



TAXAM

App zur Organisation des Studiums

Ein Projekt im Rahmen der Lehrveranstaltung
Spezielle Programmierung des Studiengangs
Wirtschaftsinformatik der HTW Berlin

Carolin Gellner

Matrikel-Nr.: 539540

Version 1.0 | 19.05.2016

Änderungsübersicht

Version	Datum	Beschreibung
0.1	17.05.2016	Initialisierung des Dokuments
0.2	18.05.2016	In Bearbeitung
1.0	19.05.2016	Fertigstellung

Inhaltsverzeichnis

1.	EINFÜHRUNG.....	3
1.1.	AUFGABENSTELLUNG	3
1.2.	IDEE ZUR UMSETZUNG DES PROJEKTS	3
1.3.	ZIELGRUPPE	3
2.	RANDBEDINGUNGEN	3
3.	FUNKTIONEN	4
4.	TECHNISCHER KONTEXT	5
5.	ANSICHTEN.....	6
5.1.	START & MENÜ	6
5.2.	SEMESTER ANLEGEN.....	7
5.3.	SEMESTER BEARBEITEN.....	8
5.4.	MODUL ANLEGEN.....	9
5.5.	MODUL BEARBEITEN.....	10
5.6.	AUFGABE ANLEGEN.....	11
6.	DATENBANK UND ANZEIGEN DER INHALTE	12
6.1.	DATEN SPEICHERN UND AUSLESEN	12
6.2.	ANZEIGEN DER AUSGELESENEN DATEN.....	13

1. Einführung

1.1. Aufgabenstellung

Im Rahmen der Lehrveranstaltung Spezielle Programmierung des Studiengangs Wirtschaftsinformatik der Hochschule für Technik und Wirtschaft ist die Konzeptionierung und Programmierung einer Android-App vorgesehen.

1.2. Idee zur Umsetzung des Projekts

Die Idee zur Umsetzung des Projekts ist die Konzeptionierung und Programmierung einer App, die zur Organisation des Studiums dient. Des Weiteren lassen sich damit Schuljahre aller Klassenstufen sowie die Berufsschule im Rahmen einer Berufsausbildung verwalten. Die App soll die Möglichkeit bieten, anfallende Aufgaben und Prüfungen zu erfassen und jederzeit im Überblick zu haben. Es soll ermöglicht werden, auf einen Blick erkennen zu können, welche Aufgabe die aktuell höchste Priorität hat.

1.3. Zielgruppe

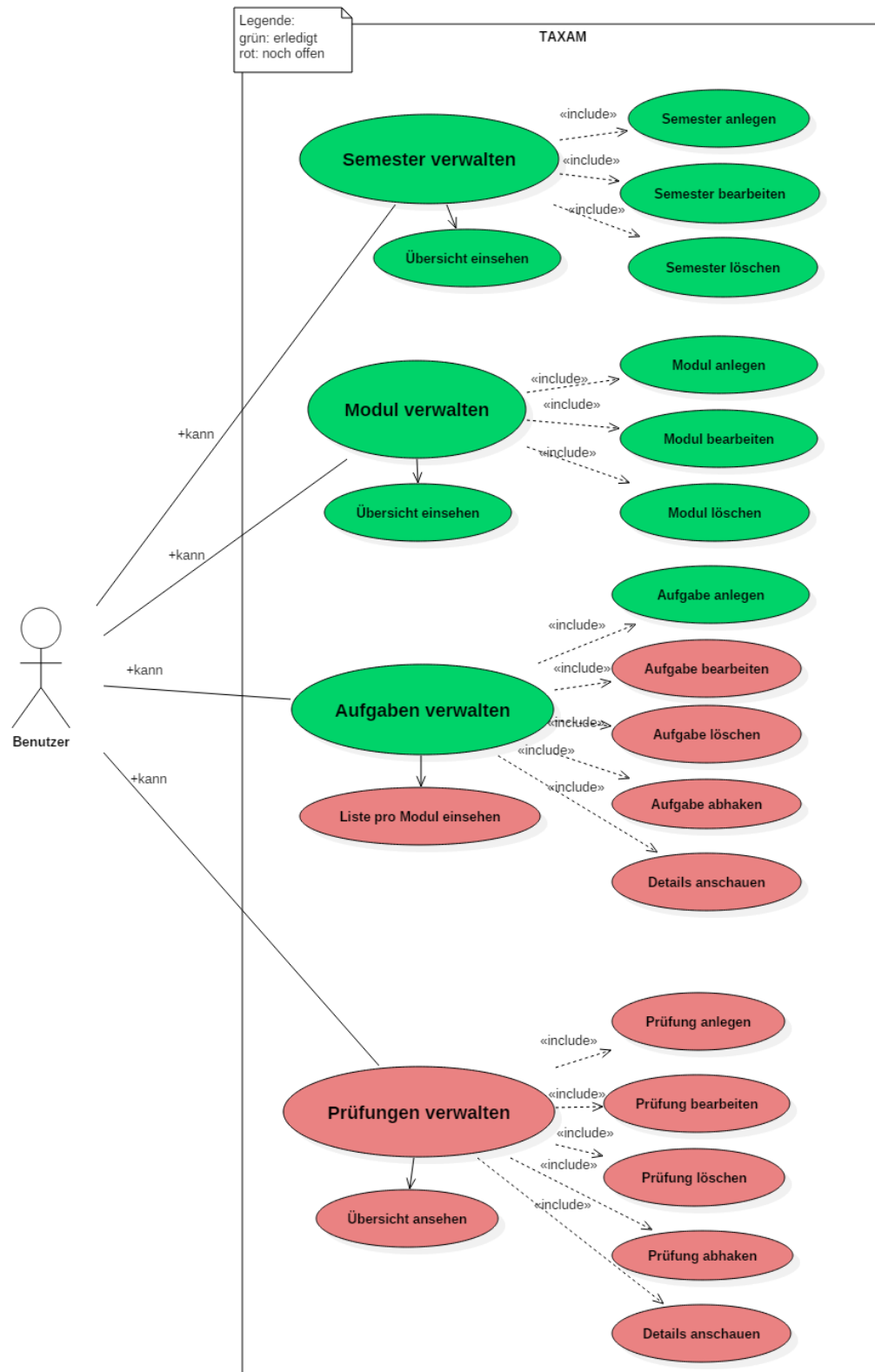
Die App richtet sich vorrangig an Studenten, aber auch an Schüler aller Klassenstufen und Auszubildende.

2. Randbedingungen

Systemvorgaben des Endgeräts	
Betriebssystem	Android 5.0
Programmiervorgaben	
Sprache	Java
Software	Android Studio 2.1 inkl. Sdk, Java Version 8
Datenbank	SQLite
Gestaltung	mind. 1 „Master-Detail“-Layout
Funktionen	Speichern und auslesen von Daten

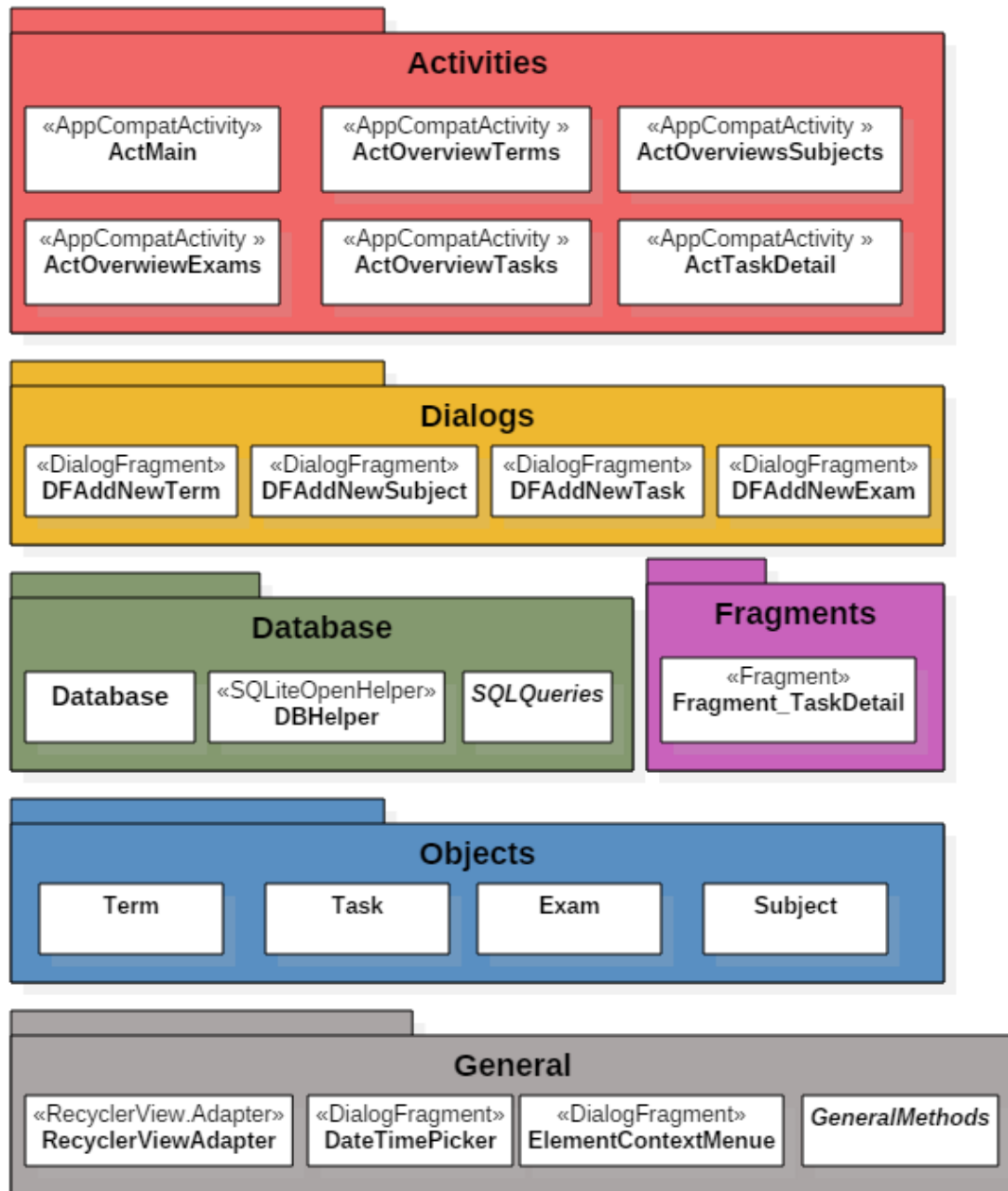
3. Funktionen

Das folgende Anwendungsfalldiagramm stellt die Hauptfunktionen dar, welche dem Benutzer in der App zur Verfügung stehen. Die grün markierten Elemente stellen Funktionen dar, die zur Zwischenabgabe (20.05.2016) des aktuellen Projektstands bereits implementiert sind. Die rot markierten Elemente werden erst im weiteren Projektverlauf realisiert.



4. Technischer Kontext

Die Darstellung zeigt eine grobe Übersicht aller Klassen des Programmcodes:

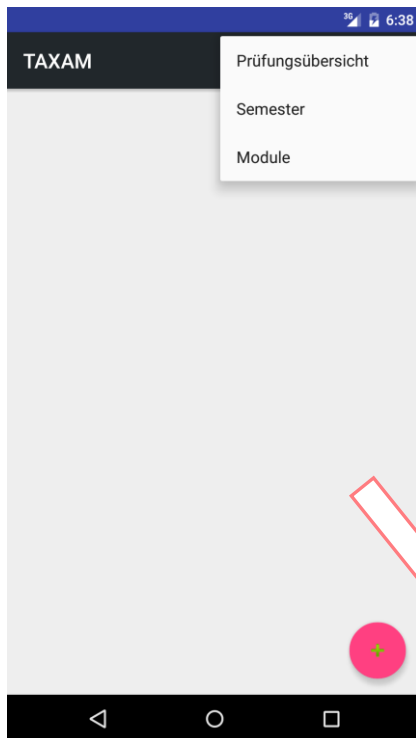


5. Ansichten

Die folgenden Abbildungen zeigen die aktuellen Ansichten zu den bisher implementierten Funktionen.

5.1. Start & Menü

Diese Sicht stellt den Startbildschirm der App dar. Derzeit sind dieser Sicht noch keine Informationen zu entnehmen



ActMain [AppCompatActivity]

```
onCreate(Bundle savedInstanceState)
onCreateOptionsMenu(Menu menu)
onItemsLoadComplete()
onOptionsItemSelected(MenuItem item)
refreshItems(android.support.v7.widget.RecyclerView rv)
setSubjectList()
startNewActivity(java.lang.Class cl)
```

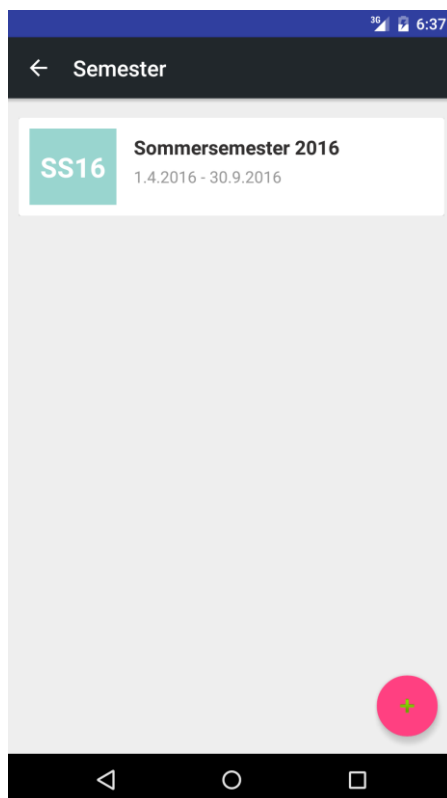
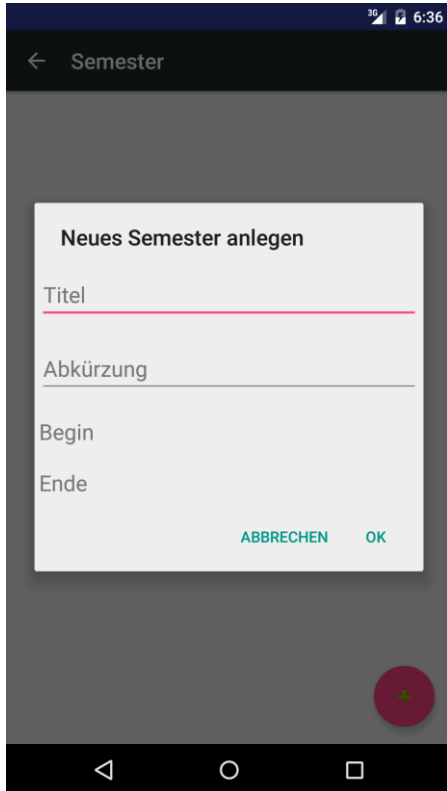
Oeffnet die Aktivität der uebergebenen Klasse



5.2. Semester anlegen

Die erste stellt ein Formular zum Erfassen eines neuen Semesters dar.

Die zweite Sicht zeigt die Übersichtsliste der erfassten Semester.



DFAddNewTerm [DialogFragment]

Methoden:

`getData(View view)`

Liest die eingegebenen Daten aus dem Dialog (Formular) aus

`getUpdatedData(View view)`

Liest die eingegebenen Daten aus dem Dialog (Formular) aus

`onCreateDialog(Bundle savedInstanceState)`

`setTermID(long ID)`

`setTextContent(View view)`

`setTitle(AlertDialog.Builder dialogBuilder)`

Legt den Titel des DialogFragments fest.

ActOverviewTerms [AppCompatActivity]

Methoden:

`onCreate(Bundle savedInstanceState)`

`onCreateOptionsMenu(Menu menu)`

`onItemsLoadComplete()`

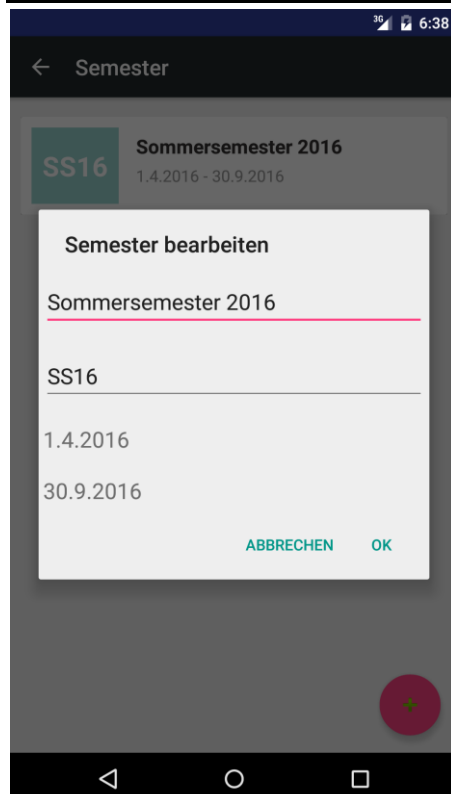
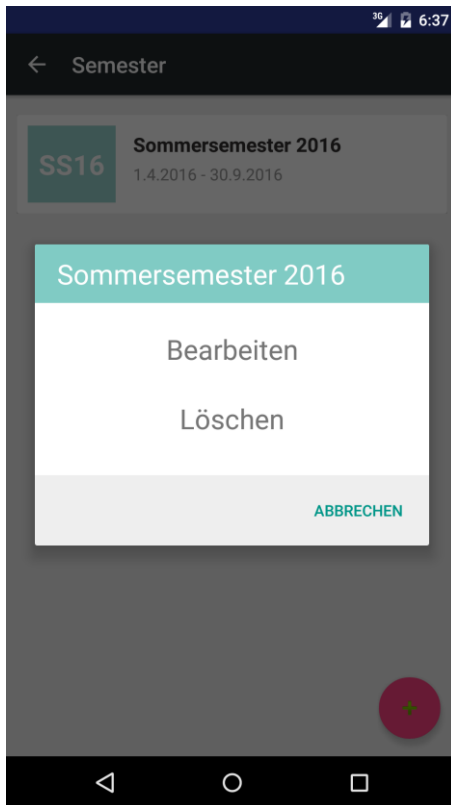
`onOptionsItemSelected(MenuItem item)`

`refreshItems(android.support.v7.widget.RecyclerView rv)`

`setTermList()`

5.3. Semester bearbeiten

Die Sicht stellt das Menü dar, welches sich öffnet, sobald der Benutzer auf ein Semester-Element klickt. Dieses Menü bietet Möglichkeiten zum Bearbeiten und Löschen eines Semesters. Die untere Sicht zeigt das „Bearbeiten“-Formular an.



ElementContextMenu [DialogFragment]

Methoden:

```
getElements(View view)
onCreateDialog(Bundle savedInstanceState)
setElementListener()
setNameCardView(long ID, java.lang.String titleCardElement)
setObject(java.lang.Class object)
```

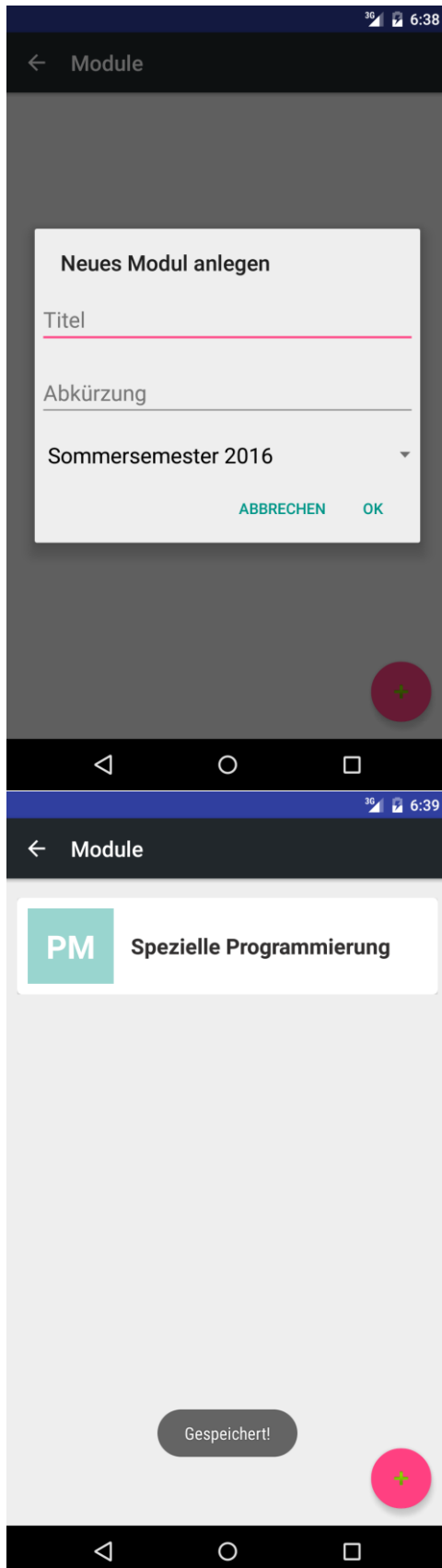
DFAddNewTerm [DialogFragment]

Methoden:

```
getData(View view)
Liest die eingegebenen Daten aus dem Dialog (Formular) aus
getUpdatedData(View view)
Liest die eingegebenen Daten aus dem Dialog (Formular) aus
onCreateDialog(Bundle savedInstanceState)
setTermID(long ID)
setTextContent(View view)
setTitle(AlertDialog.Builder dialogBuilder)
Legt den Titel des DialogFragments fest.
```

5.4. Modul anlegen

Die Abbildung zeigt das Formular zum Erfassen eines Moduls und das Ergebnis der Erfassung.



DFAddNewSubject [DialogFragment]

Methoden:

getData(View view)

Liest die eingegebenen Daten aus dem Dialog (Formular) aus

getUpdatedData(View view)

Liest die eingegebenen Daten aus dem Dialog (Formular) aus

onCreateDialog(Bundle savedInstanceState)

setSpinner(View view)

setSubjectID(long subjectID)

setDialogContent(View view)

setTitle(AlertDialog.Builder dialogBuilder)

ActOverviewSubjects [AppCompatActivity]

Methoden:

onCreate(Bundle savedInstanceState)

onCreateOptionsMenu(Menu menu)

onItemsLoadComplete()

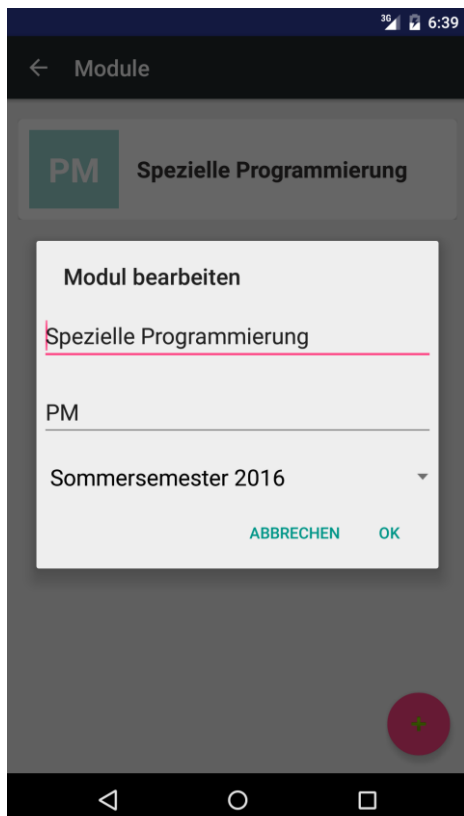
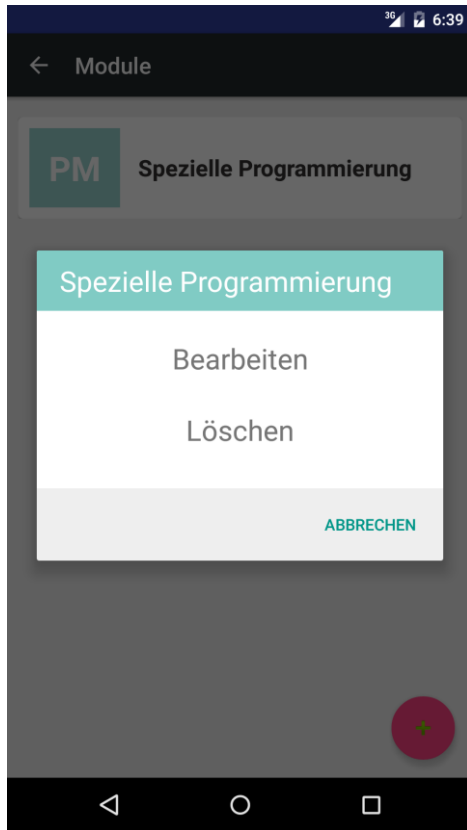
onOptionsItemSelected(MenuItem item)

refreshItems(android.support.v7.widget.RecyclerView rv)

setSubjectListList()

5.5. Modul bearbeiten

Die Sicht stellt das Menü dar, welches erscheint, wenn der Benutzer auf ein Modul-Element klickt. Das Menü bietet Möglichkeiten zum Bearbeiten und Löschen eines Moduls.



ElementContextMenu [DialogFragment]

Methoden:

```
getElements(View view)
onCreateDialog(Bundle savedInstanceState)
setElementListener()
setNameCardView(long ID, java.lang.String titleCardElement)
setObject(java.lang.Class object)
```

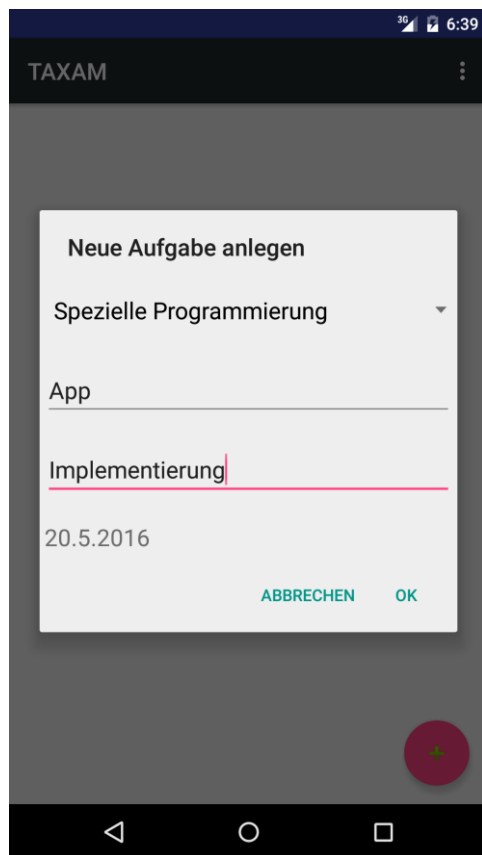
DFAddNewSubject [DialogFragment]

Methoden:

```
getData(View view)
Liest die eingegebenen Daten aus dem Dialog (Formular) aus
getUpdatedData(View view)
Liest die eingegebenen Daten aus dem Dialog (Formular) aus
onCreateDialog(Bundle savedInstanceState)
setSpinner(View view)
setSubjectID(long subjectID)
setTextContent(View view)
setTitle(AlertDialog.Builder dialogBuilder)
```

5.6. Aufgabe anlegen

Die Darstellung zeigt das Formular zum Erfassen einer Aufgabe.



The screenshot shows a mobile application interface with a dark blue header labeled 'TAXAM'. A dialog box titled 'Neue Aufgabe anlegen' is displayed in the center. The dialog contains a dropdown menu with 'Spezielle Programmierung' selected, a text input field with 'App', another text input field with 'Implementierung' (underlined in red), and a date field with '20.5.2016'. At the bottom of the dialog are two buttons: 'ABBRECHEN' and 'OK'. The background of the app is dark grey, and there is a red circular button with a green plus sign at the bottom right. The Android navigation bar is visible at the very bottom.

DFAddNewTask [DialogFragment]

Methoden:

```
getData(View view)
```

```
onCreateDialog(Bundle savedInstanceState)
```

```
setSpinner(View view)
```

6. Datenbank und Anzeigen der Inhalte

Die App hat eine Datenbank als Bestandteil. Dafür wird die SQLite-Datenbank benutzt. Diese ermöglicht das Speichern sämtlicher erfasster Daten sowie das Auslesen zum Verarbeiten und Darstellen von Daten.

6.1. Daten speichern und auslesen

Das Speichern und Auslesen von Daten aus der Datenbank wird mithilfe der beiden Klassen Database und DBHelper realisiert. Die Klasse **SqlQueries** stellt die nötigen SQL-Befehle zur Verfügung.

➤ **Database** (SQLiteDatabase)

close()

Schliesst die Datenbank

createContentValues(java.lang.Object object)

deleteSubject(long subjectID)

deleteTerm(long termId)

getAllSubjectsFromDb()

getAllTermsFromDb()

getContext()

getDatabase()

getDbHelper()

getQueryForAllSubjects()

getQueryForAllTerms()

getSubject(long subjectId)

getSubjectFromDb(long subjectId)

getTerm(long termId)

getTermFromDb(long termId)

insertData(java.lang.Object object)

open()

Oeffnet die Datenbank.

readSubjectData(Cursor cursor)

readTermData(Cursor cursor)

update(java.lang.Object object)

➤ **DBHelper** (SQLiteOpenHelper)

onCreate(SQLiteDatabase sqLiteDatabase)

Die Methode legt die Datenbank und die Tabellen an.

onUpgrade(SQLiteDatabase sqLiteDatabase, int oldVersion, int newVersion)

6.2. Anzeigen der ausgelesenen Daten

Die ausgelesenen Daten werden hauptsächlich durch sogenannte CardViews dargestellt. Die CardViews werden durch einen RecyclerViewAdapter mit den Daten gefüllt, welche sie anzeigen sollen. Eine CardView wird durch die Klasse **RecyclerViewAdapter.ViewHolder** dargestellt.

➤ RecyclerViewAdapter

`getExamList()`

`getItemCount()`

Die Methode gibt die Anzahl der Elemente in der RecyclerView zurück.

`getSubjectList()`

`getTaskList()`

`getTermList()`

`isTwoLines()`

`isWithShortcut()`

`onAttachedToRecyclerView(android.support.v7.widget.RecyclerView recyclerView)`

`onBindViewHolder(RecyclerViewAdapter.ViewHolder termViewHolder, int i)`

Die Methode weist den Elementen der CardView Werte zu

`onCreateViewHolder(ViewGroup viewGroup, int i)`

Die Methode übergibt der CardView das jeweilige Layout und erstellt einen entsprechenden ViewHolder

`setExamList(java.util.ArrayList<Exam> examList)`

`setFragmentManager(FragmentManager manager)`

`setSubjectList(java.util.ArrayList<Subject> subjectList)`

`setTaskList(java.util.ArrayList<Task> taskList)`

`setTermList(java.util.ArrayList<Term> termList)`

`setTwoLines(boolean twoLines)`

`setWithShortcut(boolean withShortcut)`