Syntax Institut

Modul 3 – Android App Entwicklung mit Kotlin

RecyclerView II

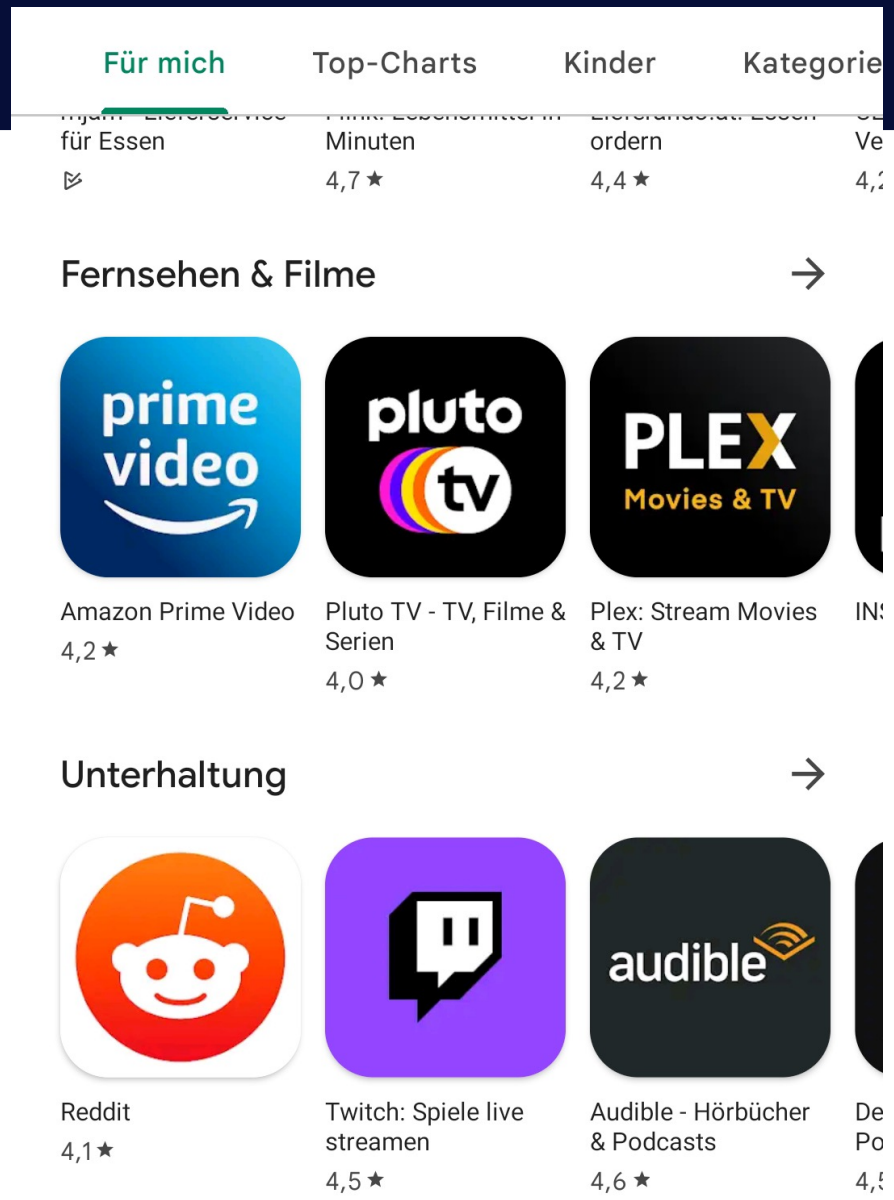
Gliederung

- Was war nochmal RecyclerView?
- Was haben wir dafür gebraucht?
- Wie wurde sie programmiert?

Quelle: https://www.memesmonkey.com/topic/recycling#&gid=1&pid=2

Was ist RecyclerView?

- Darstellung von Listen innerhalb der App
- Scrollt auch flüssig bei langen Listen mit komplexen Einträgen
- berechnet nur Einträge die gerade wirklich gezeigt werden



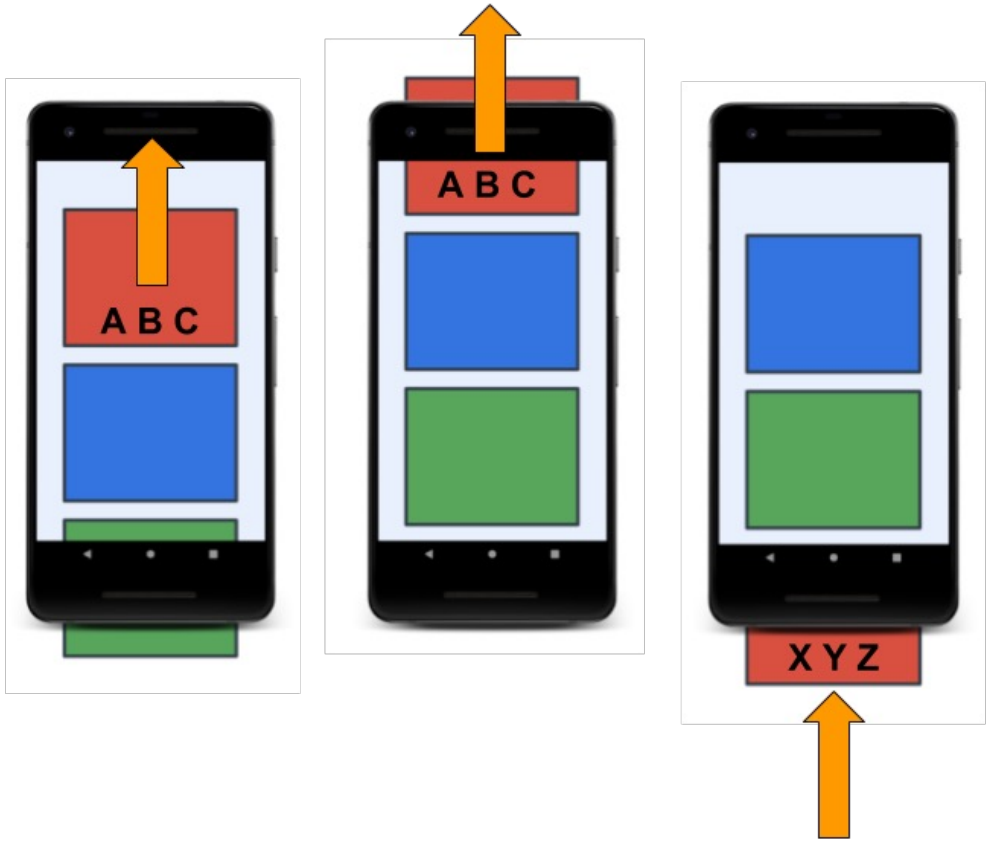
Wie funktioniert RecyclerView?

Wenn das rote Quadrat den Screen

verlässt wird die View für das nächste

Quadrat wiederverwendet

Nur der Text wird abgeändert



Quelle: https://developer.android.com/codelabs/basic-android-kotlin-training-recyclerview-scrollable-list/img/dcf4599789b9c2a1.png

Elemente der RecyclerView

- Items

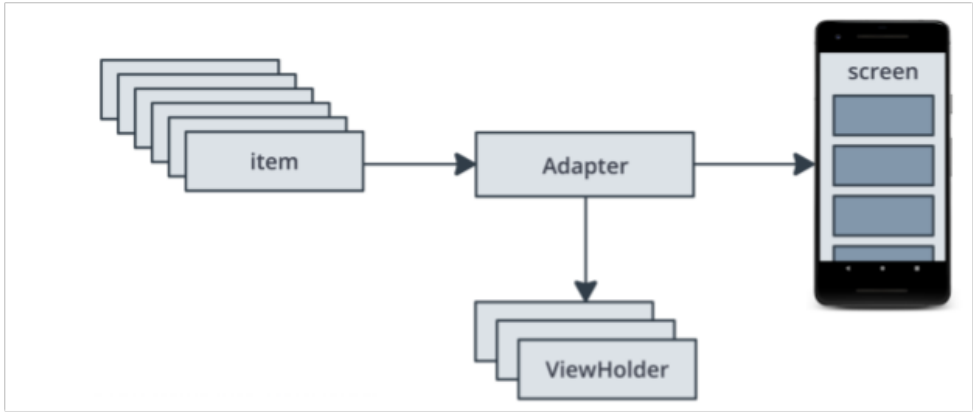
  Gegenstände einer Liste die Dargestellt werden sollen

- Adapter

  bereitet items für deren Verwendung innerhalb der RecyclerView vor

- ViewHolder

  hält items innerhalb der RecyclerView und kann immer neu befüllt werden



Quelle: https://developer.android.com/codelabs/basic-android-kotlin-training-recyclerview-scrollable-list/img/4e9c18b463f00bf7.png

Anti-Witz Liste

- Packages um Code zu organisieren
- Liste an Witzen
- Layout mit RecyclerView
- Layout für die Darstellung eines Witzes
- RecyclerView-Adapter samt ViewHolder

Neues Beispiel: Reisebüro

- Packages um Code zu organisieren
- Liste an Reisezielen (Angebot und Bild)
- Layout mit RecyclerView (GridLayoutManager)
- Layout für die Darstellung eines Reiseziels
- RecyclerView-Adapter samt ViewHolder

Data Class

Spezielle Klasse zum Speichern von Daten

- besitzt automatisch toString(), equals(), copy() Methoden
- kann keine Kinder haben

```
data class Vacation(
    val stringResource: Int,
    val imageResource: Int
)
```

Data Source

Ist für die Verwaltung von Daten zuständig

Bereitet Daten von verschiedenen Quellen und Formaten auf damit es für den Rest der App einheitlich ist

```
class Datasource {

    fun loadVacations(): List<Vacation> {
        return listOf(
            Vacation(R.string.preis1, R.drawable.urlaub1),
            Vacation(R.string.preis2, R.drawable.urlaub2),
            Vacation(R.string.preis3, R.drawable.urlaub3),
            Vacation(R.string.preis4, R.drawable.urlaub4),
            Vacation(R.string.preis1, R.drawable.urlaub5),
            Vacation(R.string.preis2, R.drawable.urlaub6),
            Vacation(R.string.preis3, R.drawable.urlaub7),
            Vacation(R.string.preis4, R.drawable.urlaub8),
            Vacation(R.string.preis1, R.drawable.urlaub9),
            Vacation(R.string.preis2, R.drawable.urlaub10),
            Vacation(R.string.preis3, R.drawable.urlaub11),
            Vacation(R.string.preis4, R.drawable.urlaub12),
            Vacation(R.string.preis1, R.drawable.urlaub1),
            Vacation(R.string.preis2, R.drawable.urlaub2),
            Vacation(R.string.preis3, R.drawable.urlaub3),
            Vacation(R.string.preis4, R.drawable.urlaub4),
            Vacation(R.string.preis1, R.drawable.urlaub5)
        )
    }
}
```

Recycler View

Kann ganz normal wie andere Elemente ins Layout eingefügt werden

- `layoutManager=”GridLayoutManager“`

  die Elemente werden in einem Grad angeordnet

- `spanCount=”2“`

  das GridLayout hat 2 Spalten

```
<androidx.recyclerview.widget.RecyclerView
    android:id="@+id/recyclerView"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:padding="16dp"
    app:layoutManager="androidx.recyclerview.widget.GridLayoutManager"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    app:spanCount="2"
    tools:itemCount="16"
    tools:listitem="@layout/list_item" />
```

Item Layout

Ein neues **list_item.xml** File bestimmt das Aussehen eines Reiseziels

```xml
<?xml version="1.0" encoding="utf-8"?>
<com.google.android.material.card.MaterialCardView
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"

    android:id="@+id/item_cardView"
    android:layout_width="150dp"
    android:layout_height="150dp"
    android:layout_margin="16dp"
    app:cardCornerRadius="10dp"
    app:cardElevation="5dp"
    app:cardPreventCornerOverlap="false">

    <androidx.constraintlayout.widget.ConstraintLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent">

        <ImageView
            android:id="@+id/item_image"
            android:layout_width="0dp"
            android:layout_height="0dp"
            android:layout_marginStart="4dp"
            android:layout_marginTop="4dp"
            android:layout_marginEnd="4dp"
            android:layout_marginBottom="24dp"
            android:adjustViewBounds="true"
            android:cropToPadding="false"
            android:scaleType="centerCrop"
            app:layout_constraintBottom_toBottomOf="parent"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintTop_toTopOf="parent"
            tools:srcCompat="@drawable/urlaub1" />

        <TextView
            android:id="@+id/item_text"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginEnd="8dp"
            android:textColor="@color/onSurface"
            android:textSize="12sp"
            android:textStyle="bold"
            app:layout_constraintBottom_toBottomOf="parent"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintTop_toBottomOf="@+id/item_image"
            tools:text="@string/preis2" />
    </androidx.constraintlayout.widget.ConstraintLayout>
</com.google.android.material.card.MaterialCardView>
```

Adapter Klasse

Organisiert mit Hilfe der ViewHolder Klasse das Recycling

```
// der Adapter braucht den Context um auf String Resourcen zuzugreifen
// und die Liste an Jokes um sie für die RecyclerView vorzubereiten
class ItemAdapter(
    private val context: Context,
    private val dataset: List<Vacation>
) : RecyclerView.Adapter<ItemAdapter.ItemViewHolder>() {

    // der ViewHolder weiß welche Teile des Layouts beim Recycling angepasst werden
    class ItemViewHolder(private val view: View) : RecyclerView.ViewHolder(view) {
        val textView: TextView = view.findViewById(R.id.item_text)
        val imageView: ImageView = view.findViewById(R.id.item_image)
    }

    // hier werden neue ViewHolder erstellt
    override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): ItemViewHolder {

        // das itemLayout wird gebaut
        val adapterLayout = LayoutInflater.from(parent.context)
            .inflate(R.layout.list_item, parent, attachToRoot: false)

        // und in einem ViewHolder zurückgegeben
        return ItemViewHolder(adapterLayout)
    }

    // hier findet der Recyclingprozess statt
    // die vom ViewHolder bereitgestellten Parameter werden verändert
    override fun onBindViewHolder(holder: ItemViewHolder, position: Int) {
        val item = dataset[position]
        holder.textView.text = context.resources.getString(item.stringResource)
        holder.imageView.setImageResource(item.imageResource)
    }

    // damit der LayoutManager weiß wie lang die Liste ist
    override fun getItemCount(): Int {
        return dataset.size
    }
}
```

MainActivity

- Zuerst werden die Reisziele in einer Variable gespeichert
- Dann wird die RecyclerView vom Layout mit dem Code verknüpft
- Unser Adapter wird erschaffen und der RecyclerView übergeben
- Da unsere RecyclerView eine fixe Größe hat kann die Performance nochmals geboostert werden

```
class MainActivity : AppCompatActivity() {

    private lateinit var binding: ActivityMainBinding

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        binding = DataBindingUtil.setContentView( activity: this, R.layout.activity_main)

        // Liste an Reisezielen wird von der Datasource geladen
        val vacations = Datasource().loadVacations()

        // recyclerView von Layout wird mit code verknüpft
        val recyclerView = binding.recyclerView

        // ItemAdapter wird als Adapter festgelegt
        recyclerView.adapter = ItemAdapter( context: this, vacations)

        // verbesserte Performance bei fixer Größe
        recyclerView.setHasFixedSize(true)
    }
}
```

Fertig

Quelle: https://www.pinterest.com/pin/492581277965142379/

RecyclerView

Wiederholung - Was haben wir heute gelernt?

1	Was ist RecyclerView?
2	Wie funktioniert sie?
3	Wie wird sie programmiert?

Syntax Institut

Quelle: https://www.quarks.de/umwelt/muell/das-solltest-du-ueber-recycling-wissen/

Viel Spaß!