Android RecyclerView

The RecyclerView makes it easy to efficiently display large sets of data. We provide the data items to be displayed and define how each individual item will be displayed and then the RecyclerView dynamically displays them.

When an item scrolls off the screen, RecyclerView doesn't destroy its view. Instead, RecyclerView reuses the view for new items that have scrolled onscreen. This reuse vastly improves performance, improving your app's responsiveness and reducing power consumption.

Using a RecyclerView to display a list of Strings

1. Add a RecyclerView to your layout like so

```
<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"
    tools:context=".MainActivity"
    >
  <androidx.recyclerview.widget.RecyclerView
      android:id="@+id/rvNames"
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      app:layout_constraintBottom_toBottomOf="parent"
      app:layout_constraintEnd_toEndOf="parent"
      app:layout_constraintStart_toStartOf="parent"
      app:layout_constraintTop_toTopOf="parent"
      />
</androidx.constraintlayout.widget.ConstraintLayout>
```

2. Create the row layout for a single item. In this case create a file called `name_list_item` inside your layouts folder

```
<?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="wrap_content">

    <TextView
        android:id="@+id/tvName"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="16dp"
        android:layout_marginTop="8dp"
        android:layout_marginEnd="16dp"
        android:layout_marginBottom="8dp"
        android:text="TextView"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

3. Create the Adapter. The adapter's role is to convert an object at a position in your list into a list row item to be displayed in the RecyclerView. The adapter requires a ViewHolder object which describes and provides access to all the views within each item row.

```
class NamesRecyclerViewAdapter(var namesList: List<String>):
RecyclerView.Adapter<NamesViewHolder>() {
    override fun onCreateViewHolder(parent: ViewGroup, viewType: Int):
NamesViewHolder {
        var itemView = LayoutInflater.from(parent.context).inflate(R.
layout.name_list_item, parent, false)
        return NamesViewHolder(itemView)
    }

    override fun onBindViewHolder(holder: NamesViewHolder, position:
Int) {
        holder.tvName.text = namesList.get(position)
    }

    override fun getItemCount(): Int {
        return namesList.size
    }
}

class NamesViewHolder(itemView: View): RecyclerView.ViewHolder(itemView)
{
    var tvName = itemView.findViewById<TextView>(R.id.tvName)
}
```

Every adapter has three primary methods that must be implemented:

`onCreateViewHolder` - To inflate the item layout and instantiate the ViewHolder.

`onBindViewHolder` - To display the individual attributes of the current item in the view.

`getItemCount` - To determine the number of items to be displayed by the RecyclerView.

4. Bind the Adapter to the RecyclerView. In the activity we will create an instance of our adapter, passing to it the list of strings that we want to display. We then bind it to our RecyclerView from the layout and our list of names gets displayed.

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

    var rvNames = findViewById<RecyclerView>(R.id.rvNames)
    rvNames.layoutManager = LinearLayoutManager(baseContext)
    var names = listOf("Dedan", "Mary", "Lydia", "JKIA", "Armani",
"Kibera", "Friday")
    var namesAdapter = NamesRecyclerViewAdapter(names)
    rvNames.adapter = namesAdapter
  }
}
```