# **Vývoj aplikácií pre mobilné zariadenia**

## Semestrálna práca

Daniel Mikulík

5ZY021

2019/2020

# Popis aplikácie

Cieľom tejto semestrálnej práce bolo vytvoriť jednoduchý To Do List (**zoznam** plánovaných aktivít). Užívateľ má možnosť pridať si aktivity s názvom, popisom a indikátorom dôležitosti vo forme ikony v troch rôznych stupňoch. Jednotlivé položky v zozname môžeme mazať potiahnutím do strany (vpravo aj vľavo). Takisto môžeme zmazať aj všetky položky naraz. Pri mazaní (jednej položky alebo všetkých ukončených naraz) vyskočí upozornenie, či chceme naozaj položku zmazať alebo vrátiť akciu späť. Každú položku si môžeme rozkliknúť a pridávať/upravovať/mazať jej podúlohy. Aj tu môžeme zmazať všetky naraz. Taktiež môžeme upraviť názov a popis danej úlohy. Jednotlivé úlohy aj podúlohy si dokážeme „odfajknúť“ ako hotové. Jednotlivé aktivity aplikácie majú príjemné meniace sa pozadie.

# Popis podobných aplikácii

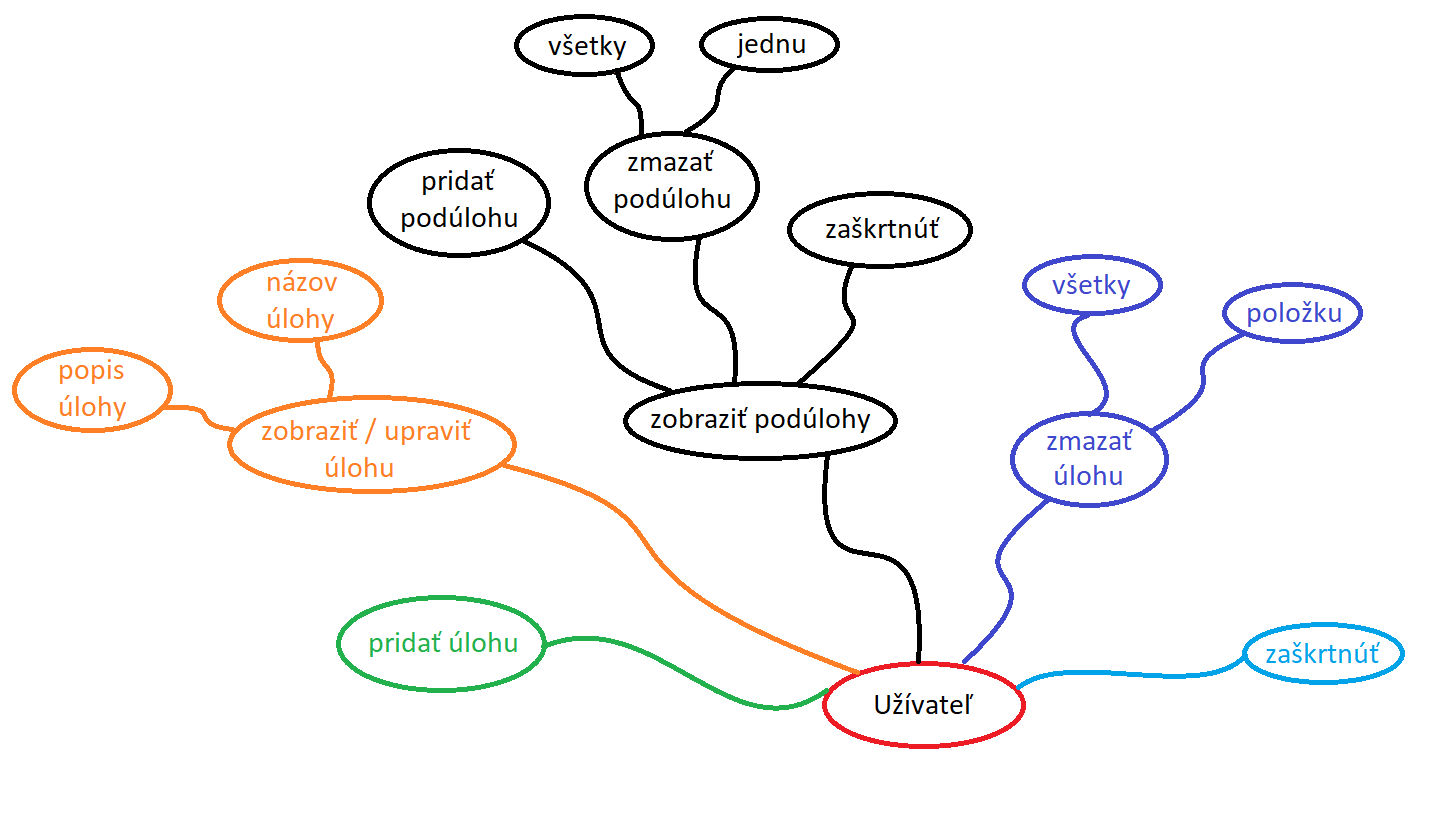
## Simple To-Do List – *Ralf Winkelmann*

Umožňuje pridávať úlohy s dátumom. Dokáže vymazať (alebo všetky ukončené) všetky naraz. Položky vie utriediť podľa dátumu ukončenia. Na rozdiel od mojej aplikácie sa tu nedajú pridávať podúlohy. Taktiež obsahuje neustále reklamy.

## Simple To Do List – *Tikl Inc*

Umožňuje pridávať a upravovať poznámky. Dokáže nastaviť pripomienku a zmeniť farbu ActionBaru. Moja aplikácia je dizajnovo príjemnejšia na používanie a dokáže pridávať podúlohy. Obsahuje reklamy, ktoré kazia dojem z aplikácie.

# Diagram prípadov použitia

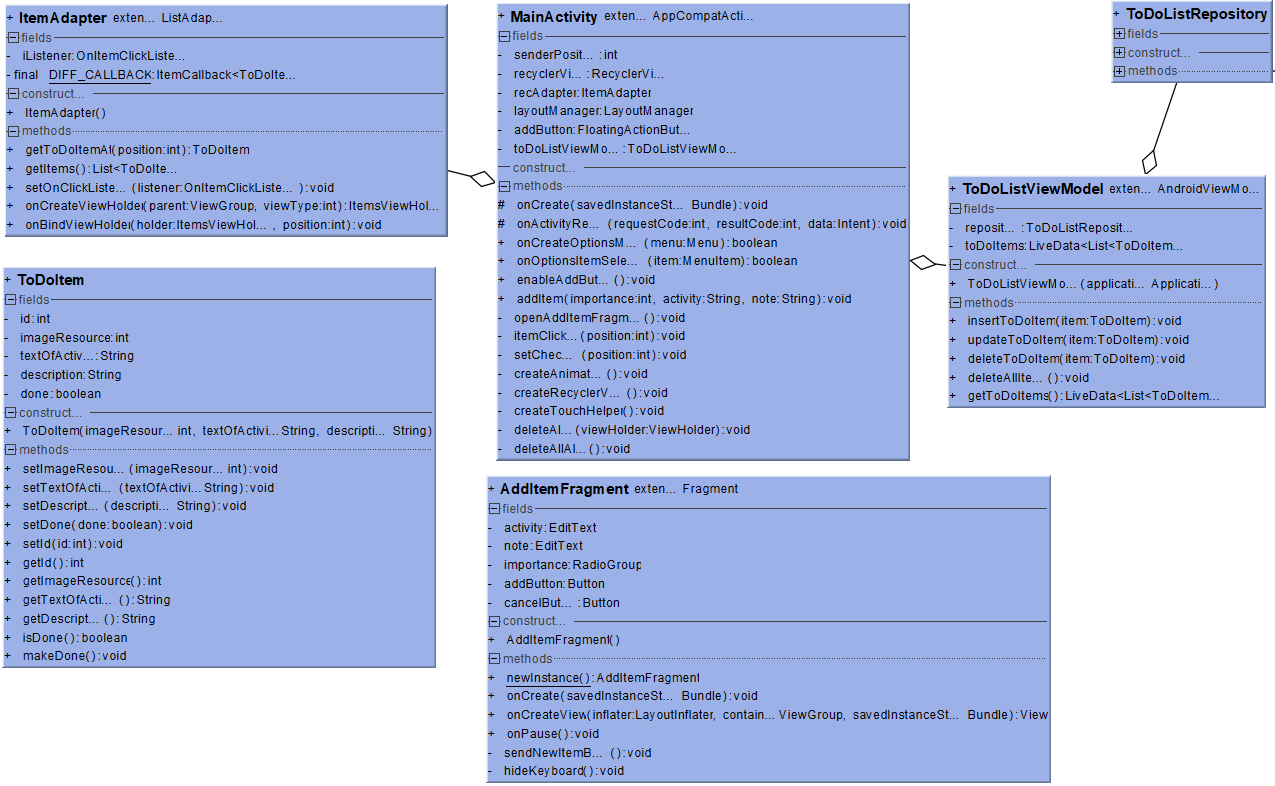


# Diagram tried

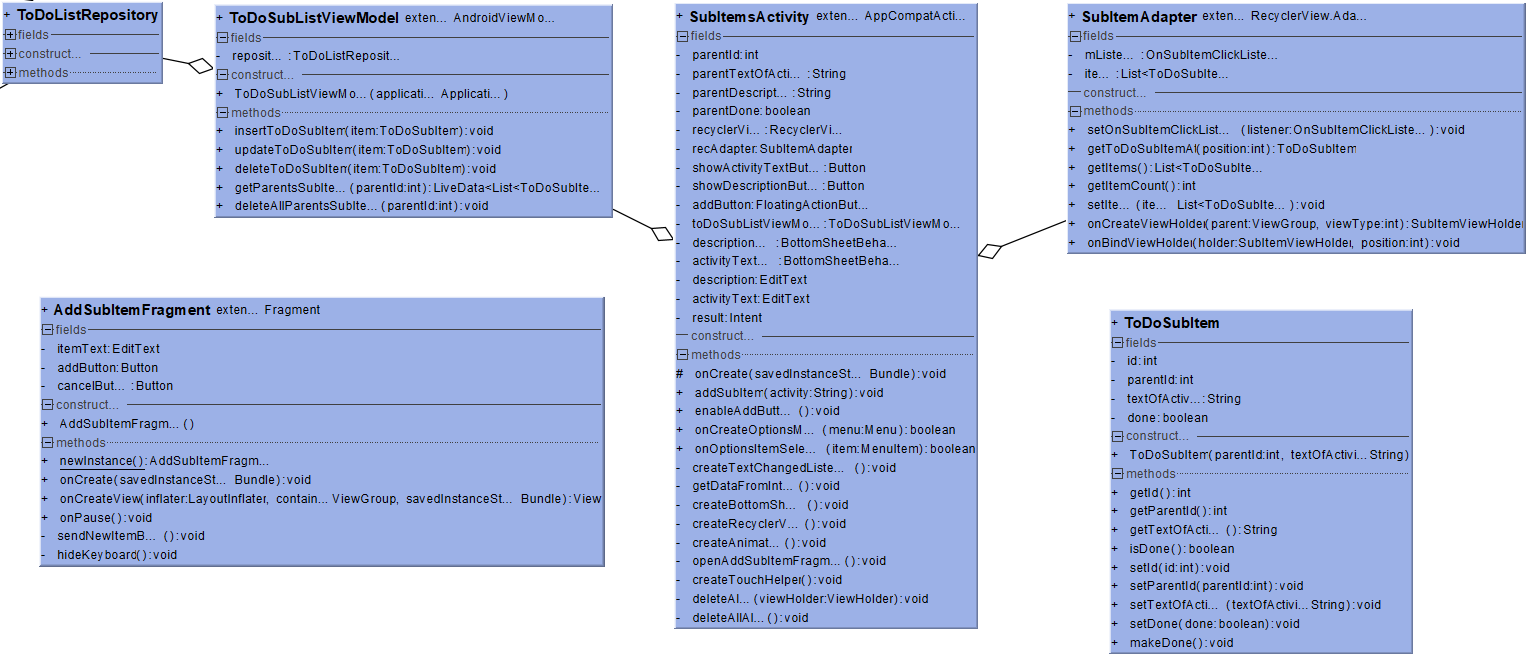
## Databáza



## MainActivity



## SubItemsActivity



# Popis implementácie

## MainActivity

Aktivita, do ktorej sa dostaneme, po spustení aplikácie. Zobrazuje sa v nej **zoznam** úloh zoradený podľa ukončenia (odfajknuté) a podľa priority. Jednotlivé úlohy (CardView) sú zobrazené v **RecyclerView**.

Za pomoci ItemTouchHelpera ich dokážeme mazať posunutím do strany. Pri mazaní vyskočí AlertDialog, ktorý nám dá na výber, či chceme úlohu naozaj vymazať, alebo ju vrátiť späť.

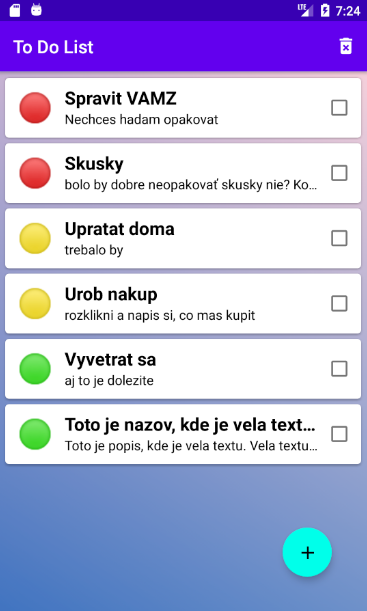
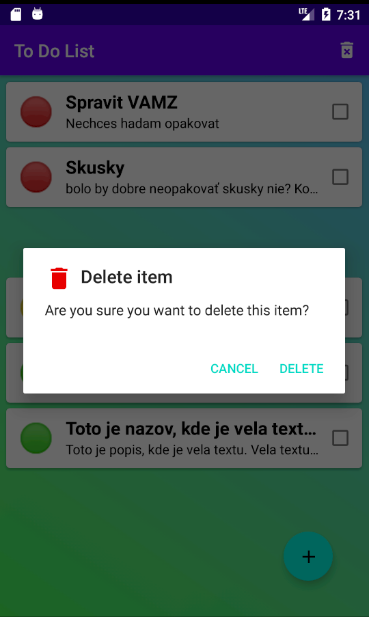
Kliknutím na ľubovoľnú úlohu sa otvorí nová aktivita, kde sa zobrazia jej podúlohy a umožňuje aj upraviť danú úlohu.

Úlohy môžeme zmazať aj všetky ukončené naraz, pomocou menu v pravom hornom rohu, zobrazeným pomocou ikony, ktoré obsahuje len jednu možnosť. Taktiež vyskočí AlertDialog.

Vpravo dole sa nachádza Floating Action Button. Jeho stlačením sa otvorí fragment, ktorý slúži na pridanie novej úlohy.

Na ošetrenie straty dát pri zmene konfigurácie je využitý **ViewModel**.

V pozadí aktivity prebieha animácia, v ktorej sa striedajú 3 rozličné gradienty.

## SubItemsActivity

Zobrazuje podúlohy, patriace úlohe, na ktorú sme klikli v MainActivity. Podúlohy sú zoradené podľa ukončenia. Taktiež sú zobrazované pomocou **RecyclerView** ako aj v MainActivity a dokážeme zmazať jednu podúlohu posunutím do strany alebo všetky ukončené kliknutím na ikonku koša v pravom hornom rohu. Pri oboch sa zobrazí potvrdzujúci AlertDialog.

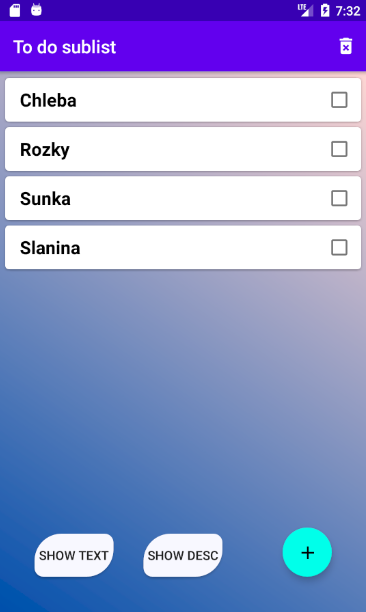
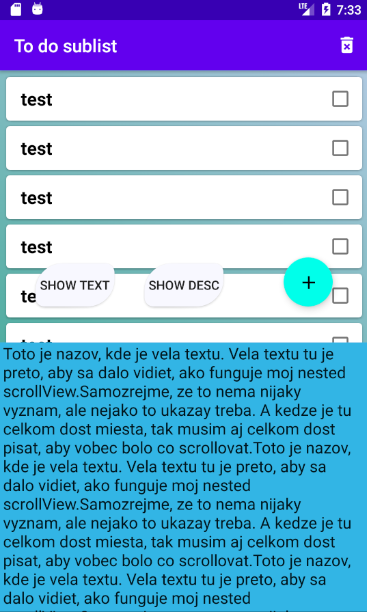
Vpravo dole je opäť Floating Action Button, ktorý otvorí fragment slúžiaci na pridanie podúlohy.

Po odfajknutí všetkých úloh, sa po návrate do MainActivity odfajkne aj rodičovská úloha. Rovnako ak boli všetky podúlohy ukončené a niektorej z nich ukončenie zrušíme, zruší sa aj ukončenie parent úlohy.

Naspodku sa nachádzajú ďalšie dva buttony, po ktorých stlačení sa vysunie bottom sheet, v ktorom si vieme pozrieť celý názov / popis parent úlohy a zmeniť ho. Výsledok sa po návrate prejaví v MainActivity. Využívam pri tom TextChangedListener.

Na ošetrenie straty dát pri zmene konfigurácie je využitý **ViewModel**.

Na pozadí je využitá rovnaká animácia, ako v MainActivity.

## AddItemFragment

Slúži na pridanie úlohy do zoznamu. Zadávame názov, popis a prioritu úlohy. Tlačidlom ADD pridáme novú úlohu, pokiaľ je zadaný názov a priorita. Inak vyskočí Toast správa a nič sa nestane. Tlačidlom CANCEL sa vrátime späť bez zmeny. Počas zobrazenia fragmentu je vypnutý Floating Action Button.

## AddSubItemFragment

Slúži na pridanie podúlohy do zoznamu. Zadáva sa len názov. Podúloha sa pridá stlačením ADD, ak má vyplnený názov. Tlačidlom CANCEL sa vrátime späť bez zmeny. Počas zobrazenia fragmentu sú vypnuté všetky tlačidlá aktivity.

## ToDoItem

Trieda – jedna úloha, ktorá má názov, popis, ikonu, prioritu a údaj o tom, či je ukončená alebo nie. Predstavuje aj tabuľku *to\_do\_item\_table* v **databáze** SQLite.

## ToDoSubItem

Trieda – jedna podúloha, ktorá má názov a údaj o tom, či je ukončená alebo nie. Predstavuje aj tabuľku *to\_do\_sub\_item\_table* v **databáze** SQLite.

## ItemAdapter

Potomok triedy **ListAdapter** (potomok triedy RecyclerView.Adapter), ktorý pri zmene svojho obsahu umožňuje porovnávať objekty na pozadí a v RecyclerView aktualizuje len tie, v ktorých nastala zmena. Obsahuje vnorený **DiffUtil**.ItemCallback, kde sú prepísané metódy porovnávania nového itemu voči starémmu. Obsahuje aj ItemViewHolder, ktorý má na starosti jednotlivé itemy (úlohy) v zozname obsahuje interface, ktorý obsahuje metódy na spracovanie klikov na checkBox v úlohe alebo calkovo na item.

## SubItemAdapter

Potomok triedy RecyclerView.Adapter, ktorý spája dáta s RecyclerView. Obsahuje aj ItemViewHolder, ktorý má na starosti jednotlivé itemy (úlohy) v zozname a obsahuje interface, ktorý obsahuje metódy na spracovanie klikov na checkBox v podúlohe.

## ActivityConstants

Obsahuje konštanty používané v aplikácií na prenos dát medzi aktivitami.

## ToDoListDao

Interface, ktorý slúži na definovanie operácií v databáze SQLite.

## ToDoListRoomDatabase

**Room**. Poskytuje priamy prístup k databáze. Ak ešte nie je databáza vytvorená, tak ju vytvorí a naplní defaultnými záznamami. Túto akciu vykonáva na pozadí za pomoci **AsyncTask**

## ToDoListRepository

Vrstva abstrakcie. Vytvára API pre ViewModel, pre zjednodušenie používania. Vykonáva akcie s databázou na pozadí pomocou **AsyncTask**.

## ToDoItemViewModel

Udržiava dáta (úlohy) UI v MainActivity ako **LiveData**. Sleduje úlohy, ktoré sa zobrazujú recyclerView a reaguje na ich zmeny, ktoré sa prejavia v UI. Berie dáta pre užívateľa z ToDoListRepository a takisto doňho posiela užívateľove interakcie. Je „lifecycle aware“, preto sa nemusíme starať o ukladanie dát v UI pri zmene konfigurácie.

## ToDoSubItemViewModel

Udržiava dáta (podúlohy) UI v SubItemsActivity ako **LiveData**. Sleduje úlohy, ktoré sa zobrazujú recyclerView a reaguje na ich zmeny, ktoré sa prejavia v UI. Berie dáta pre užívateľa z ToDoListRepository a takisto doňho posiela užívateľove interakcie. Je „lifecycle aware“, preto sa nemusíme starať o ukladanie dát v UI pri zmene konfigurácie.

Všetky zdroje, ako texty, obrázky sú uložené v zložke **res**.

Vytváral som si vlastné UI:

* aktivity – animácia, kde sa menia gradienty
* príchod fragmentu – príchod z prava
* Vlastný štýl buttonov