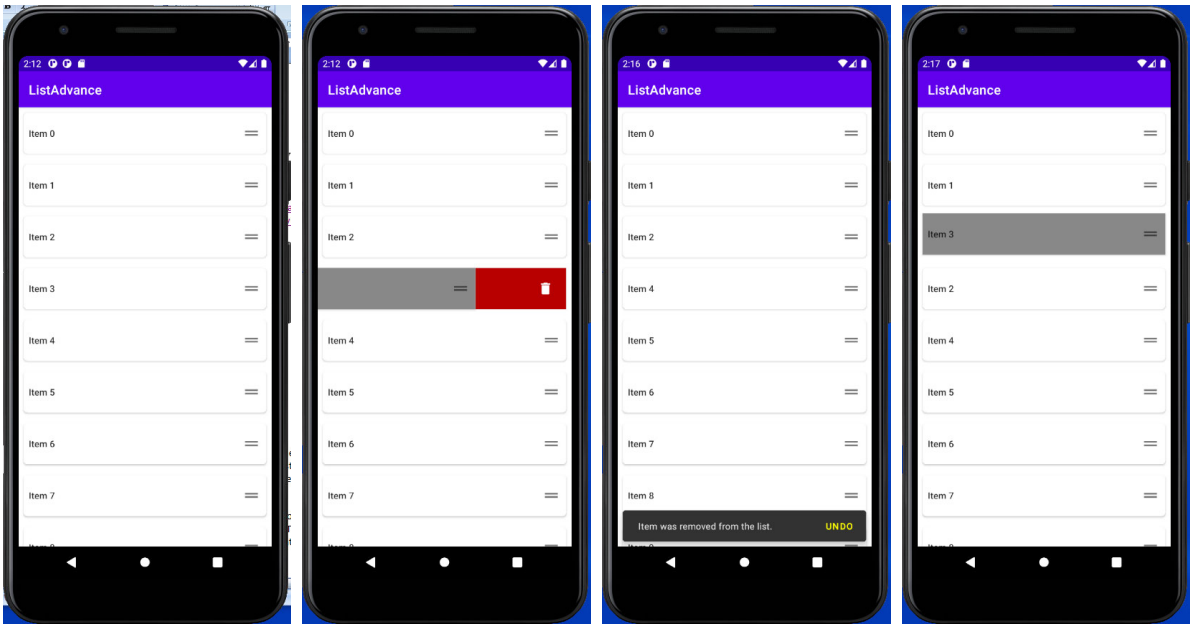
An Advance Custom List using RecyclerView

Reference: https://developer.android.com/guide/topics/ui/layout/recyclerview
https://developer.android.com/guide/topics/ui/layout/cardview
https://www.javatpoint.com/android-recyclerview-list-example
https://github.com/android/views-widgets-samples/tree/main/RecyclerView
https://www.journaldev.com/23208/android-recyclerview-drag-and-drop
https://www.journaldev.com/23164/android-recyclerview-swipe-to-delete-undo

This document describes how to do the following:

- Create a custom list
- Data is stored in a dynamic ArrayList structure
- Click to select row
- Swipe to delete row with red background and delete icon
- Snackbar to undo delete
- Drag handle icon to move row with gray highlight



List | Swipe to delete | Delete Undo | Drag to move

- Edit the activity_main.xml file
- Create the ic_drag.xml file
- Create the ic_delete.xml file
- Create and edit the cardview.xml file
- Here's another version of the complete cardview**.xml** file. This layout looks like the default listview layout.

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
```

```
    android:layout_width="match_parent"
    android:layout_height="48dp"
    android:paddingLeft="26dp"
    android:paddingRight="8dp">
        <TextView
            android:id="@+id/txtTitle"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_centerVertical="true"

android:textAppearance="@style/TextAppearance.Compat.Notification.Title"/>
        <ImageView
            android:id="@+id/dragIcon"
            android:layout_width="30dp"
            android:layout_height="30dp"
            android:layout_alignParentEnd="true"
            android:layout_alignParentRight="true"
            android:layout_centerVertical="true"
            android:src="@drawable/ic_drag"/>
    </RelativeLayout>
```

- Create and edit the StartDragListener.java file
- Edit the MainActivity.java file
- Edit the MyAdapter.java file
- Edit the ItemTouchHelperCallback.java file
- Add a new item to the list
- Adding the Fast Scroll Bar

It is highly recommended that you go through the List Basic document first before doing this advance custom list.

A custom list using RecyclerView

1. Create a new **Empty Activity** project and name it **ListAdvance**.

~~*Edit the app module's build.gradle file*~~

~~2. Add the following dependency~~

```
dependencies {
    implementation "androidx.cardview:cardview:1.0.0"
}
```

Edit the activity_main.xml file

3. Add an **android:id="@+id/constraintLayout"** id attribute in the ConstraintLayout layout block
4. Delete the TextView object
5. Add a **RecyclerView** object
    - On a new line, type **<RV**
    - Press Enter to select **androidx.recyclerview.widget.RecyclerView** from the popup
    - Press Enter to select **match_parent** for the layout_width
    - Press Enter to select **match_parent** for the layout_height
    - Type **/** to close the tag with />

- Add an id attribute by typing **id**, select **id**, select **@+id/**, type **recyclerView** for the id name

6. Here's the complete **activity_main.xml** file

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/constraintLayout"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <androidx.recyclerview.widget.RecyclerView
        android:id="@+id/recyclerView"
        android:layout_width="match_parent"
        android:layout_height="match_parent"/>

</androidx.constraintlayout.widget.ConstraintLayout>
```

Create the ic_drag.xml file

This is the drag icon

7. Create the **ic_drag.xml** file in the **res | drawable** folder.
   - Right-click on the **res | drawable** folder
   - Select **New | Drawable Resource** file
   - Type in **ic_drag** for the file name
   - Replace the file content with the following

```
<vector xmlns:android="http://schemas.android.com/apk/res/android"
    android:width="24dp"
    android:height="24dp"
    android:viewportWidth="24"
    android:viewportHeight="24"
    android:tint="?attr/colorControlNormal">
    <path
        android:fillColor="@android:color/white"
        android:pathData="M20,9H4v2h16V9zM4,15h16v-2H4V15z"/>
</vector>
```

Create the ic_delete.xml file

This is the delete icon

8. Create the **ic_delete.xml** file in the **res | drawable** folder.
   - Right-click on the **res | drawable** folder
   - Select **New | Drawable Resource** file
   - Type in **ic_drag** for the file name
   - Replace the file content with the following

```
<vector xmlns:android="http://schemas.android.com/apk/res/android"
    android:height="24dp"
```

```
    android:tint="#FFFFFF"
    android:viewportHeight="24.0"
    android:viewportWidth="24.0"
    android:width="24dp" >
    <path
        android:fillColor="#FF000000"
        android:pathData="M6,19c0,1.1 0.9,2 2,2h8c1.1,0 2,-0.9 2,-2V7H6v12zM19,4h-
3.5l-1,-1h-5l-1,1H5v2h14V4z"/>
</vector>
```

Create and edit the cardview.xml file

This is the view for each row in the list.

9. Create a new layout resource file name **cardview**. This defines the layout for a list item in the list.
    - Right-click on the **res** | **layout** folder
    - Select **New** | **Layout Resouce** File
    - Type in **cardview** for the name
10. Replace file content with the following
11. Here's the complete **cardview.xml** file

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.cardview.widget.CardView
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:id="@+id/cardView"
    android:layout_width="match_parent"
    android:layout_height="64dp"
    android:layout_margin="8dp"
    app:cardCornerRadius="6dp"
    app:cardElevation="2dp">

    <RelativeLayout
        android:id="@+id/relativeLayout"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:paddingBottom="8dp"
        android:paddingLeft="8dp"
        android:paddingRight="8dp">

        <TextView
            android:id="@+id/txtTitle"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_centerVertical="true"

android:textAppearance="@style/TextAppearance.Compat.Notification.Title"/>

        <ImageView
            android:id="@+id/dragIcon"
            android:layout_width="30dp"
            android:layout_height="30dp"
            android:layout_alignParentEnd="true"
            android:layout_alignParentRight="true"
            android:layout_centerVertical="true"
```

```
            android:src="@drawable/ic_drag"/>

    </RelativeLayout>

</androidx.cardview.widget.CardView>
```

12. Here's another version of the complete **cardview.xml** file. This layout looks like the default listview layout.

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="48dp"
    android:paddingLeft="26dp"
    android:paddingRight="8dp">
        <TextView
            android:id="@+id/txtTitle"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_centerVertical="true"

android:textAppearance="@style/TextAppearance.Compat.Notification.Title"/>
        <ImageView
            android:id="@+id/dragIcon"
            android:layout_width="30dp"
            android:layout_height="30dp"
            android:layout_alignParentEnd="true"
            android:layout_alignParentRight="true"
            android:layout_centerVertical="true"
            android:src="@drawable/ic_drag"/>
    </RelativeLayout>
```

Create and edit the StartDragListener.java file

13. Create a new interface name **StartDragListener**.
    - Right-click on the **java** | **com.example.listadvance** folder
    - Select **New** | **Java Class**
    - Type in **StartDragListener** for the name
    - Select **Interface**
    - Press Enter
14. Replace file content with the following
15. Here's the complete **StartDragListener.java** file

```
package com.example.listadvance;

import androidx.recyclerview.widget.RecyclerView;

public interface StartDragListener {
    void requestDrag(RecyclerView.ViewHolder viewHolder);
}
```

Edit the MainActivity.java file

16. In the **MainActivity** class declaration add **`implements StartDragListener`**
17. `Resolves the red error`
    - Click on the red bulb
    - Select **Implement Methods**
    - Click OK to add the **requestDrag()** method
18. There are two custom classes, **MyAdapter** and **ItemTouchHelperCallback** that we have not yet defined. They will have a red error
19. All of the other red errors (not related to the **MyAdapter** and **ItemTouchHelperCallback** classes) can be resolved by pressing **Alt+Enter** to import the needed library
20. Type in the following code
21. Here's the complete **MainActivity.java** file

```
// https://www.journaldev.com/23164/android-recyclerview-swipe-to-delete-undo
// https://www.journaldev.com/23208/android-recyclerview-drag-and-drop

package com.example.listadvance;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;
import androidx.constraintlayout.widget.ConstraintLayout;
import androidx.coordinatorlayout.widget.CoordinatorLayout;
import androidx.recyclerview.widget.ItemTouchHelper;
import androidx.recyclerview.widget.LinearLayoutManager;
import androidx.recyclerview.widget.RecyclerView;

import android.graphics.Color;
import android.os.Bundle;
import android.view.View;

import com.google.android.material.snackbar.Snackbar;

import java.util.ArrayList;

public class MainActivity extends AppCompatActivity implements StartDragListener {
    //// Basic list stuff
    RecyclerView recyclerView;
    MyAdapter myAdapter;
    ConstraintLayout constraintLayout;
    ArrayList<String> fruits = new ArrayList<>();
    // Delete and move row stuff
    ItemTouchHelper itemTouchHelper;                    // Basic
    //ItemTouchHelperCallback itemTouchHelperCallback;    // Advance with highlight
    ItemTouchHelper.Callback itemTouchHelperCallback;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        recyclerView = findViewById(R.id.recyclerView);
        recyclerView.setLayoutManager(new LinearLayoutManager(this));
```

```
        recyclerView.addItemDecoration(new DividerItemDecoration(this,
        DividerItemDecoration.VERTICAL));   // to add row separator line

        myAdapter = new MyAdapter(this, fruits);
        recyclerView.setAdapter(myAdapter);

        constraintLayout = findViewById(R.id.constraintLayout);   // Advance delete

        fruits.add("Item 0");
        fruits.add("Item 1");
        fruits.add("Item 2");
        fruits.add("Item 3");
        fruits.add("Item 4");
        fruits.add("Item 5");
        fruits.add("Item 6");
        fruits.add("Item 7");
        fruits.add("Item 8");
        fruits.add("Item 9");
        fruits.add("Item 10");
        fruits.add("Item 11");
        fruits.add("Item 12");
        fruits.add("Item 13");
        fruits.add("Item 14");
        fruits.add("Item 15");
        fruits.add("Item 16");
        fruits.add("Item 17");
        fruits.add("Item 18");
        fruits.add("Item 19");
        fruits.add("Item 20");

        itemTouchHelperCallback = new ItemTouchHelperCallback(this, myAdapter) {
            @Override
            public void onSwiped(@NonNull RecyclerView.ViewHolder viewHolder, int
direction) {
                final int position = viewHolder.getAdapterPosition();
                final String item = myAdapter.getData().get(position);
                myAdapter.deleteItem(position);
                // show snackbar to undo delete
                Snackbar snackbar = Snackbar
                        .make(constraintLayout, "Item was removed from the list.",
Snackbar.LENGTH_LONG);
                snackbar.setAction("UNDO", new View.OnClickListener() {
                    @Override
                    public void onClick(View view) {
                        myAdapter.restoreItem(item, position);
                        recyclerView.scrollToPosition(position);
                    }
                });
                snackbar.setActionTextColor(Color.YELLOW);
                snackbar.show();
            }
        };

        // initialize itemTouchHelper using custom itemTouchHelperCallback callback
        itemTouchHelper = new ItemTouchHelper(itemTouchHelperCallback);
```

```
        //// Need this for both Basic and Advance delete and move row stuff
        // attach the itemTouchHelper to the recyclerView
        itemTouchHelper.attachToRecyclerView(recyclerView);

    }   // end onCreate


////////////////////////////////////////////////////////////////////////////////
    //// Advance drag to move row stuff
    @Override
    public void requestDrag(RecyclerView.ViewHolder viewHolder) {
        itemTouchHelper.startDrag(viewHolder);
    }

}
```

22. Resolve the red MyAdapter error
    - Put your cursor on the red error
    - Click on the red bulb
    - Select **Create class MyAdapter**
    - Click OK on the popup window
    - A new **MyAdapter.java** file with the MyAdapter class is created

23. Resolve the red ItemTouchHelperCallback error
    - Put your cursor on the red error
    - Click on the red bulb
    - Select **Create class ItemTouchHelperCallback**
    - Click OK on the popup window
    - A new **ItemTouchHelperCallback.java** file with the ItemTouchHelperCallback class is created

Edit the MyAdapter.java file

24. Have the **MyAdapter** class **`extends RecyclerView.Adapter<MyAdapter.MyViewHolder>`**

25. Resolve the red MyAdapter class error
    - Put your cursor on the red class MyAdapter error
    - Click on the red bulb
    - Select **Implement methods**
    - Click OK on the popup window
    - The three methods, onCreateViewHolder, onBindViewHolder, and getItemCount, are added

26. Resolve the red MyViewHolder error
    - Put your cursor on the red MyViewHolder error
    - Click on the red bulb
    - Select **Create class MyViewHolder**
    - Click OK on the popup window
    - The MyViewHolder class is added

27. Resolve the red MyAdapter.MyViewHolder class error

- Put your cursor on the red MyAdapter.MyViewHolder error
- Click on the red bulb
- Select **Make MyViewHolder extend androidx.recyclerview.widget.RecyclerView.ViewHolder**

28. Resolve the red MyViewHolder class error
    - Put your cursor on the red public class MyViewHolder extends error
    - Click on the red bulb
    - Select **Create constructor matching super**

29. In the **onBindViewHolder** method call **relativeLayout.setOnClickListener** to detect row click

30. In the **onBindViewHolder** method call **dragIcon.setOnTouchListener** to detect drag on the move handle
31. There should be no more errors in both the MainActivity.java and MyAdapter.java files after typing in the MyAdapter.java code
32. Type in the following code
33. Here's the complete **MyAdapter.java** file

```
package com.example.listadvance;

import android.annotation.SuppressLint;
import android.content.Context;
import android.graphics.Color;
import android.view.LayoutInflater;
import android.view.MotionEvent;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ImageView;
import android.widget.RelativeLayout;
import android.widget.TextView;
import android.widget.Toast;

import androidx.annotation.NonNull;
import androidx.recyclerview.widget.RecyclerView;

import java.util.ArrayList;
import java.util.Collections;

public class MyAdapter extends RecyclerView.Adapter<MyAdapter.MyViewHolder> {
    Context context;
    private final StartDragListener startDragListener;  // Advance move on drag
    private ArrayList<String> fruits;


//////////////////////////////////////////////////////////////////////////////
    //// Basic list stuff
    public MyAdapter(MainActivity mainActivity, ArrayList<String> fruits) {
        context = mainActivity;
        startDragListener = mainActivity;   // Advance move on drag
        this.fruits = fruits;
    }
```

```
    @NonNull
    @Override
    public MyAdapter.MyViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int
viewType) {
        View view =
LayoutInflater.from(parent.getContext()).inflate(R.layout.cardview, parent, false);
        return new MyViewHolder(view);
    }

    @SuppressLint("ClickableViewAccessibility") // this is only for getting rid of
warning for the dragIcon onTouch
    // see https://stackoverflow.com/questions/47107105/android-button-has-
setontouchlistener-called-on-it-but-does-not-override-perform
    @Override
    public void onBindViewHolder(@NonNull MyAdapter.MyViewHolder holder, int
position) {
        holder.fruit.setText(fruits.get(position));
        // detect row click Method 2
//        holder.relativeLayout.setOnClickListener(new View.OnClickListener() {
//            @Override
//            public void onClick(View v) {
//                Toast.makeText(v.getContext(), position+": "+fruits.get(position),
Toast.LENGTH_SHORT).show();
//            }
//        });
        // detect drag icon click
        holder.dragIcon.setOnTouchListener(new View.OnTouchListener() {
            @Override
            public boolean onTouch(View v, MotionEvent event) {
                if (event.getAction() == MotionEvent.ACTION_DOWN) {
                    //Toast.makeText(v.getContext(), "drag handle action down",
Toast.LENGTH_SHORT).show();
                    //context.requestDrag(holder);
                    startDragListener.requestDrag(holder);
                }
                return false;
            }
        });
    }

    @Override
    public int getItemCount() {
        return fruits.size();
    }

    public class MyViewHolder extends RecyclerView.ViewHolder implements
View.OnClickListener {
        RelativeLayout relativeLayout;
        View rowView;   // Advance move on drag
        private TextView fruit;
        private ImageView dragIcon;

        public MyViewHolder(@NonNull View itemView) {
            super(itemView);
```

```
            relativeLayout = itemView.findViewById(R.id.relativeLayout);
            rowView = itemView; // Advance move on drag
            fruit = itemView.findViewById(R.id.txtTitle);
            dragIcon = itemView.findViewById(R.id.dragIcon);
            itemView.setOnClickListener(this);
        }

        // detect row click Method 1
        @Override
        public void onClick(View v) {
            int position = getLayoutPosition();
            Toast.makeText(v.getContext(), position+": "+fruits.get(position),
Toast.LENGTH_SHORT).show();
        }
    }

////////////////////////////////////////////////////////////////////////////////
    //// Advance swipe to delete row stuff
    void deleteItem(int position) {
        fruits.remove(position);
        notifyItemRemoved(position); // notify the recyclerView
    }

    public void restoreItem(String item, int position) {
        fruits.add(position, item);
        notifyItemInserted(position);
    }

    public ArrayList<String> getData() {
        return fruits;
    }


////////////////////////////////////////////////////////////////////////////////
    //// Advance drag to move row stuff
    void onRowMoved(int fromPosition, int toPosition) {
        //Collections.swap(Arrays.asList(fruits), position1, position2);
        if (fromPosition < toPosition) {
            for (int i = fromPosition; i < toPosition; i++) {
                Collections.swap(fruits, i, i + 1);
            }
        } else {
            for (int i = fromPosition; i > toPosition; i--) {
                Collections.swap(fruits, i, i - 1);
            }
        }
        notifyItemMoved(fromPosition, toPosition); // notify the recyclerView
    }

    public void onRowSelected(MyViewHolder myViewHolder) {
        myViewHolder.rowView.setBackgroundColor(Color.GRAY);
    }

    public void onRowClear(MyViewHolder myViewHolder) {
        myViewHolder.rowView.setBackgroundColor(Color.WHITE);
```

```
    }

}
```

Edit the ItemTouchHelperCallback.java file

34. Resolve the red **ItemTouchHelperCallback** class error
    - Put your cursor on the red class ItemTouchHelperCallback error
    - Click on the red bulb
    - Select **Implement methods**
    - Click OK on the popup window
    - The three methods, **getMovementFlags**, **onMove**, and **onSwiped**, are added

35. Add the keyword **abstract** to the declaration of the **ItemTouchHelperCallback** class

```
abstract public class ItemTouchHelperCallback extends ItemTouchHelper.Callback {
```

36. Comment out the **onSwiped** method. The onSwiped method is implemented in the MainActivity class. This is what makes this class **abstract**.

37. Return false for the **isLongPressDragEnabled** method to disable the default drag and drop in order to customize the drag only on the drag icon.

38. Here's the complete **ItemTouchHelperCallback.java** file

```
package com.example.listadvance;

import android.content.Context;
import android.graphics.Canvas;
import android.graphics.Color;
import android.graphics.Paint;
import android.graphics.PorterDuff;
import android.graphics.PorterDuffXfermode;
import android.graphics.drawable.ColorDrawable;
import android.graphics.drawable.Drawable;
import android.view.View;

import androidx.annotation.NonNull;
import androidx.annotation.Nullable;
import androidx.core.content.ContextCompat;
import androidx.recyclerview.widget.ItemTouchHelper;
import androidx.recyclerview.widget.RecyclerView;

// Note the extra 'abstract' keyword in the next line to make the class abstract
// See comment for the onSwiped method below
abstract public class ItemTouchHelperCallback extends ItemTouchHelper.Callback {
    //// Basic list stuff
    Context context;
    //// Advance list stuff
    private final MyAdapter myAdapter;

    // for Advance swipe to delete row stuff
```

```
    private Paint mClearPaint;
    private ColorDrawable mBackground;
    private int backgroundColor;
    private Drawable deleteDrawable;
    private int intrinsicWidth;
    private int intrinsicHeight;

    public ItemTouchHelperCallback(MainActivity mainActivity, MyAdapter myAdapter) {
        context = mainActivity;
        //// Advance list stuff
        this.myAdapter = myAdapter;
        // for Advance swipe to delete row stuff
        mBackground = new ColorDrawable();
        backgroundColor = Color.parseColor("#b80f0a");
        mClearPaint = new Paint();
        mClearPaint.setXfermode(new PorterDuffXfermode(PorterDuff.Mode.CLEAR));
        deleteDrawable = ContextCompat.getDrawable(context, R.drawable.ic_delete);
        intrinsicWidth = deleteDrawable.getIntrinsicWidth();
        intrinsicHeight = deleteDrawable.getIntrinsicHeight();
    }

    @Override
    public int getMovementFlags(@NonNull RecyclerView recyclerView, @NonNull
RecyclerView.ViewHolder viewHolder) {
        // in the makeMovementFlags parameters, the first parameter is the movements
for drag to move row,
        // and the second parameter is the movements for swipe to delete row
        return makeMovementFlags(ItemTouchHelper.UP | ItemTouchHelper.DOWN,
ItemTouchHelper.LEFT);
    }

    @Override
    public boolean onMove(@NonNull RecyclerView recyclerView, @NonNull
RecyclerView.ViewHolder viewHolder, @NonNull RecyclerView.ViewHolder target) {
        int from = viewHolder.getAdapterPosition();
        int to = target.getAdapterPosition();
        myAdapter.onRowMoved(from, to);
        return true;    // must return true to allow drag and drop reordering
    }

//    This IntemTouchHelperCallback class is abstract because the onSwiped method is
implemented in the MainActivity class
//    Note also the 'abstract' keyword in front of the class declaration
//    @Override
//    public void onSwiped(@NonNull RecyclerView.ViewHolder viewHolder, int
direction) {
//
//    }

////////////////////////////////////////////////////////////////////////////////
    //// Advance swipe to delete row stuff
    @Override
    // draw red color background when swipe to delete
    public void onChildDraw(@NonNull Canvas c, @NonNull RecyclerView recyclerView,
@NonNull RecyclerView.ViewHolder viewHolder, float dX, float dY, int actionState,
```

```
boolean isCurrentlyActive) {
        View itemView = viewHolder.itemView;
        int itemHeight = itemView.getHeight();

        boolean isCancelled = dX == 0 && !isCurrentlyActive;

        if (isCancelled) {
            clearCanvas(c, itemView.getRight() + dX, (float) itemView.getTop(),
(float) itemView.getRight(), (float) itemView.getBottom());
            super.onChildDraw(c, recyclerView, viewHolder, dX, dY, actionState,
isCurrentlyActive);
            return;
        }

        mBackground.setColor(backgroundColor);
        mBackground.setBounds(itemView.getRight() + (int) dX, itemView.getTop(),
itemView.getRight(), itemView.getBottom());
        mBackground.draw(c);

        int deleteIconTop = itemView.getTop() + (itemHeight - intrinsicHeight) / 2;
        int deleteIconMargin = (itemHeight - intrinsicHeight) / 2;
        int deleteIconLeft = itemView.getRight() - deleteIconMargin - intrinsicWidth;
        int deleteIconRight = itemView.getRight() - deleteIconMargin;
        int deleteIconBottom = deleteIconTop + intrinsicHeight;

        deleteDrawable.setBounds(deleteIconLeft, deleteIconTop, deleteIconRight,
deleteIconBottom);
        deleteDrawable.draw(c);

        super.onChildDraw(c, recyclerView, viewHolder, dX, dY, actionState,
isCurrentlyActive);
    }

    private void clearCanvas(Canvas c, Float left, Float top, Float right, Float
bottom) {
        c.drawRect(left, top, right, bottom, mClearPaint);
    }

    @Override
    public float getSwipeThreshold(@NonNull RecyclerView.ViewHolder viewHolder) {
        return 0.7f;
    }


////////////////////////////////////////////////////////////////////////////////
    //// Advance drag to move row stuff
    @Override
    // return false to customize the drag only on the drag icon
    public boolean isLongPressDragEnabled() {
        return false;
    }

    @Override
    public void onSelectedChanged(@Nullable RecyclerView.ViewHolder viewHolder, int
actionState) {
```

```
        if (actionState != ItemTouchHelper.ACTION_STATE_IDLE) {
            if (viewHolder instanceof MyAdapter.MyViewHolder) {
                MyAdapter.MyViewHolder myViewHolder =
                        (MyAdapter.MyViewHolder) viewHolder;
                myAdapter.onRowSelected(myViewHolder);
            }
        }
        super.onSelectedChanged(viewHolder, actionState);
    }


    @Override
    public void clearView(@NonNull RecyclerView recyclerView, @NonNull
RecyclerView.ViewHolder viewHolder) {
        super.clearView(recyclerView, viewHolder);
        if (viewHolder instanceof MyAdapter.MyViewHolder) {
            MyAdapter.MyViewHolder myViewHolder =
                    (MyAdapter.MyViewHolder) viewHolder;
            myAdapter.onRowClear(myViewHolder);
        }

    }
}
```

Run it

39. That's it. Run the app on an actual device.
    - Click on a row to select the row.
    - Drag a row to scroll the list.
    - Drag a row handle icon up or down to move the row.
    - Swip left on a row to delete the row. Click on Undo in the Snackbar when it pops up to undo the delete.

Add a new item to the list

Edit the activity_main.xml file

40. Add an EditText object to the layout. This allows the user to enter a fruit name.
41. Add a Button object to the layout. Clicking this button will add the fruit name in the EditText to the list.
42. Here's the complete **activity_main.xml** file

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/constraintLayout"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <EditText
        android:id="@+id/fruitName"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:hint="item"
        android:gravity="fill_horizontal"
        android:layout_marginTop="10dp"
        android:layout_marginLeft="10dp"
        android:layout_marginRight="10dp"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintEnd_toStartOf="@+id/addButton"
        android:inputType="text" />

    <Button
        android:id="@+id/addButton"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="add"
        android:layout_marginRight="10dp"
        android:layout_marginEnd="10dp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintBottom_toBottomOf="@+id/fruitName" />

    <androidx.recyclerview.widget.RecyclerView
        android:id="@+id/recyclerView"
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:paddingTop="20dp"
        app:layout_constraintTop_toBottomOf="@+id/fruitName"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintBottom_toBottomOf="parent" />
```

```
</androidx.constraintlayout.widget.ConstraintLayout>
```

Edit the MainActivity.java file

43. Write the code to handle the Add button click by adding the following code in the **onCreate** method.

```
// Handle the add button click
Button button = findViewById(R.id.addButton);
button.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        Log.d("myTag", "Add button clicked");
        EditText newFruit = findViewById(R.id.fruitName);
        String fruit = newFruit.getText().toString();
        newFruit.setText("");
        myAdapter.addItem(fruit);
    }
});
```

Edit the MyAdapter.java file

44. Add the **addItem** method in the MyAdapter class with the code to actually add the new item to the list.

```
void addItem(String fruit) {
    if (!fruit.isEmpty()) {
        fruits.add(fruit);
        notifyDataSetChanged();
        Toast.makeText(this.context, fruit + " added", Toast.LENGTH_SHORT).show();
    }
}



void addItem(String fruit) {
    Log.d("myTag", "addItem "+fruit);
    ArrayList<String> tmp = new ArrayList<String>(Arrays.asList(fruits));
    tmp.add(fruit);
    fruits = new String[tmp.size()];
    tmp.toArray(fruits);
    notifyDataSetChanged();
    // notifyItemInserted(tmp.size());
}
```

Run it

45. That's it. Run the app on an actual device.
    Type a fruit in the EditText box and press the Add button to add the item in the list.

Adding the Fast Scroll Bar

Reference: https://github.com/quiph/RecyclerView-FastScroller

Edit the activity_main.xml file

46. Set the **fastScrollEnabled** flag to true in the RecyclerView, and add the four drawable files.

```
<androidx.recyclerview.widget.RecyclerView
    android:id="@+id/recyclerView"
    android:layout_width="match_parent"
    android:layout_height="match_parent"

    android:scrollbarSize="18dp"
    app:fastScrollHorizontalThumbDrawable="@drawable/thumb_drawable"
    app:fastScrollHorizontalTrackDrawable="@drawable/line_drawable"
    app:fastScrollVerticalThumbDrawable="@drawable/thumb_drawable"
    app:fastScrollVerticalTrackDrawable="@drawable/line_drawable"
    app:fastScrollEnabled="true"/>
```

47. Add the drawable file **thumb.xml** with the following content

```
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android="http://schemas.android.com/apk/res/android"
    android:shape="rectangle">

    <corners
        android:radius="50dp" />

    <padding
        android:paddingLeft="22dp"
        android:paddingRight="22dp" />

    <solid android:color="@color/teal_700" />

</shape>
```

48. Add the drawable file **thumb_drawable.xml** with the following content

```
<selector xmlns:android="http://schemas.android.com/apk/res/android">
    <item
        android:state_pressed="true"
        android:drawable="@drawable/thumb"/>

    <item
        android:drawable="@drawable/thumb"/>
</selector>
```

49. Add the drawable file **line.xml** with the following content

```
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android="http://schemas.android.com/apk/res/android"
```

```
    android:shape="rectangle">

    <solid android:color="@android:color/darker_gray" />

    <padding
        android:top="10dp"
        android:left="10dp"
        android:right="10dp"
        android:bottom="10dp"/>
</shape>
```

50. Add the drawable file **line_drawable.xml** with the following content

```
<?xml version="1.0" encoding="utf-8"?>
<selector xmlns:android="http://schemas.android.com/apk/res/android">
    <item
        android:state_pressed="true"
        android:drawable="@drawable/line"/>

    <item
        android:drawable="@drawable/line"/>
</selector>
```

Adding the Section Indexer to the Fast Scroll Bar

The following doesn't work for a RecyclerView. It only works for a ListView. See the List using ListView document.

Edit the java file

51. Create a new class name **SectionIndexerAdapter** that extends **ArrayAdapter<String>** and implements the **SectionIndexer**

```
private class SectionIndexerAdapter extends ArrayAdapter<String> implements
    SectionIndexer {
```

52. Inside this **SectionIndexerAdapter** class declare a String array containing the section names that you want

```
    String[] sections = new String[] {
    "A","B","C","D","E","F","G","H","I","J","K","L","M","N","O","P","Q","R","S"
    ,"T","U","V","W","X","Y","Z"};
```

53. Create the constructor and override three methods that are needed inside the **SectionIndexerAdapter** class

```
public SectionIndexerAdapter(Context context, int list_item, String[] items) {
    super(context, list_item, items);
}


@Override
public Object[] getSections() {
    return sections;
}

@Override
public int getPositionForSection(int sectionIndex) {
}

@Override
public int getSectionForPosition(int position) {
}
```

54. The **getSections()** method simply returns the String array containing the section names

```
@Override
public Object[] getSections() {
    return sections;
}
```

55. The **getPositionForSection(int sectionIndex)** method returns the starting position in the data list for the given section index. The data list is the array containing the list of items. The starting position is the index of this array where you want the given section to start.

```
@Override
public int getPositionForSection(int sectionIndex) {
    if (sectionIndex == 0)      // section 0 (A) starts at index 0
        return 0;
    else if (sectionIndex == 1) // section 1 (B) starts at index 54
        return 54;
    else if (sectionIndex == 2) // section 2 (C) starts at index 79
        return 79;
    else if (sectionIndex == 3) // section 3 (D) starts at index 122
        return 122;
    else if (sectionIndex == 4) // section 4 (E) starts at index 140
        return 140;
    else if (sectionIndex == 5) // section 5 (F) starts at index
        ...
    }
}
```

56. The **getSectionForPosition(int position)** method returns the section number for the given position in the data list

```
@Override
public int getSectionForPosition(int position) {
    if(position < 54)         // positions 0 to 53 are in section 0 (A)
        return 0;
    else if(position < 79)    // positions 54 to 78 are in section 1 (B)
        return 1;
    else if(position < 122)    // positions 79 to 121 are in section 2 (C)
        return 2;
    else if(position < 140)    // positions 122 to 139 are in section 3 (D)
        return 3;
    else if(position < 148)    // positions 140 to 147 are in section 4 (E)
        return 4;
    else if(position < 176)    // positions 148 to 175 are in section 5 (F)
        ...
     }
}
```

57. Finally, create a new instance of the **SectionIndexerAdapter** in your code passing to it the context, layout, and the data array of your list

```
listView = findViewById(R.id.listView);
listView.setFastScrollEnabled(true);
listView.setAdapter(adapter);
SectionIndexerAdapter adapter = new SectionIndexerAdapter(this,
R.layout.list_item, items);
```