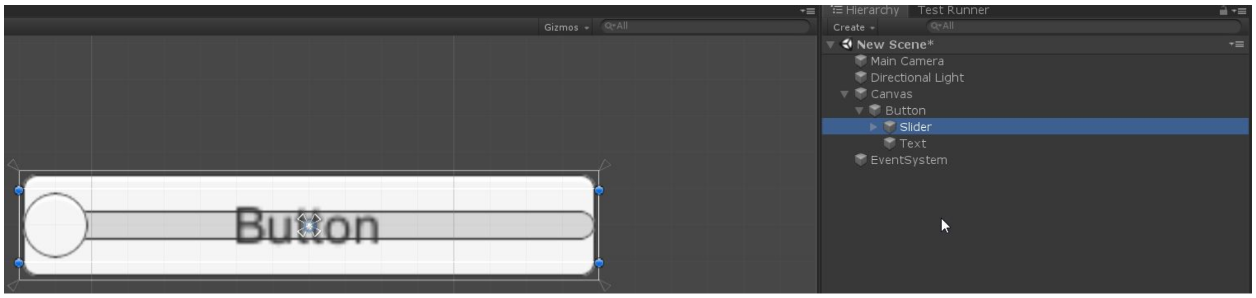
Image (Script)
Source Image UISprite
Color
Material None (Material)
Raycast Target ✓
Image Type Sliced
Fill Center ✓

Button Long Press (Script)
Interactable ✓
Transition Color Tint
Target Graphic Button (Image)
Normal Color
Highlighted Color
Pressed Color
Disabled Color
Color Multiplier 1
Fade Duration 0.1
vigation Automatic

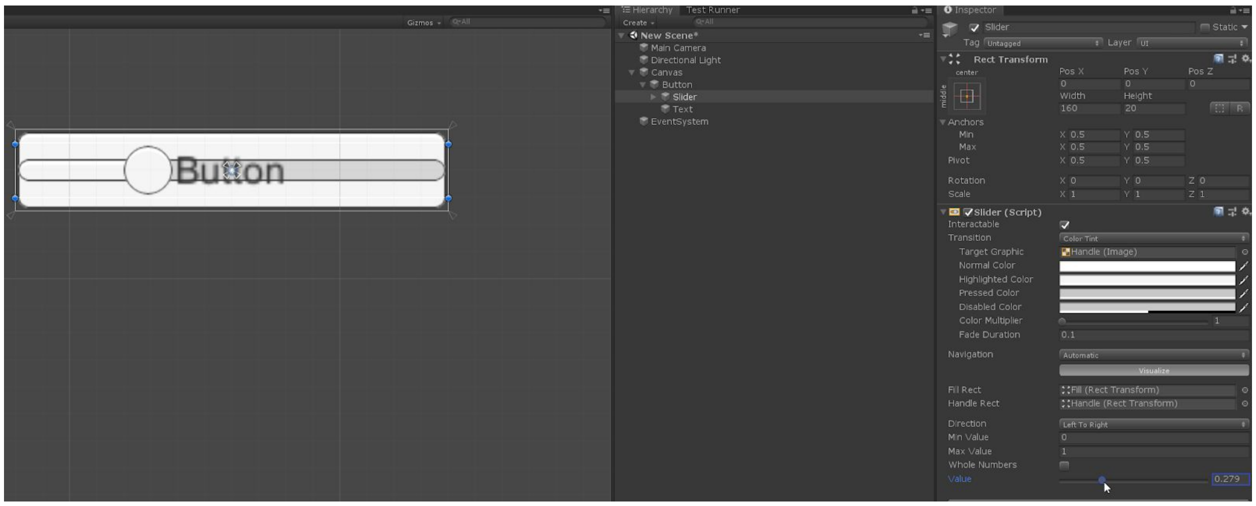
ng Press Options
quired Hold Time 1
ecute On Release ✓
n Click Long Start ()
st is Empty
n Click Long Cancel ()
st is Empty
On Click Long Complete ()
List is Empty

6. Hold and drag the **Slider** game object to position it above the **Text**, so it would not cover the button label.



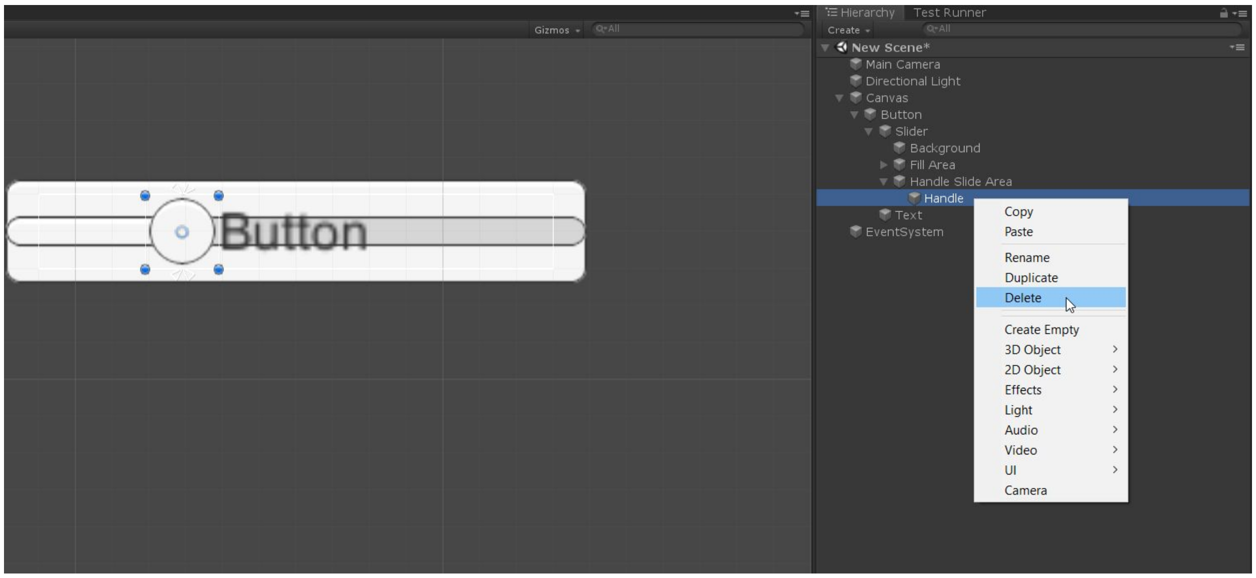


7. Set the slider value somewhere between 0 and 1, it will be easier to style it this way

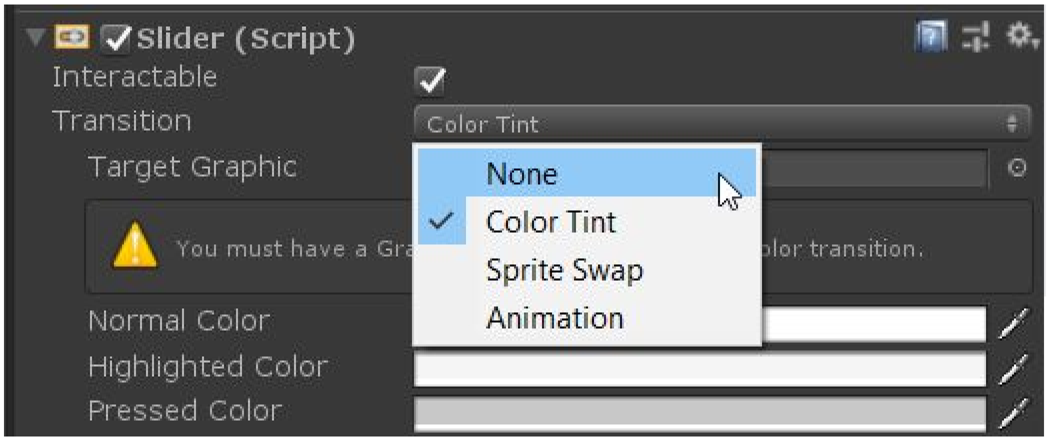


Slider Static
Tag Untagged Layer UI
Rect Transform
center / middle Pos X 0 Pos Y 0 Pos Z 0
Width 160 Height 20
Anchors
Min X 0.5 Y 0.5
Max X 0.5 Y 0.5
Pivot X 0.5 Y 0.5
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1
Slider (Script)
Interactable ✓
Transition Color Tint
Target Graphic Handle (Image)
Normal Color
Highlighted Color
Pressed Color
Disabled Color
Color Multiplier 1
Fade Duration 0.1
Navigation Automatic
Visualize
Fill Rect Fill (Rect Transform)
Handle Rect Handle (Rect Transform)
Direction Left To Right
Min Value 0
Max Value 1
Whole Numbers
Value 0.279

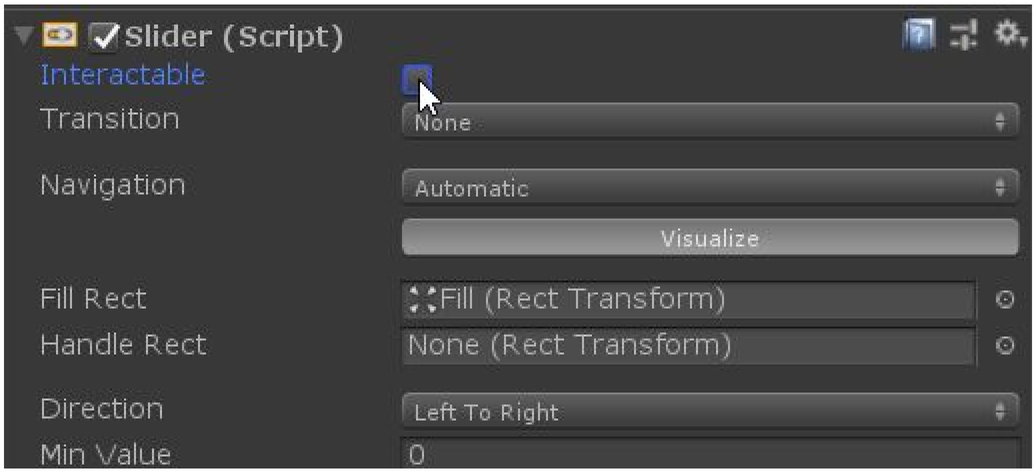
8. Remove the **Slider > Handle Slide Area > Handle** game object.



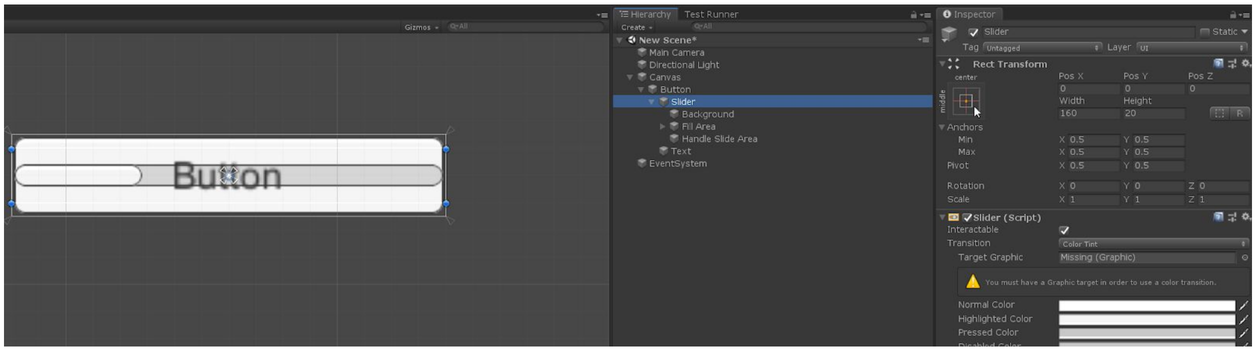
9. Change the slider **Transition** to **None**. Don't worry about the missing graphic, it's the handle we just deleted. It will disappear when transition is unset.



10. Then make the **Slider** not **Interactable** (uncheck the checkbox).

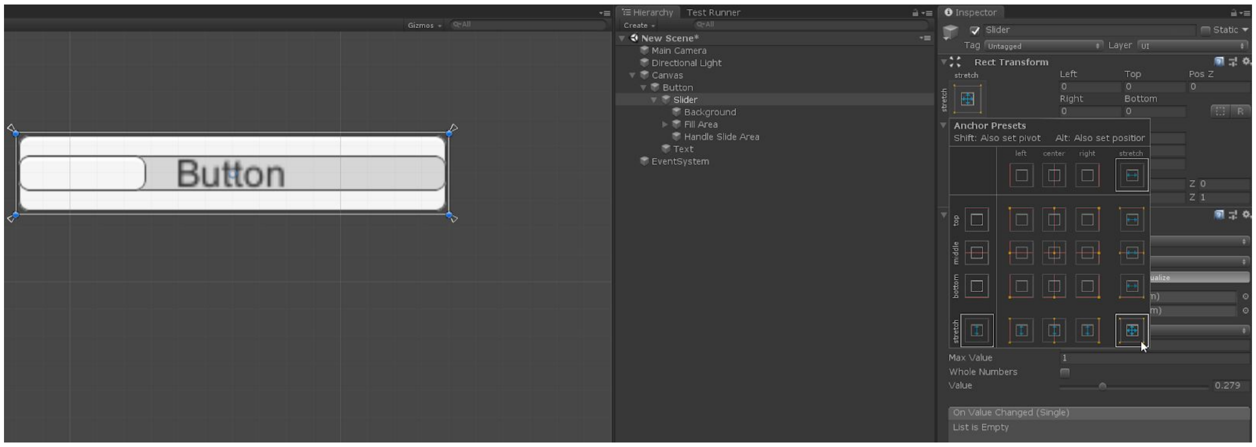


11. Click on the **Slider** game object, then on top left tool of the **Rect Transform** component in the **Inspector**



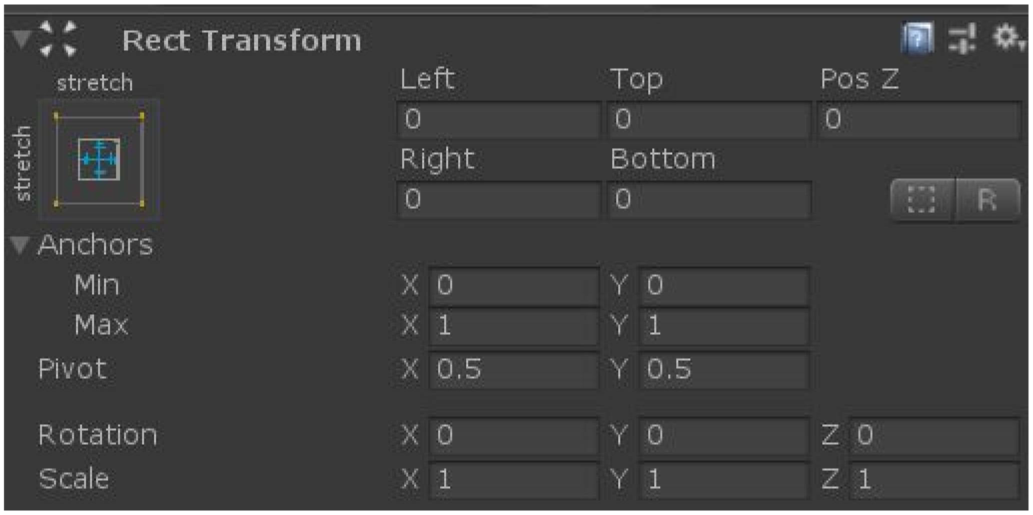
Slider — Static
Tag Untagged — Layer UI
Rect Transform
center / middle
Pos X 0 — Pos Y 0 — Pos Z 0
Width 160 — Height 20
Anchors
Min X 0.5 Y 0.5
Max X 0.5 Y 0.5
Pivot X 0.5 Y 0.5
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1
Slider (Script)
Interactable ✓
Transition Color Tint
Target Graphic Missing (Graphic)
You must have a Graphic target in order to use a color transition.
Normal Color
Highlighted Color
Pressed Color

12. Hold both **Alt** and **Shift** and click on the bottom right (**stretch** / **stretch**) option



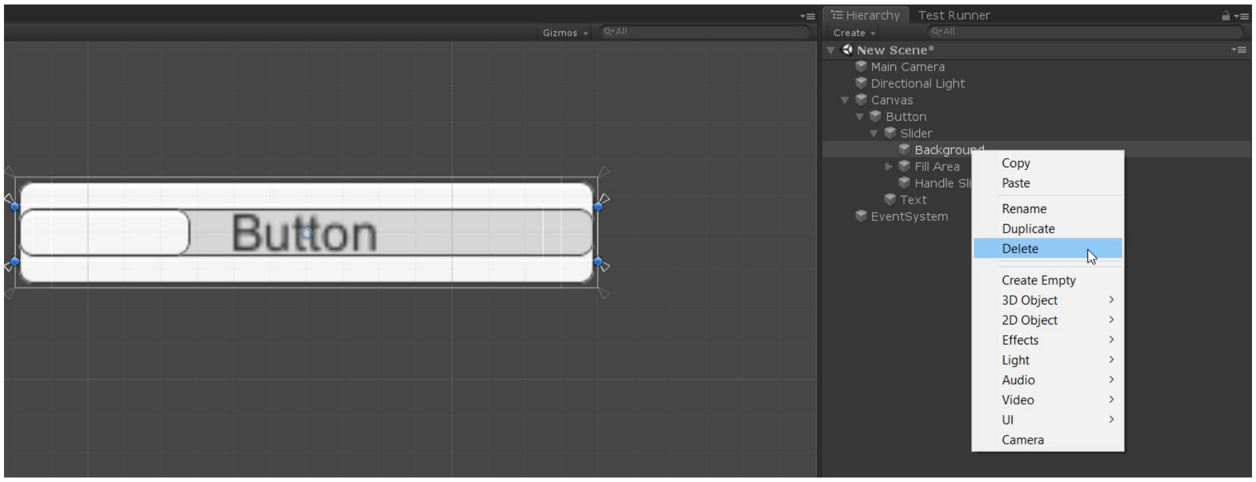
Slider — Static
Tag Untagged — Layer UI
Rect Transform
stretch / stretch
Left 0 — Top 0 — Pos Z 0
Right 0 — Bottom 0
Anchor Presets
Shift: Also set pivot — Alt: Also set position
Max Value 1
Whole Numbers
Value 0.279
On Value Changed (Single)
List is Empty

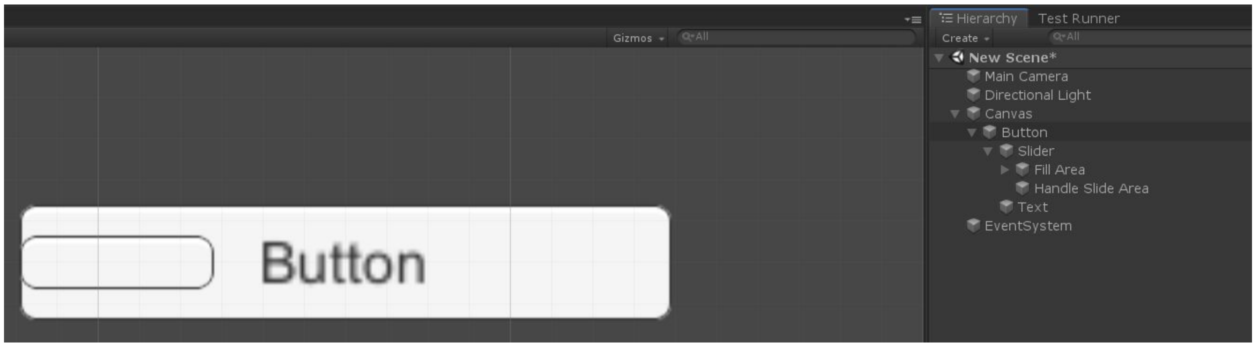
Slider Rect Transform component configuration should now look like this:



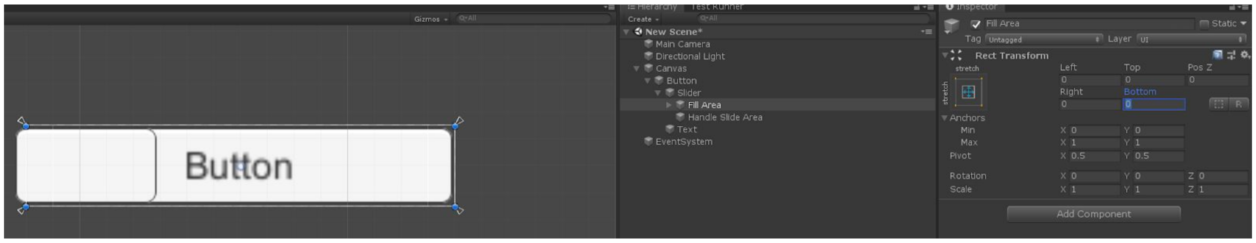
Rect Transform
Left 0 — Top 0 — Pos Z 0
Right 0 — Bottom 0
Anchors
Min X 0 Y 0
Max X 1 Y 1
Pivot X 0.5 Y 0.5
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1

13. Delete the **Slider > Background** game object.



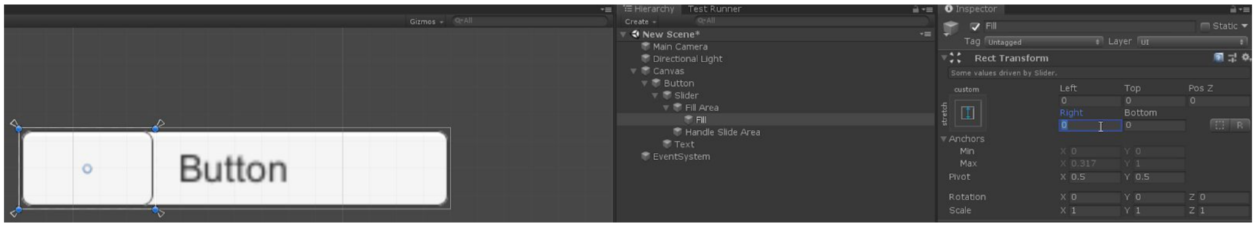


14. Select **Slider > Fill Area** game object, and set **Rect Transform Left**, **Right**, **Top** and **Bottom** values to **0**, **Anchor Min Y** to **0**, **Anchor Max Y** to **1**. The slider should become the same size as the button. Note that after you will change Anchors, **Top** and **Bottom** values will get changed automatically, so you will have to overwrite those back to **0**.



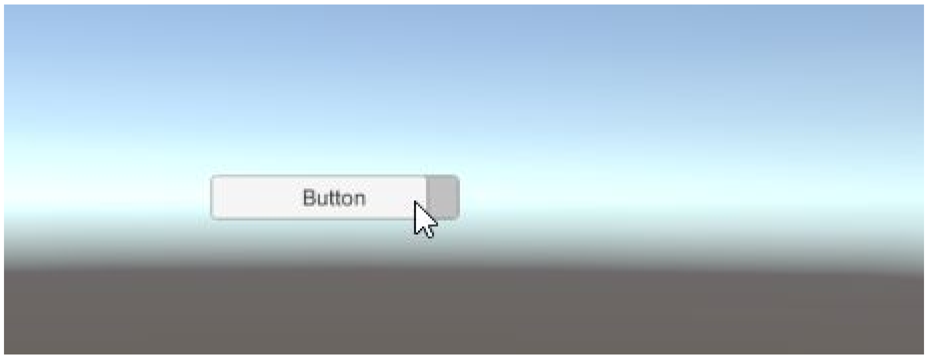
Fill Area
Tag Untagged Layer UI
Rect Transform
stretch
Left 0 Top 0 Pos Z 0
Right 0 Bottom 0
Anchors
Min X 0 Y 0
Max X 1 Y 1
Pivot X 0.5 Y 0.5
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1

15. Select **Slider > Fill Area > Fill** game object. Set **Rect Transform Left** and **Right** to **0**.



Fill
Tag Untagged Layer UI
Rect Transform
Some values driven by Slider.
custom
stretch
Left 0 Top 0 Pos Z 0
Right 0 Bottom 0
Anchors
Min X 0 Y 0
Max X 0.317 Y 1
Pivot X 0.5 Y 0.5
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1

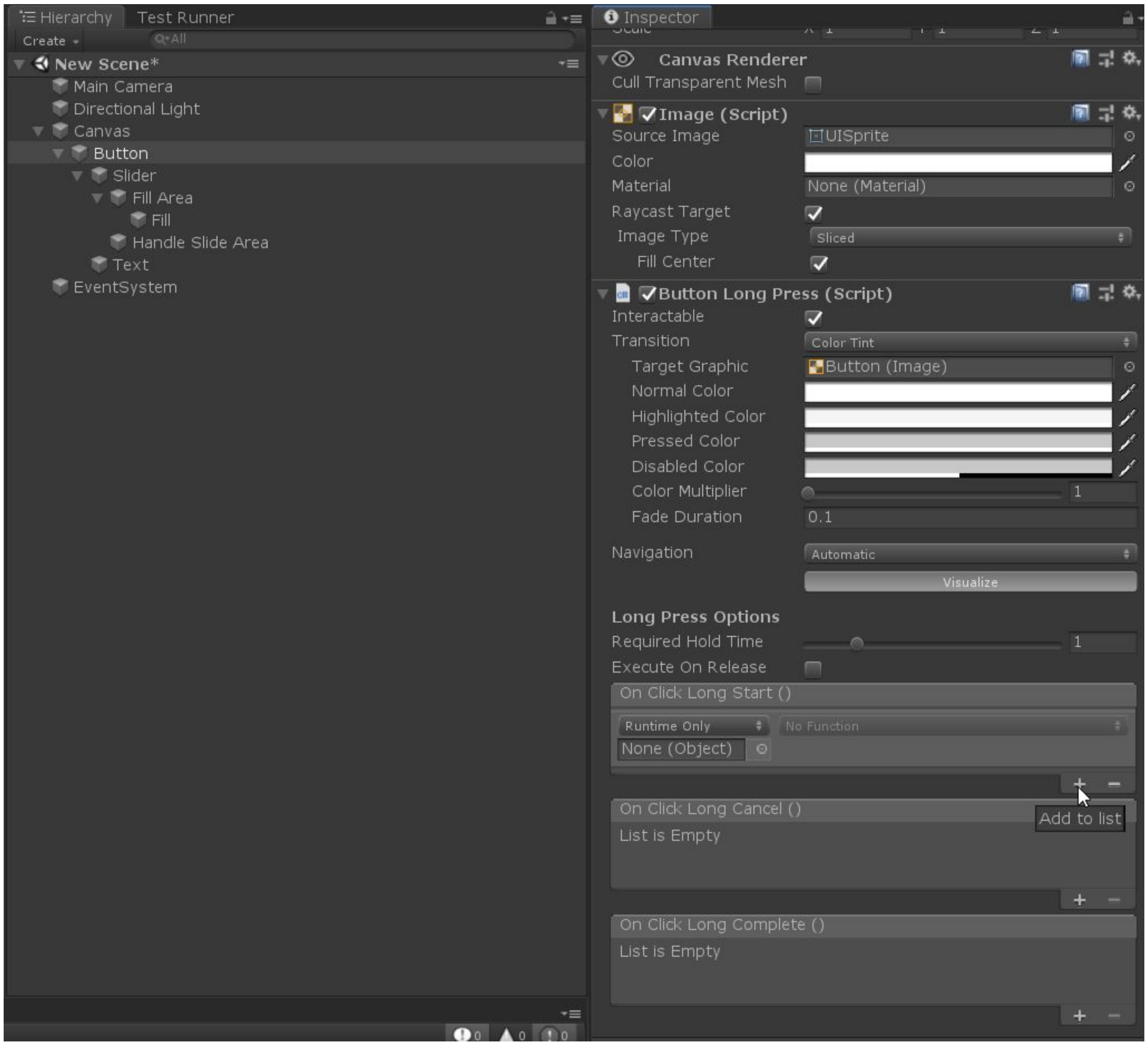
16. That’s it! Now enter play mode and test your button. It will do nothing for now, but we will add some actions soon!



Configuring a ButtonLongPress

We are going to make the button turn off the scene's Directional Light when it's being held long enough.

1. Click on your **Button** game object in the **Hierarchy**. Find **Button Long Press** component in the **Inspector** view.
2. Click on the **+** button in the bottom right side of the **On Click Long Start** block.



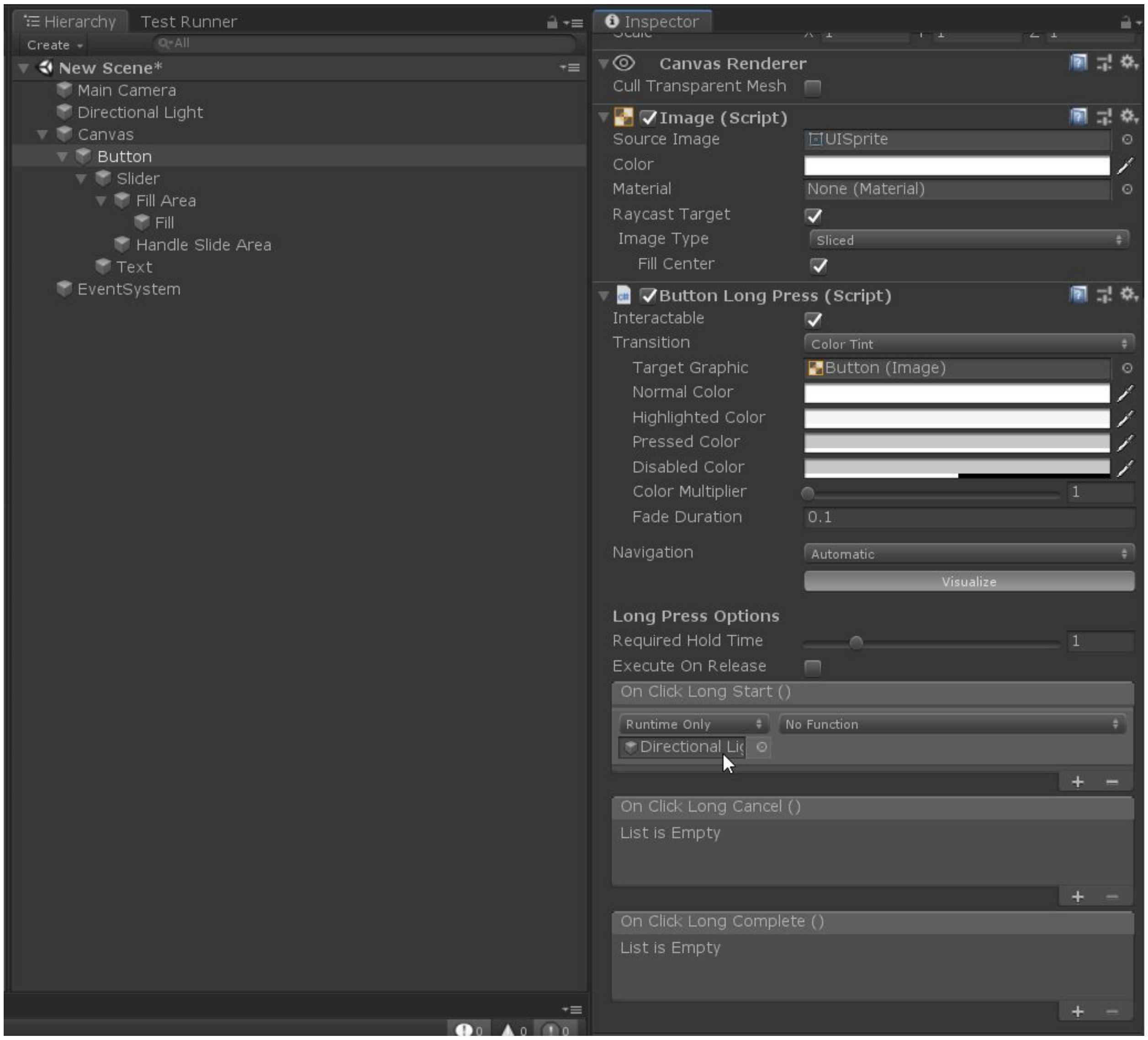
Canvas Renderer
Cull Transparent Mesh ☐

Image (Script)
Source Image: UISprite
Color
Material: None (Material)
Raycast Target ✓
Image Type: Sliced
Fill Center ✓

Button Long Press (Script)
Interactable ✓
Transition: Color Tint
Target Graphic: Button (Image)
Normal Color
Highlighted Color
Pressed Color
Disabled Color
Color Multiplier: 1
Fade Duration: 0.1
Navigation: Automatic
Visualize

Long Press Options
Required Hold Time: 1
Execute On Release ☐
On Click Long Start ()
Runtime Only | No Function
None (Object)
On Click Long Cancel ()
List is Empty
On Click Long Complete ()
List is Empty

3. Drag and drop **Directional Light** object from **Hierarchy** into the **None (Object)** slot in **On Click Long Start**.


Hierarchy | Test Runner

New Scene*
- Main Camera
- Directional Light
- Canvas
  - Button
    - Slider
      - Fill Area
        - Fill
      - Handle Slide Area
    - Text
- EventSystem

Inspector

Canvas Renderer
Cull Transparent Mesh

Image (Script)
Source Image: UISprite
Color
Material: None (Material)
Raycast Target: ✓
Image Type: Sliced
Fill Center: ✓

Button Long Press (Script)
Interactable: ✓
Transition: Color Tint
Target Graphic: Button (Image)
Normal Color
Highlighted Color
Pressed Color
Disabled Color
Color Multiplier: 1
Fade Duration: 0.1
Navigation: Automatic
Visualize

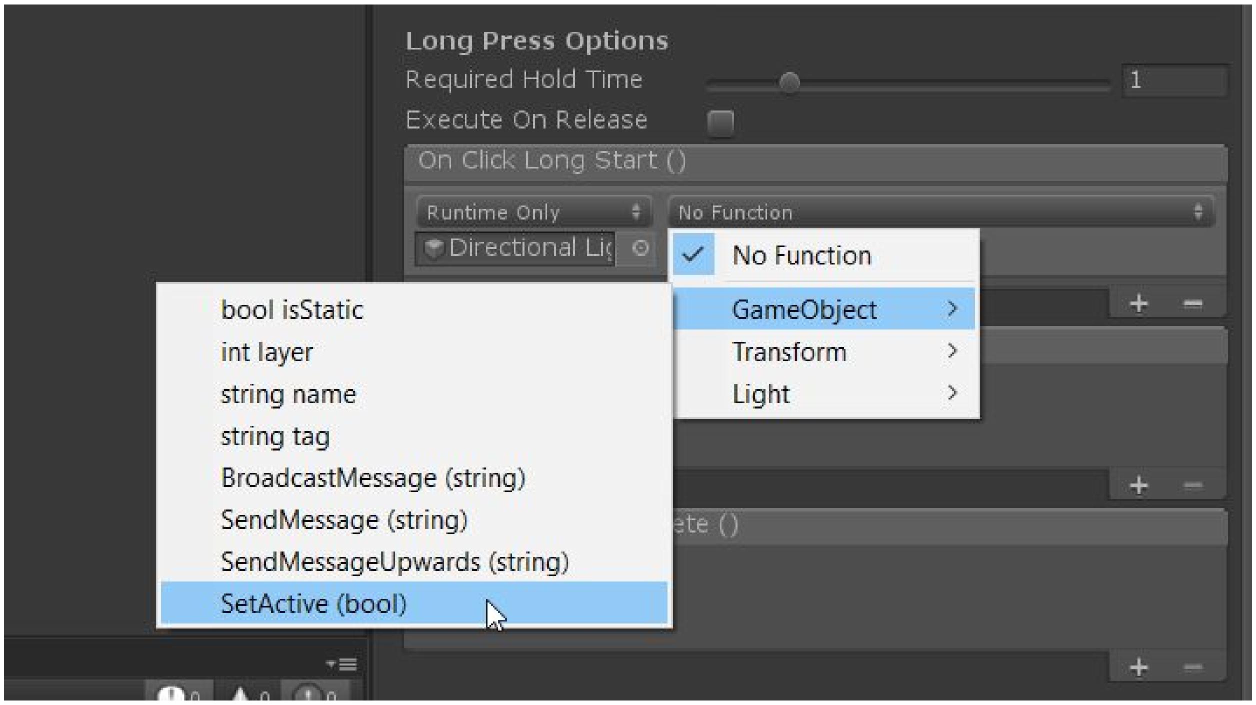
Long Press Options
Required Hold Time: 1
Execute On Release

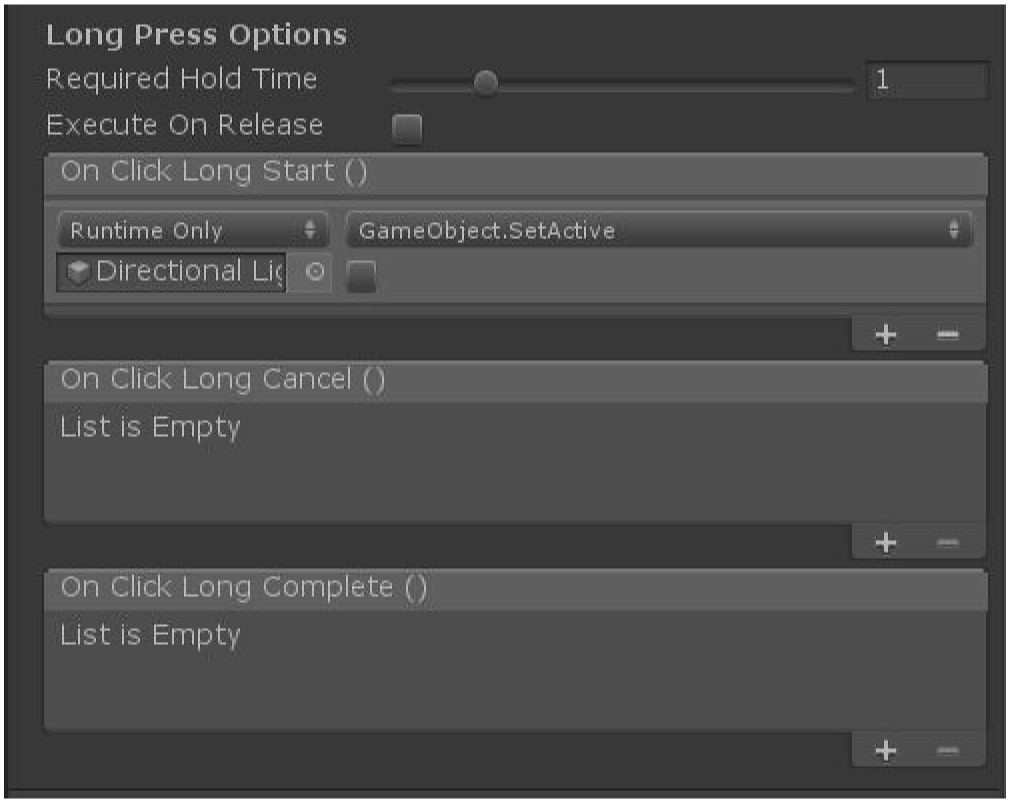
On Click Long Start ()
Runtime Only | No Function
Directional Lig

On Click Long Cancel ()
List is Empty

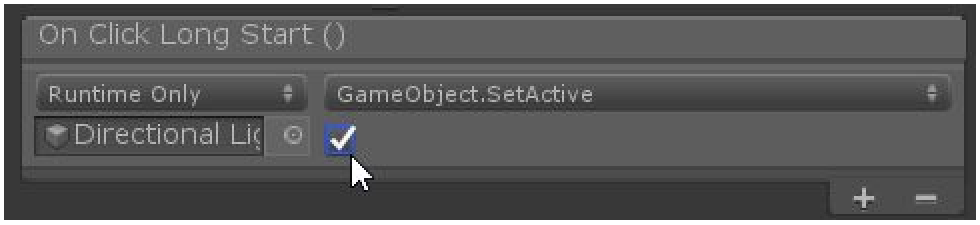
On Click Long Complete ()
List is Empty

4. Click **No Function** and select **GameObject > SetActive (bool)**.

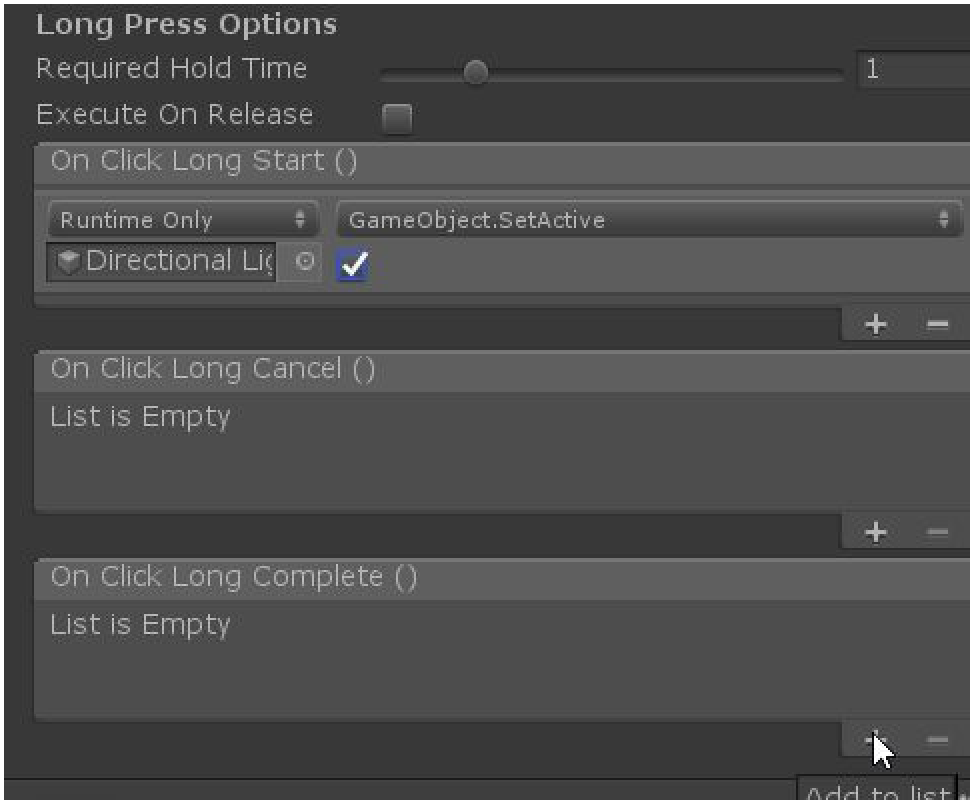


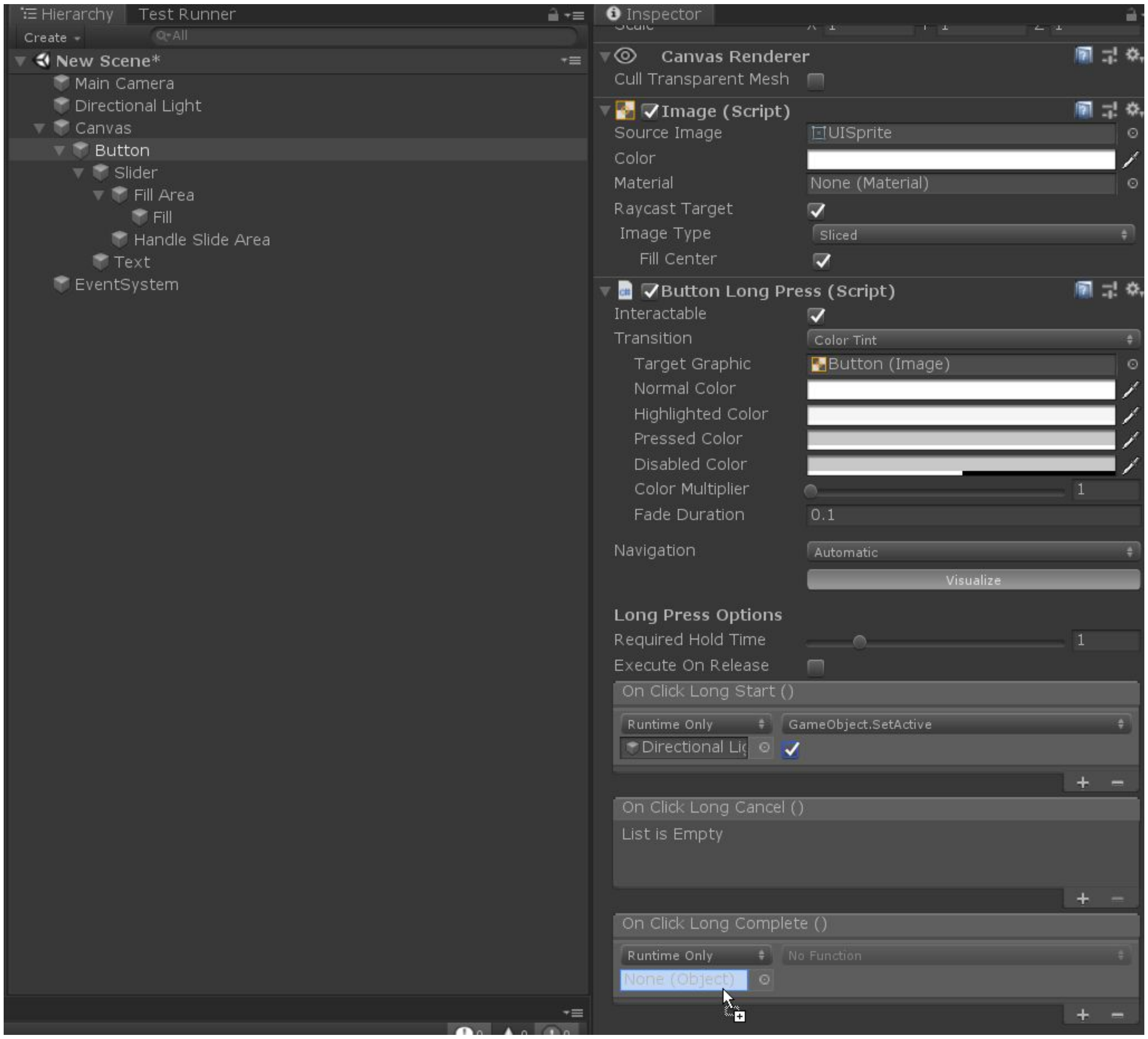


5. Check the checkbox below **GameObject.SetActive** to make the **Directional Light** active at the beginning of the long click.



6. Now repeat the same process with **On Click Long Complete** block, but leave the checkbox unchecked.


Long Press Options
Required Hold Time 1
Execute On Release ☐



New Scene*
- Main Camera
- Directional Light
- Canvas
  - Button
    - Slider
      - Fill Area
        - Fill
      - Handle Slide Area
    - Text
- EventSystem

Canvas Renderer

Cull Transparent Mesh ☐

Image (Script) ✓

Source Image	UISprite
Color	⬜
Material	None (Material)
Raycast Target	✓
Image Type	Sliced
Fill Center	✓

Button Long Press (Script) ✓

Interactable	✓
Transition	Color Tint
Target Graphic	Button (Image)
Normal Color	⬜
Highlighted Color	⬜
Pressed Color	⬜
Disabled Color	⬜
Color Multiplier	1
Fade Duration	0.1
Navigation	Automatic

Visualize

Long Press Options

Required Hold Time	1
Execute On Release	☐

On Click Long Start ()

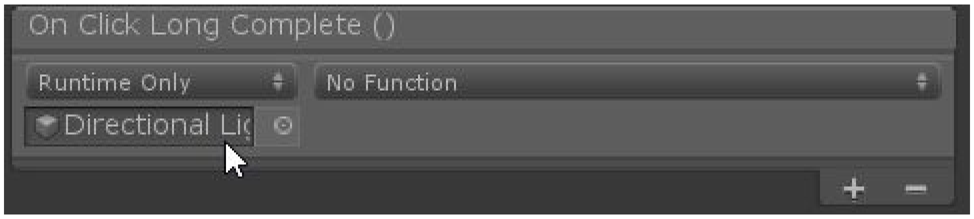
Runtime Only | GameObject.SetActive
Directional Lig | ✓

On Click Long Cancel ()

List is Empty

On Click Long Complete ()

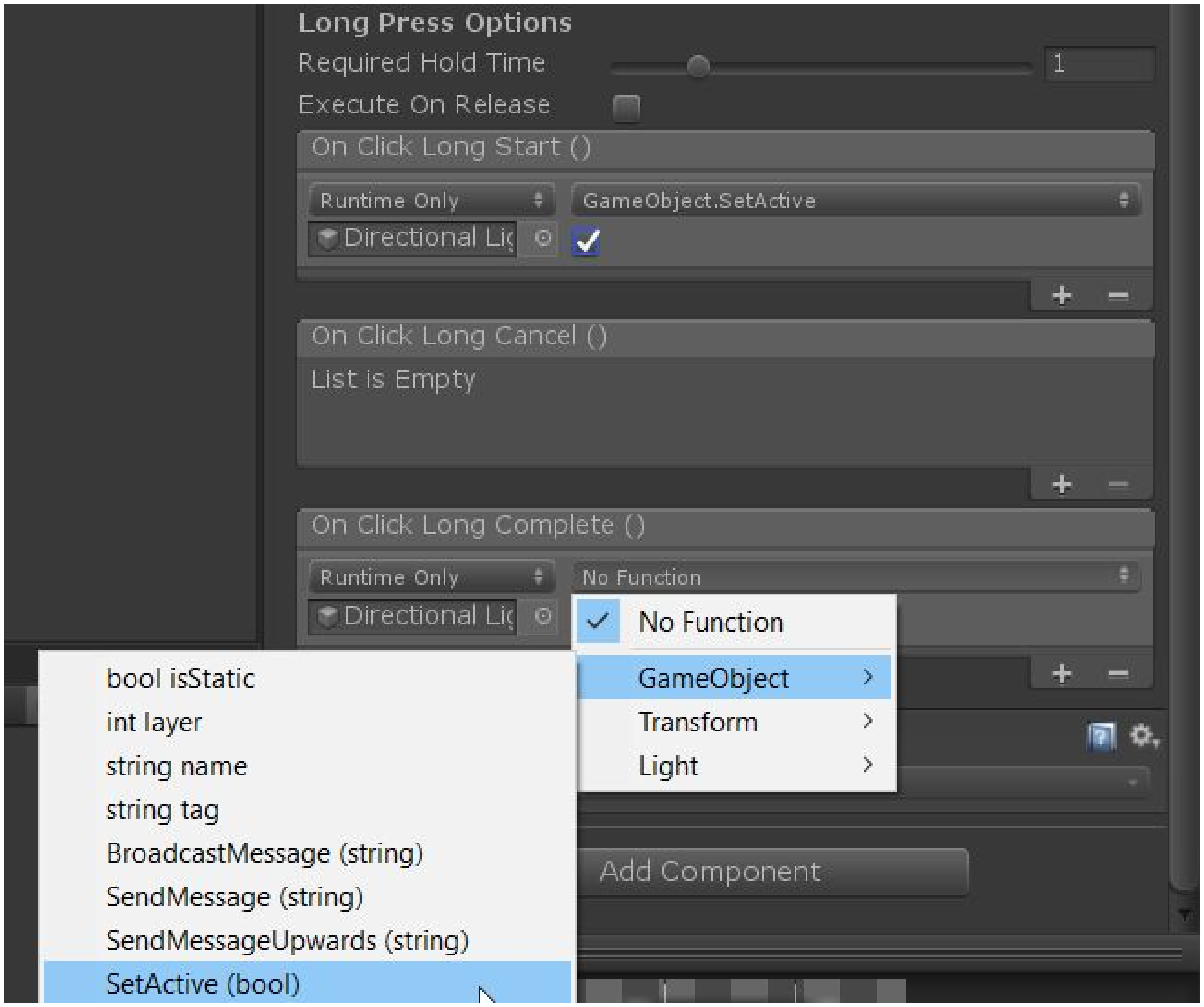
Runtime Only | No Function
None (Object)



On Click Long Complete ()

Runtime Only | No Function

Directional Lig



Long Press Options

Required Hold Time 1

Execute On Release

On Click Long Start ()

Runtime Only | GameObject.SetActive

Directional Lig ✓

On Click Long Cancel ()

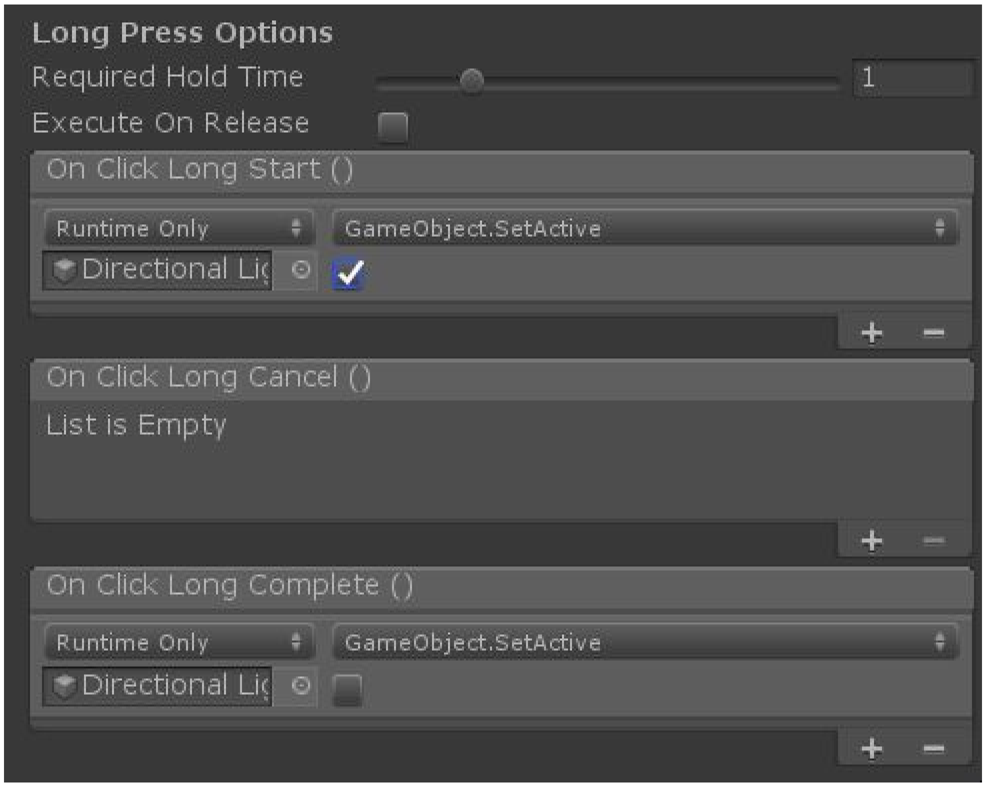
List is Empty

On Click Long Complete ()

Runtime Only | No Function

Directional Lig

7. End result should look like this.



Long Press Options

Required Hold Time 1

Execute On Release ☐

On Click Long Start ()

Runtime Only | GameObject.SetActive

Directional Lig ✓

On Click Long Cancel ()

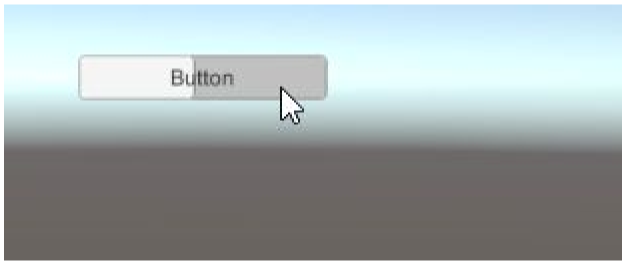
List is Empty

On Click Long Complete ()

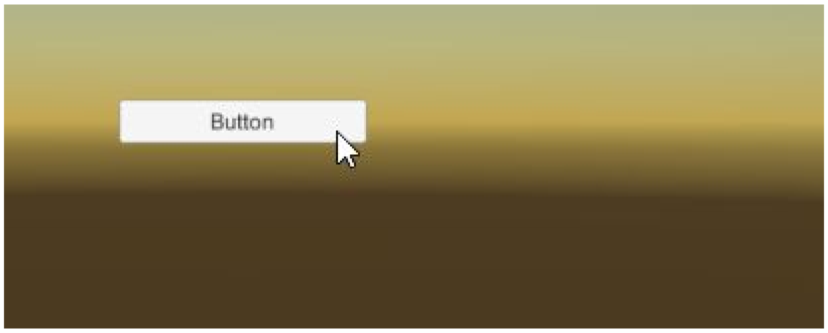
Runtime Only | GameObject.SetActive

Directional Lig ☐

8. Enter play mode again, and play around with your button! If you click shortly, light will be shining



But hold it long enough to fill the progress bar, and it will get dark:



9. That’s it! You can now build long press buttons that work with mouse, touch, keyboard navigation and gamepads!

Working with ButtonLongPress class in Scripts

You can access all the ButtonLongPress functionality directly from code. This is my preferred way! Just don't forget to add **using KodoLinija.UI** to your script first.

Example Script Using All ButtonLongPress Functions In Code

```
using UnityEngine;
using KodoLinija.UI;

namespace KodoLinija.UI.ButtonLongPressExamples
{
    public class ScriptUsageExample : MonoBehaviour
    {
        private ButtonLongPress Button;

        void Start()
        {
            // Find the button
            Button = GetComponent<ButtonLongPress>();

            // Require the button to be held for 300 ms. Default is 1s.
            Button.RequiredHoldTime = 0.3f;

            // Wait for player to release the button before calling the
            // completion callback. Default is false.
            Button.ExecuteOnRelease = false;

            // Keep repeating the action if button is still pressed.
            // Default is false.
            Button.Retrigger = true;

            // Listen to start of click event
            Button.onClickLongStart.AddListener(OnStartHold);

            // Listen to premature release event
            Button.onClickLongCancel.AddListener(OnCancelHold);

            // Listen to hold completion event. This is the main action
            // that should replace button's onClick event
            Button.onClickLongComplete.AddListener(OnCompleteHold);
        }

        void Update()
        {
            if (Button != null && Button.IsPushed)
            {
                Debug.LogFormat("Progress: {0:0} %, Hold time: {1:0.00}s",
                    Button.Progress * 100, Button.CurrentHoldTime);
            }
        }

        void OnStartHold()
        {
            Debug.Log("Started holding");
        }

        void OnCancelHold()
        {
            Debug.Log("Released too soon!");
        }

        void OnCompleteHold()
        {
            Debug.Log("Held long enough!");
        }
    }
}
```

Warning - don’t use the onClick callback

Since **ButtonLongPress** extends a **UI.Button** class, you will have an option to add listeners to the original **Button.onClick** event. This is not recommended, unless you’re absolutely sure what you’re doing. Note that onClick event would get executed even if the button was not pressed long enough to trigger the completion callback!

Issues, Bugs and Support

If you have found a problem with this asset, please send an email to help@kodolinija.com with as much details about your issue as possible, including:

- Operating system
- Target platform (Windows, Linux, Mac, Android, iOS, etc.)
- Target platform version
- Unity Version
- Input system type you are using (Legacy or new Input System)
- Expected result
- Actual result

Please include your **Unity Asset Store invoice number** in your email.