Android Developer Fundamentals

User Interaction and Intuitive Navigation

Lesson 4

4.4 Recycler View

Contents

- RecyclerView Components
- Implementing a RecyclerView

What is a Recycler View?

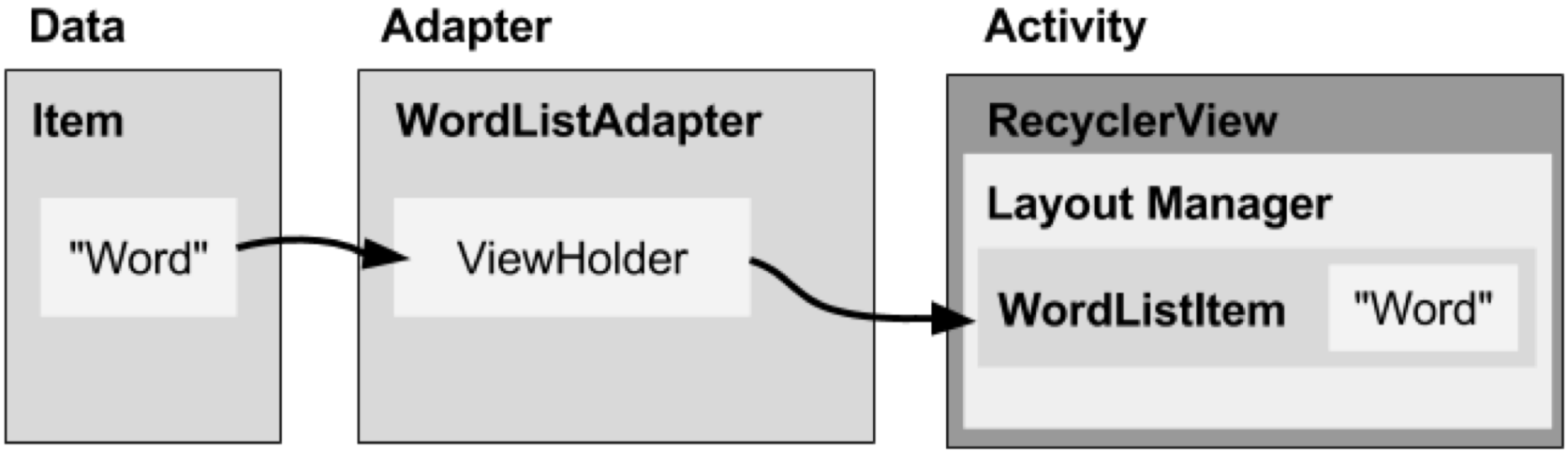
- Scrollable container for large data sets
- Efficient
  - uses and reuses limited number of views
  - Updates changing data fast
- RecyclerView

RecyclerView Components

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

How components fit together

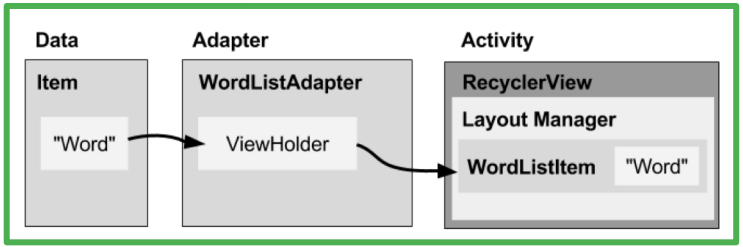


What is a layout manager?

- 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

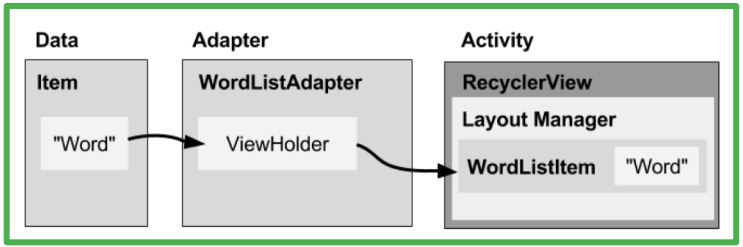
What is an adapter?

- 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



Implementing RecyclerView

Steps Summary

1. Add the RecyclerView dependency to app/build.gradle file
2. Add RecyclerView to layout
3. Create XML layout for item
4. Extend RecyclerView.Adapter
5. Extend RecyclerView.ViewHolder
6. In onCreate of activity, create a RecyclerView with adapter and layout manager

Add dependency to app/build.gradle

```
dependencies {
   ...
   compile 'com.android.support:recyclerview-v7:24.1.1'
   ...
   }
```

Add RecyclerView to XML Layout

```
<android.support.v7.widget.RecyclerView
    android:id="@+id/recyclerview"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
</android.support.v7.widget.RecyclerView>
```

Create layout for 1 list item

```
<LinearLayout …>
  <TextView
    android:id="@+id/word"
    style="@style/word_title" />
</LinearLayout>
```

Implement the adapter

```
public class WordListAdapter
    extends RecyclerView.Adapter<WordListAdapter.WordViewHolder> {


    public WordListAdapter(Context context,
                           LinkedList<String> wordList) {
        mInflater = LayoutInflater.from(context);
        this.mWordList = wordList;
    }
}
```

Adapter has 3 required methods

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

Let's take a look!

onCreateViewHolder()

```
@Override
    public WordViewHolder onCreateViewHolder(
        ViewGroup parent, int viewType) {
        // Create view from layout
        View mItemView = mInflater.inflate(
             R.layout.wordlist_item, parent, false);
        return new WordViewHolder(mItemView, this);
    }
```

onBindViewHolder()

```
@Override
    public void onBindViewHolder(
        WordViewHolder holder, int position) {
        // Retrieve the data for that position
        String mCurrent = mWordList.get(position);
        // Add the data to the view
        holder.wordItemView.setText(mCurrent);
    }
```

getItemCount()

```
@Override
public int getItemCount() {
   // Return the number of data items to display
   return mWordList.size();
}
```

Create the view holder in adapter class

```
class WordViewHolder extends RecyclerView.ViewHolder {}
```

If you want to handle mouse clicks:

```
class WordViewHolder extends RecyclerView.ViewHolder
                     implements View.OnClickListener {}
```

View holder constructor

```
public WordViewHolder(View itemView, WordListAdapter adapter) {
  super(itemView);
  // Get the layout
  wordItemView = (TextView) itemView.findViewById(R.id.word);
  // Associate with this adapter
  this.mAdapter = adapter;
  // Add click listener, if desired
  itemView.setOnClickListener(this);
}
// Implement onClick() if desired
```

Create the RecyclerView in activity's onCreate()

```
mRecyclerView = (RecyclerView)
findViewById(R.id.recyclerview);
mAdapter = new WordListAdapter(this, mWordList);
mRecyclerView.setAdapter(mAdapter);
mRecyclerView.setLayoutManager(new
LinearLayoutManager(this));
```

Practical: RecyclerView

- This is rather complex with many separate pieces. So, there is a whole practical where you implement a RecyclerView that displays a list of clickable words.

- Shows all the steps, one by one with a complete app

Learn more

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

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

What's Next?

- Concept Chapter: 4.4 C RecyclerView
- Practical: 4.4 P Create a Recycler View

END