COMP 3160
Mobile App Dev II

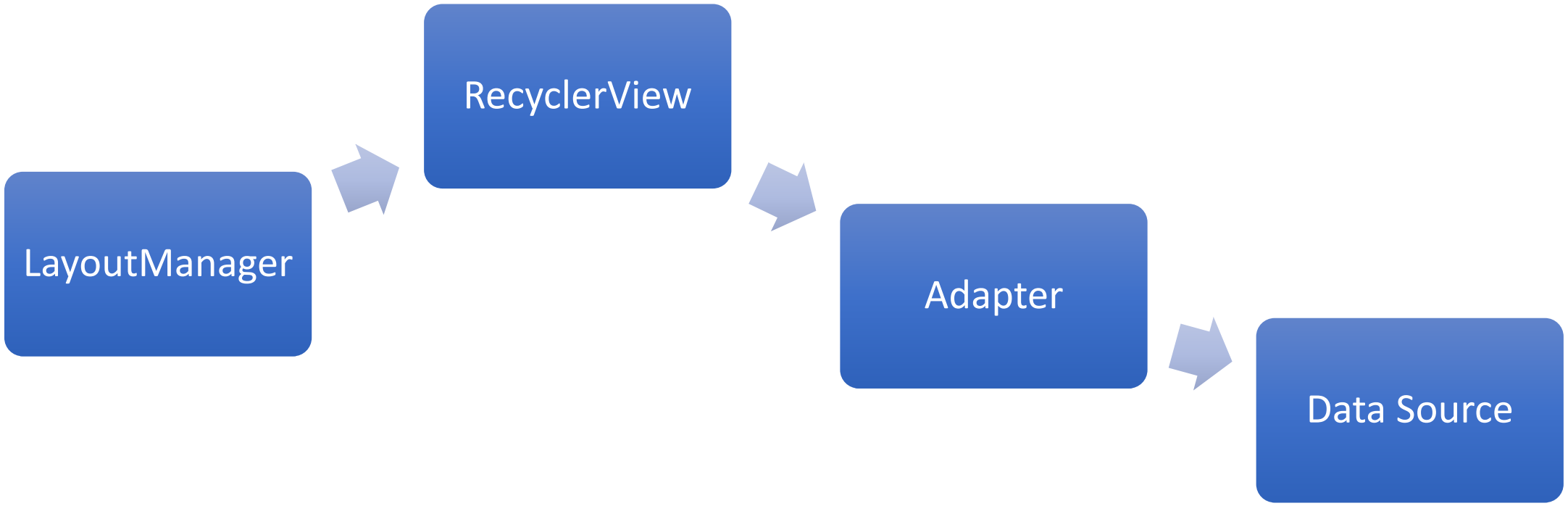
Musfiq Rahman

Winter 2018

RecyclerView Model

- Help you to design and implement a dynamic user interface that runs efficiently.
- Introduced into Android Lollipop, Google provides support library.
- Rather than creating the entire list, which would potentially cause glitches and performance problems, the RecycleView keeps some in a queue or recycle bin, for reuse.
- When you are about to scroll, the RecyclerView returns one of these previously created list items to you.
- Your code then binds the list item view with new content, then it can be scrolled in.
- Views that are scrolled out are then placed back into the queue for reuse.

Main Components



Main Components

RecyclerView Object

- The overall container for your dynamic user interface.
- add this object to your activity's or fragment's layout.
- fills itself with smaller views representing the individual items

Layout Managers

- *layout manager* arranges the items in RecyclerView.
- Use standard layout managers (such as LinearLayoutManager or GridLayoutManager), or implement your own.

Main Components

View Holder Object

- The individual items are represented by *view holder* objects.
- These objects are instances of the class you define by extending RecyclerView.ViewHolder.
- Each view holder is in charge of displaying a single item, and has its own view.

Adapter

- The view holder objects are managed by an adapter.
- Create an adapter by extending the RecyclerView.Adapter abstract class.
- The adapter
  - creates view holders as needed.
  - binds the view holders to their data.

Data Source

- Adapter generally takes a data source object and uses that data source to creates and binds view holders with data.
- You can pass any types of data source to an adapter class including database.

Benefits of Recycler View Model

- When the view is first populated, it creates and binds some view holders on either side of the list.
- As the user scrolls the list, the RecyclerView creates new view holders as necessary.
- It also saves the view holders which have scrolled off-screen, so they can be reused.
- When the displayed items change, the app notifies the adapter by calling an appropriate RecyclerView.Adapter.notify…() method. The adapter's built-in code then rebinds just the affected items.

RecyclerView Workflow

- The onCreate() method creates the RecyclerView specified by activity's layout:
  - myRecyclerView = (RecyclerView) findViewById(R.id.myrecyclerview)
- The onCreate() method creates an adapter; the adapter class is specified by the app, extending the Android Support Library's RecyclerView.Adapter class.
  - myAdapter = new CustomAdapter(myDataSet);
- The onCreate() method attaches the adapter to the RecyclerView.
  - myRecyclerView.setAdapter(myAdapter);

RecyclerView Workflow

- The adapter class's definition specifies which class it uses as its view holder:
  - public class CustomAdapter extends RecyclerView.Adapter<CustomViewHolder>
- The Android Support Library calls the adapter's onCreateViewHolder() method. That method needs to construct the view holder and set the view it uses to display its contents.
  - ```
    // Inflate the view for this view holder
    View thisItemsView =
    myInflater.inflate(R.layout.list_item_layout,
            parent, false);
    // Call the view holder's constructor, and pass the view
    //to it;
    // return that new view holder
    return new CustomViewHolder(thisItemsView);
    ```
    ```

RecyclerView Workflow

- The Android Support Library then binds the view holder to its data. It does this by calling the adapter's onBindViewHolder() method, and passing the view holder's position in the RecyclerView.
- he onBindViewHolder() method needs to fetch the appropriate data, and use it to fill in the view holder's layout.
  - **@Overridepublic**
  - **void onBindViewHolder(CustomViewHolder holder, int position) {**
  - **// Find out the data, based on this view holder's position String thisItemsName = myNameList.get(position); holder.nameTextView.setText(thisItemsName);**
  - **}**

RecyclerView Workflow

- The Android Support Library repeats this process, creating and binding new view holders until the visible portion of the RecyclerView is filled. It also creates and binds one or two more, so if the user scrolls the list, the new view holders are ready to display.

- More info: https://developer.android.com/guide/topics/ui/layout/recyclerview.html

Exercise

- Lab – 2 RecyclerView Exercise.zip on Moodle.