

Hinweis: Zu bearbeiten ist Aufgabe 1. Aufgabe 2 ist eine Bonusaufgabe

1. FILME STREAMING SERVICE - RecyclerView

In dieser Aufgabe richten wir für die uns bekannte App “WatchMe” eine effiziente `RecyclerView` mit einem `LinearLayout` ein.

Das Projekt enthält eine vorgegebene Klasse `MovieTitle`, welche alle Informationen, die ein Listeneintrag für den Filmtitel braucht, enthält. Sowie eine Klasse `Datasource`, welche alle Ressourcen für die weitere Verwendung vorbereitet.



- Öffne das Projekt “Filme Streaming Service”, die `RecyclerView` befindet sich bereits im Layout `activity_main.xml`
- Zuerst wollen wir das Layout stylen, das dann letztendlich einen Listeneintrag darstellt. Dazu befindet sich im layout Ordner die Datei `list_item.xml`
 - Füge eine `TextView` hinzu und passe die Attribute an, bis du zufrieden bist. Als Text kannst du irgendeinen Platzhalter einfügen

- Passe die Höhe, die Margins, die Transparenz und die Hintergrundfarbe des `ConstraintLayout` an, bis der Listeneintrag deinen Vorstellungen entspricht

Hinweis: Du kannst dir dein `list_item` bereits in der `RecyclerView` anzeigen lassen, indem du in der `activity_main.xml` der `RecyclerView` das Attribut `tools:listitem="@layout/list_item"` gibst

- Die Klasse `ItemAdapter` ist dafür verantwortlich, die Listeneinträge zu erstellen und ihnen die richtigen Informationen zuzuweisen.

Die innere Klasse `ItemViewHolder` erstellt ein `ViewHolder` Objekt, welches eine `ItemView` umschließt und einen Listeneintrag darstellt

- erstelle eine Variable innerhalb diese Klasse mit beliebigem Namen
- weise dieser Variable die `TextView` aus dem Layout `list_item.xml` zu
 - dafür nutze die übergebene `itemView: View`
 - über `View` Elemente ist es möglich die Funktion `findViewById (id)` aufzurufen
 - Setze die `id` für die `TextView` aus dem `list_item` Layout

Hinweis: Gib der `TextView` in deinem `list_item` Layout eine `ID`, mit der du die `TextView` hier finden kannst

- Die Funktion `onCreateViewHolder` erstellt neue `ViewHolder` Objekte.

Jedes der Objekte hat eine `ItemView` mit dem `list_item` Layout

- Um dieser `ItemView` das passende Layout zu vergeben
 - erzeuge die Variable `val itemView`
 - weise der Variablen einen `LayoutInflater` zu welches das `list_item` Layout "aufbläst"
 - **`LayoutInflater.from(<context>).inflate(<Layout>, parent, false)`**(siehe Folie 13)
 - Gebe ein neues `ItemViewHolder` Objekt zurück (`return`), welches die eben erstellte `ItemView` als Parameter erhält

- Die Funktion `onBindViewHolder` gibt der `ItemView` aus dem `ViewHolder` den richtigen Inhalt

- Die Klasse `ItemAdapter` bekommt im Konstruktor die in `Datasource` erstellte Liste mit `MovieTitles` übergeben. Nutze die als Parameter übergebene `position`, um das entsprechende `MovieTitle` Objekt aus der Liste zu holen:
`dataset[position]`

- Hole die TextView aus dem `ViewHolder` (als **holder** Parameter übergeben) und weise ihr den Text zu, der als String Ressource im `MovieTitle` Objekt gespeichert ist

Hinweis: Um in dieser Klasse auf die Ressourcen zugreifen zu können, musst du auf die `resources` des übergebenen `context` verweisen

Ressourcen können über eine eindeutige id \Rightarrow Integer Werte `R.<type>.<id>` angesprochen werden.

Beispiel:

Bilddateien, die über den Ressourcen Manager geladen werden, werden zu `drawable` Elementen, welche man wie folgt im Code abfragen kann:

```
val id: Int = R.drawable.drawableName  
<context>.getDrawable(id)
```

- Die Funktion `getItemCount` liefert lediglich die Größe der `dataset` Liste zurück, programmiere diese Funktionalität
- Abschließend müssen wir die erstellten Klassen nur noch in der `MainActivity` Klasse zusammenbringen
 - Hole die Liste aus Filmtiteln, indem du die Funktion `loadTitles` aus der Klasse `Datasource` aufrufst
 - Erstelle ein neues `ItemAdapter` Objekt und weise es dem `adapter` Attribut der `RecyclerView` zu
 - Weise dem Attribut `LayoutManager` einen geeigneten Layoutmanager zu

Hinweis: übergebe hier als Kontext immer `this`

- Führe die App aus, wenn alles richtig implementiert ist, solltest du nun eine Filmtitelliste in der `RecyclerView` sehen!

Viel Erfolg!



2. FILME STREAMING SERVICE - RecyclerView

In dieser Bonusaufgabe fügen wir den Listeneinträgen einen Like-Button hinzu, mit dem die Filme als Favorit markiert werden können.



- Arbeite weiter in dem Projekt aus Aufgabe 1
- Ergänze das `list_item` Layout um einen `ImageButton` und passe die Attribute entsprechend an. Die beiden Bilder für die beiden Zustände findest du im Resource Manager
- Passe die `MovieTitle` Klasse an, indem du ihr im Konstruktor einen weiteren Boolean Parameter hinzufügst, in dem der Zustand des Like-Buttons gespeichert ist
- Füge den `MovieTitle` Objekten in der Klasse `Datasource` den zweiten Parameter hinzu, den du mit `false` initialisierst
- Passe schlussendlich den Adapter an, in dem du die Klasse `ItemAdapter` anpasst
 - Füge in der inneren Klasse einen Parameter für den Like-Button ein
 - Hole in der Funktion `onBindViewHolder` zusätzlich zur `TextView` auch den Like-Button aus dem `ViewHolder` Objekt
 - Setze das richtige Bild für den Like-Button, abhängig von dem im `MovieTitle` Objekt gespeicherten Boolean Wert
 - Richte für den Like-Button einen `onClick`Listener ein, in dem der Boolean Wert umgekehrt wird und ebenfalls das Bild angepasst wird

Hinweis: Um Code-Dopplung zu vermeiden, eignet sich hierfür eine kleine Hilfsfunktion, die für den Bildwechsel verantwortlich ist

- Führe die App aus, jeder Listeneintrag sollte nun einen Like-Button besitzen und dessen Wert auch richtig anzeigen, wenn der Listeneintrag erst außerhalb des Bildschirms und dann wieder auf den Bildschirm gescrollt wird

Viel Erfolg! 