30.6.2017

App: Categorizer

Projektdokumentation

Hochschule Bremerhaven – University of Applied Sciences
Studiengang Wirtschafts-/Informatik

Sommersemester 2017

Veranstaltung: Programmierung mobiler Endgeräte

Dozent: Dipl. Ing. Wolfgang Wissel

Autoren: Kevin Kussyk, 32603, kevin.kussyk@abat.de
Thomas Fahrenholz, 32460, schrul.thomas@gmail.com

Inhaltsverzeichnis

1. Einleitung

Das vorliegende Dokument wurde von Kevin Kussyk und Thomas Fahrenholz verfasst, ist die Projektdokumentation zum Projekt der Wahlpflichtveranstaltung „Programmierung mobiler Endgeräte“ und stellt zusammen mit der Abgabe des Projekts die Prüfungsleistung für dieses Modul dar.

Die Auswahl eines Themas für das Projekt ist nicht beschränkt, muss aber ein paar Kriterien erfüllen. Anforderungen an den Projektinhalt sind die Folgenden:

- Saubere Gliederung der Pakete
- Zentrale Speicherung von Beschriftungen
- Es sollte zudem vorhanden sein:
    - Liste
    - Intent
    - Datenbank
    - Fragment
    - Möglichkeit Eingabefehler abzufangen

Das Projekt soll insgesamt als *.rar Datei abgegeben werden.

Wir haben uns für das Thema Kategorisierung entschieden, wobei unsere App unter dem Namen „Categorizer“ geführt wird. Die Details und Features zur App werden im weiteren Verlauf dieser Dokumentation genauer aufgeführt.

In dieser Dokumentation werden zu Beginn die Projektorganisation und das Produkt kurz beschrieben. Darüber hinaus werden u.a. noch Produktanforderungen, sowie die Produktkomponenten detailliert beschrieben. Abschließend gibt es noch eine Reflexion unsererseits zum Projekt.

2. Projektorganisation

2.1. Teamorganisation

Um die Projektarbeit im Team besser organisieren und uns absprechen zu können, haben wir das Kommunikationsmedium „WhatsApp“ verwendet. Hierrüber haben wir die Zeitpunkte für unsere Treffen zum Projekt festgelegt und Statusaktualisierungen über Arbeitsergebnisse oder den Arbeitsfortschritt ausgetauscht.

2.2. Tools und Techniken

Neben der Entwicklungsumgebung, haben wir für eine konsistente und versionierte Datenablage den Dienst GitHub (GitHub Inc, 2017) verwendet. Um das Projekt mit GitHub zu verbinden und zu verwalten haben wir entsprechende Tools eingesetzt. Weitere Informationen zur Entwicklungsumgebung werden im gleichnamigen Kapitel dieser Dokumentation aufgeführt.

3. Produktbeschreibung

Die App „Categorizer“ dient der einfachen Erstellung von Kategorien verschiedener Themen für Benutzer eines Android Systems und der Zuordnung diverser Items zu diesen Kategorien.

Dabei können Kategorien und Items über Gesten, Interaktionsdialoge oder Menüs bearbeitet, erstellt oder gelöscht werden. Des Weiteren erhält der Benutzer je nach Aktion oder Fehler entsprechend eine Rückmeldung bzw. Benachrichtigung.

Der Benutzer kann die App sowohl im Portrait als auch im Landschaftsmodus verwenden. Zudem passen sich die Inhalte, sowie Schriftgrößen der Displaygröße, Pixeldichte bzw. Auflösung dynamisch an.

Darüber hinaus ist es möglich im Einstellungsmenü das Farbschema zu ändern oder die Sprache zwischen Deutsch und Englisch umzuschalten.

Die App Daten werden in einer internen Datenbank lokal abgespeichert.

Eine detaillierte Beschreibung der Features und Anforderungen der App, sowie der Teilkomponenten können den Kapiteln „Anforderungsdefinition“ und „Detaillierte Beschreibung der Produktkomponenten“ entnommen werden.

4. Entwicklungsumgebung

Für die Entwicklung unserer Android App verwendeten wir die vorgegebene Entwicklungsumgebung Android Studio von Google Inc auf unseren Windows Laptops. Um die App auf unseren eigenen und aktuellen Geräten verwenden zu können, haben wir die Android SDK's von Android 6.0 (Marshmallow) bis Android 7.1.1 (Nougat) und somit API Level 23 – 25 dazu installiert.

Zum Testen der App haben wir zum einen ein virtuelles Android Gerät im Android Virtual Device Manager verwendet und zum anderen hauptsächlich unsere eigenen Hardware Geräte: ein Samsung Galaxy S6 Edge und ein Huawei P8 Lite.

Damit wir auch sinnvoll getrennt voneinander programmieren, unsere Ergebnisse einfach zusammenführen aber auch versionsbasierte Projektzustände erstellen konnten, haben wir uns für die Versionsverwaltung unseres Projektes mit GitHub entschieden. Dazu haben wir für unser Projekt zunächst ein Repository und verschiedene Branches auf GitHub erstellt. Anschließend konnten wir unsere Arbeitsfortschritte immer mit dem Server synchronisieren, verschiedene Versionen der App erstellen, Ergebnisse zusammenführen und ggf. zu einem älteren Projektstand zurückkehren.

5. Anforderungsdefinition

In diesem Kapitel werden unsere eigenen Anforderungen an unsere App „Categorizer“ festgehalten.

5.1. Schnittstellen

5.1.1. Benutzerschnittstellen

Das Display des jeweiligen Smartphones oder aber auch Tablets des Benutzers ist die Schnittstelle der App zum Benutzer. Hierrüber werden die Daten der App visualisiert und der Benutzer kann darüber mit der App interagieren.

5.1.2. Hardwareschnittstellen

Die App wird auf einem Smartphone oder Tablet installiert. Dabei werden die App-Daten lokal gespeichert.

5.1.3. Softwareschnittstellen

Die App wird unter einem Android Betriebssystem ausgeführt.

5.2. Produktfeatures

In diesem Abschnitt werden die Features der App kurz aufgelistet:

- Der Benutzer kann verschiedene Kategorien erstellen.
- Der Benutzer kann verschiedene Items den einzelnen Kategorien hinzufügen.
- Kategorien besitzen ein Popup-Menü, worüber neue Items hinzugefügt oder die Kategorie bearbeitet oder gelöscht werden kann.
- Kategorien und Items innerhalb der Kategorien können bearbeitet und gelöscht werden. Hierbei kann der Löschvorgang durch langes Drücken auf das jeweilige Item oder durch das Popup-Menü (bei Kategorien) bzw. dem Löschsymbol (bei Items) gestartet werden.
- Die Bearbeitung der Kategorie findet in einem separaten Fenster statt. Dort werden zudem zusätzlich die bisherigen Items der Kategorie aufgelistet.
- Ein Item kann durch Drücken bearbeitet werden. Die Bearbeitung findet ebenfalls in einem separaten Fenster statt.
- Neben dem Titel eines Items, kann man diesem noch eine textuelle Beschreibung hinzufügen.
- Änderungen im Bearbeitungsmodus werden durch die „Zurück“-Taste verworfen oder durch Betätigen des Speichersymbols gespeichert.

- Die Inhalte der Kategorien können ein- und ausgeklappt werden.
- Die App ist sowohl im Portrait als auch im Landschaftsformat nutzbar.
- Die Anzahl der möglichen Kategorien und Items ist nicht limitiert.
- Über ein Aktionsmenü werden die Kategorien erstellt. Ebenfalls können hierüber die Items erstellt und der jeweiligen Kategorie zugeordnet werden.
- Im Hauptmenü der App kann man die App-Einstellungen oder Informationen zu den App-Autoren aufrufen.
- In den App-Einstellungen kann das Farbschema (Nachtmodus) verändert und die Sprache zwischen Deutsch und Englisch umgestellt werden.
- Beim Wechsel des Farbschemas passen sich alle Elemente dementsprechend an.
- Nach verschiedenen App-Ereignissen (z.B. dem Erstellen einer Kategorie) werden dem Benutzer entsprechende Benachrichtigungen angezeigt.
- An geeigneten Stellen in der App werden dem Benutzer Informations-, Warnungs-, Alarm- oder Interaktionsdialoge angezeigt.
- Je nach Displaygröße, Pixeldichte oder Auflösung des Zielgerätes werden die Schriftgrößen dynamisch angepasst.

5.3. Softwaresystemattribute

5.3.1. Benutzerfreundlichkeit

- Die Schriftgrößen sollen für den Benutzer leserlich sein.
- Die Menüeinträge und die Navigation zu diesen sollen für den Benutzer so einfach und verständlich wie möglich gehalten werden.
- Die allgemeine Bedienung der App soll der aktuellen Apps angelehnt sein und somit dem Benutzer vertraut vorkommen.
- Der Benutzer wird von der App an notwendigen Stellen Rückmeldung zu seinen getätigten oder zu tätigenden Aktionen erhalten.
- Der Nachtmodus soll die App-Farben dem dunkleren Umfeld so anpassen, dass die Augen des Benutzers weniger strapaziert werden.
- Der Benutzer kann zwischen den Sprachen Deutsch und Englisch wählen.
- Ist die Systemsprache nicht Deutsch soll die Sprache der App standardmäßig auf Englisch umgeschaltet werden.

5.3.2. Zuverlässigkeit

- Bei möglichen Fehlerfällen bzw. nicht durchführbaren Operationen wird dem Benutzer eine entsprechende Rückmeldung gegeben.

5.3.3. Verfügbarkeit

- Der Zugriff auf die App wird gewährleistet sein, solange das Zielgerät in Betrieb ist.

5.3.4. Sicherheit

- Konkrete Sicherheitsaspekte der App werden nicht betrachtet, da diese nicht Teil der Modulveranstaltung sind.
- Der Zugriff auf die App und die dort enthaltenen Daten werden von den Sicherheitseinstellungen des Gerätes durch den Benutzer geregelt.
- Eine Kategorie mit dem gleichen Namen einer bereits vorhandenen Kategorie, wird nicht erstellt werden können.
- Ein Item wird nur erstellt werden können, sofern bereits mindestens eine Kategorie existiert.
- Es werden keine Titel für Kategorien oder Items erlaubt, die leer sind oder nur aus Leerzeichen bestehen.
- Überflüssige Leerzeichen am Anfang und am Ende eines Titels werden entfernt.
- Es werden keine Zeilenumbrüche in Titeln erlaubt.
- Die Titellängen werden auf maximal 20 Zeichen limitiert.

5.3.5. Wartbarkeit

- Das Modul wird für projektfremde Personen (vermutlich) schwer zu warten sein, da man sich dafür dementsprechend in das Projekt reinarbeiten muss.

5.3.6. Portabilität

- Die App wird primär für Android Betriebssysteme der Versionen 6.0 (Marshmallow) bis 7.1.1 (Nougat) bzw. API Level 23-25 entwickelt sein.
- Die App soll (im besten Fall) auf allen Displaygrößen, Auflösungen und Pixeldichten der oben genannten Versionen funktionsfähig sein.
- Die App wird primär auf unseren eigenen Smartphones (Samsung Galaxy S6 Edge und Huawei P8 Lite) getestet.

5.4. Datenbank Anforderungen

Der Benutzer des Smartphones bzw. Tablets hat dafür Sorge zu tragen, dass ausreichend Speicher für die App und der internen Datenbank auf dem Gerät zur Verfügung steht. Ebenso ist dieser dafür verantwortlich, die Datensicherheit seiner Apps zu gewährleisten und den Zugang zum Speicherort der App unberechtigten Personen zu verwehren.

Darüber hinaus werden die Daten der App mittels der internen Datenbankanwendung SQLite verwaltet.

6. Detaillierte Beschreibung der Produktkomponenten

6.1. Projektstruktur

Wir haben für unsere Projektordnerstruktur für verschiedene Dateiarten oder Ressourcen die Namen so erweitert, sodass die Kategorie bzw. Zuordnung alleine schon aus dem Namen der Datei erkennbar ist. Beispielsweise erhalten alle Java Klassen am Ende ihres eigentlichen Namens die Bezeichnung „Class“ falls es sich um eine „normale“ Java Klasse handelt oder die Bezeichnung „Activity“ falls es eine Activity ist. Ebenso existieren extra Bezeichnungen für die XML Dateien im „res/layout“ Ordner. So steht z.B. jeder Activity ein „activity_“ voran und jedem Layout ein „layout_“.

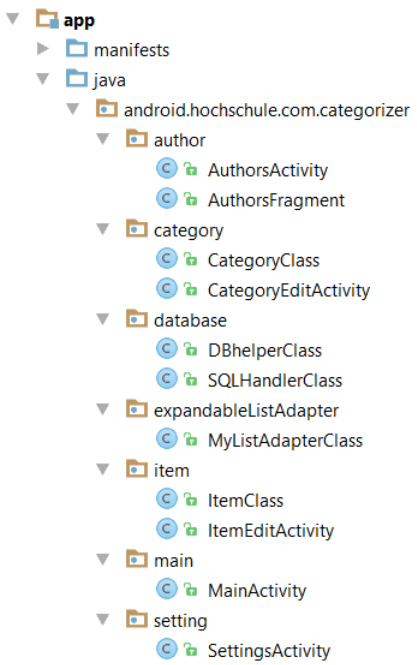

Abbildung 1 Projektstruktur

Darüber hinaus haben wir die Java Klassen kategorisiert und in entsprechende Pakete untergliedert (siehe Abbildung 1).

Weiterhin enthalten alle Klassen in Form eines Kommentars zu Beginn des Codes eine kurze Information darüber, wofür diese Klasse genau da ist bzw. was sie tut. Zudem sind Funktionen und ausgewählte Zeilen im Code zur besseren Verständlichkeit kommentiert.

6.2. Kategorien

Eine Kategorie besitzt einen Namen bzw. einen Titel und eine Liste von Items, welche dieser Kategorie angehören. Diese Informationen sind in der Java Klasse „CategoryClass“ definiert. Zudem implementiert die Klasse das Interface „Parcelable“ und die dazugehörigen Funktionen, um in der App Kategorien zwischen verschiedenen Activities senden und empfangen zu können.

6.3. Items

Items besitzen ebenfalls einen Namen bzw. einen Titel und dazu noch eine Beschreibung. Diese Informationen sind in der Java Klasse „ItemClass“ definiert und ebenso wie die Kategorien, implementiert diese Klasse auch das Interface „Parcelable“, um Items zwischen Activities senden und empfangen zu können.

6.4. Hauptmenü

In der App existiert ein Hauptmenü, welches sich in der sog. „Action Bar“ also der oberen Leiste der App befindet. Hierüber können zwei Menüeinträge aufgerufen werden (siehe Abbildung 2):

- Einstellungen
- Autoren

Die Einstellungen werden im nächsten Kapitel genauer erläutert. Der Eintrag „Autoren“ öffnet die „AuthorsActivity“ in der in einem Fragment die Entwickler der App aufgelistet werden.

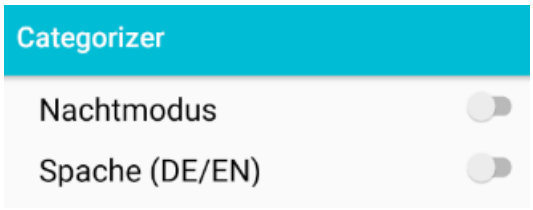

Abbildung 2 Einstellungen

6.5. Einstellungen

Im Einstellungsmenü besteht die Möglichkeit den Nachtmodus zu aktivieren und die Sprache zwischen Deutsch und Englisch umzuschalten.

6.5.1. Nachtmodus

In der „SettingsActivity“ besteht die Möglichkeit einen Nachtmodus für die App zu aktivieren. Da die App standardmäßig mit hellen Farben ausgelegt ist, kann es zu späterer Stunde den Bediener evtl. zu sehr blenden. Deshalb werden im Nachtmodus alle weißen und türkisen Farben durch verschiedene Grautöne ersetzt (siehe Abbildung 3). Zudem ändert sich die Schriftfarbe in den meisten Situationen von Schwarz zu Weiß. Der Nachtmodus kann jederzeit vom Benutzer deaktiviert oder aktiviert werden, wobei dies über einen „Switch“ geändert werden kann. Sobald der Modus geändert wird, ist es notwendig, dass die Activity aktualisiert wird, um die Farben anzupassen. Da die Activity durch den Hardware „Zurück Button“ nicht vollständig neu geladen wird, wird beim Wechsel des Farbschemas die „MainActivity“ aufgerufen und dem Benutzer zusätzlich zur optischen Farbänderung der App ein Toast mit der Information dieser Farbänderung angezeigt.

Um das Farbschema zu ändern, werden sog. Styles verwendet. Hierfür haben wir neben dem Basis Style noch zusätzlich eins für den Nachtmodus in der „styles.xml“ angelegt und den entsprechenden Elementen Farben aus der „colors.xml“ zugewiesen. Das Style muss in der „onCreate“ Methode einer Activity gesetzt

werden. Wichtig dabei ist, dass das Style gesetzt wird bevor man die „Content View“ setzt. Dies muss in jeder Activity stattfinden, wo das Style eingesetzt werden soll.

Um sich den Modus zu merken, wird die „SharedPreferences“-Klasse genutzt. Sie dient dazu, Einstellungen beim Neustarten der App zu laden und beim Beenden zu speichern. Die entsprechende Variable wurde in der „MainActivity“ erzeugt und wird von jeder Activity importiert, damit alle auf dieselbe Variable zugreifen können.

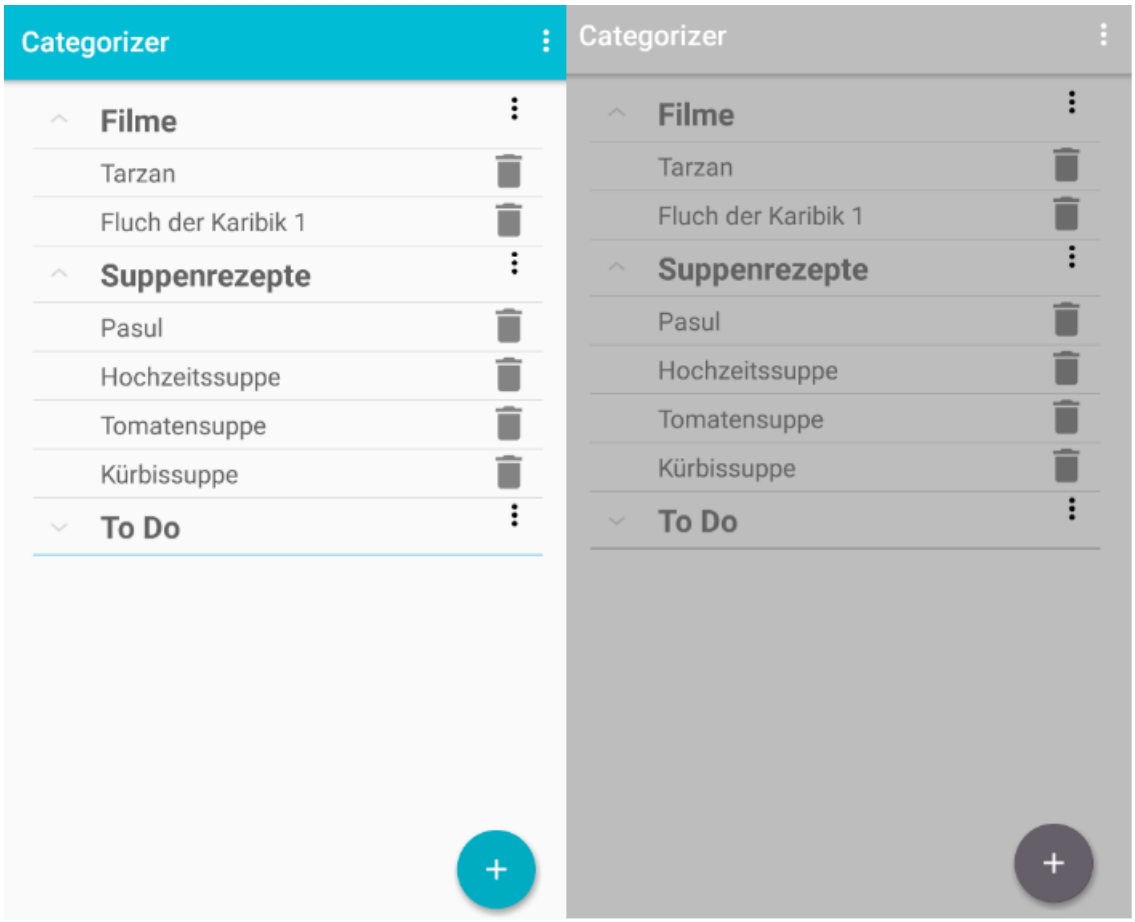

Abbildung 3 Beide Farbmodi

6.5.2. Sprache

Damit der Benutzer bei Bedarf die App nicht nur in der deutschen Sprache sondern auch in der englischen Sprache bedienen kann, existiert ein weiterer Switch in den Einstellungen für den Wechsel der Sprache. Mit dem Switch kann man die App entsprechend auf Englisch bzw. Deutsch umstellen. Ähnlich wie bei dem Nachtmodus muss die „MainActivity“ hierfür neu gestartet werden, damit die Änderung wirkt. Über den Sprachwechsel wird der Benutzer ebenfalls mittels einem Toast informiert. Auch wurde wieder die „SharedPreferences“-Klasse verwendet, um die Spracheinstellung zu der App zu speichern.

Unter AndroidStudio ist es relativ leicht, verschiedene Übersetzungen zu der vorhandenen „strings.xml“ bzw. Standardsprache zu erstellen. Ein Problem hierbei ist aber, dass die App-Sprache über die Systemsprache des Gerätes ausgesucht wird. Existiert in der jeweiligen „strings.xml“ keine Übersetzung einer Sprache, welche das Android System als aktuelle Spracheinstellung verwendet, so wird die Standardsprache (Deutsch) der App verwendet. Um dies zu unterbinden bzw. stattdessen die App in englischer Sprache zu starten, haben wir dies dementsprechend umgesetzt. Vor allem, da die Virtuellen Android Emulatoren der Entwicklungsumgebung als Systemsprache Englisch verwenden.

6.6. Floating Action Menu

Zwar existiert im Android Studio die Möglichkeit einen sog. „Floating Action Button“ zu erstellen, leider gibt es aber keine Möglichkeit mehrere dieser Buttons in einem Menü zu Schachteln. Allerdings existieren mehrere Fremd-Bibliotheken, die so eine Art Menü zur Verfügung stellen. Daher verwenden wir so eine Bibliothek. Wir nutzen die Menüstruktur des GitHub Repositories „FloatingActionButton“ (Tarianyk, 2016). Hiermit können wir mehrere Floating Action Buttons dem Menü hinzufügen.

Mit dieser Möglichkeit haben wir ein Menü erstellt, in dem zwei Menüeinträge existieren (siehe Abbildung 4):

- Neue Kategorie
- Neues Item

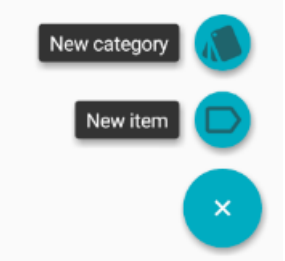

Abbildung 4 Floating Action Menu

Es können hierüber also sowohl Kategorien als auch Items erstellt werden. Wählt man einen der beiden Menüeinträge, wird der Benutzer zur Eingabe eines Titels aufgefordert. Dabei

wird überprüft ob der Titel nicht leer ist, nicht nur aus Leerzeichen besteht und nur maximal 20 Zeichen enthält. Zeilenumbrüche sind nicht möglich. Überflüssige Leerzeichen am Anfang und am Ende des Titels werden entfernt. Der Benutzer wird dementsprechend darauf hingewiesen, falls die Eingabe nicht korrekt ist.

Zudem wird bei der Erstellung einer neuen Kategorie überprüft, ob der neue Titel bereits in einem der vorhandenen Kategorien existiert. Ist dies der Fall, kann die neue Kategorie nicht erstellt werden, da wir keine Kategorien selben Namens zulassen. Der Benutzer erhält in diesem Fall eine Rückmeldung.

Bei der Erstellung eines neuen Items wird zunächst überprüft. ob bereits eine Kategorie existiert. Falls nicht wird der Benutzer aufgefordert zunächst eine Kategorie zu erstellen. Existiert bereits wenigstens eine Kategorie, kann der Benutzer nach Titeleingabe des Items die Kategorie wählen, der das Item zugeordnet werden soll.

Damit das Floating Action Menu auch benutzt werden kann, muss das entsprechende Git Repository importiert werden:

- In „build.gradle“: compile 'com.github.clans:fab:1.6.4'
- In der Main Activity:
  - import com.github.clans.fab.FloatingActionButton;
  - import com.github.clans.fab.FloatingActionMenu;
- Zugriff auf das Menü oder den Button in der XML Datei:
  - <com.github.clans.fab.FloatingActionMenu …>
  - <com.github.clans.fab.FloatingActionButton …>

6.7. Liste der Kategorien und deren Inhalte

Die Kernkomponente der App besteht aus einer Liste, die auf- und zu klappbar ist („Expandable List View“). Hierrüber werden letztendlich alle Kategorien und ihre Items aufgelistet. Diese Liste befindet sich in der „MainActivity“ und wird beim Start der App dem Benutzer sofort als Übersicht angezeigt. Dabei kann durch einen Tipp auf die Kategorie die dort enthaltene Liste der Items entweder ein- oder ausgeklappt werden.

Um diese Liste selbst zu gestalten und mit Daten und verschiedenen Funktionen zu füllen, existieren hierfür eine Adapter Klasse („MyListAdapterClass“) und zwei Layout XML Dateien („layout_child_item.xml“ und „layout_group_item.xml“). In jeweils einer

Layout Datei wird das Layout der Items oder der Kategorien erstellt. So besitzen sowohl die Kategorien als auch die Items je eine Textansicht für den Titel und ein Symbol welches angetippt werden kann und eine bestimmte Funktion erfüllt.

Damit diese Layout Dateien auch der Liste später zugeordnet werden, werden diese in der Adapter Klasse der Liste hinzugefügt. Zudem sorgt die Klasse dafür, dass die Liste korrekt aufgebaut wird und die Inhalte an richtiger Stelle angezeigt werden. Darüber hinaus werden im Adapter die Referenzen zum aktuellen Kontext und zu den vorhandenen Kategorien übergeben. Neben dieser Basisimplementierung haben wir den Adapter bzw. einige Elemente in diesem mit zusätzlichen Funktionen versehen.

So wird hier festgelegt welche Funktionen das Symbol für die Gruppe und das für die Items erfüllen soll. Hinter dem Mülltonnensymbol bei jedem Item steht die Funktionalität des Löschens des jeweiligen Items, bei dem die Mülltonne angetippt wurde. Hierfür wird dem Benutzer zunächst ein Dialog angezeigt, welcher erstmal bestätigt werden muss bevor der Löschvorgang tatsächlich stattfindet.

Tippt man auf das Drei-Punkte Symbol bei einer Kategorie, wird dem Benutzer ein Popup-Menü mit den folgenden Einträgen geöffnet (siehe Abbildung 5):

- Hinzufügen
- Bearbeiten
- Löschen

Mittels „Hinzufügen“ wird direkt der jeweiligen Kategorie ein Item hinzugefügt. Dabei wird die Activity zur Bearbeitung eines Items geöffnet (und die benötigten Informationen mitgeschickt) sodass der Benutzer die Item Informationen direkt eingeben kann.

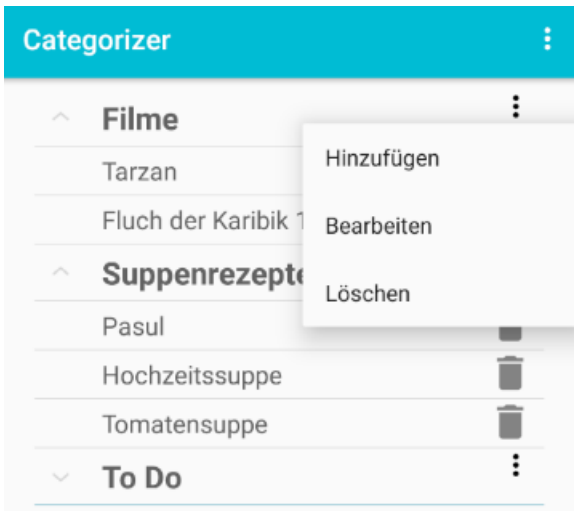

Abbildung 5 Popup-Menü

Wählt der Benutzer den Eintrag „Bearbeiten“, kann er den Titel der jeweiligen Kategorie bearbeiten. Hierfür wird die Activity zur Bearbeitung einer Kategorie gestartet und die jeweilige Kategorie Information mitübertragen.

Wird der Eintrag „Löschen“ gewählt, kann der Nutzer die Kategorie samt Inhalt löschen. Vorher wird dem Benutzer jedoch ein Bestätigungsdialog angezeigt.

All die oben genannten Funktionen sind in der Adapter Klasse implementiert, wobei für die Erstellung der Basisinformationen ein Tutorial verwendet wurde (Vasudevan, 2013). Weitere Funktionalitäten der Liste wurden in der Main Activity eingerichtet. Hier existiert die Liste der Kategorien und ebenfalls eine Hashmap, welche als Schlüssel den Namen der Kategorie und als Datum das Kategorie-Objekt enthält. Die Hashmap dient der schnellen Identifikation vorhandener Kategorien. Bei der Kreierung der Main Activity wird u.a. die Liste initialisiert. Hierbei wird unser selbstgeschriebener Adapter gesetzt und anschließend werden noch Funktionalitäten für die Items und Kategorien erstellt.

So reagieren Items auf ein einfaches Tippen. Dementsprechend ermöglicht das Antippen eines Items dessen Bearbeitung. Hierfür wird das Item an die Bearbeitungsaktivität versendet. Darüber hinaus öffnet ein langes antippen auf eine Kategorie oder ein Item einen Dialog, worüber der Benutzer die Kategorie oder das Item löschen kann.

6.8. Kategorien bearbeiten

Die Activity „CategoryEditActivity“ wird geöffnet, sobald eine Kategorie bearbeitet werden soll. Hierfür empfängt sie die Informationen der aktuell zu bearbeitenden Kategorie, speichert diese temporär ab und zeigt diese an. So werden im oberen Bereich der Activity der Kategorie Name und darunter eine Liste aller enthaltenen Items angezeigt (siehe Abbildung 6).

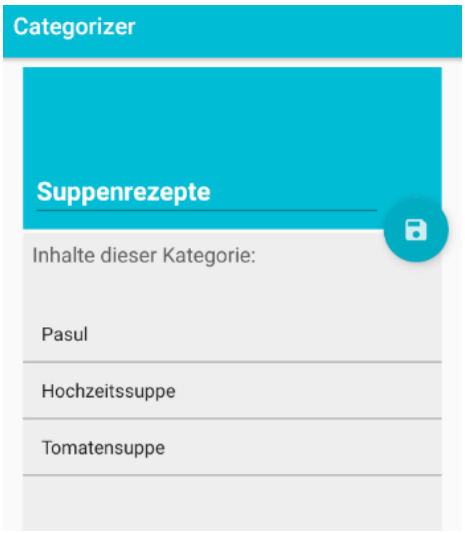

Abbildung 6 Kategorie editieren

Die Aktualisierung des Kategorienamens kann durch das Antippen des Speichern Symbols durchgeführt werden. Hierbei wird aber zunächst noch folgendes geprüft:

- Ist der Titel leer?
- Besteht der Titel nur aus Leerzeichen?

Falls einer der beiden Fällte zutrifft, erhält der Benutzer dementsprechend einen Dialog als Rückmeldung und der Speichervorgang findet nicht statt bis der Benutzer

die Eingabe korrigiert oder das Ganze abbricht. Bei der Eingabe des Titels werden zudem nur maximal 20 Zeichen akzeptiert. Weiterhin werden beim Speichern überflüssige Leerzeichen am Anfang und Ende des Titels gelöscht.

6.9. Items bearbeiten

Die Activity „ItemEditActivity“ wird geöffnet sobald ein Item bearbeitet werden soll. Hierfür empfängt sie die Informationen des aktuell zu bearbeitenden Items, speichert diese temporär ab und zeigt diese an. Im oberen Bereich der Activity wird der Item Titel angezeigt und darunter das Beschreibungsfeld des Items (siehe Abbildung 7).

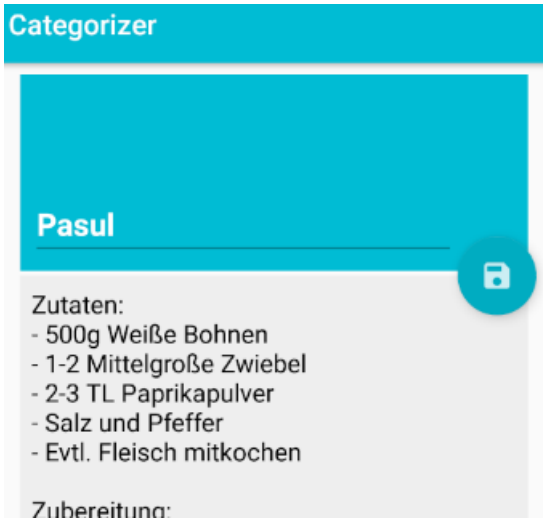

Abbildung 7 Item editieren

Das Item kann ebenfalls wie die Kategorie mittels des Speicher Symbols abgespeichert werden. Auch hier wird schließlich überprüft ob der Titel leer ist oder nur aus Leerzeichen besteht. Zudem werden auch hier nur 20 Zeichen im Titel zugelassen und überflüssige Leerzeichen am Anfang und Ende des Titels gelöscht.

6.10. Dialoge

Wir verwenden Dialoge in notwendigen und kritischen Situationen, um den Benutzer auf verschiedene Fehler und nicht umkehrbare Aktionen aufmerksam zu machen oder um durch verschiedene Interaktionsschritte zu führen.

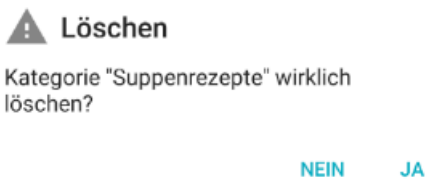

Abbildung 8 Dialog Abfrage beim Löschen

So werden Dialoge u.a. in folgenden Situationen verwendet:

- Für die Erstellung von Items und Kategorien, wo Daten vom Benutzer erwartet werden
- Beim Löschen von Items oder Kategorien (siehe Abbildung 8)
- Bei keiner bzw. fehlerhafter Eingabe
- Bei nicht möglichen Aktionen (z.B. Kategorie mit bereits existierendem Namen erstellen)

Die Dialoge dienen also primär dazu dem Benutzer für seine Vorgänge ein Feedback zu geben bzw. mögliche Folgen mitzuteilen oder Aktionen erst gegen Bestätigung durchzuführen, da diese ausversehen ausgelöst sein könnten.

6.11. Toasts

Toasts werden unsererseits verwendet, um dem Benutzer nebenbei Benachrichtigungen mit Informationen zu seinen durchgeführten Aktionen anzuzeigen (siehe Abbildung 9). So werden beispielsweise Benachrichtigungen erstellter, gelöschter, aktualisierter Kategorien und Items und Benachrichtigungen zu geänderten Einstellungen angezeigt.

Abbildung 9 Toast bei Erstellung eines Items

6.12. Datenbank

Für die permanente Speicherung der App Daten, also den Kategorien und Items, verwenden wir die interne SQLite Datenbank. Für die Realisierung der Datenspeicherung und des Ladens der Daten existieren primär für die Datenbank zwei Klassen: „DBhelperClass“ und „SQLHandlerClass“.

Die „DBhelperClass“ ist das Kernstück der Datenbank. Sie leitet sich von der Klasse „SQLiteOpenHelper“ ab und definiert das Datenbankschema, es werden also Datenbank-, Tabellen- und Spaltennamen für die Kategorien und Items festgelegt. Zudem wird die Klasse verwendet, um die Tabellen in der Datenbank zu erstellen.

Die „SQLHandlerClass“ enthält Funktionalitäten, welche auf die Datenbank anwendbar sind. So enthält sie die Basismethoden für den Zugriff zur Datenbank, nämlich zum einen die Möglichkeit die Datenbank entweder im Schreib- oder Lesemodus zu öffnen, oder die Datenbank zu schließen. Weiterhin enthält sie Methoden zum Hinzufügen von Items oder Kategorien und zum Laden aller Kategorien, sowie zum Laden aller Items einer spezifischen Kategorie. Als letzte Methode existiert noch eine, welche die Tabellen löscht und wieder neu erstellt.

Wird die App gestartet, dann werden in der Main Activity zunächst alle Daten aus der Datenbank geladen, sofern welche vorhanden sind, und lokal gespeichert. Um zu garantieren, dass alle Daten wieder zurück in die Datenbank geschrieben werden, falls die App geschlossen wird oder ähnliches, wird dies vorher abgefangen (in der „onStop“ Methode) und die Daten in die Datenbank gespeichert.

6.13. Ressourcen

Alle Ressourcen die wir verwenden, werden zentral in den von Android Studio dafür vorgesehenen Ordnern gespeichert. Das bewirkt, dass Änderungen einfach an zentraler Stelle unkompliziert vorgenommen werden können ohne diese an jeder Stelle im Quellcode ändern zu müssen.

So sind z.B. all unsere Beschriftungen und Infotexte in der „string.xml“ gespeichert und die englische Übersetzung ebenso unter dem entsprechenden Identifier. Auch unsere App-Farben haben wir zentraler Stelle gespeichert. So verwendet unsere App z.B. als Grundfarbe ein Türkis für den Tagesmodus und Grautöne für den Nachtmodus.

Die Textgrößen unserer App verweisen auf die Ressource „dimens.xml“. Hierbei werden je nach Displaygröße, Auflösung oder Pixeldichte der Geräte die Textgrößen dementsprechend angepasst.

Die verschiedenen Symbole, die wir in unserer App verwenden, z.B. in den Dialogen, werden ebenfalls zentral abgelegt. Diese können unter Ressourcen in „drawable“ gefunden werden.

Weiterhin liegen hier noch die Layout Dateien, Menüs und die Start Icons der App (siehe Abbildung 10).

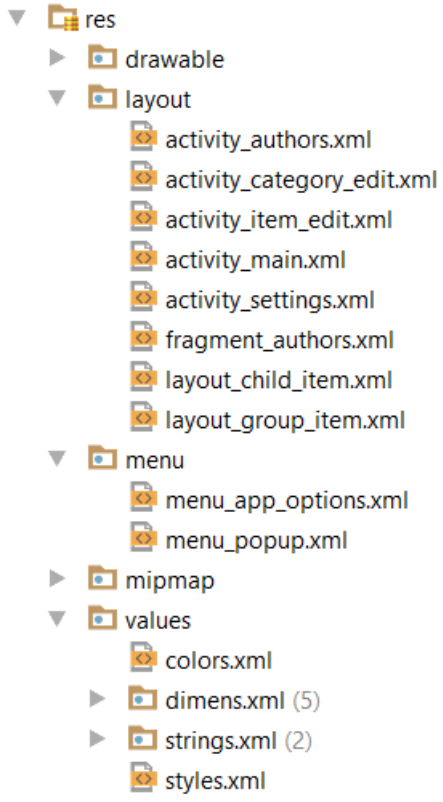

Abbildung 10 Ressourcen Struktur

7. Reflexion

Abschließend und reflektierend lässt sich sagen, dass das Modul sehr interessant war und wir eine Menge im Bereich Android-Programmierung unter Android Studio gelernt haben. Nach dieser Veranstaltung betrachtet man Android Apps nun aus einem ganz anderen Blickwinkel. So fällt einem z.B. immer mehr auf, dass sehr viele Apps in ihrer Grundstruktur (vermutlich) aus Listen bestehen.

Das Modul hat uns also einen guten Einblick in die Welt der Android-Programmierung gegeben und unsern Horizont in diesem Bereich erweitert. Wir haben viele Grundlagen gelernt, um „kleine“ App’s selbst schreiben zu können, ebenso wie man diese Apps über Android-Studio entwickelt.

Darüber hinaus lief unsere Teamarbeit sehr gut ab. Unsere Kommunikation fand größtenteils über WhatsApp und bzw. oder bei kurzen „Meetings“ während der Mittagspause statt. So haben wir unsere internen Aufgaben gut verteilen und von Problemen und Erfolgen im Projekt dem anderen berichten können. Da wir GitHub als Versionierungsdienst genutzt haben, konnten wir beide von überall aus gleichzeitig und unabhängig am Projekt arbeiten und die Ergebnisse zwischenzeitlich zusammenführen. Zudem wollen wir anmerken, dass die Dokumentation von Android Studio sehr gut ist, sodass wir bei verschiedenen Problemen nach kurzem Zeitaufwand einen Lösungsansatz (auch wenn dieser nicht immer hilfreich war) für unsere Probleme finden und die Umsetzung testen konnten.

Abschließend können wir sagen, dass wir mit unserer App sehr zufrieden sind, uns die Programmierung dieser App viel Spaß bereitet hat und Apps nun von einem anderen Blickwinkel aus betrachtet werden können.

Quellen

GitHub Inc. (2017). Retrieved 06 21, 2017, from https://github.com/.

Google Inc. (2017). Retrieved 06 21, 2017, from https://developer.android.com/studio/index.html.

Tarianyk, D. (2016, 05 28). *Android Floating Action Button based on Material Design specification*. Retrieved 05 29, 2017, from https://github.com/Clans/FloatingActionButton

Vasudevan, N. (2013, 05 03). *Android: Expandable List View Example*. Retrieved 05 22, 2017, from http://theopentutorials.com/tutorials/android/listview/android-expandable-list-view-example/

Eigenständigkeitserklärung

Wir erklären hiermit,

- dass wir die vorliegende Arbeit ohne fremde Hilfe und ohne Verwendung anderer, als der angegebenen Hilfsmittel verfasst haben
- dass wir sämtliche verwendeten Quellen erwähnt und korrekt zitiert haben.

_____

Kevin Kussyk

_____

Thomas Fahrenholz