

SnapScrollerPlugin

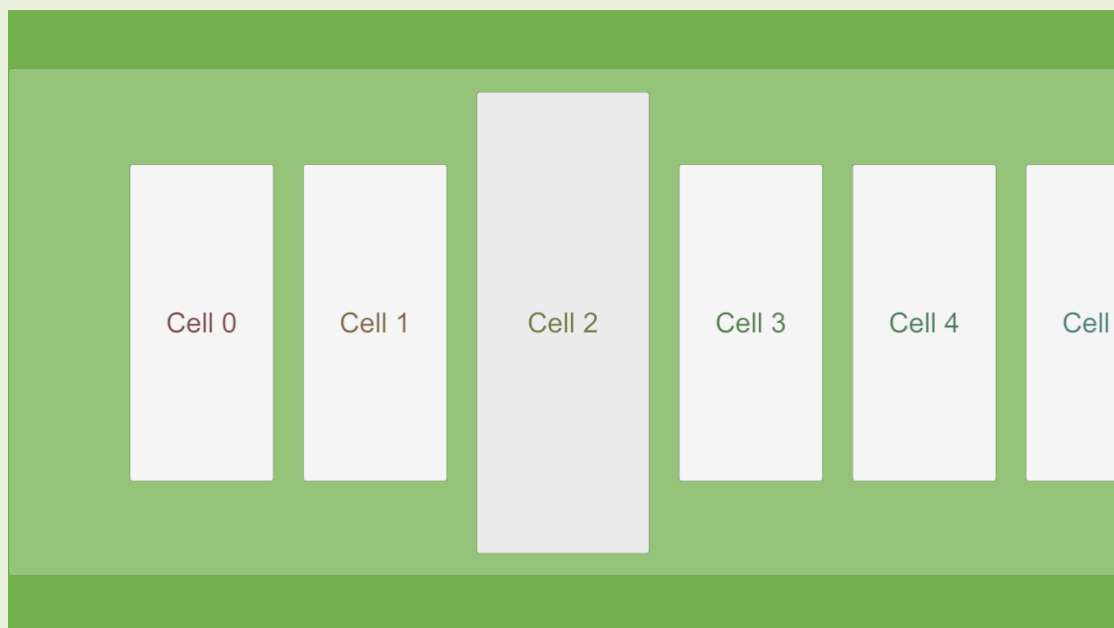
User Manual

魚丸®蕾娜 RenaWevin

SnapScrollerPlugin is a plugin that allows you to automatically position items on ScrollRect in Unity, and the positioned object will have a zoom effect.

This plugin has the following features:

- ◆ Automatic positioning for items in the center of ScrollRect
- ◆ Automatic scaling of items
- ◆ Use object pool
- ◆ Support for infinite loop scrolling (optional)



Contents

In the PDF file, you can directly click on the following title to navigate to the paragraph.

Get Started	3
Install SnapScrollerPlugin	3
Most Lightweight Usage.....	4
Customized Usage.....	11
Parameter Explanation	18
SnapScroller script.....	18
SnapScrollerCell script	20

Get Started

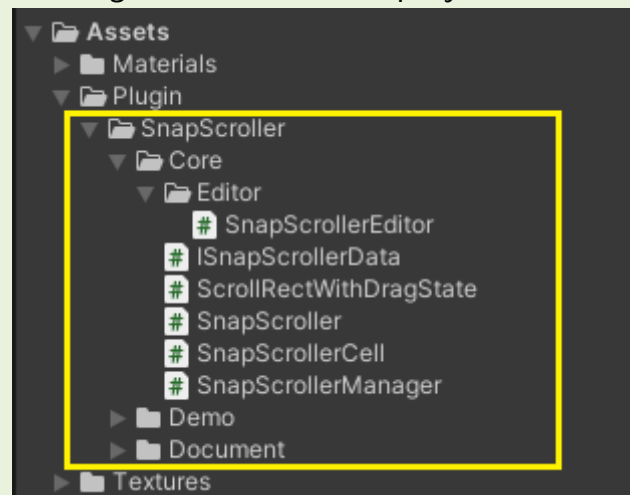
Install SnapScrollerPlugin

You can install this plugin from the Unity Asset Store.

After installation, there should be files as shown in the following figure.

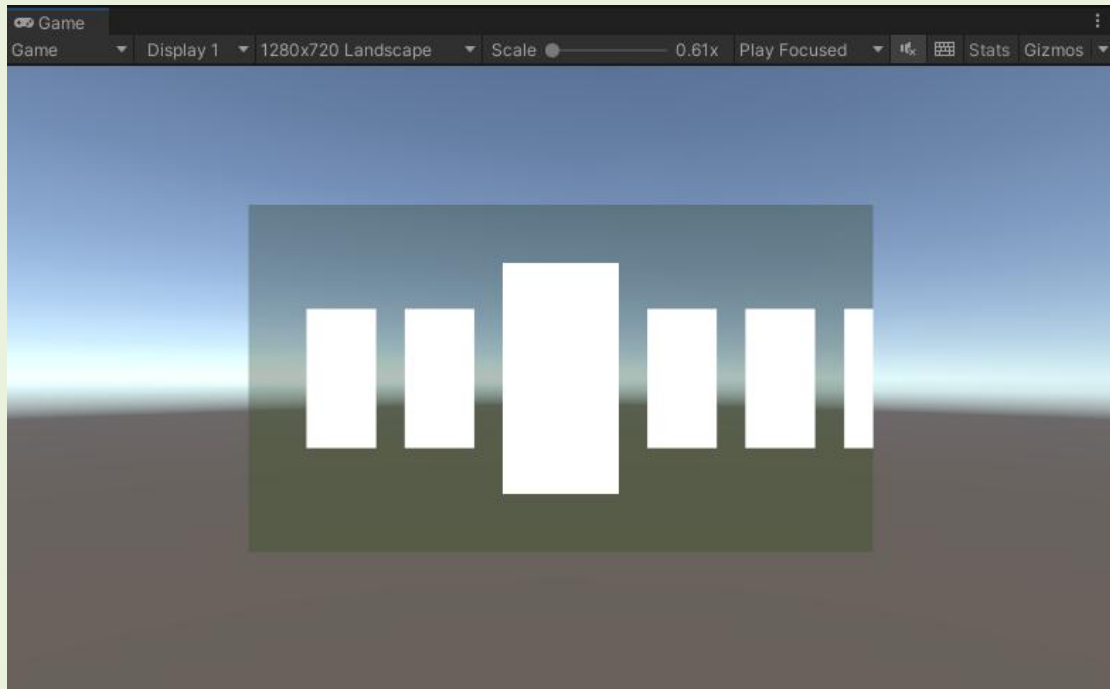
Please make sure that all the following files are present in the “Core” folder under the “SnapScroller” directory.

The “Demo” and “Document” folders can be deleted when they are no longer needed for the project.



Most Lightweight Usage

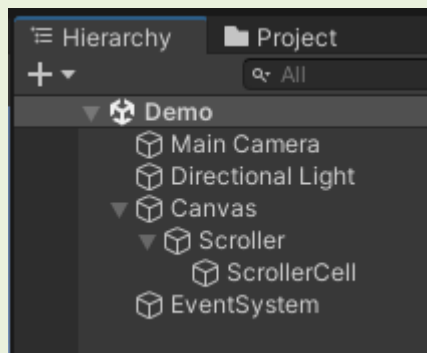
The following will show the lightest usage of this plugin, which will create a scroller with 10 blank objects as shown in the following image.



1. Create scene objects

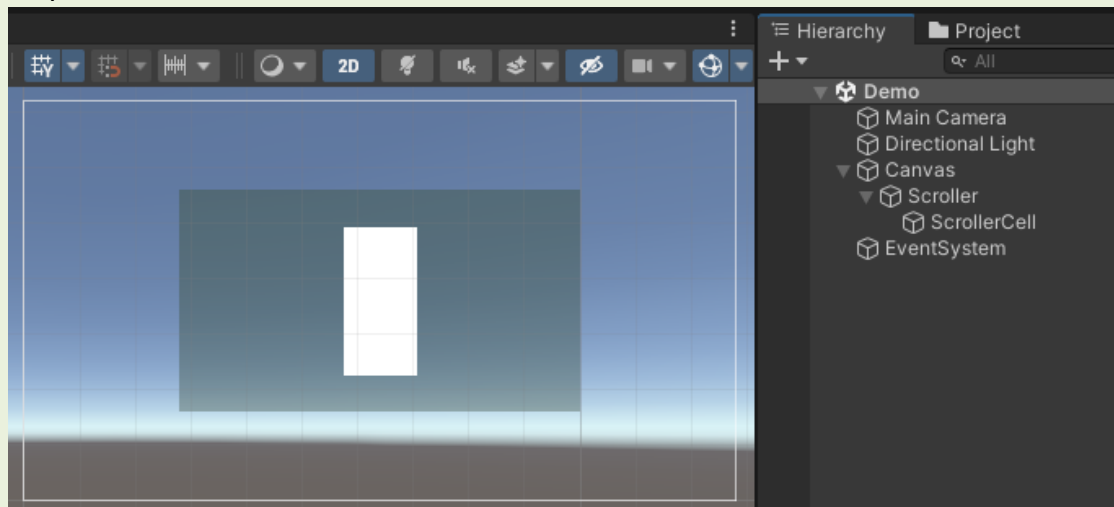
This step creates a new scene, adds the necessary UI Canvas and EventSystem, and then adds Scroller and ScrollerCell empty GameObjects under the Canvas object.

The final state is as shown in the following image.



2. Add components to UI

This step will add components to GameObjects we created in previous step.

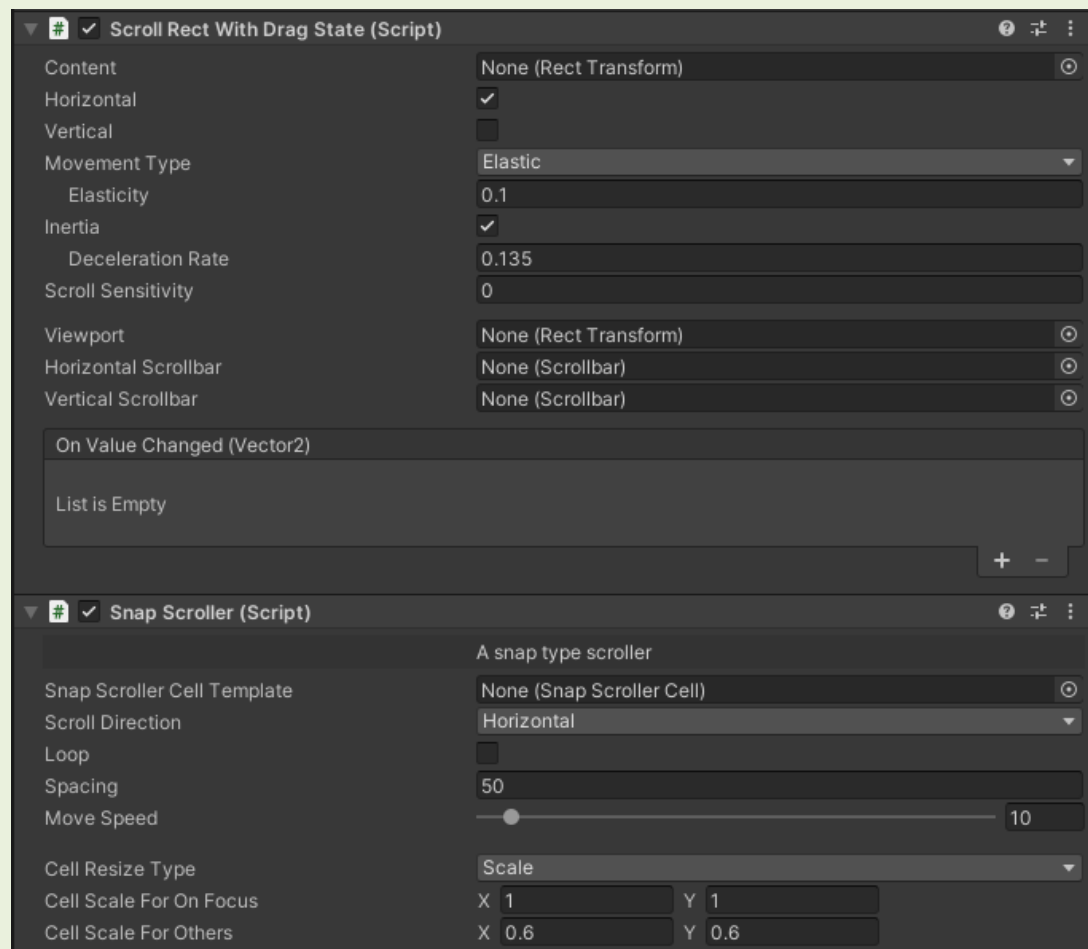


Scale the Scroller object (sliding area) and ScrollerCell object (single item) to the size you want.

(The canvas size in this demo is 1920×1080, the Scroller size is 1080×600, and the Cell size is 200×400.)

Add the "SnapScroller" script to the Scroller object,

"ScrollRectWithDragState" will be added automatically at the same time, and some parameters can be adjusted at this time.



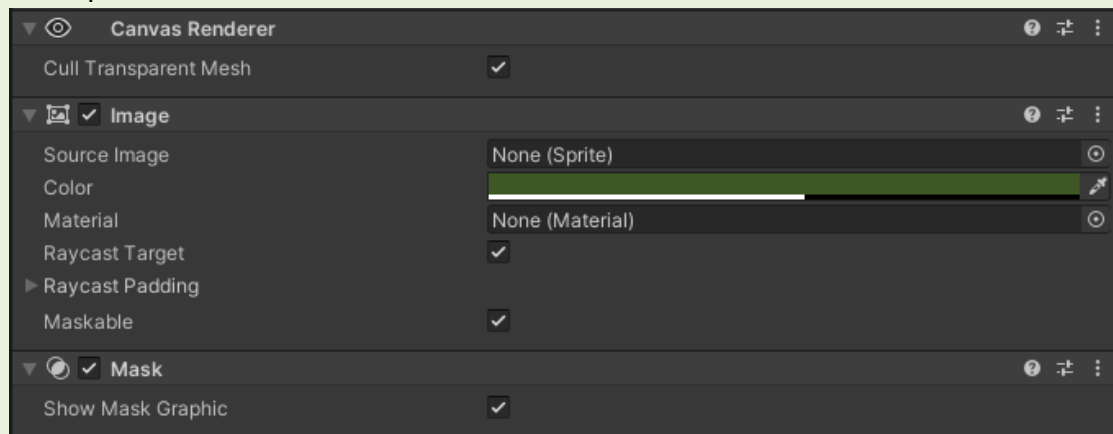
According to the sliding direction:

For horizontal scrolling, please set the ScrollerDirection of SnapScroller to Horizontal, then check the Horizontal option and uncheck the Vertical option of ScrollRectWithDragState.

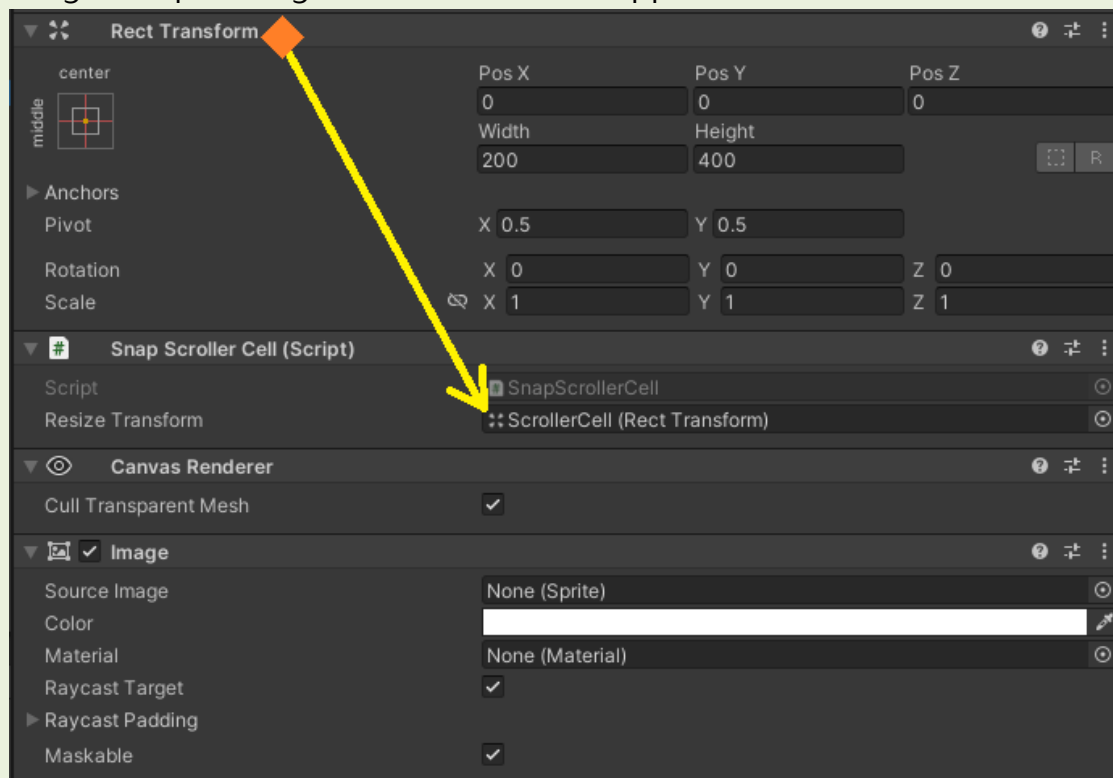
For vertical scrolling, please set the ScrollerDirection of SnapScroller to Vertical, then check the Vertical option and uncheck the Horizontal option of ScrollRectWithDragState.

Then add Unity's built-in Image component and Mask component to the Scroller object to receive the touch, and hide Cell objects which outside the range. At the same time, please ensure that the Raycast Target of the Image must be checked to receive the touch sliding event.

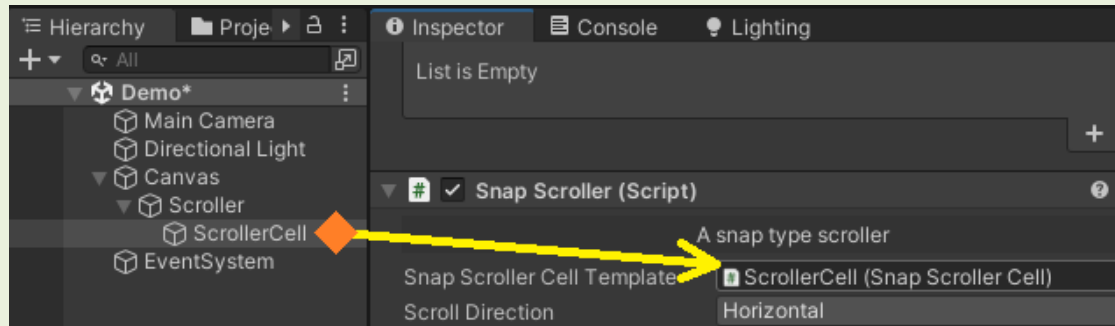
The Image here can also give the Scroller background color or image, examples are as follows.



Add the "SnapScrollerCell" script and Unity's built-in Image component to the ScrollerCell object, then drag the RectTransform component of this object to the SnapScrollerCell script, so that it can adjust its size; and the Image component gives the Cell a white appearance.



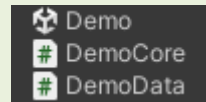
Finally, go back to the Scroller object, drag the ScrollerCell object to the Snap Scroller Cell Template field of the SnapScroller script, to allow the Scroller use this Cell as a Prefab to display the contents.



3. Create necessary scripts

This step will create the following scripts: "DemoData" and "DemoCore" .

The following are the contents of the two scripts:



DemoData - used to give an empty data container

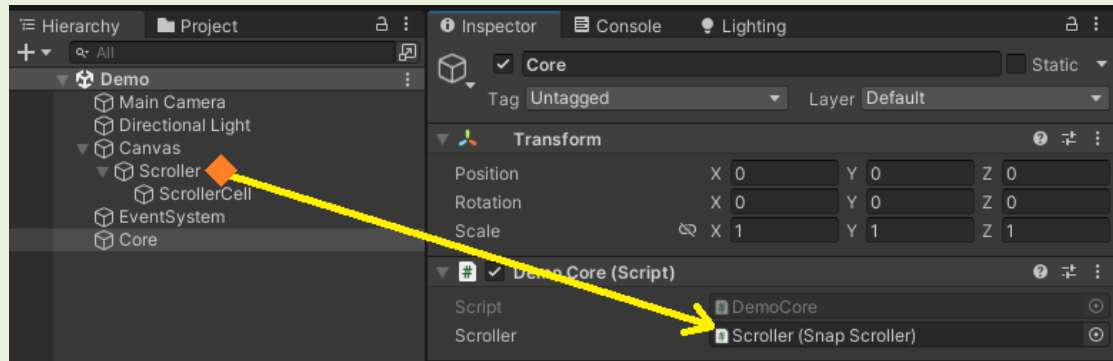
```
1 using RW.UI.SnapScrollerPlugin;
2
3 public class DemoData : ISnapScrollerData {
4
5 }
```


DemoCore - used to initialize the Scroller and determine the number of cell objects

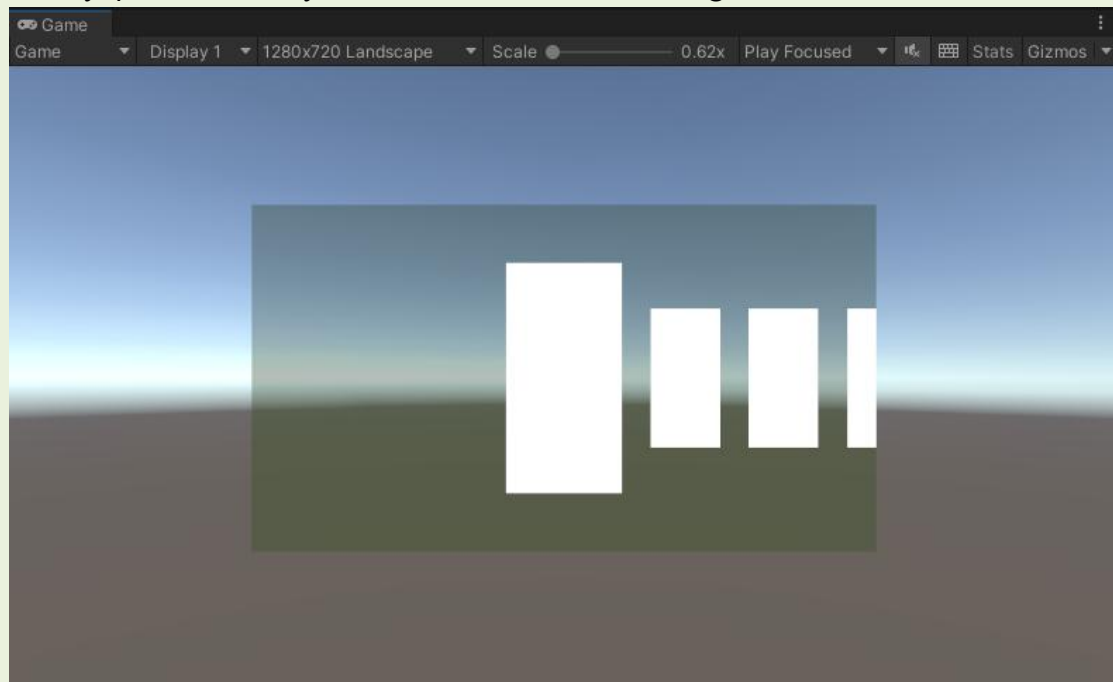
```
1 using RW.UI.SnapScrollerPlugin;
2 using UnityEngine;
3
4 public class DemoCore : MonoBehaviour {
5
6     private SnapScrollerManager manager;
7
8     [SerializeField]
9     private SnapScroller scroller;
10
11     void Start() {
12         manager = new SnapScrollerManager();
13         scroller.SetManager(manager);
14         manager.ClearData();
15         for (int i = 0; i < 10; i++) {
16             manager.AddData(new DemoData());
17         }
18         scroller.RefreshData();
19     }
20 }
```

4. Add scripts to objects

Add an empty game object named “Core” at the top level of the scene, attach the DemoCore script to it, and drag the Scroller object into the Scroller field of the DemoCore script.



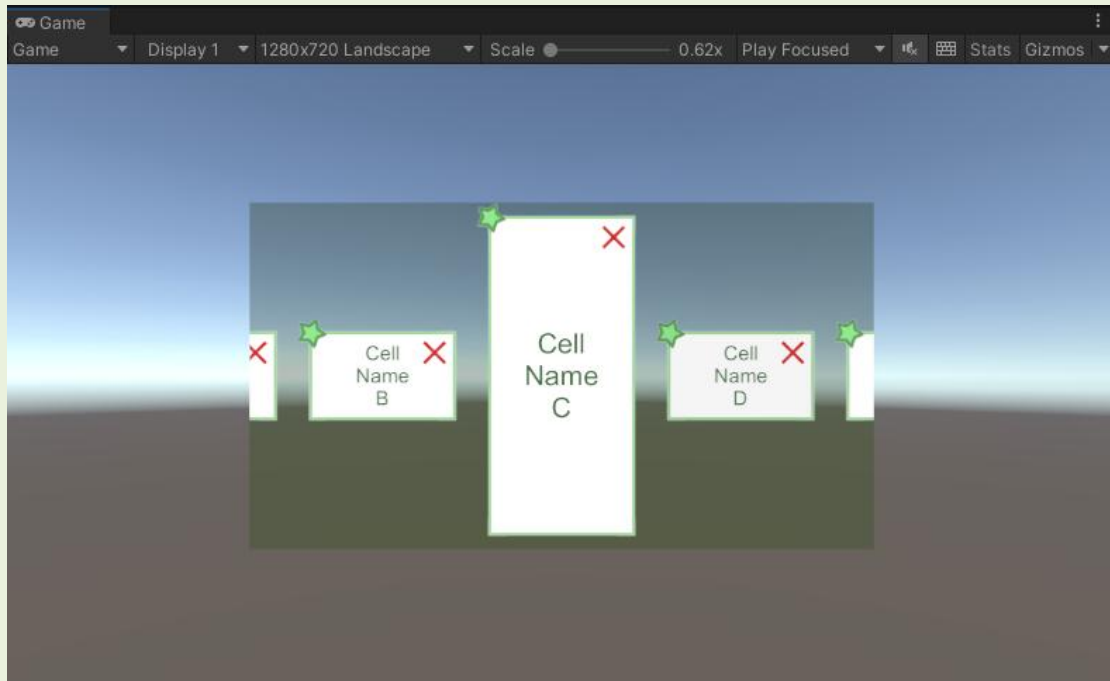
Finally, press the Play button  to start the game, and this is all done.



You can scroll left and right, and there will be 10 white Cell boxes in total.

Customized Usage

The following will show the customized usage of this plugin, which will create a scroller with 10 objects, each item has a name from A to J, can be selected by clicking on it, and every item could be deleted.

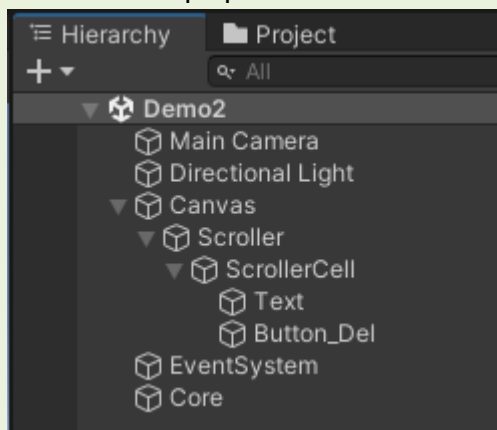


1. Create scene objects and basic UI

This step will use the same method as the previous example.

Create a new scene, add the necessary Canvas and EventSystem for the UI, and then add the following objects:

- ♦ Scroller: The main part of the Scroller.
- ♦ ScrollerCell: The Cell object.
- ♦ Text: The text displayed in the Cell.
- ♦ Button_Del: The button used to delete the item.
- ♦ Core: Script placement area to control the initial state of data.



2. Create scripts

In this step, we will add the following three scripts: "DemoData2" ,
"DemoCore2" , and "DemoCell2" .

The following are the contents of these scripts:



DemoData2 - used to provide data container

```
1 using RW.UI.SnapScrollerPlugin;
2
3 public class DemoData2 : ISnapScrollerData {
4     public string text;
5 }
```

DemoCore2 - used to initialize the Scroller and determine the content and quantity of cell data

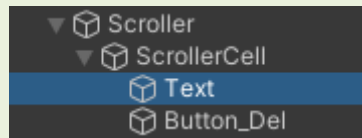
```
1 using RW.UI.SnapScrollerPlugin;
2 using UnityEngine;
3
4 public class DemoCore2 : MonoBehaviour {
5
6     private string[] names = {
7         "A", "B", "C", "D", "E", "F", "G", "H", "I", "J"
8     };
9
10    private SnapScrollerManager manager;
11
12    [SerializeField]
13    private SnapScroller scroller;
14
15    void Start() {
16        manager = new SnapScrollerManager();
17        scroller.SetManager(manager);
18        manager.ClearData();
19        for (int i = 0; i < 10; i++) {
20            manager.AddData(new DemoData2() {
21                text = $"Cell\nName\n{names[i]}"
22            });
23        }
24        scroller.RefreshData();
25    }
26 }
```

DemoCell2 - used to customize the UI displayed on each cell object

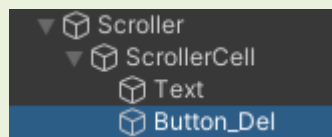
```
1  using RW.UI.SnapScrollerPlugin;
2  using UnityEngine;
3  using UnityEngine.UI;
4
5  public class DemoCell2 : SnapScrollerCell {
6
7      [SerializeField]
8      private Text text;
9      [SerializeField]
10     private Button btn_Select;
11     [SerializeField]
12     private Button btn_Del;
13
14     void Start() {
15         btn_Select.onClick.AddListener(OnClick_Btn_Select);
16         btn_Del.onClick.AddListener(OnClick_Btn_Del);
17     }
18
19     public override void OnSetData() {
20         DemoData2 data = GetData as DemoData2;
21         if (data == null) { return; }
22         text.text = data.text;
23     }
24
25     private void OnClick_Btn_Select() {
26         scroller.ScrollToIndex(cellIndex);
27     }
28
29     private void OnClick_Btn_Del() {
30         manager.RemoveDataAt(cellIndex);
31         scroller.RefreshData();
32     }
33
34 }
```

3. Add components and scripts to UI

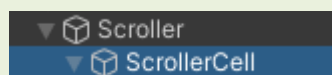
In this step, we will add UI components and scripts to GameObjects, and will skip all style adjustment steps (color, sliding direction, text content, etc.).



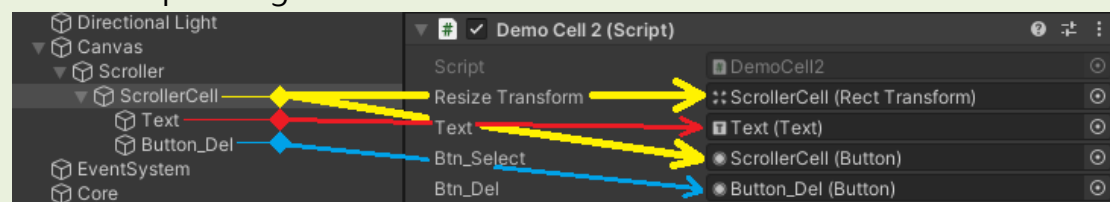
In the Text object under the Cell, add Unity's built-in Text component, and adjust the display settings of this component in the Cell (size, text color, font size, etc.).



Next, add Unity's built-in Button component into Button_Del object, and add some appearance for it (such as Image or Text). The click event of button has been determined earlier in the script.



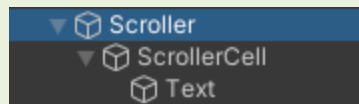
Next, set up the ScrollerCell object. First, add the DemoCell2 script to it, and also add the built-in Unity Image and Button components to display the frame and act as the selection button. Drag the following objects to the corresponding fields in the ScrollerCell:



ScrollerCell object → Resize Transform, and Btn_Select field

Text object → Text field

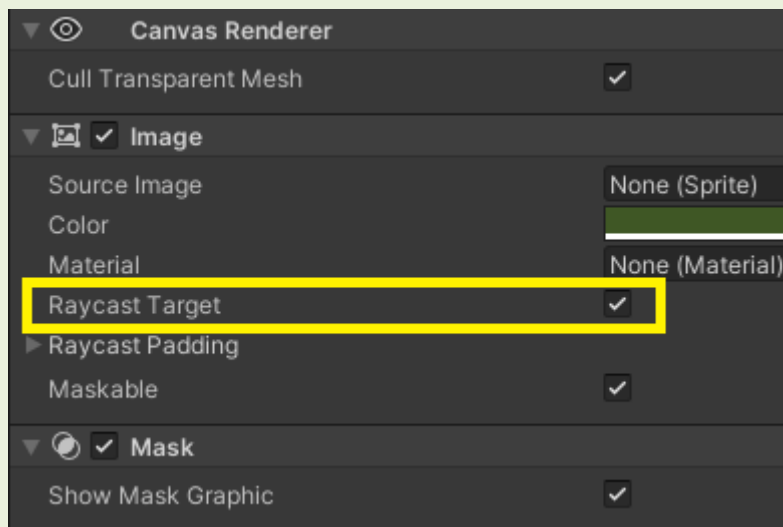
Button_Del object → Btn_Del field



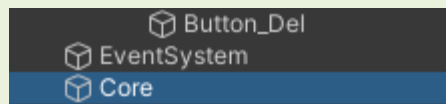
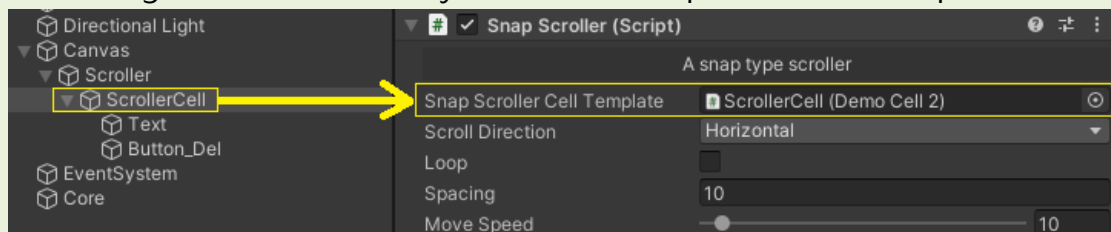
In the Scroller object, add the SnapScroller script, then ScrollRectWithDragState will be added automatically.

And add Unity's built-in Image component and Mask component to this object, to receive touch event and hide Cell objects which outside the range.

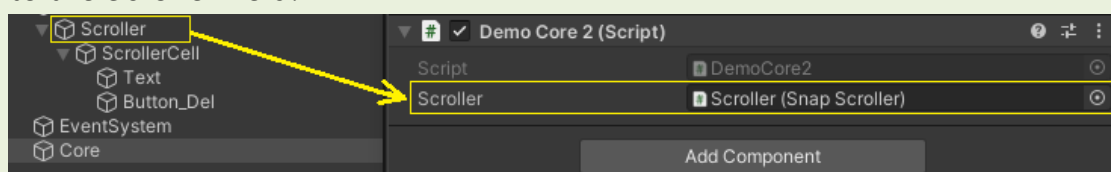
At the same time, make sure the Raycast Target of Image component must be checked to receive the touch event.




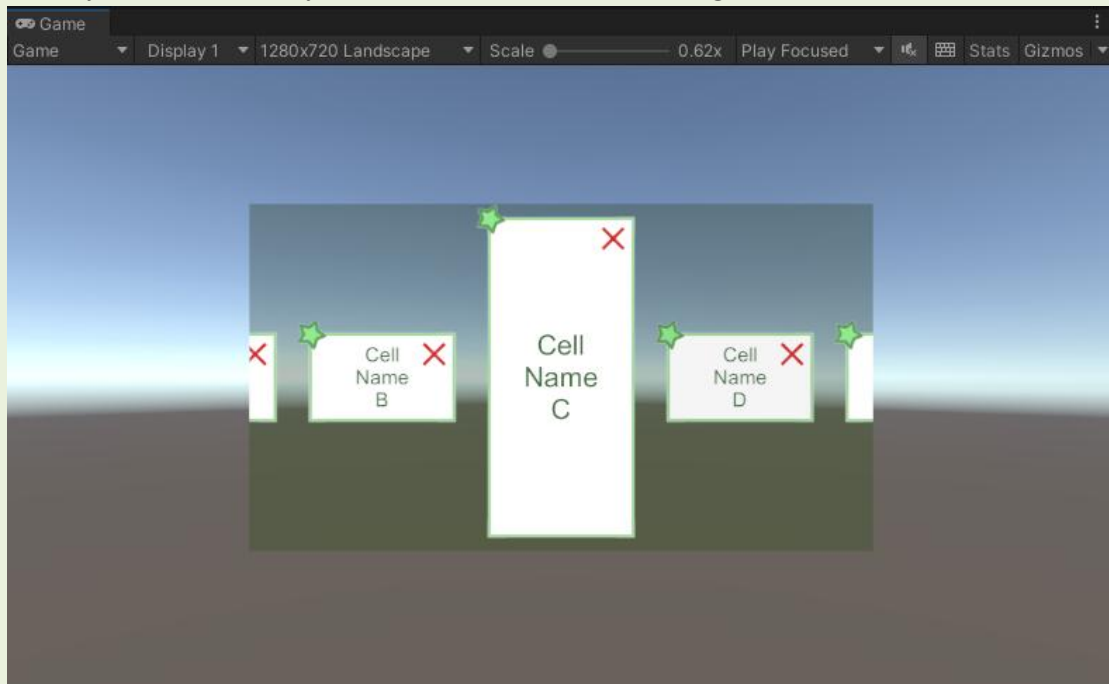
Then drag the ScrollerCell object into the SnapScrollerCellTemplate field.



Add the DemoCore2 script in the Core object, and drag the Scroller object to the Scroller field.



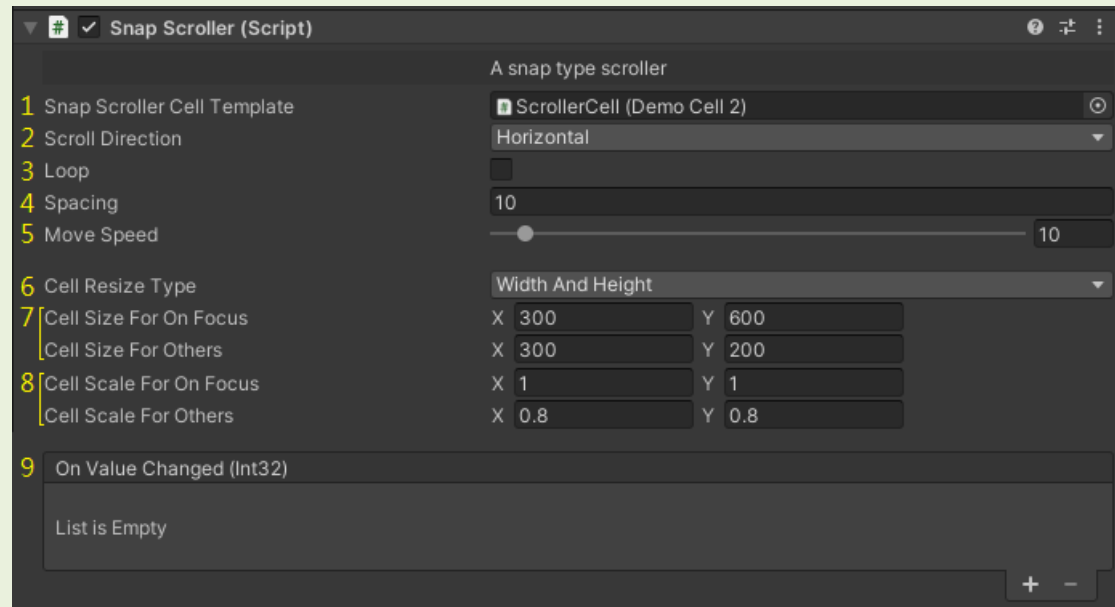
Finally, press the Play button  to start the game, and this is all done.



Each Cell in the Demo will have a name from A to J. Click Cell objects to quickly scroll to the Cell, and click the red cross in the upper right corner to delete the data.

Parameter Explanation

SnapScroller script



1. Snap Scroller Cell Template

Required, it is used to specify the template of the Cell.

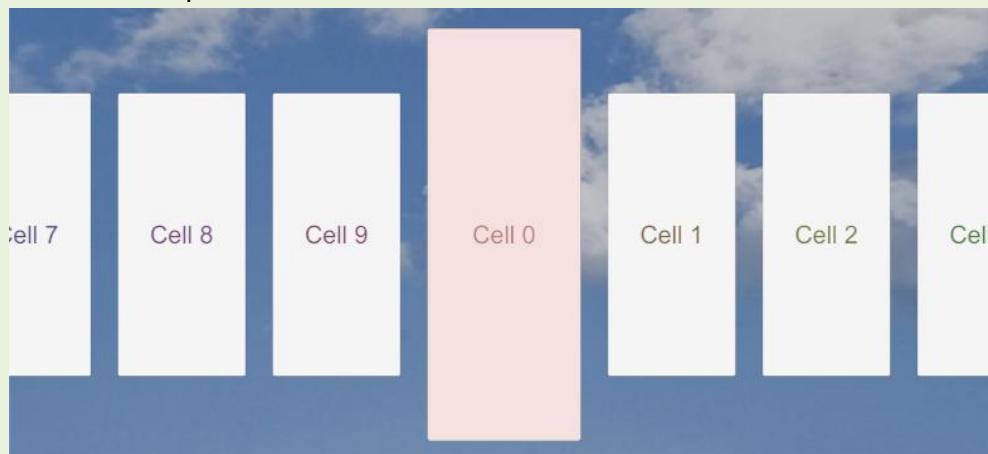
2. Scroll Direction

The scroll direction of scroller, requires the cooperation of Scroll Rect.

Horizontal: Left and Right. Vertical: Up and Down.

3. Loop

When it's enabled, the scroller will have a head-to-tail loop, the last item will be positioned next to the first item.



4. **Spacing**

The interval between each cell in pixels.

5. **Move Speed**

The movement speed of the scroller, used when user releases the mouse/finger or when calling the `scroller.ScrollTo` methods. The Scroller will move the objects at the speed set by this.

6. **Cell Resize Type**

The method used to resize Cells, used when the Scroller's position is moved.

None: No scaling at all.

Scale: Will use the `localScale` of Cell object to scale.

Width And Height: Will change the width and height of Cell object.

7. **Cell Size For On Focus / Others**

This parameter will only appear when Cell Resize Type is selected as Width And Height.

It will determine the width and height of the Cell object when it is Focused (at the center of scroller) or not.

X parameter represents width, Y parameter represents height.

8. **Cell Scale For On Focus / Others**

This parameter will only appear when Cell Resize Type is selected as Scale.

It will determine the `localScale` of the Cell object when it is Focused (at the center of scroller) or not.

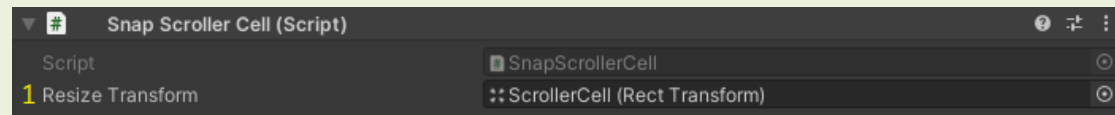
9. **On Value Changed (Int32)**

This is an event that will be called once when the focused cell index changes. The method takes an integer parameter which is the index of the currently focused cell after the change.

SnapScrollerCell script

Only the default parameters of the object are listed here.

If additional parameters are needed, please override this class and add fields.



1. **Resize Transform**

Used to specify the Rect Transform that will be modified when the Cell object is scaled by the Scroller.