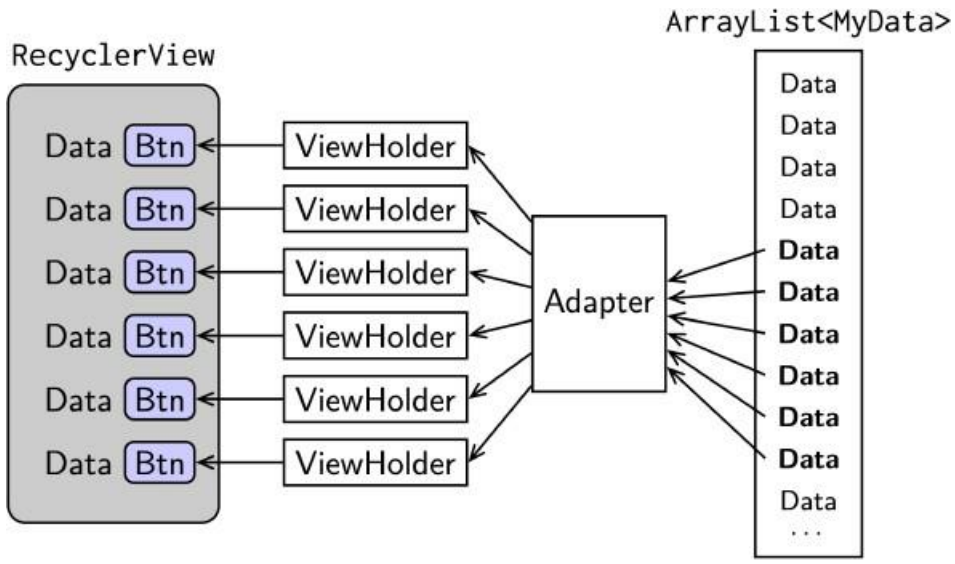
IT2010 – Mobile Application Development
BSc (Hons) in Information Technology
2nd Year
Faculty of Computing
SLIIT
2025 - Tutorial

Recycler View

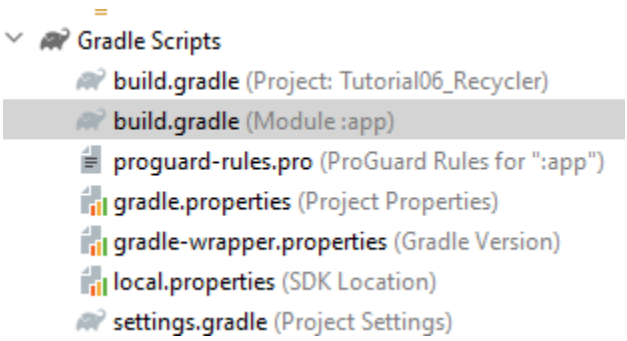
RecyclerView is a versatile UI component introduced in the Android Lollipop release, part of the Android Support Library. It is designed to display large data sets or data that changes frequently, optimizing performance by recycling the layout items that scroll off the screen. Unlike its predecessor, **ListView**, **RecyclerView** is more efficient and offers greater flexibility in terms of layout and decorations.

To use **RecyclerView**, developers typically need to implement an **Adapter** and a **ViewHolder**. The **Adapter** provides the data, while the **ViewHolder** holds the references to the UI components within each item layout. One of the notable features of **RecyclerView** is its ability to easily support different types of layouts like linear, grid, and staggered grids, by using different **LayoutManager** implementations.



Note: "Data Btn " is just an example of what could be in each list row (a TextView and a Button).

1. Add the following dependency to the Build.gradle(app) file. And click sync now.

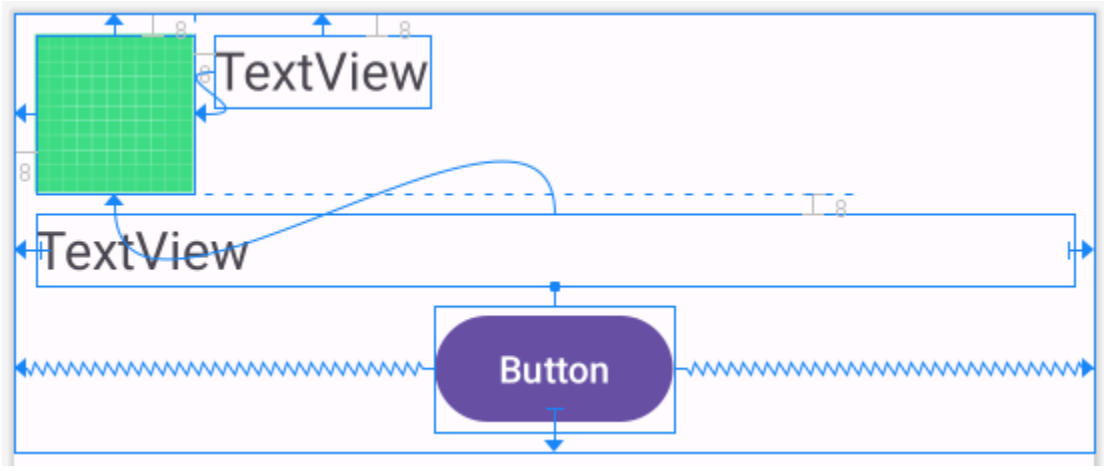


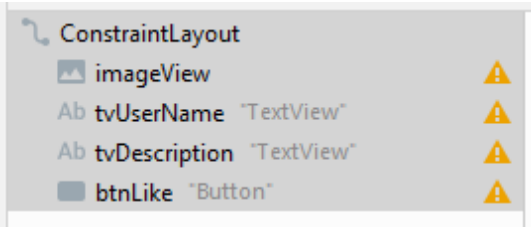
implementation 'androidx.recyclerview:recyclerview:1.3.1'

2. Create a data class named Post and implement the following.

```
data class Post(
   val userName:String,
   val description:String,
   val likes:Int
)
```

3. Create a xml layout named list_item_layout. And design it as follows.





4. Create a view holder. For that create a Kotlin class named MyDataVH and implement it as follows.

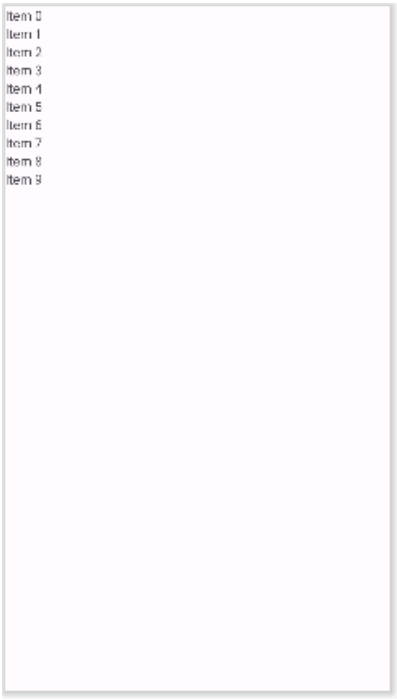
```
class MyDataVH(itemView: View) : RecyclerView.ViewHolder(itemView) {
   val tvUserName:TextView = itemView.findViewById(R.id.tvUserName)
   val tvDescription:TextView = itemView.findViewById(R.id.tvDescription)
   val btnLike:Button = itemView.findViewById(R.id.btnLike)
}
```

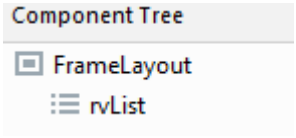
5. Create an Adapter to bind the data with the view. For that create a Kotlin Class named MyDataAdapter and implement it as follows:

```
class MyDataAdapter(private val data:List<Post>): RecyclerView.Adapter<MyDataVH>() {

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

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

override fun onBindViewHolder(holder: MyDataVH, position: Int) {
   val singleData = data[position]
   var toggle = true

   holder.tvUserName.text = singleData.userName
   holder.tvDescription.text = singleData.description
   holder.btnLike.text = "${singleData.likes}E   "

   holder.btnLike.setOnClickListener {

      if(toggle){
         holder.btnLike.text = "${singleData.likes + 1}E   "
         toggle = false
      }else{
         holder.btnLike.text = "${singleData.likes}E"
         toggle = true
      }
```

```
    }
}

}
```

6. To use RecyclerView on Activity/fragment, RecyclerView layout is needed. Design the Main Activity as follows.





9. To display data on the RecyclerView lets add some dummy data in the Main Activity. Paste the below code before the onCreate method

```
private val postsList = mutableListOf<Post>()

init {
    postsList.add(Post("Hannah Ice", "Anyone else here from New York?", 62))
    postsList.add(Post("Jane Smith", "Sunsets are so beautiful.", 37))
    postsList.add(Post("George Hill", "Anyone else here from New York?", 32))
    postsList.add(Post("Hannah Ice", "At the beach with friends.", 32))
    postsList.add(Post("Charlie Davis", "Sunsets are so beautiful.", 57))
    postsList.add(Post("Eddie Foster", "At the beach with friends.", 64))
    postsList.add(Post("Bob Brown", "This is my first post on Android Social Media app", 19))
    postsList.add(Post("John Doe", "Loving the new app update!", 90))
```

```
    postsList.add(Post("Fay Green", "Look at my new dog!", 63))
    postsList.add(Post("Diana Evans", "Anyone else here from New York?", 38))

}
```

10. implement the following code in the onCreate method

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

        val rvList:RecyclerView = findViewById(R.id.rvList)
        rvList.layoutManager = LinearLayoutManager(this)
        val adapter = MyDataAdapter(postsList)
        rvList.adapter = adapter
    }
```

11. Default LinearLayoutManager is having the vertical orientation you can change it as follows if you need the horizontal orientation. Try the following and check the output.

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

    val rvList:RecyclerView = findViewById(R.id.rvList)
    rvList.layoutManager = LinearLayoutManager(this,LinearLayoutManager.HORIZONTAL,false)
    val adapter = MyDataAdapter(postsList)
    rvList.adapter = adapter
}
```

13. next you can try the GridLayout manager as follows.

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

    val rvList:RecyclerView = findViewById(R.id.rvList)
    //rvList.layoutManager = LinearLayoutManager(this,LinearLayoutManager.HORIZONTAL,false)
    rvList.layoutManager = GridLayoutManager(this,2,
       GridLayoutManager.VERTICAL,false)
    val adapter = MyDataAdapter(postsList)
    rvList.adapter = adapter
  }
```

14. To dynamically adjust the width and height of the posts you can update the MyDataVH as follows:

```
class MyDataVH(itemView: View, parent:ViewGroup) : RecyclerView.ViewHolder(itemView) {

  val hSize = parent.measuredWidth /2
  val vSize = parent.measuredHeight /2

  init {
    itemView.layoutParams.height = vSize - 16
    itemView.layoutParams.width = hSize - 16
  }

  val tvUserName:TextView = itemView.findViewById(R.id.tvUserName)
  val tvDescription:TextView = itemView.findViewById(R.id.tvDescription)
  val btnLike:Button = itemView.findViewById(R.id.btnLike)
}
```