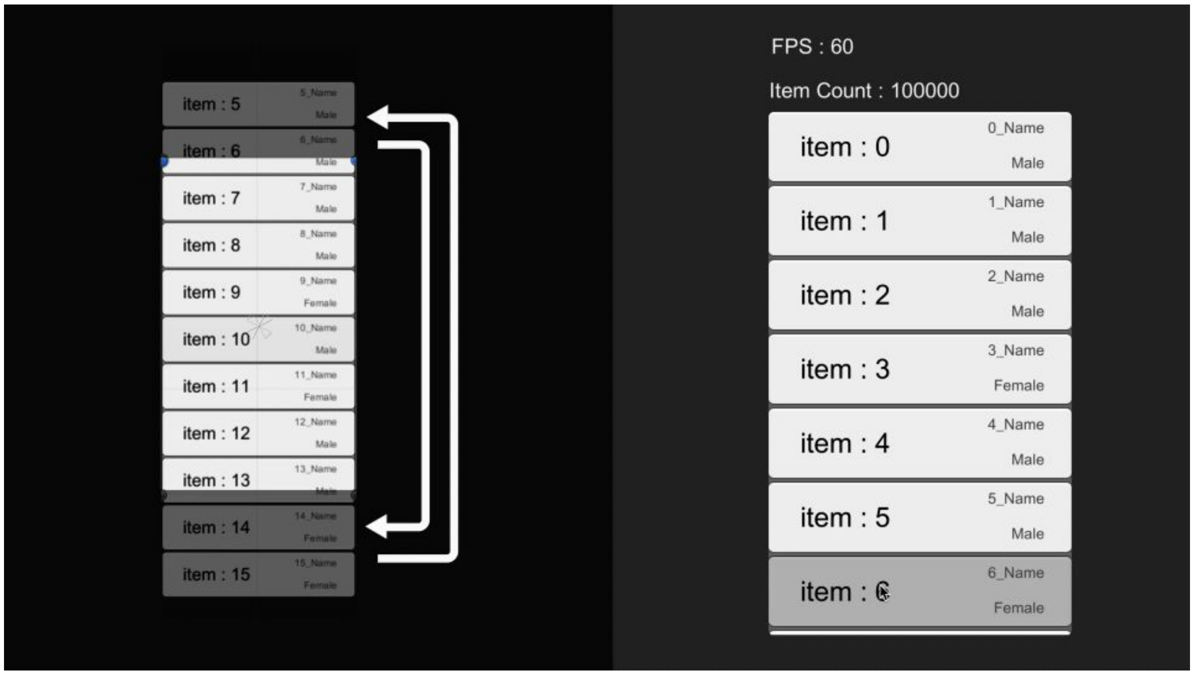
Recyclable Scroll Rect

https://twitter.com/polyandcode
https://polyandcode.com
https://www.facebook.com/Polyandcode
https://www.instagram.com/polyandcode

Summary

Using the default Scroll-Rect to create lists with a huge number of items results in a laggy performance. Especially when creating a list with hundreds or thousands of items, it becomes impossible to use the Scroll Rect with the default approach i.e instantiating that many items. *Recyclable Scroll Rect* reuses or recycles the least number of cells required to fill the viewport. As a result, a huge number of items can be shown in the list without any performance hit. Vertical, Horizontal and Grid layouts are supported.



Quickstart Guide :

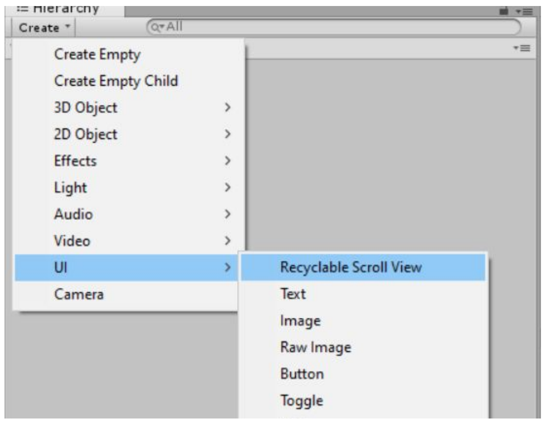
Check the Demo scenes for a complete example

The usage and structure are similar to Native iOS *TableViewController*. There are mainly two parts in setting up a Recyclable Scroll Rect; Prototype cell and DataSource. Following are the steps to set up a *Recyclable Scroll Rect* in detail:

1. **Recyclable Scroll View**
2. **Prototype cell**
3. **Cell class**
4. **Datasource**

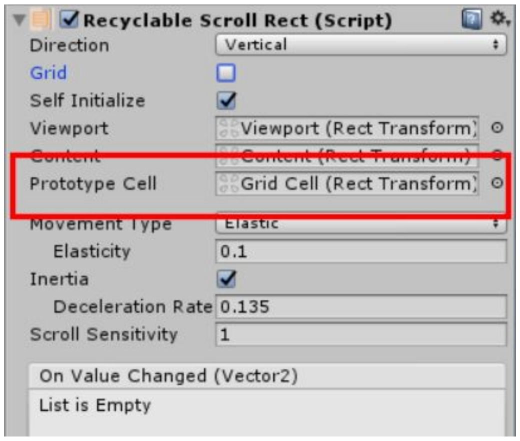
1. Recyclable Scroll View

You can create a *Recyclable Scroll View* by going to *Create -> UI -> Recyclable Scroll View*.



2. Prototype Cell

A Prototype cell is basically the cell layout for your list. A prototype cell can be in the hierarchy as the content's child or it can be a prefab. Don't worry about disabling it if it is present in the hierarchy, it will not show up in play mode. The prototype cell must be assigned to the *Recyclable Scroll Rect*


Recyclable Scroll Rect (Script)
Direction: Vertical
Grid: ☐
Self Initialize: ✓
Viewport: Viewport (Rect Transform)
Content: Content (Rect Transform)
Prototype Cell: Grid Cell (Rect Transform)
Movement Type: Elastic
Elasticity: 0.1
Inertia: ✓
Deceleration Rate: 0.135
Scroll Sensitivity: 1
On Value Changed (Vector2)
List is Empty

3. Cell class

Once you create your desired Prototype cell, assign it to the *Recyclable Scroll Rect* component. Now you will need to create a *Cell* script and attach it to the Prototype Cell. This script must be a *Monobehaviour* implementing *ICell* interface. The purpose of a Cell script is to configure the cell as the list is scrolled or updated. You must keep reference to the UI items that are required to be updated according to your data source.

Check *DemoCell* class for reference

```
public class DemoCell : MonoBehaviour, ICell
{
    //UI
    public Text nameLabel;
    public Text genderLabel;
    public Text idLabel;

    //Model
    private ContactInfo _contactInfo;
    private int _cellIndex;

    //This is called from the SetCell method in DataSource
    public void ConfigureCell(ContactInfo contactInfo,int cellIndex)
    {
        _cellIndex = cellIndex;
        _contactInfo = contactInfo;

        nameLabel.text = contactInfo.Name;
        genderLabel.text = contactInfo.Gender;
        idLabel.text = contactInfo.id;
    }
}
```

4. Data source

The next step is to create a Data source class. A Data source is responsible for data-related operations in the *Recyclable Scroll Rect*. These are the number of items in the list and how a cell should be configured according to the data. To create a Data source, implement the *IRecyclableScrollRectDataSource* interface and its methods :

• **GetItemCount**: This method tells *Recyclable Scroll Rect* the length of the List.

• **SetCell** : This method is responsible for configuring the cell UI according to your data. A cell is received as a parameter in this method with its index in the list. Using these, the necessary UI configuration can be done for the cell. The received cell is of *ICell* type. It must be cast to the implemented Cell type before using.

After the creation of a Cell and Data source, the last step is to assign the Data source instance to the *Recyclable Scroll Rect*. The assignment must be done in *Awake* or before the *Recyclable Scroll Rect*'s Start method (Check *others* section below for details on self-initialization).

Check *RecyclableScrollerDemo* class for reference

```
public class RecyclableScrollerDemo : MonoBehaviour,IRecyclableScrollRectDataSource
{
    [SerializeField]
    RecyclableScrollRect _recyclableScrollRect;

    [SerializeField]
    private int _dataLength;

    //Dummy data List
    private List<ContactInfo> _contactList = new List<ContactInfo>();

    //Recyclable scroll rect's data source must be assigned in Awake.
    private void Awake()
    {
        InitData();
        _recyclableScrollRect.DataSource = this;
    }

    #region DATA-SOURCE

    /// <summary>
    /// Data source method. return the list length.
    /// </summary>
    public int GetItemCount()
    {
        return _contactList.Count;
    }
    /// <summary>
    /// Called for a cell every time it is recycled
    /// Implement this method to do the necessary cell configuration.
    /// </summary>
    public void SetCell(ICell cell, int index)
    {
        //Casting to the implemented Cell
        var item = cell as DemoCell;
        item.ConfigureCell(_contactList[index],index);
    }

    #endregion
}
```

Others

Self-Initialize

The *Recyclable Scroll Rect* initializes on its own in its *Start* method. If you wish to initialize it yourself you can turn off the component's *self initialize* property and call the *Initialize* method whenever required. Make sure the Data-source is assigned before initializing.

Reloading Data

If a new data-source is assigned after initialization, call the *ReloadData()* function. Alternatively *ReloadData(IRecyclableScrollRectDataSource dataSource)* can also be used for assigning the data-source and reloading data.