RecyclerView & Adapters in Android

- This is a **very practical Android Kotlin topic**
- **RecyclerView-based contact list app** helps you to understand how the RecyclerView, Adapter, and ViewHolder pattern work together.

Concept Overview

1. RecyclerView

- A flexible and efficient view used to display large lists or grids of items.
- It **recycles item views** (hence the name) to improve performance.
- It needs an **Adapter** to bind data and a **LayoutManager** to arrange items (e.g., LinearLayoutManager, GridLayoutManager).

2. Adapter

- The Adapter acts as a **bridge between data and the RecyclerView**.
- It creates and binds the individual item views to your dataset.

3. ViewHolder

- The **ViewHolder** holds references to the item view's components (like TextViews, ImageViews).
- It avoids calling findViewById() repeatedly, improving performance.

Note:

Component	Purpose
RecyclerView	Displays the list efficiently
Adapter	Binds data to the RecyclerView
ViewHolder	Holds references to views for performance
LayoutManager	Arranges the items (linear or grid layout)

Example App: Simple Contact List App

Create a simple 2-screen Android app displaying a list of contacts using RecyclerView.

Step 1: Create a New Project

Name: ContactListApp
Language: **Kotlin**
Minimum SDK: API 24+

Step 2: Update activity_main.xml

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

```
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    tools:context=".MainActivity">

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

Step 3: Create a Data Model – Contact.kt

```
package com.example.contactlistapp

data class Contact(
    val name: String,
    val phone: String
)
```

Step 4: Create Item Layout – item_contact.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical"
    android:padding="12dp"
    android:background="?android:attr/selectableItemBackground">

    <TextView
        android:id="@+id/textViewName"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:textStyle="bold"
        android:textSize="18sp"
        android:text="Name" />

    <TextView
        android:id="@+id/textViewPhone"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Phone"
        android:textSize="16sp" />
</LinearLayout>
```

Step 5: Create Adapter – ContactAdapter.kt

```
package com.example.contactlistapp

import android.view.LayoutInflater
import android.view.View
import android.view.ViewGroup
import android.widget.TextView
import androidx.recyclerview.widget.RecyclerView

class ContactAdapter(private val contactList: List<Contact>) :
    RecyclerView.Adapter<ContactAdapter.ContactViewHolder>() {

    class ContactViewHolder(itemView: View) : RecyclerView.ViewHolder(itemView) {
        val nameText: TextView = itemView.findViewById(R.id.textViewName)
        val phoneText: TextView = itemView.findViewById(R.id.textViewPhone)
    }

    override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): ContactViewHolder {
        val view = LayoutInflater.from(parent.context)
            .inflate(R.layout.item_contact, parent, false)
        return ContactViewHolder(view)
    }

    override fun onBindViewHolder(holder: ContactViewHolder, position: Int) {
        val contact = contactList[position]
        holder.nameText.text = contact.name
        holder.phoneText.text = contact.phone
    }

    override fun getItemCount(): Int = contactList.size
}
```

Step 6: Main Activity – MainActivity.kt

```
package com.example.contactlistapp

import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import androidx.recyclerview.widget.LinearLayoutManager
import androidx.recyclerview.widget.RecyclerView

class MainActivity : AppCompatActivity() {
    private lateinit var recyclerView: RecyclerView
```

```
    private lateinit var contactAdapter: ContactAdapter

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        recyclerView = findViewById(R.id.recyclerViewContacts)
        recyclerView.layoutManager = LinearLayoutManager(this)

        val contactList = listOf(
            Contact("Amit Sharma", "9876543210"),
            Contact("Neha Verma", "9998877665"),
            Contact("Ravi Kumar", "8887766554"),
            Contact("Priya Singh", "7776655443"),
            Contact("Vikas Gupta", "9997766552")
        )

        contactAdapter = ContactAdapter(contactList)
        recyclerView.adapter = contactAdapter
    }
}
```

Step 7: Run the App

You will see a **scrollable list of contacts** with their names and phone numbers.
Each list item is efficiently managed using the **RecyclerView + Adapter + ViewHolder** pattern.

RecyclerView + Intents (Contact List App with Details Screen)

Extend your Contact List App so that tapping a contact opens a **details screen** showing that person's name and phone number — using **Intents** to pass data between activities.

Objective

When a user taps on a contact in the list:

1. The app opens a **new screen (ContactDetailsActivity)**.
2. The selected contact's **name** and **phone number** are displayed.

Step-by-Step Implementation

Step 1: Create a New Activity

In Android Studio:

Right-click app → New → Activity → Empty Activity → Name it ContactDetailsActivity

It will generate:

- ContactDetailsActivity.kt
- activity_contact_details.xml

Step 2: Update activity_contact_details.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:gravity="center"
    android:orientation="vertical"
    android:padding="20dp">

    <TextView
        android:id="@+id/textViewDetailName"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Name"
        android:textStyle="bold"
        android:textSize="22sp"
        android:padding="10dp" />

    <TextView
        android:id="@+id/textViewDetailPhone"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
```

```
        android:text="Phone"
        android:textSize="18sp" />
</LinearLayout>
```

Step 3: Update ContactDetailsActivity.kt

```
package com.example.contactlistapp

import android.os.Bundle
import android.widget.TextView
import androidx.appcompat.app.AppCompatActivity

class ContactDetailsActivity : AppCompatActivity() {

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_contact_details)

        val nameText: TextView = findViewById(R.id.textViewDetailName)
        val phoneText: TextView = findViewById(R.id.textViewDetailPhone)

        // Receive data from Intent
        val name = intent.getStringExtra("contact_name")
        val phone = intent.getStringExtra("contact_phone")

        nameText.text = name
        phoneText.text = phone
    }
}
```

Step 4: Update ContactAdapter.kt

```
Modify it to include a click listener to open the detail screen.
package com.example.contactlistapp

import android.content.Intent
import android.view.LayoutInflater
import android.view.View
import android.view.ViewGroup
import android.widget.TextView
import androidx.recyclerview.widget.RecyclerView
```

```
class ContactAdapter(private val contactList: List<Contact>) :
    RecyclerView.Adapter<ContactAdapter.ContactViewHolder>() {

    inner class ContactViewHolder(itemView: View) : RecyclerView.ViewHolder(itemView) {
        val nameText: TextView = itemView.findViewById(R.id.textViewName)
        val phoneText: TextView = itemView.findViewById(R.id.textViewPhone)
    }

    override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): ContactViewHolder {
        val view = LayoutInflater.from(parent.context)
            .inflate(R.layout.item_contact, parent, false)
        return ContactViewHolder(view)
    }

    override fun onBindViewHolder(holder: ContactViewHolder, position: Int) {
        val contact = contactList[position]
        holder.nameText.text = contact.name
        holder.phoneText.text = contact.phone

        // Handle item click
        holder.itemView.setOnClickListener {
            val context = holder.itemView.context
            val intent = Intent(context, ContactDetailsActivity::class.java)
            intent.putExtra("contact_name", contact.name)
            intent.putExtra("contact_phone", contact.phone)
            context.startActivity(intent)
        }
    }

    override fun getItemCount(): Int = contactList.size
}
```

Step 5: Update AndroidManifest.xml

Ensure that the new activity is declared (it's usually auto-added, but verify):

```
<activity android:name=".ContactDetailsActivity" />
```

Step 6: Run the App

Observe the following:

- A scrollable **contact list**.
- Tap on any contact → Opens a new screen displaying their **name** and **phone number**.

Note:

Component	Description
RecyclerView	Displays a scrolling list of contacts.
Adapter	Bridges the data (contacts) and UI.
ViewHolder	Holds each list item's views efficiently.
Intent	Used to navigate and pass data between activities.
putExtra() / getStringExtra()	Used to send and retrieve the contact details.

Activity:

Try to add contact images or **a search bar** to filter contacts dynamically
It would make it look more like a real-world contact app.