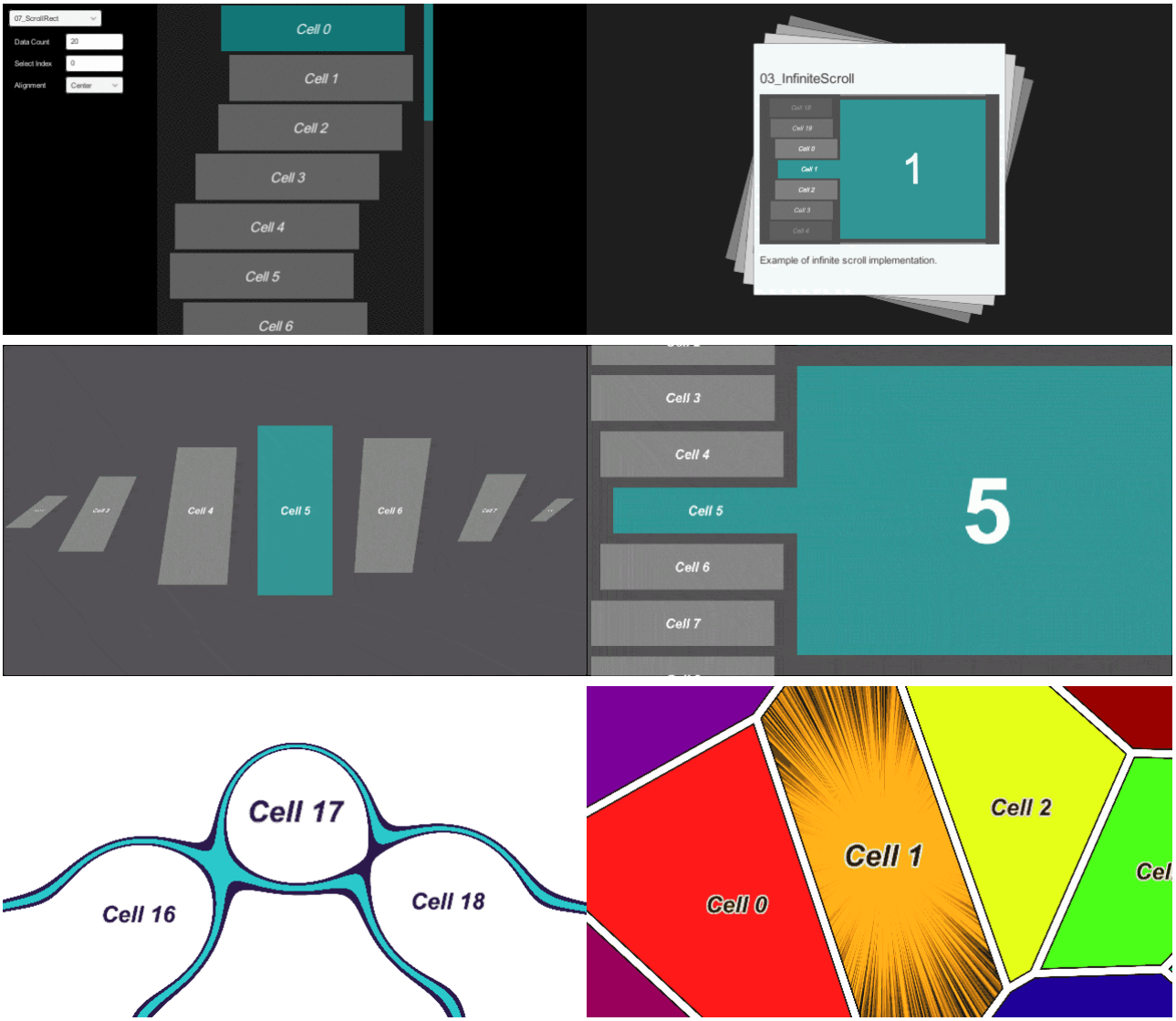
FancyScrollView

A generic ScrollView component that can implement highly flexible animations. Infinite scrolling is also supported.



Requirements

Installation

GitHub Releases

Download FancyScrollView.unitypackage from Releases and import it into Unity Project.

Unity Asset Store

Import packages from the Asset Store into Unity Project.

Unity Package Manager *(Example scenes not included)*

Add a reference to the repository in the Packages / manifest.json file in the project directory .

```
{
  "dependencies": {
    "jp.setchi.fancyscrollview":
"https://github.com/setchi/FancyScrollView.git#upm"
  }
}
```

Features

You can freely implement scroll animation

When FancyScrollView updates the scroll position, it passes each cell the normalized position of the viewport range. On the cell side, the scroll position and appearance are controlled by the cell itself based on a value between 0.0 and 1.0 . The sample implements scrolling movement using Animator and mathematical expressions.

It works lightly even if the number of data is large

Only the number of cells required for display is generated and the cells are reused. With Demo , you can check the operation while actually increasing the number of data. In FancyScrollRect and FancyGridView , you can also specify the margin before cells are reused during scrolling .

Messages can be exchanged freely between cells and scroll view

Through Context, it is possible to simply implement the process of detecting that a cell has been clicked in the scroll view or issuing an instruction to the cell from the scroll view. An implementation example (Examples / 02_FocusOn) is included, so please refer to it.

Scroll and jump to a specific cell

You can also specify the number of seconds to move and Easing. For more information API Documentation of Class Scroller Please refer to.

You can fine-tune the scrolling behavior

You can set scroll behavior such as inertia and deceleration rate. For more information API Documentation of Class Scroller Please refer to.

Supports snapping

When snapping is enabled, it moves to the nearest cell just before scrolling stops. You can specify the speed threshold at which snapping begins, the number of seconds to move, and Easing. FancyScrollRect and FancyGridView do not support snapping.

Supports infinite scroll

Infinite scrolling can be implemented by setting the following in Inspector

Infinite scrolling can be implemented by setting the following in Inspector.

1. When Loop of FancyScrollView is turned on, the cells circulate so that the last cell is arranged before the first cell, and the first cell is arranged after the last cell.
2. When using the Scroller used in the sample, set the Movement Type to Unrestricted to make the scroll range unlimited. Infinite scrolling can be achieved by combining with 1.

An implementation example (Examples / 03_InfiniteScroll) is included. FancyScrollRect and FancyGridView do not support infinite scrolling.

Examples

See FancyScrollView / Examples .

Name	Description
01_Basic	This is an example of the simplest configuration.
02_FocusOn	This is an example of focusing on the left and right cells with a button.
03_InfiniteScroll	An implementation example of infinite scroll.
04_Metaball	This is an example of metaball implementation using a shader.
05_Voronoi	This is an example of Voronoi implementation using a shader.
06_LoopTabBar	This is an example of switching screens with tabs.
07_ScrollRect	An example implementation of the ScrollRect style with a scroll bar.
08_GridView	This is an example of grid layout implementation.
09_LoadTexture	An implementation example that loads and displays a texture.

Usage

In the simplest configuration,

- Object for passing data to the cell
- cell
- Scroll view

Implementation is required.

Implementation

Defines an object for passing data to cells.

```
public class ItemData
{
    public string Message { get; }
```

```
    public ItemData(string message)
    {
        Message = message;
    }
}
```

FancyScrollViewCell Inherit and implement your own cell.

```
using UnityEngine;
using UnityEngine.UI;
using FancyScrollView;

public class MyCell : FancyScrollViewCell<ItemData>
{
    [SerializeField] Text message = default;

    public override void UpdateContent(ItemData itemData)
    {
        message.text = itemData.Message;
    }

    public override void UpdatePosition(float position)
    {
        // position は 0.0 ~ 1.0 の値です
        // position に基づいてスクロールの外観を自由に制御できます
    }
}
```

FancyScrollView Inherit and implement your own scroll view.

```
using UnityEngine;
using System.Linq;
using FancyScrollView;

public class MyScrollView : FancyScrollView<ItemData>
{
    [SerializeField] Scroller scroller = default;
    [SerializeField] GameObject cellPrefab = default;

    protected override GameObject CellPrefab => cellPrefab;

    void Start()
    {
        scroller.OnValueChanged(base.UpdatePosition);
    }

    public void UpdateData(IList<ItemData> items)
    {
        base.UpdateContents(items);
        scroller.SetTotalCount(items.Count);
```

```
    }
}
```

Fills the scroll view with data.

```
using UnityEngine;
using System.Linq;

public class EntryPoint : MonoBehaviour
{
    [SerializeField] MyScrollView myScrollView = default;

    void Start()
    {
        var items = Enumerable.Range(0, 20)
            .Select(i => new ItemData($"Cell {i}"))
            .ToArray();

        myScrollView.UpdateData(items);
    }
}
```

See Examples and API Documentation for other details .

Author

setchi

License

MIT