Android Developer Fundamentals

Recycler View

Contents

- RecyclerView Components
- Implementing a RecyclerView

What is a Recycler View?

- Scrollable container for large data sets
- Efficient
  - uses and reuses limited number of views
  - Updates changing data fast
- RecyclerView

All Components overview

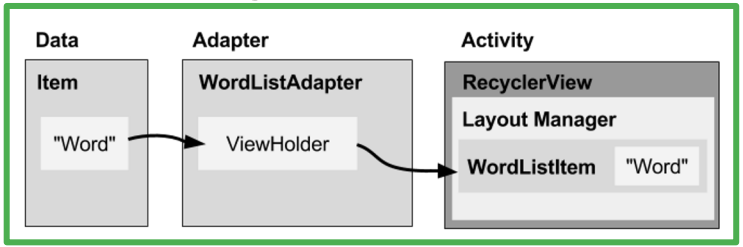
- **Data**
- **RecyclerView** scrolling list for list items—RecyclerView
- **Layout** for one item of data—XML file
- **Layout manage**r handles the organization of UI components in a view—Recyclerview.LayoutManager
- **Adapter** connects data to the RecyclerView—RecyclerView.Adapter
- **View holder** has view information for displaying one item—RecyclerView.ViewHolder

What is a layout manager?

- All view groups have layout managers
- Positions item views inside a RecyclerView.
- Reuses item views that are no longer visible to the user
- Built-in layout managers include LinearLayoutManager, GridLayoutManager, and StaggeredGridLayoutManager
- For RecyclerView, extend RecyclerView.LayoutManager

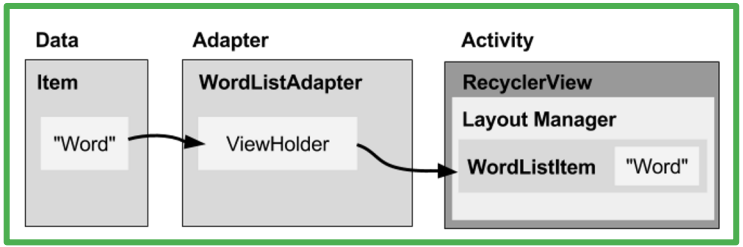
What is an adapter?

- Helps incompatible interfaces work together, for example, takes data from a database Cursor and puts them as strings into a view
- Intermediary between data and view
- Manages creating, updating, adding, deleting item views as the underlying data changes
- RecyclerView.Adapter



What is a view holder?

- Used by the adapter to prepare one view with data for one list item
- Layout specified in an XML resource file
- Can have clickable elements
- Is placed by the layout manager
- RecyclerView.ViewHolder



Steps Summary

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

View holder class

```
class MyViewHolder extends RecyclerView.ViewHolder {
    TextView textView;

    public MyViewHolder(View itemView) {
        super(itemView);

        // Get the layout
        textView = itemView.findViewById(R.id.textView);
    }
}
```

Adapter class: required methods

- `onCreateViewHolder()`
- `onBindViewHolder()`
- `getItemCount()`

Create the view holder

```
class MyAdapter extends RecyclerView.Adapter<MyViewHolder> {
    public MyViewHolder onCreateViewHolder(...) {
        MyViewHolder holder;

        View itemView = //load layout for one item
        holder = new MyViewHolder(itemView);

        return holder;
    }
}
```

Fill data to view holder

```
class MyAdapter extends RecyclerView.Adapter<MyViewHolder> {
    public void onBindViewHolder(MyViewHolder holder,
                                 int position, ...) {

        holder.textView.setText(data[position]);

    }
}
```

Create the RecyclerView in activity's onCreate()

```
recyclerView = findViewById(R.id.recyclerview);
adapter = new MyAdapter(this, data);
recyclerView.setAdapter(adapter);
recyclerView.setLayoutManager(
        new LinearLayoutManager(this));
```

Learn more

- RecyclerView
- RecyclerView class
- RecyclerView.Adapter class
- RecyclerView.ViewHolder class
- RecyclerView.LayoutManager class

https://stackoverflow.com/questions/26728651/recyclerview-vs-listview

END