Mobile_Application_Development_lab

Lab Notes von Philipp Wolfmajer & Matthias Schreiner

22.02.2019 - Lab 1

- Überblick über Android allgemein
- Entwicklungsumgebung eingerichtet (Android Studio)
- AndroidManifest.xml angesehen
- Graddle Module:app (betrifft nur aktuelle Modul)
    - applicationid ist wichtig für den Play Store, bei Änderung bedeutet das -> neue App
    - dependencies
- Projektweites Graddle File
    - wenn z.B. neues Android Wear App dazu entwickelt wurde
- unterschiedliche Events die eine App haben kann (onCreate, etc.)
- Activities sind Ansichten der App
- Intent -> sind Nachrichten die andere Activities aufrufen
    - Intent mit Result verwendet beim Aufruf immer einen Request Code damit man das Resultat zuordnen kann (z.B. Pfad zu einem Bild)
- Actions -> Android kümmert sich um das App das eine Aktion ausführt (z.B. ACTION_CALL)

01.03.2019 - Lab 2 Online

- Starten mit AirKoality APP
    - Leere App von Grund erstellen (keine Vorlage gewählt)
- Importieren der notwendigen Ressourcen von der e-learning Plattform
- Anlegen und implementieren folgender Klassen
    - MainActivity.java
    - SplashActivity.java – Ladebildschirm mit Verzögerung (delay)
- Anlegen und implementieren folgender Ressourcen
    - activity_main.xml
    - activity_splash.xml
- Editieren der Farben, Strings und Styles in den jeweiligen Files unter /res/values/
- Hinzufügen der Activities in der AndroidManifest.xml Datei
- strings.xml kennengelernt um Texte einer App zentral verwalten zu können

08.03.2019 - Lab 3

Context -> weiß was in der App gerade "abgeht" (Java Objekt)

- Activity oder Application
- Immer Context der aktuellen Activity verwenden

Best Practise kennengelernt um große Listen darzustellen

Liste bekommt einen Adapter und stellt dann nur Listenelemente dar, die gerade am Bildschrim angezeigt werden.

Card View Layout ausprobiert

Toast Messages implementiert -> Pop up Nachrichten im unteren Bereich des Bildschirms

15.03.2019 - Lab 4

RecyclerView -> Items der View wiederverwendbar machen - vorsicht bei ausgewählten Items (Adapter muss sich Position merken) - onBindViewHolder: Selection über Index merken und Hintergrundfarbe je nach Index anpassen, da die RecyclerView wieder verwednet wird.

Fragments

- Sind gekapselte Elemente, damit keine unnötigen Elemente geladen werden. (Details siehe Folie)
- ursprünglich für Tablets entwickelt
- Darstellung von z.B. Listen und Inhalt gleichzeitig
- Fragments erstellen (Klassen und Layout) und ManActivity.java auseindader teilen

Storage

- unterschiedliche Möglichkeiten der Speicherung
    - local storage (im App Verzeichnis)
    - external storage
    - shared preferences (key - value pairs, internes XML file)
    - SQLite -> single file database
    - Permissions beim Speichern beachten
    - mit SharedPreferences das letzte Fragment laden

13.04.2019 - Lab 5

- SQ-Lite implementierung angeschaut aber unser Projekt wird auf Room aufgesetzt
    - Room arbeitet mit Annotationen -> viel weniger Code im vergleich zu SQ-Lite
- Network
    - HttpsGet Methoden implementiert (waren vorgegeben)

10.05.2019 - Lab 5

- Unterschied Service - Background Task
    - Service: Wenn etwas im Hintergrund laufen muss (z.B. Nuki App)
    - BT: z.B Musik Player, immer wenn der User nicht mit der App Interagieren will

Service sollte immmer eine Aktion die er ausführt in einem extra Thread auslagern.

Service wird im Manifest deklariert Um einen Foregroundservice zu starten muss das innerhalb von 5 Sekunden nach "start Service" (Intent)

Notifications benötigen immer einen Channel und Channel ID (ab Android O verfügbar)

17.05.2019 - Online

Map Fragment implementiert

• Neue Permission ACCESS_FINE_LOCATION für GPS Zugriff-> wieder in AndroidManifest.xml

ActivityMain • Überprüfung Permission bei startLocationService hinzufügen

18.05.2019 - Online

- Clustering von Map Fragments um Performance der App zu optimieren -> Je nach Zoomfaktor werden MapMarker gruppiert. (ClusterManager)
- Content Provider kennengelernt
    - wird verwendet um Daten für andere Apps bereitzustellen (z.B. Kontakte)
    - besteht aus URI - Content Resolver - Content Provider - Persistierung
    - Messdaten aus AirKoality in andere App übertragen (als PNG) [App die den Intent entgegennimmt muss den Filtype unterstützen]
- Wie kann eine App published werden?
    - Apk-File direkt (unsicher und unschön)
    - Market Places
        - Google Play
        - Amazon
    - Um im Play Store zu publishen muss man einen Developer Account anlegen (einmalige Zahlung von 25$)
    - max. 100MB
    - Target API ab Version 26
    - App signing
        - App wird mit einem Zertifikat signiert (Release oder Debug)
        - alle folgenden Updates der App müssen mit dem gleichen Key signiert werden