适用于 UGUI 2.4 的超级 ScrollView

包装概述

在 SuperScrollView 包中，包含三个主要组件：LoopListView2，LoopGridView 和 LoopStaggeredGridView。LoopListView2 主要用于 ListView，LoopGridView 主要用于 GridView，所有项目的大小均相同，而 LoopStaggeredGridView 主要用于 StaggeredGridView。StaggeredGridView 是 GridView，对于垂直 GridView，项目具有不同的高度（对于水平 GridView，项目具有不同的宽度），例如 www.pinterest.com 好像。

LoopListView2 和 LoopGridView 和 LoopStaggeredGridView 是附加到 UGUI ScrollRect 的相同游戏对象的组件。它们帮助 UGUI ScrollRect 支持高性能和节省内存的任何计数项目。

本文档将首先介绍 LoopListView2 组件，然后介绍 LoopGridView 组件，然后介绍 LoopStaggeredGridView 组件。

LoopListView2 概述

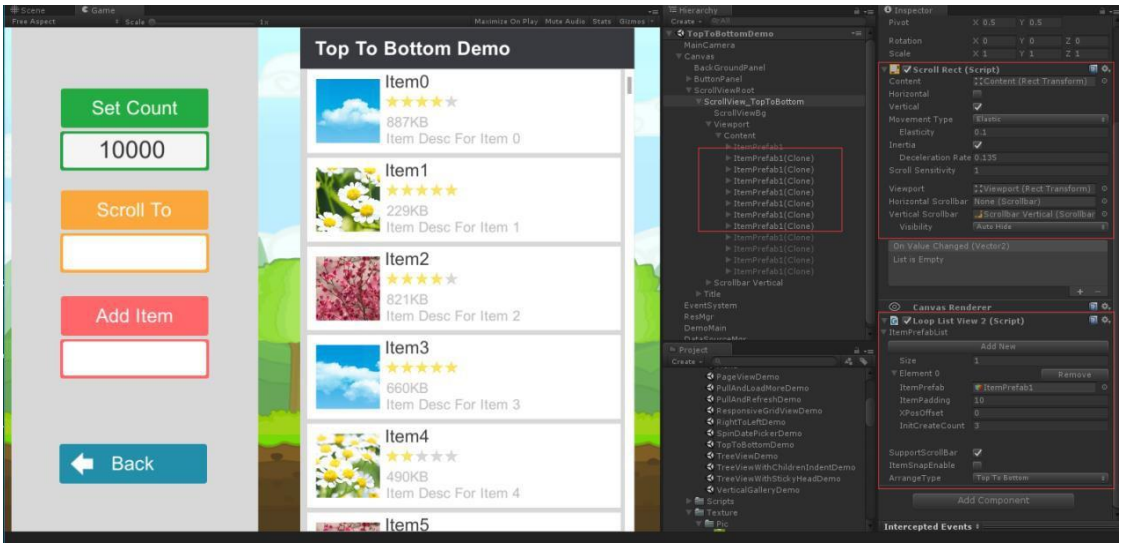
对于具有 10,000 个项目的 ScrollRect，LoopListView2 不会真正创建 10,000 个项目，而是根据视口的大小创建一些项目。

例如，当 ScrollRect 向上移动时，LoopListView2 组件将检查最顶层的项目的位置，一旦最顶层的项目不在视口中，则 LoopListView2 组件将回收最顶层的项目，同时检查最底层的项目的位置。 ，并且最下面的项目位于视口的底部附近后，LoopListView2 组件将调用 onGetItemByIndex 处理程序创建一个新项目，然后将新创建的项目放在最下面的项目下，因此新创建的项目成为新的最下面的项目。

每个项目都可以使用不同的预制件，并且可以具有不同的高度/宽度和填充物。

在具有路径的文件夹中，有几个示例可以帮助您学习 LoopListView2 组件：资产-> SuperScrollView->演示->场景-> ListView。

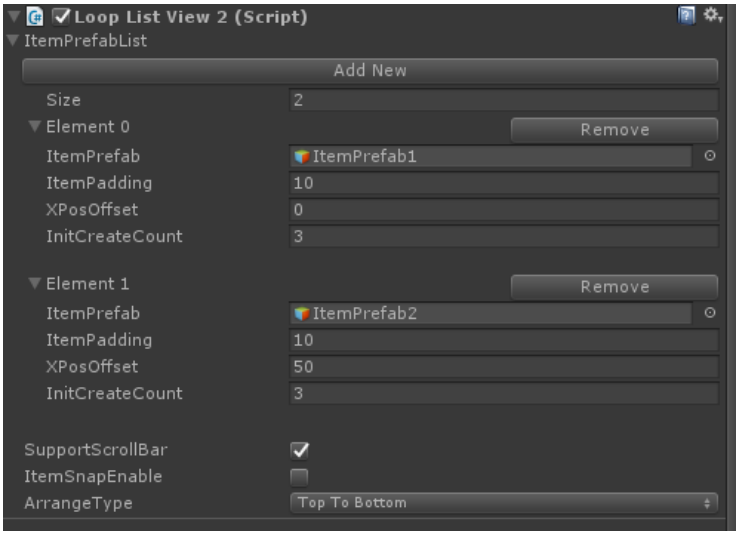
下图是 TopToButtom 排列的 scrollrect 的样子：



在上图中，scrollrect 有 10000 个项目，但实际上只有 7 个真正创建的项目。

LoopListView2 检查器设置

在 Inspector 中，要确保 LoopListView2 组件运行良好，需要设置几个参数：



ItemPrefabList：这是要克隆的现有项目。

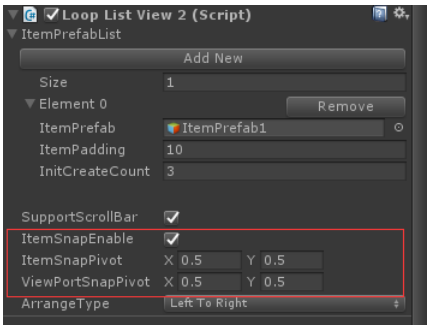
每个项目可以使用不同的预制件，并且每个预制件可以具有不同的默认填充（scrollrect 中每个项目之间的间距量）。而且，每个预制件可以具有不同的默认 x 本地 pos（用于“垂直滚动”）或默认 y 本地（用于“水平滚动”），在此称为（X / Y）PosOffset。每个预制件都有一个用于获取和回收操作的池，InitCreateCount 是启动时在池中创建的计数。

实际上，在运行时，每个项目都可以具有不同的填充和（X / Y）PosOffset，您可以在 onGetItemByIndex 回调中更改项目的填充和（X / Y）PosOffset。您可以通过 LoopListViewItem2 获取/设置它们。填充和 LoopListViewItem2。StartPosOffset.

重要说明：所有 itemPrefab 的 localScale 必须为（1,1,1）。

SupportScrollbar：如果选中，则 LoopListView2 组件将支持滚动条。

ItemSnapEnable:



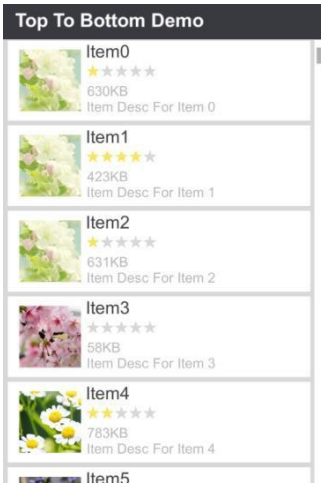
如果选中，则 LoopListView2 组件将尝试将项目捕捉到视口中的配置位置。

ItemSnapPivot 是项目的捕捉枢轴点的位置，定义为矩形本身大小的一部分。0,0 对应于左下角，而 1,1 对应于右上角。

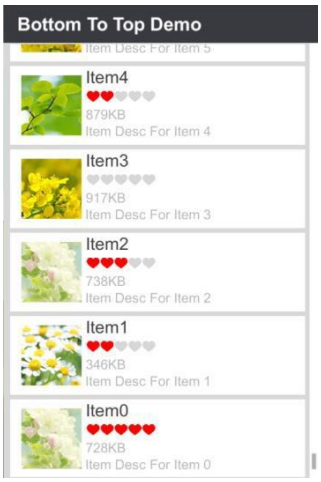
ViewPortSnapPivot 是 ScrollRect ViewPort 的捕捉枢轴点的位置，定义为矩形本身大小的一部分。0,0 对应于左下角，而 1,1 对应于右上角。

ArrangeType：滚动方向，有四种类型：

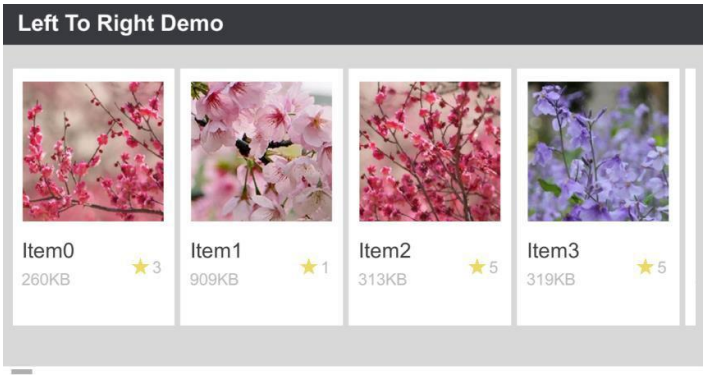
(1) TopToBottom：此类型用于垂直 screctrect，并且在 scrollrect 视口中，item0，item1，…itemN 从上到下一个一个地放置，如下所示：



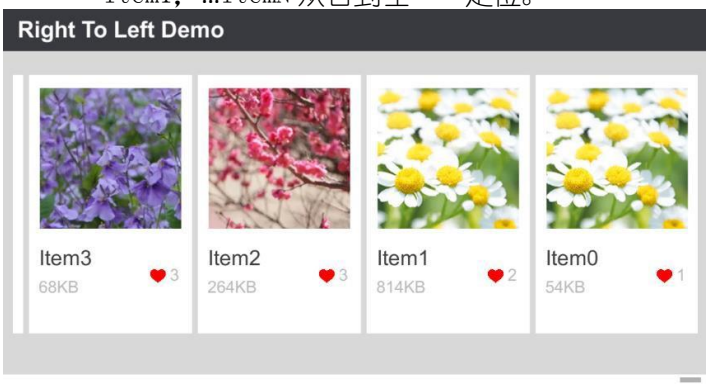
(2) BottomToTop：此类型用于垂直 screctrect，并且在 scrollrect 视口中，item0，item1，…itemN 从下到上一个一个地定位，如下所示：



(3) LeftToRight：此类型用于水平水平滚动矩形，并且在滚动矩形视口中将 item0，item1，…itemN 从左到右一一定位。



(4) RightToLeft：此类型用于水平水平滚动矩形，并且在滚动矩形视口中将 item0，item1，…itemN 从右到左一一定位。



LoopListView2 重要的公共方法

```
public void InitListView(int itemTotalCount,
    System.Func<LoopListView2, int, LoopListViewItem2> onGetItemByIndex,
    LoopListViewInitParam initParam = null)
```

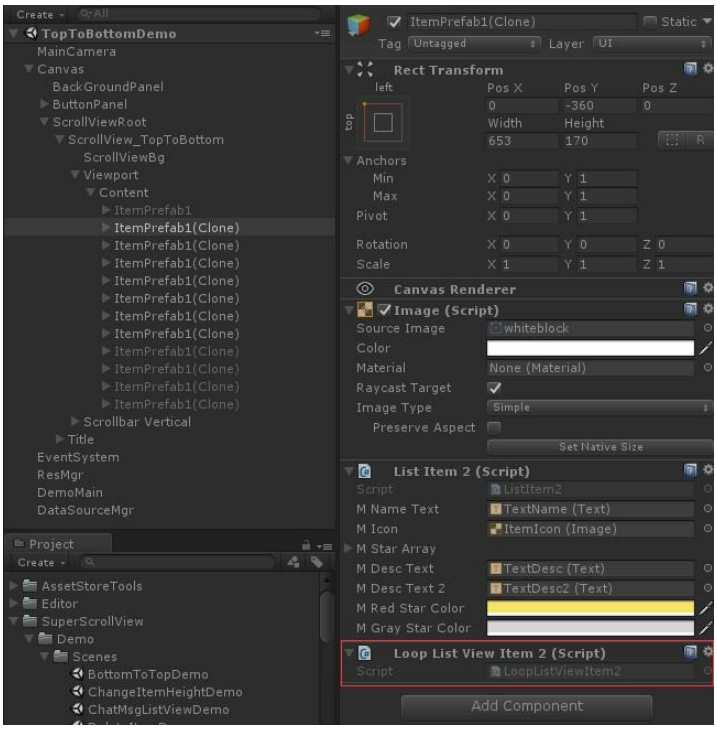
InitListView 方法是启动 LoopListView2 组件。有 3 个参数：itemTotalCount：滚动视图中的总项目数。如果此参数设置为-1，则表示存在无限项，并且不支持滚动条，并且 ItemIndex 可以从

-MaxInt 至+ MaxInt。如果此参数的值设置为> = 0，则 ItemIndex 只能是 0 到 itemTotalCount -1。

onGetItemByIndex：当某项进入 scrollrect 视口时，将调用此 Action 以商品的索引为参数，可以创建商品并更新其内容。

LoopListViewItem2 是 onGetItemByIndex 的返回值

每个创建的项目都会自动附加一个 LoopListViewItem2 组件：



LoopListViewItem2 组件非常简单：

```
public class LoopListViewItem2 : MonoBehaviour
{
    int mItemIndex = -1;
    int mItemId = -1;
    LoopListView2 mParentListView = null;
    bool mIsInitHandlerCalled = false;
    string mItemPrefabName;
    RectTransform mCachedRectTransform;
    float mPadding;
```

mItemIndex 属性指示项目在列表中的索引，如上所述，如果 itemTotalCount 设置为-1，则 mItemIndex 可以从-MaxInt 到+ MaxInt。如果

itemTotalCount 的值设置为> = 0，则 mItemIndex 只能从 0 到 itemTotalCount
-1.
mItemId 属性指示项目的 ID。在创建或从池中获取项目时设置此属性，并且在将项目回收回到池中之前将不再更改。
以下代码是 onGetItemByIndex 的示例：

```
LoopListViewItem2 OnGetItemByIndex(LoopListView2 listView, int itemIndex)
{
    如果 (itemIndex <0 || itemIndex> = DataSourceMgr.Get.TotalItemCount)
    {
        返回 null;
    }
    //获取要显示的数据
    ItemData itemData = DataSourceMgr.Get.GetItemDataByIndex(itemIndex);
    if(itemData == null)
    {
        返回 null;
    }
    / *获取新商品。每个项目都可以使用不同的预制，NewListViewItem 的参数为
    预制名称。
    并且所有预制件都应在 LoopListView2 检查器设置的 ItemPrefabList 中列出* /
    LoopListViewItem2 item = listView.NewListViewItem("ItemPrefab1");
    ListItem2 itemScript = item.gameObject.GetComponent<ListItem2>();//获得自己的组件
    // IsInitHandlerCalled 为 false 表示此项目是新建的，但未从池中获取。
    如果 (item.IsInitHandlerCalled ==假)
    {
        item.IsInitHandlerCalled = true;
        itemScript.Init () ; //在此处初始化项目，例如添加按钮单击事件侦听器。
    }
    //更新要显示的项目内容，例如图像，文本。itemScript.SetItemData(itemData);
    归还物品;
}
```

```
public LoopListViewItem2 NewListViewItem(string itemPrefabName)
```

此方法用于获取新项目，该新项目是来自名为 itemPrefabName 的预制件的克隆。此方法通常在 onGetItemByIndex 中使用。

```
public void SetListItemCount(int itemCount, bool resetPos = true)
```

此方法可用于在运行时设置滚动视图的项目总数。如果此参数设置为-1，则表示存在无限项，并且不支持滚动条，并且 ItemIndex 可以从 -MaxInt 到+ MaxInt。如果此参数的值设置为> = 0，则 ItemIndex 只能是 0 到 itemTotalCount -1。
如果将 resetPos 设置为 false，则此方法完成后，scrollrect 的内容位置将不会更改。

```
public LoopListViewItem2 GetShownItemByItemIndex(int itemIndex)
```

通过 itemIndex 获取可见项。如果该项不可见，则此方法返回 null。

```
public LoopListViewItem2 GetShownItemByIndex(int index)
```

所有可见项都存储在 List <LoopListViewItem2>中，该列表名为 mItemList；此方法是通过可见项列表中的索引获取可见项。参数索引是从 0 到 mItemList.Count。

```
public int ShownItemCount
{
    get
    {
        return mItemList.Count;
    }
}
```

获取所有可见项目的总数。

```
public void RefreshItemByItemIndex(int itemIndex)
```

通过 itemIndex 更新项目。如果第 itemIndex 个项目不可见，则此方法将不执行任何操作。否则，此方法将首先调用 onGetItemByIndex (itemIndex) 以获取更新的项目，然后重新放置所有可见项目的位置。

```
public void RefreshAllShownItem()
```

此方法将更新所有可见项目。

```
public void MovePanelToItemIndex(int itemIndex, float offset)
```

此方法会将 scrollrect 内容的位置移动到（第 itemIndex 个项目的位置+ offset），在当前版本中，itemIndex 从 0 到 MaxInt，offset 从 0 到 scrollrect 视口大小。

```
public void OnItemSizeChanged(int itemIndex)
```

对于垂直 scrollrectrect，当可见项目的高度在运行时发生变化时，应调用此方法以使 LoopListView2 组件将所有可见项目的位置重新定位。
对于水平 scrollrect，当可见项的宽度在运行时更改时，则此方法

应该调用该方法，以使 LoopListView2 组件重新定位所有可见项的位置。

```
public void FinishSnapImmediately()
```

快速移动将立即完成。

```
public int CurSnapNearestItemIndex
```

获取具有视口捕捉点的最近项目索引。

```
public void SetSnapTargetItemIndex(int itemIndex)
```

设置捕捉目标项目索引。PageViewDemo 中使用此方法。

```
public void ClearSnapData()
```

清除当前的捕捉目标，然后 LoopScrollView2 将自动捕捉到 CurSnapNearestItemIndex。

LoopGridView 概述

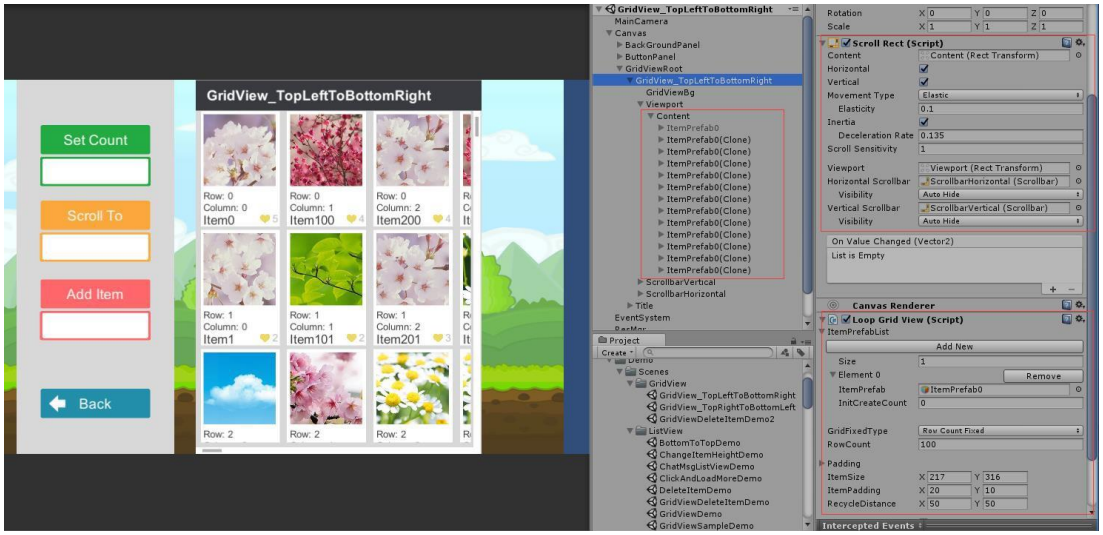
LoopGridView 组件附加到 UGUI ScrollRect 的相同游戏对象，并且主要用于 GridView，其中所有项目的大小均相同，并且具有固定计数的 Row 或固定计数的 Column。ScrollRect 可以同时水平和垂直滚动。对于具有 10,000 个项目（例如 100 行* 100 列）的 GridView，LoopGridView 不会真正创建 10,000 个项目，而是根据视口的大小创建一些项目。

例如，当 ScrollRect 向上移动时，LoopGridView 组件将检查最上一行的位置，一旦最上一行离开视口，则 LoopGridView 组件将回收最上一行的所有项目，并同时检查最下一行的位置，并且最下一行在视口的底部附近后，LoopGridView 组件将调用 onGetItemByRowColumn 处理函数以创建新行，并将视口中的所有项目放置在新行中，然后将新创建的行放置在最下一行，因此新创建的行将成为新的最下一行。

每个项目都可以使用不同的预制件。

在具有以下路径的文件夹中，有几个示例可帮助您学习 LoopGridView 组件：资产->SuperScrollView->演示->场景-> GridView。

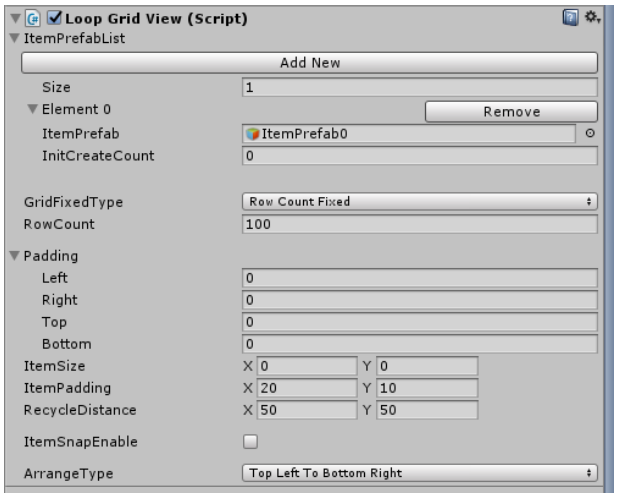
下图是从 TopLeft 到 BottomRight（100 行* 100 列）排列的 ScrollRect 的样子：



在上图中，GridView 有 10000 个项目，但实际上只有几个项目真正创建了。

LoopGridView 检查器设置

在 Inspector 中，要确保 LoopGridView2 组件运行良好，需要设置几个参数：



ItemPrefabList：这是要克隆的现有项目。

每个项目都可以使用不同的预制件。每个预制件都有一个用于获取和回收操作的池，InitCreateCount 是启动时在池中创建的计数。

重要说明：所有 itemPrefab 的 localScale 必须为（1,1,1）。ScrollRect 的内容游戏对象的 localScale 也需要为（1,1,1）。

GridFixedType：该值可以是 ColumnCountFixed 或 RowCountFixed。此属性类似于 UGUI GridLayoutGroup.constraint 和 GridLayoutGroup.startAxis，这是应该将项目沿哪个轴放置在最前面。

RowCount / ColumnCount：如果 GridFixedType 为 ColumnCountFixed，则在此处可以设置 GridView 的列数；如果 GridFixedType 为 RowCountFixed，则在此处将设置 GridView 的行数。此属性类似于 UGUI GridLayoutGroup。约束计数。

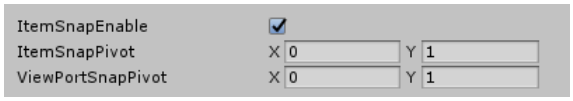
填充：项目和视口之间的空间，此属性类似于 UGUI GridLayoutGroup.Padding。

ItemSize：项目的大小。如果 x 或 y 设置为 0，则在初始化 LoopGridView 组件时，ItemSize 将自动设置 ItemPrefabList 的第一项的大小。

ItemPadding：GridView 中每个项目之间的间隔量。

RecycleDistance：物品将在回收时离开视口的距离。

ItemSnapEnable:



如果选中，则 LoopGridView 组件将尝试将项目捕捉到视口中的配置位置。

ItemSnapPivot 是项目的捕捉枢轴点的位置，定义为矩形本身大小的一部分。0,0 对应于左下角，而 1,1 对应于右上角。

ViewPortSnapPivot 是 ScrollRect ViewPort 的捕捉枢轴点的位置，定义为矩形本身大小的一部分。0,0 对应于左下角，而 1,1 对应于右上角。

ArrangeType：滚动方向，有四种类型：
TopLeftToBottomRight ， BottomLeftToTopRight, TopRightToBottomLeft, BottomRightToTopLeft.

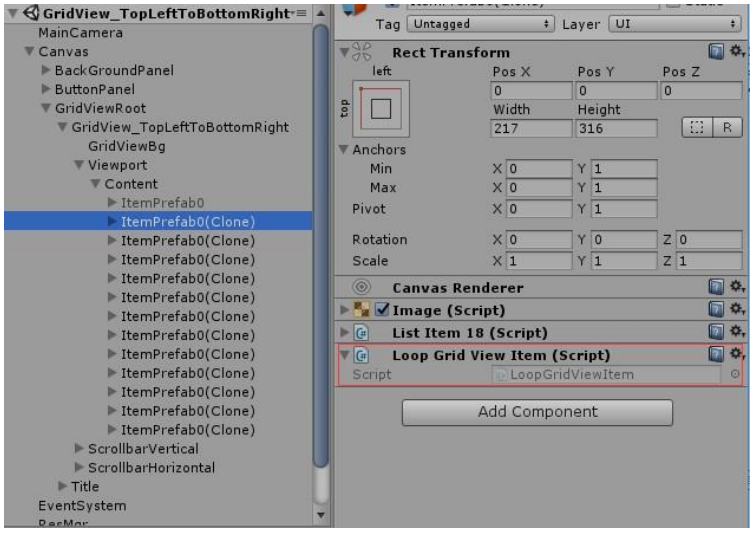
LoopGridView 重要的公共方法

```
public void InitGridView(int itemTotalCount,
    System.Func<LoopGridView,int,int,int, LoopGridViewItem> onGetItemByRowColumn,
    LoopGridViewSettingParam settingParam = null,
    LoopGridViewInitParam initParam = null)
```

InitGridView 方法是启动 LoopGridView 组件。有 4 个参数：itemTotalCount：GridView 中的总项目数。必须将此参数设置为> = 0，并且 ItemIndex 可以从 0 到 itemTotalCount -1。
onGetItemByRowColumn：当某个项目进入 ScrollRect 视口时，此操作将为以项目（itmeIndex，row，column）作为参数调用，以创建项目并更新其内容。
settingParam：此参数是 LoopGridViewSettingParam 对象。您可以使用此参数来覆盖“检查器设置”中的值，例如：

```
LoopGridViewSettingParam settingParam = new LoopGridViewSettingParam();
settingParam.mItemSize = new Vector2(500, 500);
settingParam.mItemPadding = new Vector2(40, 40);
settingParam.mPadding = new RectOffset(10, 20, 30, 40);
settingParam.mGridFixedType = GridFixedType.RowCountFixed;
settingParam.mFixedRowOrColumnCount = 6;
mLoopGridView.InitGridView(DataSourceMgr.Get.TotalItemCount, OnGetItemByIndex, settingParam);
```

LoopGridViewItem 是 onGetItemByRowColumn 的返回值
每个创建的项目都有一个自动附加的 LoopGridViewItem 组件：



LoopGridViewItem 组件非常简单：

```
public class LoopGridViewItem : MonoBehaviour
{
    // indicates the item's index in the list the mItemIndex can only be from 0 t
    int mItemIndex = -1;
    // the row index, the item is in. starting from 0.
    int mRow = -1;
    // the column index, the item is in. starting from 0.
    int mColumn = -1;
    //indicates the item's id.
    //This property is set when the item is created or fetched from pool,
    //and will no longer change until the item is recycled back to pool.
    int mItemId = -1;
    LoopGridView mParentGridView = null;
    bool mIsInitHandlerCalled = false;
    string mItemPrefabName;
    RectTransform mCachedRectTransform;
```

mItemIndex 属性指示项目在 GridView 中的索引。

mItemId 属性指示项目的 ID。在创建或从池中获取项目时设置此属性，并且在将项目回收回到池中之前将不再更改。

以下代码是 onGetItemByRowColumn 的示例：

```
LoopGridViewItem OnGetItemByRowColumn (LoopGridView gridView, int itemIndex, int 行, int 列)
{
    //获取要显示的数据
    ItemData itemData = DataSourceMgr.Get.GetItemDataByIndex(itemIndex);
    如果 (itemData == null)
    {
        返回 null;
    }
    /*
    得到一个新项目。每个物品都可以使用不同的预制件，
    NewListViewItem 的参数是预制名称。
    并且所有预制件都应在 LoopGridView 检查器设置的 ItemPrefabList 中列出
    */
    LoopGridViewItem 项目= gridView.NewListViewItem (“ ItemPrefab0”) ;
    ListItem18 itemScript = item.GetComponent <ListItem18> () ; //获取自己的组件
    // IsInitHandlerCalled 为 false 表示此项目是新建的，但未从池中获取。如果
     (item.IsInitHandlerCalled ==假)
    {
        item.IsInitHandlerCalled = true;
        itemScript.Init () ; //在此处初始化项目，例如添加按钮单击事件侦听器。
    }
    //更新要显示的项目内容，例如图像，文本。
    itemScript.SetItemData(itemData, itemIndex, row, column);归还
    物品;
}
```

```
public LoopGridViewItem NewListViewItem(string itemPrefabName)
```

此方法用于获取新物品，该新物品是来自名为的预制件的克隆

itemPrefabName.此方法通常在 onGetItemByRowColumn 中使用。

```
public void SetListItemCount(int itemCount, bool resetPos = true)
```

此方法可用于在运行时设置 GridView 的项目总数。如果将 resetPos 设置为 false，则此方法完成后，ScrollRect 的内容位置将不会更改。

```
public LoopGridViewItem GetShownItemByItemIndex(int itemIndex)
```

通过 itemIndex 获取可见项。如果该项不可见，则此方法返回 null。

```
public LoopGridViewItem GetShownItemByRowColumn(int row, int column)
```

通过（行，列）获取可见项。如果该项不可见，则此方法返回 null。

```
public void RefreshItemByItemIndex(int itemIndex)
```

通过 itemIndex 更新项目。如果第 itemIndex 个项目不可见，则此方法将不执行任何操作。否则，此方法将调用 onGetItemByRowColumn 以获取更新的项目

```
public void RefreshItemByRowColumn(int row,int column)
```

通过（行，列）更新项目。如果该项目不可见，则此方法将无效。否则，此方法将调用 onGetItemByRowColumn 以获取更新的项目

```
public void RefreshAllShownItem()
```

此方法将更新所有可见项目。

```
public void MovePanelToItemByIndex(int itemIndex, float offsetX = 0, float offsetY = 0)
```

此方法会将 ScrollRect 内容的位置移动到（itemIndex-th item 的位置+ offset）。

```
public void MovePanelToItemByRowColumn(int row,int column, float offsetX = 0,float offsetY = 0)
```

此方法会将面板的位置移动到（（行，列）项目的位置+ offset）。

```
public void SetGridFixedGroupCount(GridFixedType fixedType,int count)
```

在运行时更改 GridFixedType 和固定的行/列数。
您还可以在运行时通过 SetItemSize（Vector2），SetItemPadding（Vector2），SetPadding（RectOffset）更改 itemSize，itemPadding，contentPadding。

```
public void FinishSnapImmediately()
```

快速移动将立即完成。

```
public RowColumnPair CurSnapNearestItemRowColumn
```

使用视口捕捉点获取最近的项目的行和列。

```
public void SetSnapTargetItemRowColumn(int row, int column)
```

设置捕捉目标项目的行和列。

```
public void ClearSnapData()
```

清除当前的捕捉目标，然后 LoopGridView 将自动捕捉到 CurSnapNearestItemRowColumn。

LoopStaggeredGridView 概述

LoopStaggeredGridView 主要用于 StaggeredGridView，对于垂直 GridView，项目具有不同的高度（对于水平 GridView，项目具有不同的宽度）。因此，如果 GridView 的所有项目都具有相同的大小，则最好使用 LoopGridView。

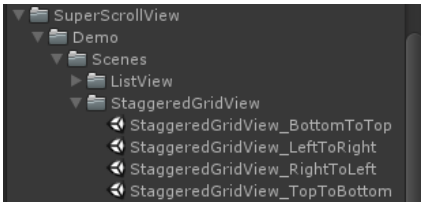
LoopStaggeredGridView 与 LoopListView2 相似，对于具有 10,000 个项目的 ScrollRect，LoopStaggeredGridView 不会真正创建 10,000 个项目，而是根据视口的大小创建一些项目。

当 ScrollRect 向上移动时，对于垂直 GridView，LoopStaggeredGridView 组件将检查每列的最顶层项目的位置，并且一旦列的最顶层项目不在视口中，则 LoopStaggeredGridView 组件将回收最顶层的项目。

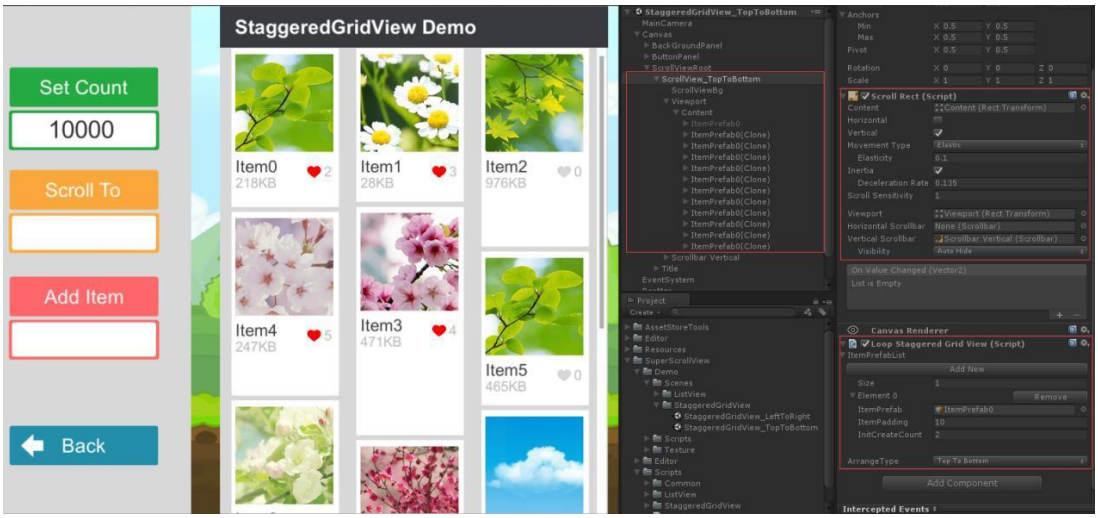
并且同时检查每列的最底层项目的位置，并且一旦列的最底层项目在视口的底部附近，LoopStaggeredGridView 组件将调用 onGetItemByIndex 处理程序以创建一个新项目，然后将其放置在新创建的项目上在最短列的最下端项目下方的列，即该列的最下端项目的底部位置在所有列的最下位位置中的最高位置。总体而言，所有项目都从左到右，从上到下排列。

每个项目都可以使用不同的预制件。但是对于垂直 GridView，项目可以具有不同的高度，但宽度必须相同。对于水平 GridView，项目可以具有不同的宽度，但高度必须相同。

在具有路径的文件夹中，有两个示例可帮助您学习 LoopStaggeredGridView 组件：资产-> SuperScrollView->演示->场景-> StaggeredGridView。



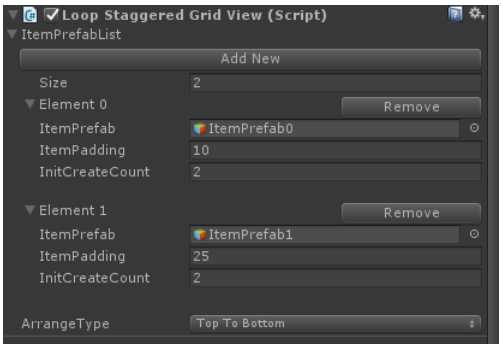
下图是 TopToButtom 排列的具有 3 列的 StaggeredGridView 的样子：



在上图中，scrollrect 有 10,000 个项目，但实际上，实际创建的项目只有 11 个。

LoopStaggeredGridView 检查器设置

在 Inspector 中，要确保 LoopStaggeredGridView 组件运行良好，需要设置几个参数：



ItemPrefabList：这是要克隆的现有项目。
每个项目可以使用不同的预制件，并且每个预制件可以具有不同的默认填充（scrollrect 中每个项目之间的间距量）。每个预制件都有一个用于获取和回收操作的池，InitCreateCount 是启动时在池中创建的计数。
实际上，在运行时，每个项目可以具有不同的填充，您可以在 onGetItemByIndex 回调中更改项目的填充。您可以通过 LoopStaggeredGridViewItem 获取/设置它们。填充
重要说明：所有 itemPrefab 的 localScale 必须为（1,1,1）。

ArrangeType：滚动方向，与 ListView 相同。有四种类型：TopToBottom，BottomToTop，LeftToRight，RightToLeft。

LoopStaggeredGridView 重要的公共方法

```
public void InitListView(int itemTotalCount, GridViewLayoutParam layoutParam,
    System.Func<LoopStaggeredGridView, int, LoopStaggeredGridViewItem> onGetItemByItemIndex,
    StaggeredGridViewInitParam initParam = null)
```

InitListView 方法是启动 LoopStaggeredGridView 组件。有 4 个参数：
itemTotalCount：滚动视图中的总项目数，此参数应> = 0。

```
public class GridViewLayoutParam
{
    public int mColumnOrRowCount = 0;
    public float mItemWidthOrHeight = 0;
    public float mPadding1 = 0;
    public float mPadding2 = 0;
    public float[] mCustomColumnOrRowOffsetArray = null;
```

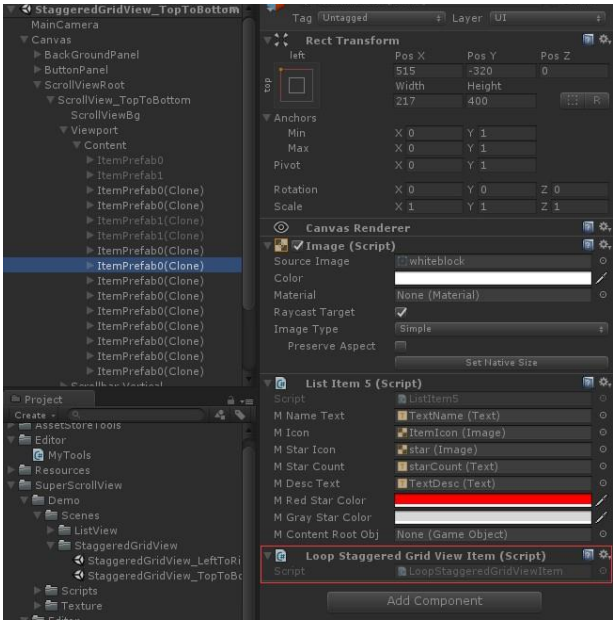
layoutParam：此类非常简单，您需要一个新的 GridViewLayoutParam 实例并设置所需的值。
对于垂直 GridView，mColumnOrRowCount 是列数，

mItemWidthOrHeight 是项目的宽度，mPadding1 是视口的左边距，mPadding2 是视口的右边距。对于水平 GridView，mColumnOrRowCount 是行数，mItemWidthOrHeight 是项目的高度，mPadding1 是视口的上边距，mPadding2 是视口的下边距。如果 mCustomColumnOrRowOffsetArray 为 null，也就是说，您不设置此参数的值，则 GiriView 将排列所有平均的列或行。如果 mCustomColumnOrRowOffsetArray 不为 null，则数组的值为每个列/行的 XOffset / YOffset，并且 mCustomColumnOrRowOffsetArray.length 必须与 mColumnOrRowCount 相同。

onGetItemByItemIndex：当某个项目进入 scrollrect 视口时，此操作将为以项目的索引作为参数调用，以创建项目并更新其内容。

LoopStaggeredGridViewItem 是 onGetItemByItemIndex 的返回值

每个创建的项目都有一个自动附加的 LoopStaggeredGridViewItem 组件：



LoopStaggeredGridViewItem 组件非常简单：

```
public class LoopStaggeredGridViewItem : MonoBehaviour
{
    int mItemIndex = -1;
    int mItemIndexInGroup = -1;
    int mItemId = -1;
    float mPadding;
    bool mIsInitHandlerCalled = false;
    string mItemPrefabName;
```

mItemIndex 属性指示项目在 GridView 中的索引，范围可以从 0 到 itemTotalCount -1.

mItemIndexInGroup 属性指示项目在列或行中的索引。这里的"组"一词是：对于垂直 GridView，组是一列，对于水平 GridView，组是一行。

mItemId 属性指示项目的 ID。在创建或从池中获取项目时设置此属性，并且在将项目回收回到池中之前将不再更改。

以下代码是 onGetItemByItemIndex 的示例：LoopStaggeredGridViewItem OnGetItemByItemIndex (LoopStaggeredGridView gridView, int itemIndex)

```
{
    如果 (itemIndex <0 || itemIndex> = DataSourceMgr.Get.TotalItemCount)
    {
        返回 null;
    }
    //获取要显示的数据
    ItemData itemData = DataSourceMgr.Get.GetItemDataByIndex(itemIndex);
    if(itemData == null)
    {
        返回 null;
    }
    / *获取新商品。每个项目都可以使用不同的预制，NewListViewItem 的参数为
    预制名称。
    所有预制件都应在 LoopStaggeredGridView 检查器设置的 ItemPrefabList 中列出* /
    LoopStaggeredGridViewItem item = gridView.NewListViewItem("ItemPrefab1");ListItem2
    itemScript = item.gameObject.GetComponent<ListItem2>();//得到你自己的
   零件
    // IsInitHandlerCalled 为 false 表示此项目是新建的，但未从池中获取。
    如果 (item.IsInitHandlerCalled ==假)
    {
        item.IsInitHandlerCalled = true;
        itemScript.Init () ; //在此处初始化项目，例如添加按钮单击事件侦听器。
    }
    //更新要显示的项目内容，例如图像，文本。itemScript.SetItemData(itemData);
    归还物品;
}
```

```
public LoopStaggeredGridViewItem NewListViewItem(string itemPrefabName)
```

此方法用于获取新项目，该新项目是来自名为 itemPrefabName 的预制件的克隆。此方法通常在 onGetItemByItemIndex 中使用。

```
public void SetListItemCount(int itemCount, bool resetPos = true)
```

此方法可用于在运行时设置 GridView 的项目总数。如果将 resetPos 设置为 false，则此方法完成后，scrollrect 的内容位置将不会更改。

```
public LoopStaggeredGridViewItem GetShownItemByItemIndex(int itemIndex)
```

通过 itemIndex 获取可见项。如果该项不可见，则此方法返回 null。

```
public void RefreshItemByItemIndex(int itemIndex)
```

通过 itemIndex 更新项目。如果第 itemIndex 个项目不可见，则此方法将不执行任何操作。否则，此方法将首先调用 onGetItemByItemIndex（itemIndex）以获取更新的项目，然后重新放置同一组（即同一列/行）中所有可见项目的位置。

```
public void RefreshAllShownItem()
```

此方法将更新所有可见项目。

```
public void MovePanelToItemIndex(int itemIndex, float offset)
```

此方法会将 scrollrect 内容的位置移动到（itemIndex-th item 的位置+ offset）

```
public void OnItemSizeChanged(int itemIndex)
```

对于垂直 scrollrectrect，当可见项目的高度在运行时发生变化时，则应调用此方法以使 LoopStaggeredGridView 组件将所有可见项目在同一组（即同一列/行）中的位置重新定位。对于水平 scrollrectrect，当可见项目的宽度在运行时发生变化时，则应调用此方法，以使 LoopStaggeredGridView 组件将所有可见项目在同一组（即同一列/行）中的位置重新定位。