ĐẠI HỌC BÁCH KHOA HÀ NỘI
VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

Mobile Programing

Chapter 4.5. Recycle View

Note

- This slide is based on Google Android code labs slides
- Original slides: https://drive.google.com/drive/folders/1eu-LXxiHocSktGYpG04PfE9Xmr_pBY5P

4.5 RecyclerView

Contents

- `RecyclerView` Components
- Implementing a `RecyclerView`

What is a RecyclerView?

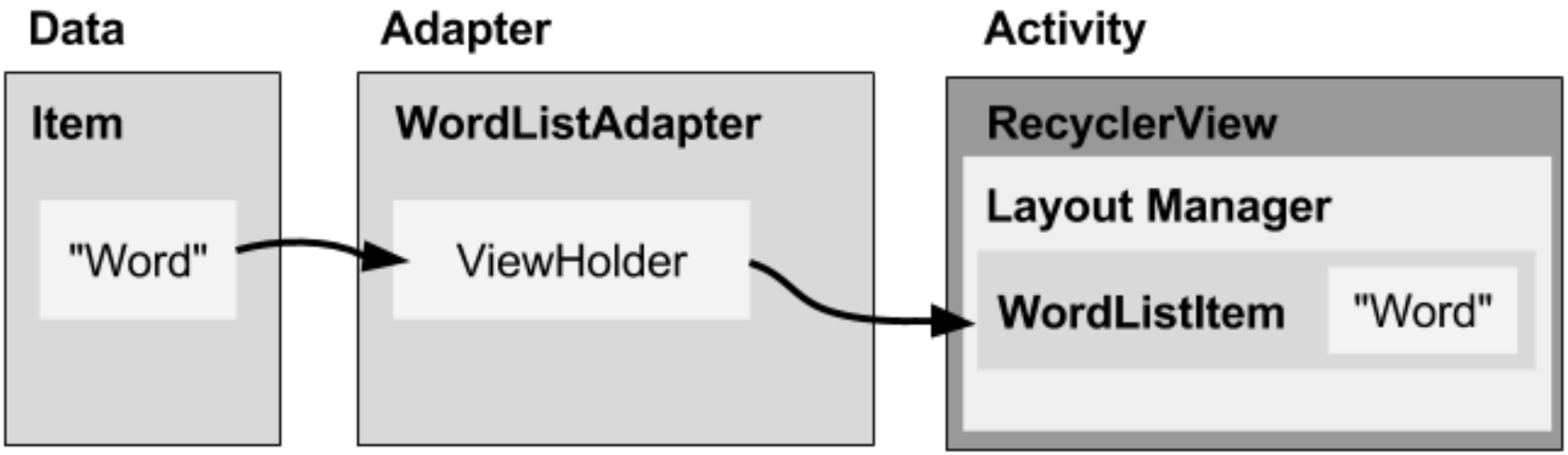
- `RecyclerView` is scrollable container for large data sets
- Efficient
  - Uses and reuses limited number of `View` elements
  - Updates changing data fast

RecyclerView 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`
- **ViewHolder** has view information for displaying one item—`RecyclerView.ViewHolder`

How components fit together overview



Compare to ListView

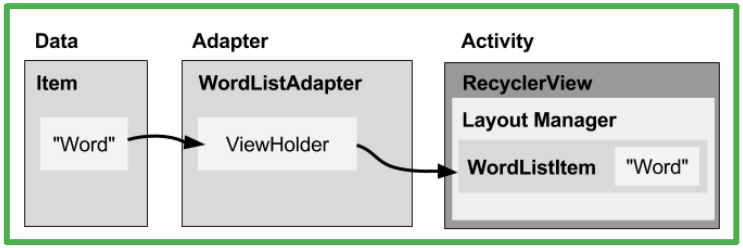
https://guides.codepath.com/android/using-the-recyclerview

What is a layout manager?

- Each `ViewGroup` has a layout manager
- Use to position `View` items inside a `RecyclerView`
- Reuses `View` items that are no longer visible to the user
- Built-in layout managers
  - `LinearLayoutManager`
  - `GridLayoutManager`
  - `StaggeredGridLayoutManager`
- Extend `RecyclerView.LayoutManager`

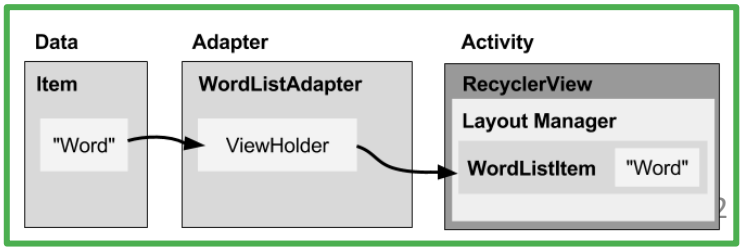
What is an adapter?

- Helps incompatible interfaces work together
  - Example: Takes data from database `Cursor` and prepares strings to put into a `View`
- Intermediary between data and `View`
- Manages creating, updating, adding, deleting `View` items as underlying data changes
- `RecyclerView.Adapter`



Adapter
What is a ViewHolder?

- 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 `RecyclerView` dependency to build.gradle if needed
2. Add `RecyclerView` to layout
3. Create XML layout for item
4. Extend `RecyclerView.Adapter`
5. Extend `RecyclerView.ViewHolder`
6. In `Activity` `onCreate()`, create `RecyclerView` with adapter and layout manager

Add dependency to app/build.gradle

Add `RecyclerView` dependency to build.gradle if needed:

```
dependencies {
   ...
   compile 'com.android.support:recyclerview-v7:26.1.0'
   ...
   }
```

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
Item layout

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

Implement the adapter
Adapter: Create

```
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
Adapter: onCreateViewHolder()

- `onCreateViewHolder()`
- `inBindViewHolder()`
- `getItemCount()`

Let's take a look!

onCreateViewHolder()
Adapter: 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()

Adapter: 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()
Adapter: getItemCount()

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

Create the view holder in adapter class

Adapter: ViewHolder Class

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

If you want to handle mouse clicks:

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

ViewHolder: Constructor

```
public WordViewHolder(View itemView, WordListAdapter adapter) {
  super(itemView);
  // Get the layout
  wordItemView = 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 RecyclerView

Create the RecyclerView in Activity onCreate()

```
mRecyclerView =
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

What's Next?

- Concept Chapter: 4.5 RecyclerView
- Practical: 4.5 RecyclerView

END