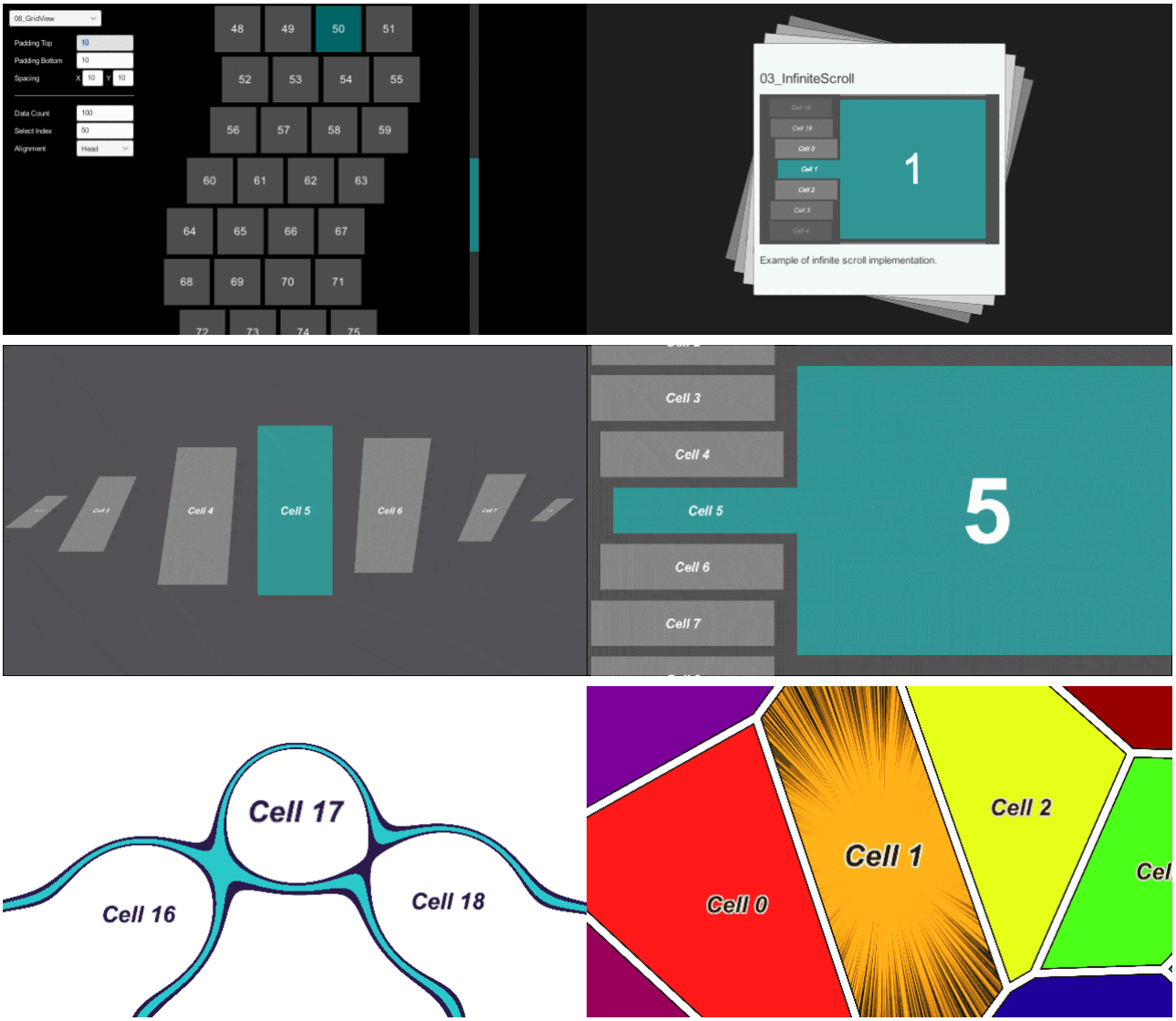
FancyScrollView

A generic ScrollView component that can implement highly flexible animations. It also supports infinite scrolling.



Requirements

Installation

OpenUPM

Add packages from the OpenUPM registry to Unity Project.

```
openupm add jp.setchi.fancyscrollview
```

Unity Package Manager

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

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

Unity Asset Store

Import the package from the Asset Store into Unity Project.

Features

You can implement scroll animation freely

When FancyScrollView updates the scroll position, it passes each cell the normalized position of the viewport range. On the cell side, the position and appearance during scrolling are controlled by the cell itself based on the value from 0.0 to 1.0 . The sample uses Animator and formulas to implement the scrolling movement.

Works lightly even with large numbers of data

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

You can freely exchange messages between cells and scroll view

Via Context, it is possible to simply implement the process of detecting that a cell is clicked in the scroll view and issuing instructions to the cell from the scroll view. Please refer to the implementation example (Examples / 02_FocusOn).

Scroll and jump to specific cells

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

Scroll behavior can be set in detail

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

Supports snap

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

Supports infinite scroll

You can implement infinite scrolling by making the following settings in Inspector.

1. If you turn on Loop in FancyScrollView, cells will cycle so that the last cell is before the first cell, and the first cell is 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 realized by combining with 1.

An implementation example (Example / 03_InfiniteScroll) is included, so please refer to this as well.
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 implementation example that focuses on the left and right cells with buttons.
03_InfiniteScroll	This is an implementation example of infinite scroll.
04_Metaball	This is an example of mounting a metaball using a shader.
05_Voronoi	This is a Voronoi implementation example using a shader.
06_LoopTabBar	This is an example of switching screens with tabs.
07_ScrollRect	Example implementation of ScrollRect style with scrollbar.
08_GridView	This is an example of grid layout implementation.
09_LoadTexture	This is an example of loading and displaying textures.

Usage

In the simplest configuration,

- An object for passing data to the cell
- cell
- Scroll view

Requires implementation.

Implementation

Defines an object for passing data to a cell.

```
class ItemData
{
```

```
{
    public string Message { get; }

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

FancyCell And implement your own cell.

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

class MyCell : FancyCell<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 And implement your own scroll view.

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

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;

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 more details.

Author

setchi

License

MIT