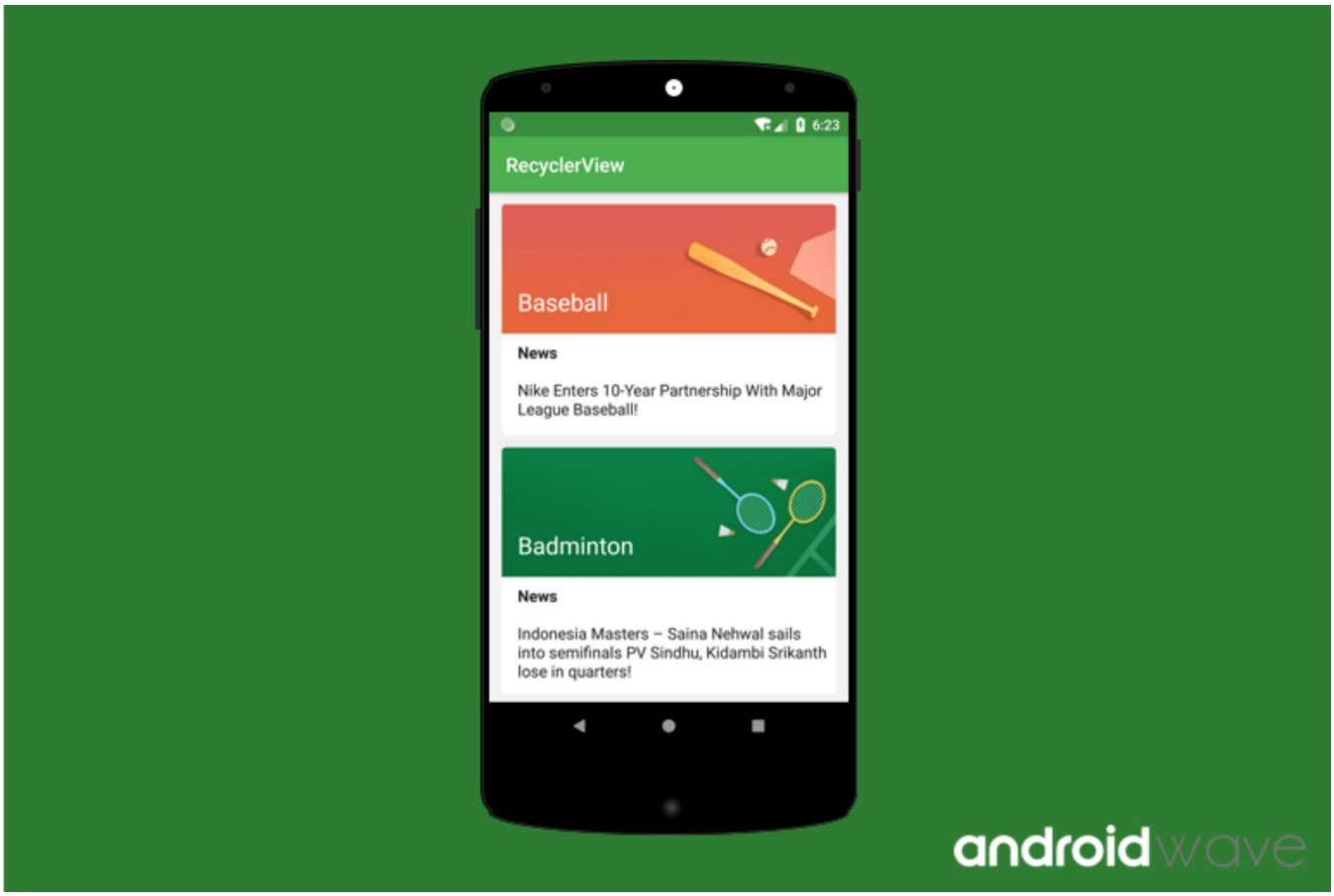
UI/UX

RecyclerView in Android Example Best Practices

BY MORRIS - JANUARY 27, 2019



Introduction

RecyclerView is the most powerful, advanced and flexible version of ListView. In RecyclerView is used several different components work together to display data. Using the standard layout managers (such as LinearLayoutManager or GridLayoutManager), you can create the list and grid both very easily.

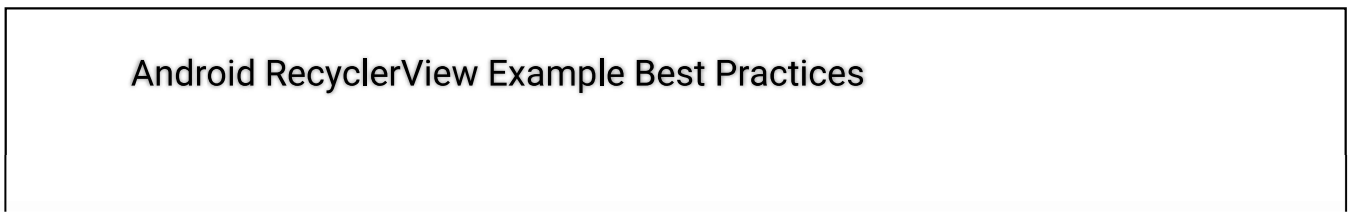
RecyclerView model is a lot of optimized and powerful. As the suggest RecyclerView is used the reusable cell when scrolling up and down. Suppose the view is displaying list positions 0 to 9 the RecyclerView has bind those view holders when user scroll down RecyclerView reuse the cell and bind with holder. One more improvement in RecyclerView is that allows us to set layout managers at runtime so we can manage to scroll, such as vertical or horizontal and manage view in such as list, grid, and Staggered layout.

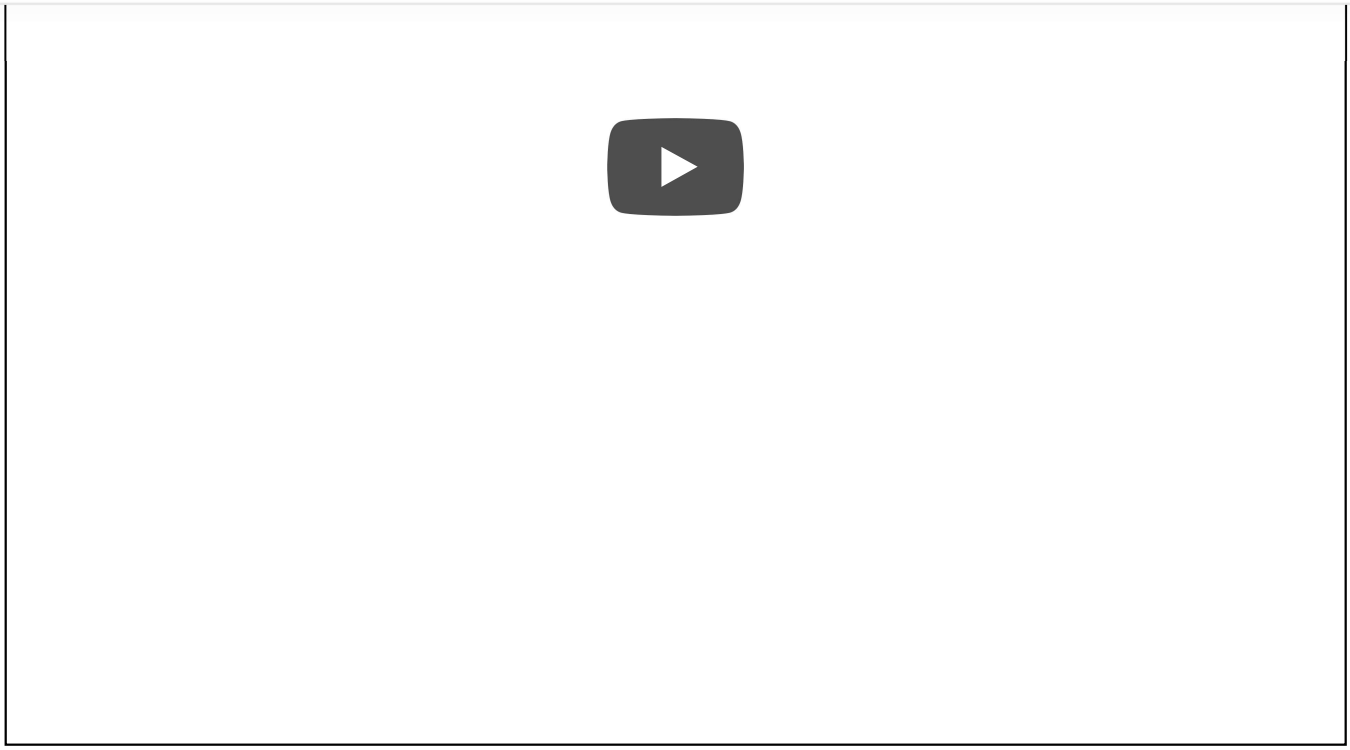
Prerequisite

You need to have knowledge of following classes that used in RecyclerView implementation

- **Adapter –** Provides the data model and responsible for rendering the views for the individual cell
- **ViewHolder –** Contains instances for all views that are filled by the data of the entry
- **LayoutManager –** Allows us to set the LayoutManagers at runtime. eg. LinearLayoutManager, StaggeredLayoutManager, GridLayoutManager

RecyclerView Android Example (Demo App)





Add the support library

Open the **build.gradle** file in app module and add followings dependency

```
dependencies {
  implementation fileTree(dir: 'libs', include: ['*.jar'])
  implementation 'androidx.appcompat:appcompat:1.1.0'
  implementation 'androidx.constraintlayout:constraintlayout:1.1.3'
  testImplementation 'junit:junit:4.12'
  androidTestImplementation 'androidx.test.ext:junit:1.1.1'
  androidTestImplementation 'androidx.test.espresso:espresso-core:3.2.0'

  implementation 'androidx.cardview:cardview:1.0.0'
  implementation 'androidx.recyclerview:recyclerview:1.2.0-alpha01'
  implementation 'com.google.android.material:material:1.2.0-alpha03'
  implementation 'com.google.code.gson:gson:2.8.5'
  // ButterKnife for view binding
  implementation 'com.jakewharton:butterknife:10.1.0'
  annotationProcessor 'com.jakewharton:butterknife-compiler:10.1.0'
  // Glide for image loading
  implementation 'com.github.bumptech.glide:glide:4.10.0'
  annotationProcessor 'com.github.bumptech.glide:compiler:4.10.0'
}
```

```
}
```

Open the color.xml in from res=>value folder and add color values

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
  <color name="colorPrimary">#008577</color>
  <color name="colorPrimaryDark">#00574B</color>
  <color name="colorAccent">#FF9800</color>
  <color name="colorPrimaryText">#212121</color>
  <color name="colorSecondaryText">#757575</color>
  <color name="colorIcons">#FFFFFF</color>
  <color name="colorDivider">#BDBDBD</color>
  <color name="black_effective">#2a2a2a</color>
</resources>
```

Go to string.xml from res => values=> string.xml and add some string values

Open string.xml add string values and demo content arrays for showing in the list.

```
<resources>
  <string name="app_name">RecyclerView Example</string>

  <string name="title_placeholder">Title</string>
  <string name="news_placeholder">News</string>
  <string name="sports_info_placeholder">Here is some news</string>
  <string name="empty_screen">We do not have anything\nhere right now. Please
  <string name="btn_retry">Retry</string>

  <string-array name="sports_images">
    <item>https://androidwave.com/media/images/img_baseball.jpg</item>
    <item>https://androidwave.com/media/images/img_badminton.jpg</item>
    <item>https://androidwave.com/media/images/img_basketball.jpg</item>
    <item>https://androidwave.com/media/images/img_bowling.jpg</item>
    <item>https://androidwave.com/media/images/img_cycling.jpg</item>
    <item>https://androidwave.com/media/images/img_golf.jpg</item>
    <item>https://androidwave.com/media/images/img_running.jpg</item>
    <item>https://androidwave.com/media/images/img_soccer.jpg</item>
    <item>https://androidwave.com/media/images/img_swimming.jpg</item>
    <item>https://androidwave.com/media/images/img_tabletennis.jpg</item>
    <item>https://androidwave.com/media/images/img_tennis.jpg</item>
  </string-array>
  <string-array name="sports_titles">
    <item>Baseball</item>
    <item>Badminton</item>
    <item>Basketball</item>
    <item>Bowling</item>
```

```
    </string-array>

    <string-array name="sports_info">
      <item>Nike Enters 10-Year Partnership With Major League Baseball! </item>
      <item>Indonesia Masters - Saina Nehwal sails into semifinals PV Sindhu, K
      <item>College Basketball 2018-19 Player of the Year Power Rankings Everyo
      <item>Local bowler lands 132 perfect games, shares his secret for success
      <item>Sierra solos to win at womens Cadel Evans Great Ocean Road Race!</i
      <item>Tiger Woods Friday live blog: Woods overcomes mid-round hiccup to m
      <item>The 5km fun run was organised by Ironman England as part of its ann
      <item>Soccer insider notes: Liverpool not the only Premier League club ey
      <item>Chloe Grimme, a USA Swimming Scholastic All American from Land O La
      <item>Table Tennis Federation Revokes Soumyajit Ghoshs Suspension, 9 Mont
      <item>2019 Australian Open odds, predictions for mens finals!</item>
    </string-array>
</resources>
```

Create a class with name BaseViewHolder

Go to src and create a new Java class with **BaseViewHolder** which extends **RecyclerView.ViewHolder. We will use BaseViewHolder class instance of RecyclerView holder class, It improves your code quality and eliminates** boilerplate code.

```
package com.recyclerviewexample;

import android.view.View;
import androidx.recyclerview.widget.RecyclerView;

public abstract class BaseViewHolder extends RecyclerView.ViewHolder {

  private int mCurrentPosition;

  public BaseViewHolder(View itemView) {
    super(itemView);
  }

  protected abstract void clear();

  public void onBind(int position) {
    mCurrentPosition = position;
    clear();
  }

  public int getCurrentPosition() {
    return mCurrentPosition;
```

Let's start writing an adapter class withhold the list item and render on RecyclerView in Android.

Let's Create a Model Class with named Sport.java

Create a new **POJO** class, and define some entity such as title, subtitle, and news discerption with getter setter.

```
package com.recyclerviewexample;

import com.google.gson.annotations.SerializedName;

public class Sport {

  @SerializedName("imageUrl")
  private String mImageUrl;
  @SerializedName("info")
  private String mInfo;
  @SerializedName("subTitle")
  private String mSubTitle;
  @SerializedName("title")
  private String mTitle;

  public Sport(String mImageUrl, String mInfo, String mSubTitle, String mTitle
    this.mImageUrl = mImageUrl;
    this.mInfo = mInfo;
    this.mSubTitle = mSubTitle;
    this.mTitle = mTitle;
  }

  public String getImageUrl() {
    return mImageUrl;
  }

  public void setImageUrl(String imageUrl) {
    mImageUrl = imageUrl;
  }

  public String getInfo() {
    return mInfo;
  }

  public void setInfo(String info) {
    mInfo = info;
  }

  public String getSubTitle() {
    return mSubTitle;
```

```
  public String getTitle() {
    return mTitle;
  }

  public void setTitle(String title) {
    mTitle = title;
  }
}
```

Create a xml layout with *list_item.xml*

Just go to res directory and create an xml layout for displaying a single row of RecyclerView. Which contains some ImageView, TextView along CardView.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:card_view="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_margin="@dimen/_4dp"
    android:background="@color/colorIcons"
    >

  <androidx.cardview.widget.CardView
      android:id="@+id/card_view"
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:layout_gravity="center"
      android:elevation="3dp"
      android:padding="@dimen/card_view_margin"
      card_view:cardCornerRadius="@dimen/card_radius"
      >

    <RelativeLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        >

      <ImageView
          android:id="@+id/thumbnail"
          android:layout_width="match_parent"
          android:layout_height="130dp"
          android:background="?attr/selectableItemBackgroundBorderless"
          android:scaleType="fitXY"
          />
```

```
            android:layout_alignParentStart="true"
            android:layout_marginStart="@dimen/_8dp"
            android:layout_marginBottom="8dp"
            android:padding="@dimen/_8dp"
            android:text="@string/title_placeholder"
            android:textColor="@color/colorIcons"
            />

        <TextView
            android:id="@+id/newsTitle"
            style="@style/TextAppearance.AppCompat.Subhead"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_below="@+id/thumbnail"
            android:layout_alignParentStart="true"
            android:layout_marginStart="@dimen/_8dp"
            android:padding="@dimen/_8dp"
            android:text="@string/news_placeholder"
            android:textStyle="bold"
            />

        <TextView
            android:id="@+id/newsInfo"
            style="@style/TextAppearance.AppCompat.Subhead"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_below="@+id/newsTitle"
            android:layout_alignParentStart="true"
            android:layout_marginStart="@dimen/_8dp"
            android:layout_marginBottom="@dimen/_8dp"
            android:padding="@dimen/_8dp"
            android:text="@string/sports_info_placeholder"
            android:layout_alignParentLeft="true"
            android:layout_marginLeft="@dimen/_8dp"
            />

    </RelativeLayout>

  </androidx.cardview.widget.CardView>

</LinearLayout>
```

Create an XML layout that renders at the time when no item added in the list. so create XML class with named is *item_empty_view.xml*

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

```
        android:id="@+id/tv_message"
        style="@style/TextAppearance.AppCompat.Headline"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:layout_marginEnd="8dp"
        android:layout_marginBottom="8dp"
        android:gravity="center"
        android:text="@string/empty_screen"
        android:textColor="@color/colorSecondaryText"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        />

    <Button
        android:id="@+id/buttonRetry"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="8dp"
        android:layout_marginTop="16dp"
        android:layout_marginEnd="8dp"
        android:background="@color/colorAccent"
        android:text="@string/btn_retry"
        android:textColor="@color/colorIcons"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/tv_message"
        />


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

Create a new java class with named is *SportAdapter*.java

In Java source folder create a **SportAdapter.java** file which extend RecyclerView.Adapter<BaseViewHolder> , In Adapter class ***onCreateViewHolder()*** is responsible to inflate ***list_tem****.xml* and ***item_empty_view****.xml* of each row and ***onBindViewHolder()*** is bind the data of each row

```
package com.recyclerviewexample;

import android.content.Intent;
import android.net.Uri;
```

```
import androidx.recyclerview.widget.RecyclerView;
import butterknife.BindView;
import butterknife.ButterKnife;
import com.bumptech.glide.Glide;
import java.util.List;

public class SportAdapter extends RecyclerView.Adapter<BaseViewHolder> {
  private static final String TAG = "SportAdapter";
  public static final int VIEW_TYPE_EMPTY = 0;
  public static final int VIEW_TYPE_NORMAL = 1;

  private Callback mCallback;
  private List<Sport> mSportList;

  public SportAdapter(List<Sport> sportList) {
    mSportList = sportList;
  }

  public void setCallback(Callback callback) {
    mCallback = callback;
  }

  @Override
  public void onBindViewHolder(BaseViewHolder holder, int position) {
    holder.onBind(position);
  }

  @Override
  public BaseViewHolder onCreateViewHolder(ViewGroup parent, int viewType) {

    switch (viewType) {
      case VIEW_TYPE_NORMAL:
        return new ViewHolder(
            LayoutInflater.from(parent.getContext()).inflate(R.layout.list_it
      case VIEW_TYPE_EMPTY:
      default:
        return new EmptyViewHolder(
            LayoutInflater.from(parent.getContext())
                .inflate(R.layout.item_empty_view, parent, false));
    }
  }

  @Override
  public int getItemViewType(int position) {
    if (mSportList != null && mSportList.size() > 0) {
      return VIEW_TYPE_NORMAL;
    } else {
      return VIEW_TYPE_EMPTY;
    }
  }
```

```
    }
  }

  public void addItems(List<Sport> sportList) {
    mSportList.addAll(sportList);
    notifyDataSetChanged();
  }

  public interface Callback {
    void onEmptyViewRetryClick();
  }

  public class ViewHolder extends BaseViewHolder {

    @BindView(R.id.thumbnail)
    ImageView coverImageView;

    @BindView(R.id.title)
    TextView titleTextView;

    @BindView(R.id.newsTitle)
    TextView newsTextView;

    @BindView(R.id.newsInfo)
    TextView infoTextView;

    public ViewHolder(View itemView) {
      super(itemView);
      ButterKnife.bind(this, itemView);
    }

    protected void clear() {
      coverImageView.setImageDrawable(null);
      titleTextView.setText("");
      newsTextView.setText("");
      infoTextView.setText("");
    }

    public void onBind(int position) {
      super.onBind(position);

      final Sport mSport = mSportList.get(position);

      if (mSport.getImageUrl() != null) {
        Glide.with(itemView.getContext())
            .load(mSport.getImageUrl())
            .into(coverImageView);
      }

      if (mSport.getTitle() != null) {
        titleTextView.setText(mSport.getTitle());
```

```
        if (mSport.getInfo() != null) {
          infoTextView.setText(mSport.getInfo());
        }

        itemView.setOnClickListener(v -> {
          if (mSport.getImageUrl() != null) {
            try {
              Intent intent = new Intent();
              intent.setAction(Intent.ACTION_VIEW);
              intent.addCategory(Intent.CATEGORY_BROWSABLE);
              intent.setData(Uri.parse(mSport.getImageUrl()));
              itemView.getContext().startActivity(intent);
            } catch (Exception e) {
              Log.e(TAG, "onClick: Image url is not correct");
            }
          }
        });
      }
    }

    public class EmptyViewHolder extends BaseViewHolder {

      @BindView(R.id.tv_message)
      TextView messageTextView;
      @BindView(R.id.buttonRetry)
      TextView buttonRetry;

      EmptyViewHolder(View itemView) {
        super(itemView);
        ButterKnife.bind(this, itemView);
        buttonRetry.setOnClickListener(v -> mCallback.onEmptyViewRetryClick());
      }

      @Override
      protected void clear() {

      }
    }
}
```

In Adapter class you are seeing a callback interface which responsible to retry to call API or method in case network failure and server failure. So call back method will call on click of Retr button

```
buttonRetry.setOnClickListener(v -> mCallback.onEmptyViewRetryClick());
```

```
<?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:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#f2f2f2"
    tools:context=".MainActivity"
    >

  <androidx.recyclerview.widget.RecyclerView
      android:id="@+id/mRecyclerView"
      android:layout_width="0dp"
      android:layout_height="0dp"
      android:layout_marginStart="8dp"
      android:layout_marginTop="8dp"
      android:layout_marginEnd="8dp"
      android:layout_marginBottom="8dp"
      app:layout_constraintBottom_toBottomOf="parent"
      app:layout_constraintEnd_toEndOf="parent"
      app:layout_constraintStart_toStartOf="parent"
      app:layout_constraintTop_toTopOf="parent"
      />

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

Open the MainActivity and add below code

Basically following things we have to do in this class

- Bind view by id using ***ButterKnife***.
- Setup LinearLayoutManager and DefaultItemAnimator on RecyclerView.
- Prepare demo content (In your you can fetch date from the remote server using retrofit).
- Set the Adapter on ***RecyclerView***
- Implement Callback methods and call Retry method in case of no item in list

```
package com.recyclerviewexample;

import android.graphics.drawable.Drawable;
```

```
import androidx.recyclerview.widget.RecyclerView;
import butterknife.BindView;
import butterknife.ButterKnife;
import com.recyclerviewexample.utils.CommonUtils;
import com.recyclerviewexample.utils.DividerItemDecoration;
import java.util.ArrayList;

public class MainActivity extends AppCompatActivity implements SportAdapter.Ca

  @BindView(R.id.mRecyclerView)
  RecyclerView mRecyclerView;
  SportAdapter mSportAdapter;

  LinearLayoutManager mLayoutManager;

  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    ButterKnife.bind(this);
    setUp();
  }

  private void setUp() {
    mLayoutManager = new LinearLayoutManager(this);
    mLayoutManager.setOrientation(RecyclerView.VERTICAL);
    mRecyclerView.setLayoutManager(mLayoutManager);
    mRecyclerView.setItemAnimator(new DefaultItemAnimator());
    Drawable dividerDrawable = ContextCompat.getDrawable(this, R.drawable.divi
    mRecyclerView.addItemDecoration(new DividerItemDecoration(dividerDrawable
    mSportAdapter = new SportAdapter(new ArrayList<>());

    prepareDemoContent();
  }

  private void prepareDemoContent() {
    CommonUtils.showLoading(MainActivity.this);
    new Handler().postDelayed(() -> {
      //prepare data and show loading
      CommonUtils.hideLoading();
      ArrayList<Sport> mSports = new ArrayList<>();
      String[] sportsList = getResources().getStringArray(R.array.sports_title
      String[] sportsInfo = getResources().getStringArray(R.array.sports_info
      String[] sportsImage = getResources().getStringArray(R.array.sports_ima
      for (int i = 0; i < sportsList.length; i++) {
        mSports.add(new Sport(sportsImage[i], sportsInfo[i], "News", sportsLis
      }
      mSportAdapter.addItems(mSports);
      mRecyclerView.setAdapter(mSportAdapter);
    }, 2000);
  }
```

Add Divider DividerItemDecoration

You are seeing I have added divider between to cell by using DividerItemDecoration. You can manage color and space easily by using below code

Create java class with named DividerItemDecoration which extends RecyclerView.ItemDecoration

```
package com.recyclerviewexample.utils;

import android.graphics.Canvas;
import android.graphics.Rect;
import android.graphics.drawable.Drawable;
import android.view.View;
import androidx.recyclerview.widget.LinearLayoutManager;
import androidx.recyclerview.widget.RecyclerView;

public class DividerItemDecoration extends RecyclerView.ItemDecoration {


  private Drawable mDivider;
  private int mOrientation;

  /**
   * Sole constructor. Takes in a {@link Drawable} to be used as the interior
   * divider.
   *
   * @param divider A divider {@code Drawable} to be drawn on the RecyclerVie
   */
  public DividerItemDecoration(Drawable divider) {
    mDivider = divider;
  }

  /**
   * Draws horizontal or vertical dividers onto the parent RecyclerView.
   *
   * @param canvas The {@link Canvas} onto which dividers will be drawn
   * @param parent The RecyclerView onto which dividers are being added
   * @param state The current RecyclerView.State of the RecyclerView
   */
  @Override
  public void onDraw(Canvas canvas, RecyclerView parent, RecyclerView.State s
    if (mOrientation == LinearLayoutManager.HORIZONTAL) {
      drawHorizontalDividers(canvas, parent);
```

```
     * Determines the size and location of offsets between items in the parent
     * RecyclerView.
     *
     * @param outRect The {@link Rect} of offsets to be added around the child
     * view
     * @param view The child view to be decorated with an offset
     * @param parent The RecyclerView onto which dividers are being added
     * @param state The current RecyclerView.State of the RecyclerView
     */
    @Override
    public void getItemOffsets(Rect outRect, View view, RecyclerView parent,
        RecyclerView.State state) {
      super.getItemOffsets(outRect, view, parent, state);

      if (parent.getChildAdapterPosition(view) == 0) {
        return;
      }

      mOrientation = ((LinearLayoutManager) parent.getLayoutManager()).getOrien
      if (mOrientation == LinearLayoutManager.HORIZONTAL) {
        outRect.left = mDivider.getIntrinsicWidth();
      } else if (mOrientation == LinearLayoutManager.VERTICAL) {
        outRect.top = mDivider.getIntrinsicHeight();
      }
    }

    /**
     * Adds dividers to a RecyclerView with a LinearLayoutManager or its
     * subclass oriented horizontally.
     *
     * @param canvas The {@link Canvas} onto which horizontal dividers will be
     * drawn
     * @param parent The RecyclerView onto which horizontal dividers are being
     * added
     */
    private void drawHorizontalDividers(Canvas canvas, RecyclerView parent) {
      int parentTop = parent.getPaddingTop();
      int parentBottom = parent.getHeight() - parent.getPaddingBottom();

      int childCount = parent.getChildCount();
      for (int i = 0; i < childCount - 1; i++) {
        View child = parent.getChildAt(i);

        RecyclerView.LayoutParams params = (RecyclerView.LayoutParams) child.ge

        int parentLeft = child.getRight() + params.rightMargin;
        int parentRight = parentLeft + mDivider.getIntrinsicWidth();

        mDivider.setBounds(parentLeft, parentTop, parentRight, parentBottom);
        mDivider.draw(canvas);
      }
```

```
     * @param canvas The {@link Canvas} onto which vertical dividers will be
     * drawn
     * @param parent The RecyclerView onto which vertical dividers are being
     * added
     */
    private void drawVerticalDividers(Canvas canvas, RecyclerView parent) {
        int parentLeft = parent.getPaddingLeft();
        int parentRight = parent.getWidth() - parent.getPaddingRight();

        int childCount = parent.getChildCount();
        for (int i = 0; i < childCount - 1; i++) {
            View child = parent.getChildAt(i);

            RecyclerView.LayoutParams params = (RecyclerView.LayoutParams) child.ge

            int parentTop = child.getBottom() + params.bottomMargin;
            int parentBottom = parentTop + mDivider.getIntrinsicHeight();

            mDivider.setBounds(parentLeft, parentTop, parentRight, parentBottom);
            mDivider.draw(canvas);
        }
    }
}
```

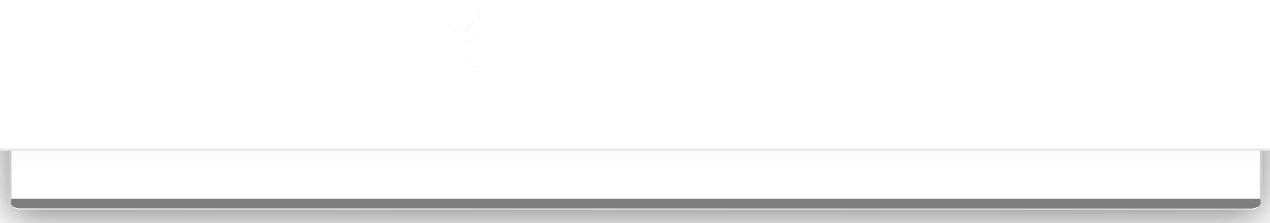
Create a drawable class and manage space and divider color here

```
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android="http://schemas.android.com/apk/res/android">
    <size android:height="@dimen/_4dp" />
    <solid android:color="#EEEEEE" />
</shape>
```

Finally set on RecyclerView

```
    Drawable dividerDrawable = ContextCompat.getDrawable(this, R.drawable.divid
        mRecyclerView.addItemDecoration(new DividerItemDecoration(dividerDrawable
```

After following all above just RUN the project and use app. If you have any queries, feel free to ask



RELATED POSTS

Drag and drop RecyclerView item Android

WebView Example in Android (Kotlin)

Subscribe Login

Join the discussion

5 COMMENTS Oldest

H. Schultz
1 year ago

Great overview and work on using RecyclerView. Since I haven't been programming apps for that long, this is a very good introduction which I will try out in the next few days. Unfortunately the change in Android Studio to Androidx delayed my project development considerably, because the names have

H. Schultz

0 0 Reply

> **alper**
> Reply to H. Schultz 10 months ago
>
> yes
>
> 0 0 Reply

phyuthinkyi
9 months ago

I don't find CommonUtils class under the package import com.recyclerviewexample.utils.CommonUtils;

0 0 Reply

mete
5 months ago

how can ı change drawable image instead of image url ? If possibler please help me

0 0 Reply

> **Morris** Author
> Reply to mete 5 months ago
>
> imageview.setImageDrawable(getResources().getDrawable(R.drawable.xxx));
>
> 0 0 Reply