

CS 473 - MDP

Mobile Device Programming

© 2021 Maharishi International University

All course materials are copyright protected by international copyright laws and remain the property of the Maharishi International University. The materials are accessible only for the personal use of students enrolled in this course and only for the duration of the course. Any copying and distributing are not allowed and subject to legal action.



Maharishi International
University

CS 473 - MDP

Mobile Device Programming

MS.CS Program
Department of Computer Science
Renuka Mohanraj , Ph.D.



Maharishi International
University

CS 473 – MDP

Mobile Device Programming

Lesson-6

User Input Controls – Day 2

Advanced UI – RecyclerView and CardView



Maharishi International
University

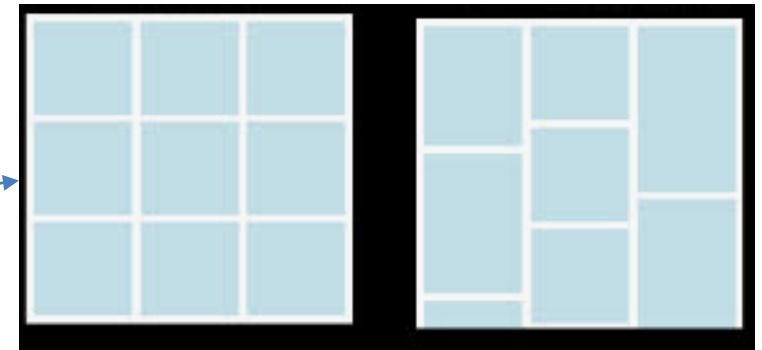
Contents

- What is a Recycler View?
- RecyclerView Components Overview
- Implementing RecyclerView Steps
- Hands on Example – 1 using RecyclerView to show the Book List
- CardView
- Hands on Example – 2 using RecyclerView, CardView and Listener Implementation

What is a Recycler View?

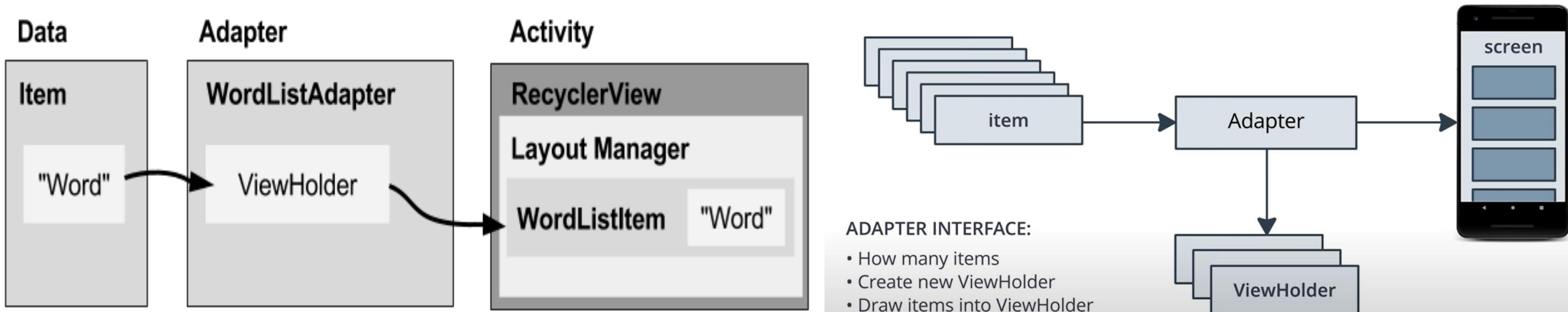
- Scrollable container for large data sets
- Efficient
 - uses and reuses limited number of views
 - Updates changing data fast
- RecyclerView also provides a choice of three built-in layout managers to control the way in which the list items are presented to the user
 - LinearLayoutManager
 - GridLayoutManager
 - StaggeredGridLayoutManager

WordListSQL
Android Performance
ViewHolder
LinkedList
Data model
SQLiteOpenHelper
SQLiteDatabase
Android Studio
AsyncTask
ListView
Adapter
inflate



RecyclerView Components Overview

- **Data**
- **RecyclerView** scrolling list — [RecyclerView](#)
- **Layout** for one item of data—XML file
- **Layout manager** handles the organization of UI components in a view—[Recyclerview.LayoutManager](#)
- **Adapter** connects data to the RecyclerView—[RecyclerView.Adapter](#)
- **View holder** has view information for displaying one item—[RecyclerView.ViewHolder](#)



Implementing RecyclerView Steps

1. Add the RecyclerView just drag the RecyclerView in the Layout
2. Create XML layout for item
3. Extend RecyclerView.Adapter
4. Extend RecyclerView.ViewHolder
5. In onCreate of activity, create a RecyclerView with adapter and layout manager

Refer step by step implementation and Demo codes from Lesson6 Day2

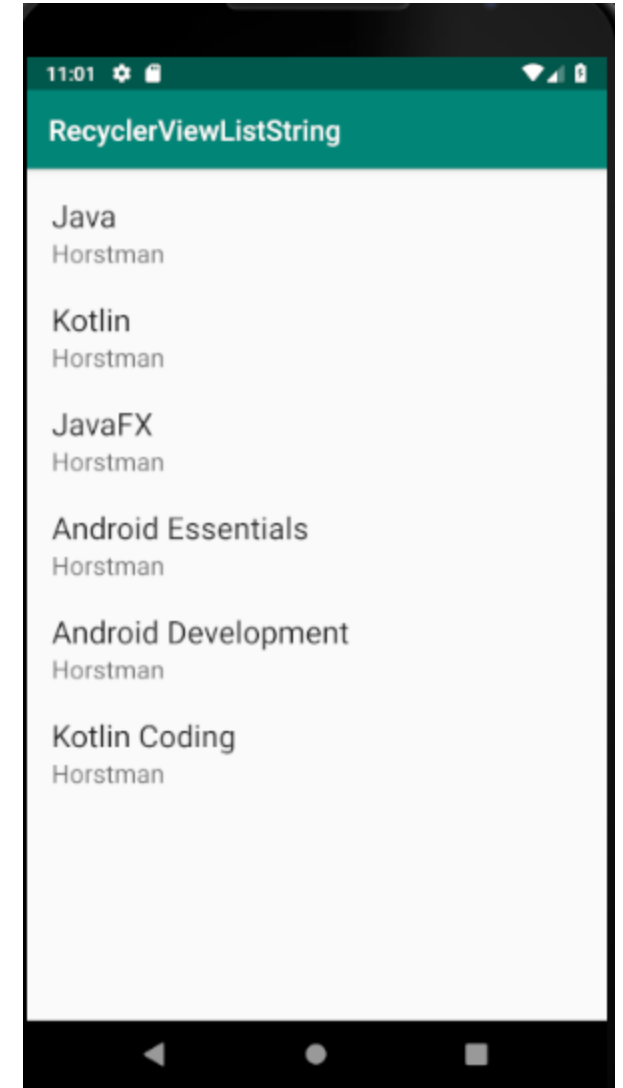
Hands on Example - 1

Problem :

Show the RecyclerView with the list of Books.

Book is data class with the attributes book name and author.

Refer : Lesson6/InClassRCVTest



Adding RecyclerView in the layout

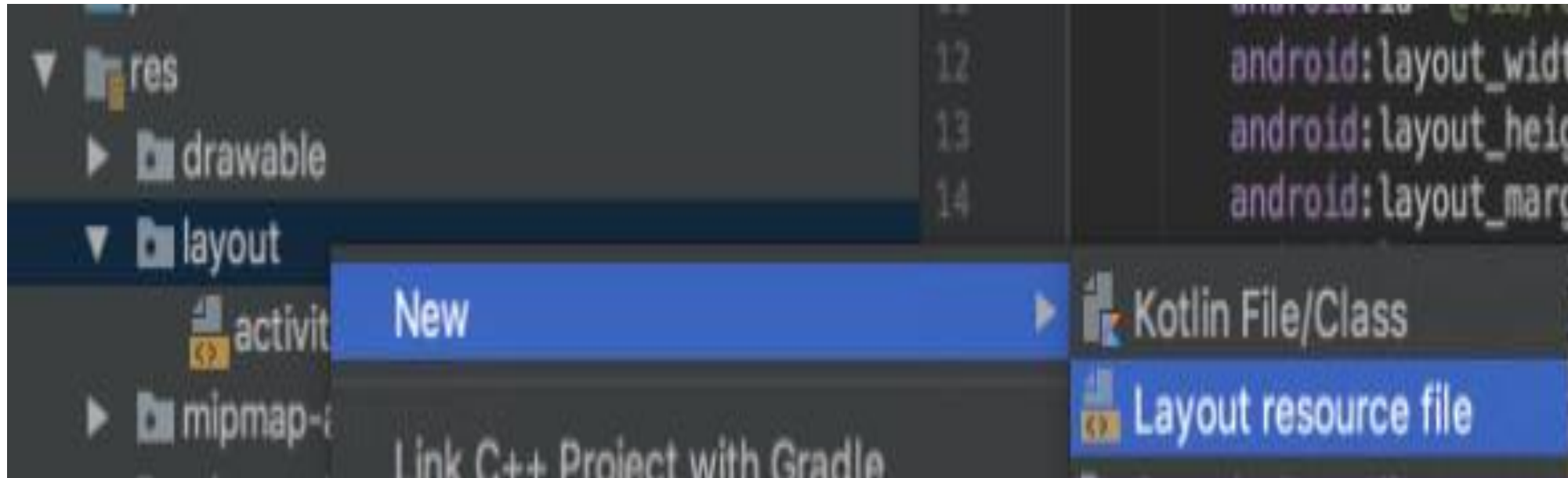
Step:1 In your activity_main.xml layout add the RecyclerView as below

Add RecyclerView to XML Layout

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

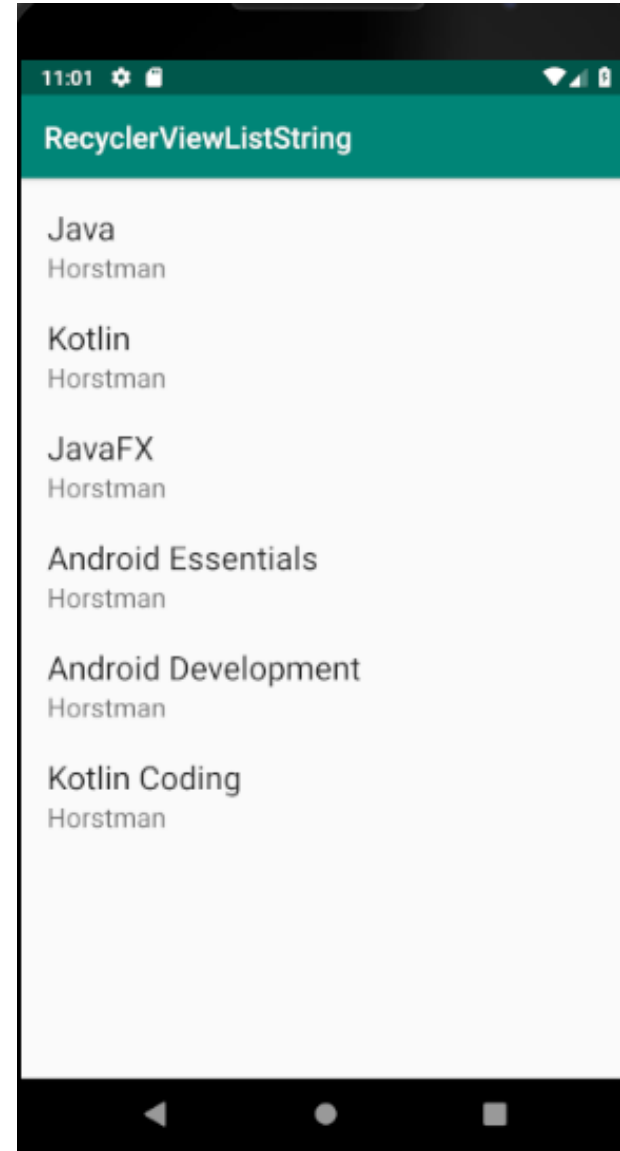
Creating new layout to display the list item

Step 2. Create layout to show the Book item as per the screenshot and give a name as item_layout.xml



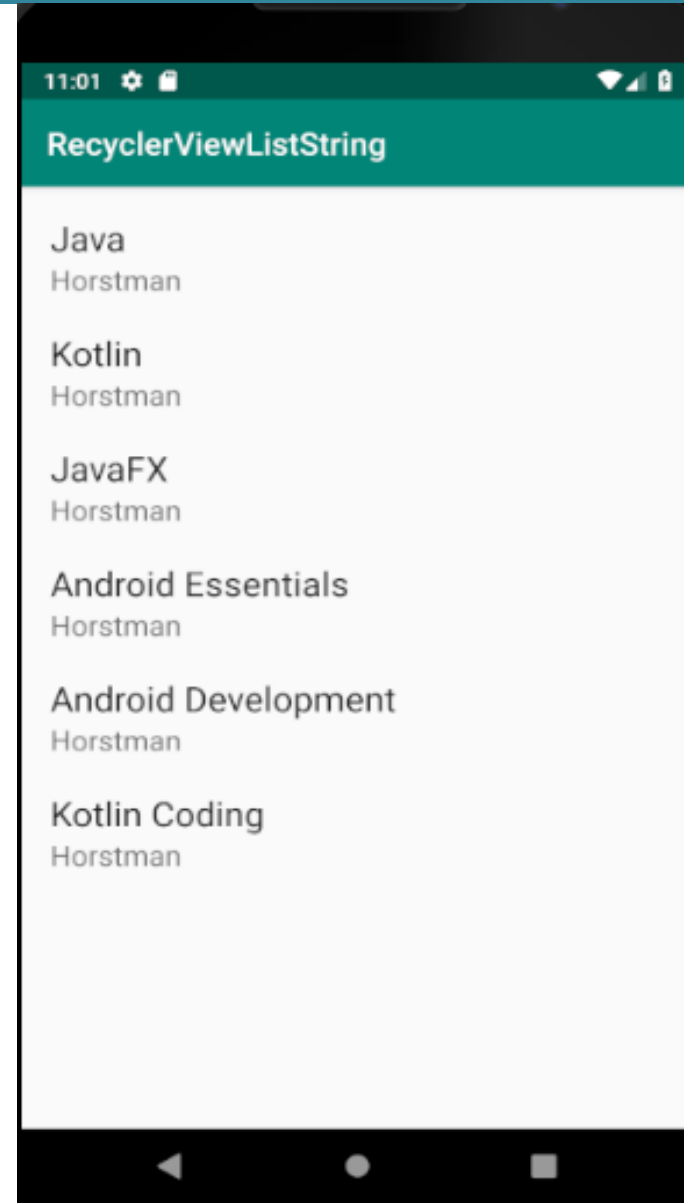
item_layout.xml

```
<LinearLayout xmlns:android="http://schemas.android.com
/apk/res/android"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:padding="10dp">
    <TextView
        android:textAppearance="@style/
Base.TextAppearance.AppCompat.Large"
        android:id="@+id/name"
        android:text="Book Name"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />
```



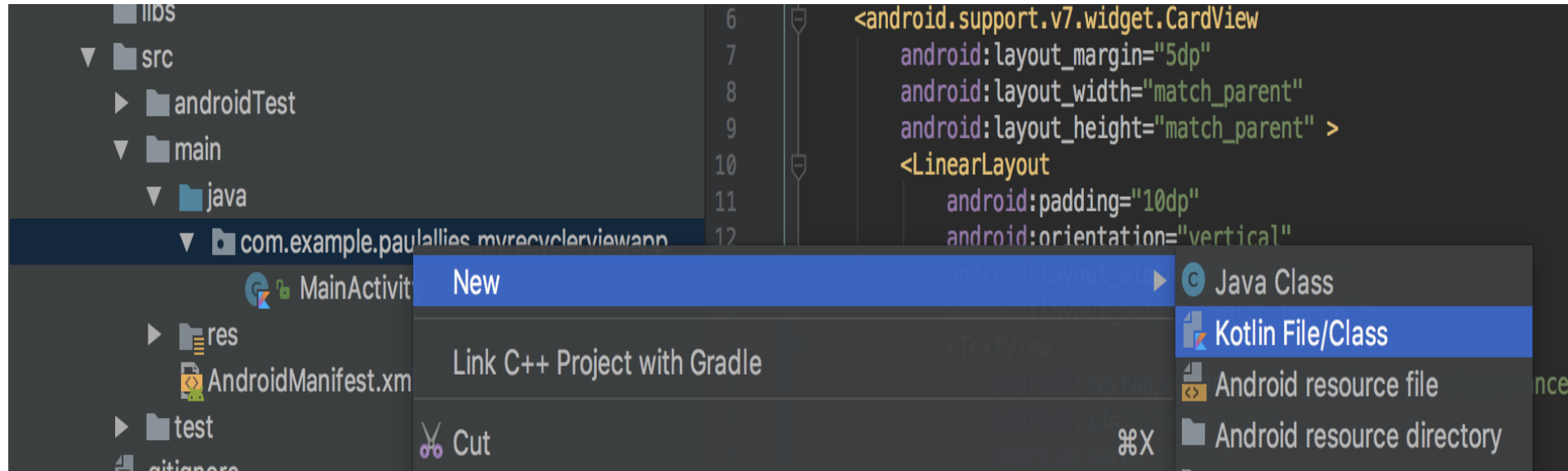
item_layout.xml

```
<TextView
    android:textAppearance="@style/
Base.TextAppearance.AppCompat.Medium"
    android:text="Author"
    android:id="@+id/author"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"/>
</LinearLayout>
```



Create a Data Class for Book

- Create Book.kt file as per the screenshot.



Book.kt file code

```
data class Book(val name: String, val author: String)
```

Implementing RecyclerView

Implement the adapter by creating MyAdapter.kt

Adapter has 3 required methods need to implement

- `onCreateViewHolder()`
 - This method creates and returns a `ViewHolder` object initialized with the view that is to be used to display the data. This view is typically created by inflating the XML layout file.
- `onBindViewHolder()`
 - This method is passed the `ViewHolder` object created by the `onCreateViewHolder()` method together with an integer value indicating the list item that is about to be displayed. Contained within the `ViewHolder` object is the layout assigned by the `onCreateViewHolder()` method. It is the responsibility of the `onBindViewHolder()` method to populate the views in the layout with the text and graphics corresponding to the specified item and to return the object to the `RecyclerView` where it will be presented to the user.
- `getItemCount()`
 - This method must return a count of the number of items that are to be displayed in the list.

Step 3: Implement the adapter in MyAdapter.kt

```
class MyAdapter(var blist: ArrayList<Book>) : RecyclerView.Adapter<MyAdapter.MyViewHolder>() {
    override fun onCreateViewHolder(parent: ViewGroup, viewType: Int):
                                   MyAdapter.MyViewHolder {
        val view =
            LayoutInflater.from(parent.context).inflate(R.layout.item_list,parent,false)
        return MyViewHolder(view)
    }
    override fun onBindViewHolder(holder: MyAdapter.MyViewHolder, position: Int) {
        holder.tv_name.text = books[position].bname
        holder.tv_auth.text = books[position].bauth
    }
    // return Size of the list of data.
    override fun getItemCount(): Int {
        return blist.size
    }
    inner class MyViewHolder(itemView: View):RecyclerView.ViewHolder(itemView){
        var tv_name : TextView = itemView.findViewById(R.id.name)
        var tv_auth : TextView = itemView.findViewById(R.id.author)
    }
}
```

Step4: Implementing RecyclerView in MainActivity.kt

```
class MainActivity : AppCompatActivity() {  
    private lateinit var binding: ActivityMainBinding  
    override fun onCreate(savedInstanceState: Bundle?) {  
        super.onCreate(savedInstanceState)  
        binding = ActivityMainBinding.inflate(layoutInflater)  
        setContentView(binding.root)  
        val books = ArrayList<Book>()  
        books.add(Book("Java", "Horstman"))  
        books.add(Book("Kotlin", "Joshua Bloch"))  
        books.add(Book("JavaFX", "Herbert"))  
        books.add(Book("Android Essentials", "Kathy"))  
        books.add(Book("Android Development", "Bruce"))  
        // Set the Layout Manager  
        binding.recyclerView1.layoutManager = LinearLayoutManager(this)  
        // Create an object for the MyAdapter  
        val adapter = MyAdapter(books)  
        // Set adapter to your RecyclerView  
        binding.recyclerView1.adapter = adapter  
    }  
}
```


CardView

- Android 7.0 introduces a new widget called CardView.
- CardView wraps a layout and will often be the container used in a layout for each item within a ListView or RecyclerView.
- Appeared with shadow effects and rounded corners.
- CardView API is an easy way for you to show information inside cards that have a consistent look across the platform.



Hands on Example - 2

- This example uses
 - RecyclerView displays the Fruit list
 - Each item displayed using CardView. Each row displayed with Fruit image, name of the fruit and small description about the fruit.
 - To display the image in circle shape needs third party library xml code as mentioned below.

```
<de.hdodenhof.circleimageview.CircleImageView  
    android:id="@+id/imageView"  
    android:layout_width="90dp"  
    android:layout_height="90dp"  
    android:src="@mipmap/ic_launcher"  
    app:civ_border_width="2dp"  
    app:civ_border_color="#FF000000"  
/>
```

Include the below line in build.gradle dependencies

//Getting Image in a Circular View

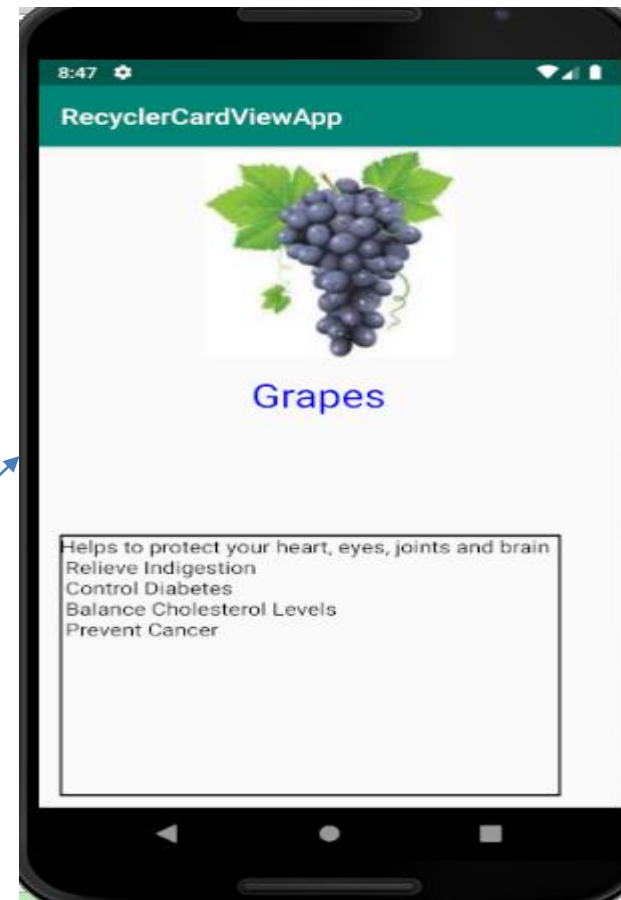
implementation 'de.hdodenhof:circleimageview:3.1.0'

Refer without third party library to show the image in different shapes Demo: CircularViewCustom



Hands on Example - 2

- Once clicking each item, opens another activity gives detail description about the fruit. Here is the screen after clicking Grapes.
- Refer : Lesson6\CardViewDemo



Adding CardView Layout

XML Code for the CardView

```
<androidx.cardview.widget.CardView
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:padding="20dp"
    android:id="@+id/card_view"
    app:cardBackgroundColor="#e4b9f1"
    app:cardCornerRadius="12dp"
    android:layout_marginBottom="4dp">
```

ListView vs RecyclerView

- ListView
 - ListView provides only one type of view i.e Vertical ListView, and there is no way to implement the others such as Horizontal ListView, GridView etc.
- RecyclerView :
 - RecyclerView, with faster performance, is the advanced version of ListView which can be used to implement Vertical List, Horizontal List and Gridview with the help of Layout Managers.
 - RecyclerView is a container for **displaying large data sets** that can be scrolled very efficiently by maintaining a limited number of views.

Main Point 4

- **RecyclerView** is flexible and efficient version of ListView. It is a container for rendering larger data set of views that can be recycled and scrolled very efficiently. **RecyclerView** is like traditional ListView widget, but with more flexibility to customizes and optimized to work with larger datasets.
- ***Science of Consciousness: pure awareness is an abstraction of individual awareness; each individual provides a specific, concrete realization of unbounded and unmoving pure awareness. Pure awareness is more efficient and powerful.***

UNITY CHART

CONNECTING THE PARTS OF KNOWLEDGE WITH THE WHOLENESS OF KNOWLEDGE

Living fully developed human potential

1. There are many ways to take Android user input just as Creative Intelligence synthesizes parts for completeness of living.
 2. The many parts of an Android program function together in a cohesive whole as an Android app just as Creative Intelligence binds together delicate impulses of life.
-

3. **Transcendental Consciousness:** TC is the identity of each individual, located at the source of thought.
4. ***Impulses within the Transcendental field:*** *These impulses are perfectly balanced to create only the desired effect, no more and no less.*
5. ***Wholeness moving within Itself:*** *In Unity Consciousness, one's unbounded identity is recognized to be the final truth about every object. All objects are seen to have the same ultimate identity,| even though differences on the surface still remain.*

