Android 3

Importieren von neuen Bibliotheken

Es können neue Bibliotheken in das Projekt ganz einfach über das Gradle-File importiert werden. Gradle ist das Buildsystem und kümmert sich um das Bauen des Projektes. Eine neue Library kann über das `build.gradle`-File importiert werden, indem die neue Abhängigkeit als `dependency`, wie im Beispiel unten gezeigt, hinzugefügt wird.

```
dependencies {
    // ...
    implementation 'com.google.android.material:material:1.1.0'
    // ...
}
```

Implementierung einer eigenen Tabnavigation

Zuerst muss die Abhängigkeit `'com.google.android.material:material:1.1.0'` installiert werden. Danach können im MainActivity XML-File die Views der Bibliothek verwendet werden.

```
<com.google.android.material.bottomnavigation.BottomNavigationView
        android:id="@+id/bnvMain"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_alignParentBottom="true"
        app:menu="@menu/main_menu"
        />
```

Das `app:menu` muss auf eine Ressource zeigen, dass dem `menu`-Typ entspricht. In diesem neu erzeugten `menu`-Ressourcen Ordner können neue Menüs hinzgefügt werden.

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android">
    <item
        android:title="Locations"
        android:icon="@drawable/ic_location_city_black_24dp" />
    <item
        android:title="Map"
        android:icon="@drawable/pin" />
</menu>
```

Adressierung von Elementen

Activities die eine zugewiesene ID im XML-File haben (`android:id=@+id/someId`) können über den Java-Code in der zugehörigen Java-Implementierung adressiert werden. Mit der Methode `findViewById` können Elemente über deren ID gesucht werden. Diese Methode gibt eine Referenz auf das Objekt zurück, die das Element im XML referenziert, wie im Beispiel unten gezeigt:

```
TextView textView = findViewById(R.id.tvMain);

BottomNavigationView bottomNavigationView = findViewById(R.id.bnvMain);

bottomNavigationView.setOnNavigationItemSelectedListener(item -> {
    textView.setText(item.getTitle());
    return true;
});
```

Im oben gezeigten Code werden die Referenzen zu den Views von einem Textview und der Navigationsleiste erstellt. Danach wird ein sogenannter Action-Listener auf die Navigationsleiste erzeugt, der immer aufgerufen wird, wenn ein neues Element in der Leiste ausgewählt wird. Danach wird dem Textelement der Titel des Tabs zugewiesen.

Context

Ein Context gibt die Möglichkeit auf die Ressourcen und globale Informationen zuzugreifen. Ein Context ist entweder eine Activity oder eine Application.

Application

Es gibt die Möglichkeit eine Klasse zu erstellen, die die Applikation darstellt. Man muss von der Application ableiten und kann im `android:name=".MyCustomApplication"`-Attribut des `Application`-Tag instanzieren. Sobald die Applikation gestartet wird, wird die Klasse instanziert und lebt bis die Applikation beendet wird. Sie kann hilfreich sein, um Crash-Reports zu generieren oder Ressourcen, die Activity übergreifend verwenden sollen.

Liste von Elementen

Listen können nicht mit unendlich vielen Elementen erstellt werden, da sie aufgrund des hohen Speicherverbrauchs zu einem Crash führen würden. Deswegen braucht man einen sogenannten Recycler-View, der nur eine bestimmte Anzahl an Elementen darstellt. Für das Recycling wird eine Klasse benötigt, die vom `RecyclerView.Adapter<T>` ableitet, wobei `T` eine Klasse ist, die einen ViewHolder repräsentiert. Der Viewholder bestimmt, wie das Element auszusehen hat und der Adapter selbst kümmert sich um das Recycling. Der Viewholder wird vom Adapter erzeugt.