# 介绍

RecyclerView是更先进、更灵活的ListView版本。

RecyclerView通过使用不同的组件协同工作以显示数据。把Recyclerview对象作为一个整体容器添加到你的布局中，RecyclerView通过你提供的一个布局管理器来把views填充到它自己里面，你可以使用标准的布局管理器（LinearLayoutManager或GridLayoutManager），或者使用你自己实现的布局管理器。

列表中的views由 view holder对象表示，这些对象是你自己定义的，继承自RecyclerView.ViewHolder。每个view holder负责在一个view上显示一项item，例如，如果您的列表显示音乐收藏，则每个view holder可能代表一张专辑。当用户滚动浏览列表时，RecyclerView会获取屏幕外的view并将其重新绑定到正在滚动到屏幕上的数据。

view holder对象由adapter管理，adapter继承自RecyclerView.Adapter。adapter根据需要创建view holders。adapter还将它们的数据绑定到view holders，通过将view holder分配到某个位置并调用adapter的onBindViewHolder()方法来实现。

RecyclerView进行了大量优化工作：

* 在首次填充列表时，例如视图正在显示列表位置 0 – 9的内容，则RecyclerView将创建一些新的view holders并绑定到列表上，并且可能还会创建新view holder绑定位置10，这样，如果用户滚动列表，则下一个元素就做好要显示的准备了。
* 当用户滚动列表时，RecyclerView根据需要创建新的view holder。它还保存了滚动到屏幕外的view holders，因此可以重复使用它们。如果用户切换了他们的滚动方向，则可以把滚出屏幕外的view holders带回来。另一方面，如果用户继续沿相同方向滚动，则离开屏幕时间最长的view holders可以重新绑定到新数据。view holders不需要创建或inflated它的view。
* 当显示的items更改时，可以通过调用适当的RecyclerView.Adapter.notify…()方法来通知adapter。然后，适配器的内置代码仅重新绑定受影响的items。

# 示例

1. 可通过File🡪New🡪Fragment🡪 Fragment（List）的方式创建一个带RecyclerView的Fragment，然后只需要把这个Fragment放到Activity中显示即可。(注：这个Activity需要实现Fragment里面的OnListFragmentInteractionListener接口)
2. 官方示例代码：
   1. <https://github.com/android/views-widgets-samples/tree/master/RecyclerView> （Java）
   2. <https://github.com/android/views-widgets-samples/tree/master/RecyclerViewKotlin> （Kotlin）

# 添加支持库

dependencies {

implementation 'com.android.support:recyclerview-v7:28.0.0'

}

androidx使用：androidx.recyclerview:recyclerview:1.0.0

# RecylerView使用简单示例

示例如下：

<?xml version="1.0" encoding="utf-8"?>  
<!-- A RecyclerView with some commonly used attributes -->  
<android.support.v7.widget.RecyclerView

xmlns:android="http://schemas.android.com/apk/res/android"  
    android:id="@+id/my\_recycler\_view"  
    android:scrollbars="vertical"  
    android:layout\_width="match\_parent"  
    android:layout\_height="match\_parent"/>

在Activity中获取RecyclerView引用，并设置布局管理器和adapter

class MainActivity : AppCompatActivity() {  
 private lateinit var recyclerView: RecyclerView  
 private lateinit var viewAdapter: RecyclerView.Adapter<\*>  
 private lateinit var viewManager: RecyclerView.LayoutManager  
  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 setContentView(R.layout.*activity\_main*)  
 val myDataset = *arrayOf*("1", "2", "3", "4", "5", "6")  
 viewManager = LinearLayoutManager(this)  
 viewAdapter = MyAdapter(myDataset)  
 recyclerView = findViewById<RecyclerView>(R.id.*my\_recycler\_view*).*apply* {  
 *// use this setting to improve performance if you know that changes  
 // in content do not change the layout size of the RecyclerView* setHasFixedSize(true)  
 *// use a linear layout manager  
 layoutManager* = viewManager  
 *// specify an viewAdapter (see also next example)  
 adapter* = viewAdapter  
  
 }  
 }  
}

**class** MyAdapter(**private val myDataset**: Array<String>) :  
 RecyclerView.Adapter<MyAdapter.MyViewHolder>() {  
  
 **class** MyViewHolder(**val textView**: TextView) : RecyclerView.ViewHolder(textView)  
  
 *// 创建新View (由布局管理器调用)* **override fun** onCreateViewHolder(parent: ViewGroup,  
 viewType: Int): MyAdapter.MyViewHolder {  
 *// create a new view* **val** textView = View.inflate(parent.*context*, android.R.layout.*simple\_list\_item\_1*, **null**) **as** TextView  
 *// set the view's size, margins, paddings and layout parameters  
 // ...* **return** MyViewHolder(textView)  
 }  
  
 *// 替换一个View的内容 (由布局管理器调用)* **override fun** onBindViewHolder(holder: MyViewHolder, position: Int) {  
 holder.**textView**.*text* = **myDataset**[position]  
 }  
  
 *// 返回你的数据集合的size (由布局管理器调用)* **override fun** getItemCount() = **myDataset**.**size**}

如果列表需要更新，请调用RecyclerView.Adapter对象的notifyItemChanged()方法。

提示：ListAdapter可以在数据更改时自动更新列表

# 自定义RecyclerView

## 修改布局

RecyclerView使用一个布局管理器来定位屏幕上的各个item，并决定何时重用不再对用户可见的项目。要重用（或回收）一个view，布局管理器可能会要求适配器用与数据集不同的元素替换视图的内容。以这种方式回收视图可避免创建不必要的视图或执行昂贵的findViewById()查找，从而提高了性能。Android支持库包括三个标准布局管理器，每个管理器都提供许多自定义选项：

* LinearLayoutManager将items排列在一维列表中。使用RecyclerView和LinearLayoutManager提供类似旧版ListView式的功能。
* GridLayoutManager将items排列在二维网格中，例如棋盘上的正方形。使用RecyclerView和GridLayoutManager提供类似旧版GridView式的功能。
* StaggeredGridLayoutManager 将物品排列在二维网格中，每列与以前的列略有偏移，就像一面美国国旗中的星星一样。

如果这些布局管理器都不符合您的需求，则可以通过继承RecyclerView.LayoutManager 抽象类来创建自己的布局管理器。

## 添加item动画

每当item更改时，RecyclerView 使用一个animator更改其外观。这个animator是继承自抽象RecyclerView.ItemAnimator类的对象。默认情况下，RecyclerView使用DefaultItemAnimator来提供动画。如果要提供自定义动画，则可以通过继承RecyclerView.ItemAnimator来定义你自己的animator。

## 启用list-item selection

[recyclerview-selection](https://developer.android.com/reference/androidx/recyclerview/selection/package-summary.html) 库使用户可以在[RecyclerView](https://developer.android.com/reference/androidx/recyclerview/widget/RecyclerView.html)上使用触摸或鼠标输入来选择列表中的items 。您保留对所选项目的视觉呈现的控制。您还可以保留对控制选择行为的策略的控制，例如可以选择的items以及可以选择的items数量。

要将selction support添加到[RecyclerView](https://developer.android.com/reference/androidx/recyclerview/widget/RecyclerView.html) 实例，请按照下列步骤操作：

1. 确定要使用的selection key type，然后构建一个 [ItemKeyProvider](https://developer.android.com/reference/androidx/recyclerview/selection/ItemKeyProvider.html)。

您可以使用三种key types来标识选定的items：[Parcelable](https://developer.android.com/reference/android/os/Parcelable.html)（ 以及所有子类，如 [Uri](https://developer.android.com/reference/android/net/Uri.html)）、[String](https://developer.android.com/reference/java/lang/String.html)和[Long](https://developer.android.com/reference/java/lang/Long.html)。有关selection-key types的详细信息，请参见 [SelectionTracker.Builder](https://developer.android.com/reference/androidx/recyclerview/selection/SelectionTracker.Builder.html)。

1. 实现[ItemDetailsLookup](https://developer.android.com/reference/androidx/recyclerview/selection/ItemDetailsLookup.html)。

[ItemDetailsLookup](https://developer.android.com/reference/androidx/recyclerview/selection/ItemDetailsLookup.html)允许selction库访问关于[RecyclerView](https://developer.android.com/reference/androidx/recyclerview/widget/RecyclerView.html) items给定 [MotionEvent](https://developer.android.com/reference/android/view/MotionEvent.html)的信息。对于由[ItemDetails](https://developer.android.com/reference/androidx/recyclerview/selection/ItemDetailsLookup.ItemDetails.html) 实例备份（或从[RecyclerView.ViewHolder](https://developer.android.com/reference/androidx/recyclerview/widget/RecyclerView.ViewHolder.html)实例提取）的实例， 它实际上是工厂 。

英文：

[ItemDetailsLookup](https://developer.android.com/reference/androidx/recyclerview/selection/ItemDetailsLookup.html) enables the selection library to access information about [RecyclerView](https://developer.android.com/reference/androidx/recyclerview/widget/RecyclerView.html) items given a [MotionEvent](https://developer.android.com/reference/android/view/MotionEvent.html). It is effectively a factory for [ItemDetails](https://developer.android.com/reference/androidx/recyclerview/selection/ItemDetailsLookup.ItemDetails.html) instances that are backed up by (or extracted from) a [RecyclerView.ViewHolder](https://developer.android.com/reference/androidx/recyclerview/widget/RecyclerView.ViewHolder.html) instance.

1. 更新[RecyclerView](https://developer.android.com/reference/androidx/recyclerview/widget/RecyclerView.html)中[Views](https://developer.android.com/reference/android/view/View.html)的item，以反映用户已选择或不选择它。

selection库没有为选择的items提供默认的视觉装饰，当你实现 [onBindViewHolder()](https://developer.android.com/reference/androidx/recyclerview/widget/RecyclerView.Adapter.html#onBindViewHolder(VH,%20int))时，你必须提供这个。推荐的方法如下：

* + 在中 [onBindViewHolder()](https://developer.android.com/reference/androidx/recyclerview/widget/RecyclerView.Adapter.html#onBindViewHolder(VH,%20int))中，在View对象上用true或false（取决于item是否是selected）调用[setActivated()](https://developer.android.com/reference/android/view/View.html#setActivated(boolean)) （不是[setSelected()](https://developer.android.com/reference/android/view/View.html#setSelected(boolean)) ）
  + 更新视图的样式以表示激活状态。我们建议您使用 [颜色状态列表资源](https://developer.android.com/guide/topics/resources/color-list-resource.html) 来配置样式。

1. 使用[ActionMode](https://developer.android.com/reference/androidx/appcompat/view/ActionMode.html)向用户提供工具来执行该选择的动作。

注册一个[SelectionTracker.SelectionObserver](https://developer.android.com/reference/androidx/recyclerview/selection/SelectionTracker.SelectionObserver.html)来通知当选择更改时。当一个selection是第一次创建时，开始[ActionMode](https://developer.android.com/reference/androidx/appcompat/view/ActionMode.html)向用户表示该选择，并提供特定于选择的操作。例如，您可以在[ActionMode](https://developer.android.com/reference/androidx/appcompat/view/ActionMode.html)栏中添加一个删除按钮，并连接该栏中的后退箭头以清除选择。当选择变为空时（如果用户上次清除选择），请不要忘记终止操作模式。

1. 执行任何解释的辅助动作

在事件处理流水线的结尾，库可以确定用户正在尝试通过轻击来激活一个项目，还是正在尝试拖放一个项目或一组选定的项目。通过注册适当的侦听器来对这些解释做出反应。有关更多信息，请参见 [SelectionTracker.Builder](https://developer.android.com/reference/androidx/recyclerview/selection/SelectionTracker.Builder.html)。

1. 使用[SelectionTracker.Builder](https://developer.android.com/reference/androidx/recyclerview/selection/SelectionTracker.Builder.html)组装一切

下面的示例显示如何使用[Long](https://developer.android.com/reference/java/lang/Long.html) selection key将这些片段放在一起 ：

var tracker = SelectionTracker.Builder(  
    "my-selection-id",  
    recyclerView,  
    StableIdKeyProvider(recyclerView),  
    MyDetailsLookup(recyclerView),  
    StorageStrategy.createLongStorage())  
        .withOnItemActivatedListener(myItemActivatedListener)  
        .build()

为了构建一个 SelectionTracker 实例，您的应用程序必须提供相同的 RecyclerView.Adapter，你用来初始化RecyclerView 到 SelectionTracker.Builder。因此，您很可能需要在SelectionTracker 创建实例后将实例创建RecyclerView.Adapter后 注入到您的 实例中 RecyclerView.Adapter 。否则，您将无法从onBindViewHolder() 方法中检查项目的选定状态 。

1. 在活动生命周期 事件中包括选择。

为了跨越保存选择状态 的活动生命周期 事件，您的应用程序必须调用选择跟踪的 onSaveInstanceState() ，并 onRestoreInstanceState() 从活动的方法 onSaveInstanceState()和 onRestoreInstanceState() 分别的方法。您的应用还必须向SelectionTracker.Builder 构造函数提供唯一的选择ID 。此ID是必需的，因为一个活动或一个片段可能具有多个不同的，可选择的列表，所有这些都需要以其保存状态持久保存。

# 额外资源

在[Sunflower](https://github.com/googlesamples/android-sunflower) demo app中使用了RecyclerView。

# 更新Adapter的方式

在data的set方法中调用通知方法



# 填充布局时不要用View.inflate方法

请使用LayoutInflate.from(context).inflate(R.layout.item, parent, false)，这个方法会用到parent参数，虽然后面参数是false，它会把parent当成item的父布局，这样item的根元素的layout打头的元素就会起作用了。