Mobil programozási alapok

Feladat nyilvántartó mobilalkalmazás

(Task Manager)

Sikora Dávid Ádám
IRE699

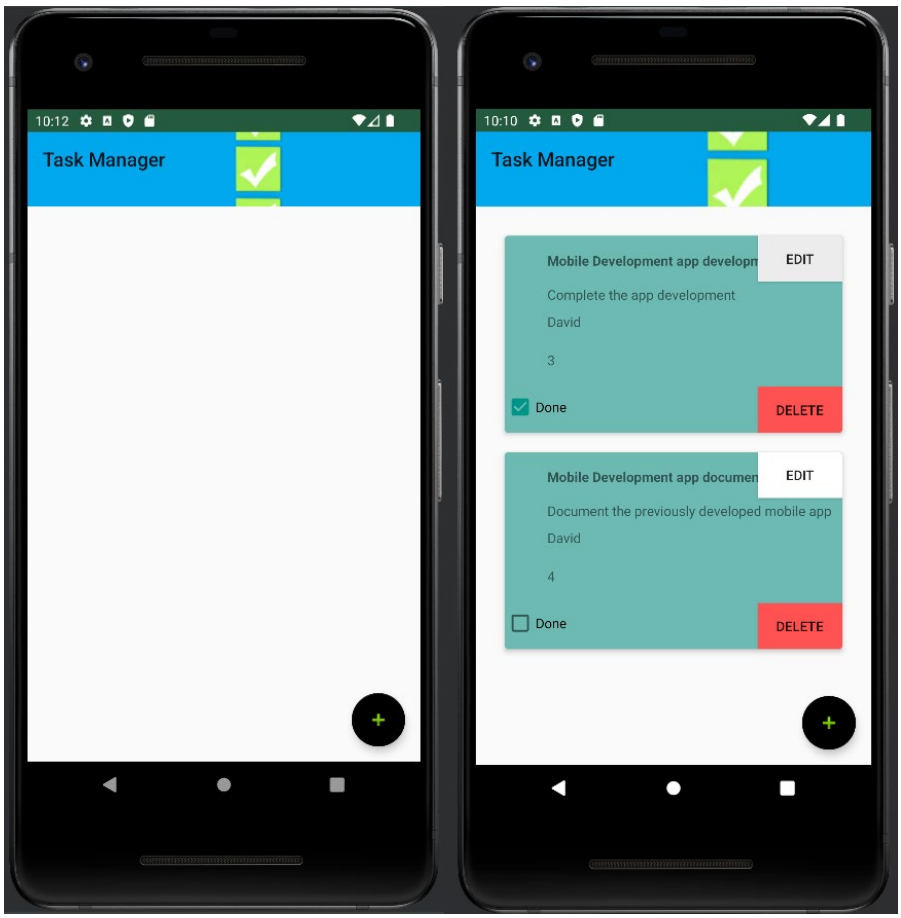
Tartalomjegyzék

1. Felhasználói felület – UI 3
1.1. A feladatok listázása 3
1.2. A feladatok felvitele és módosítása 4
2. A programkód 5
2.1. Adatbázis 5
2.2. A *Task* egyed 6
2.3. TaskItemDAO 6
2.4. Main Activity 7
2.5. TaskDialog 7
2.6. TaskAdapter 8

1. Felhasználói felület – UI

A Task Manager nevezetű alkalmazás egy egyszerű, helyi adatbázison alapuló feladat menedzsmentre alkalmas app. A felhasználó létre tud hozni új feladatokat, késznek nyilvánítani őket, szerkeszteni, végül a nem szükségeseket törölheti is.

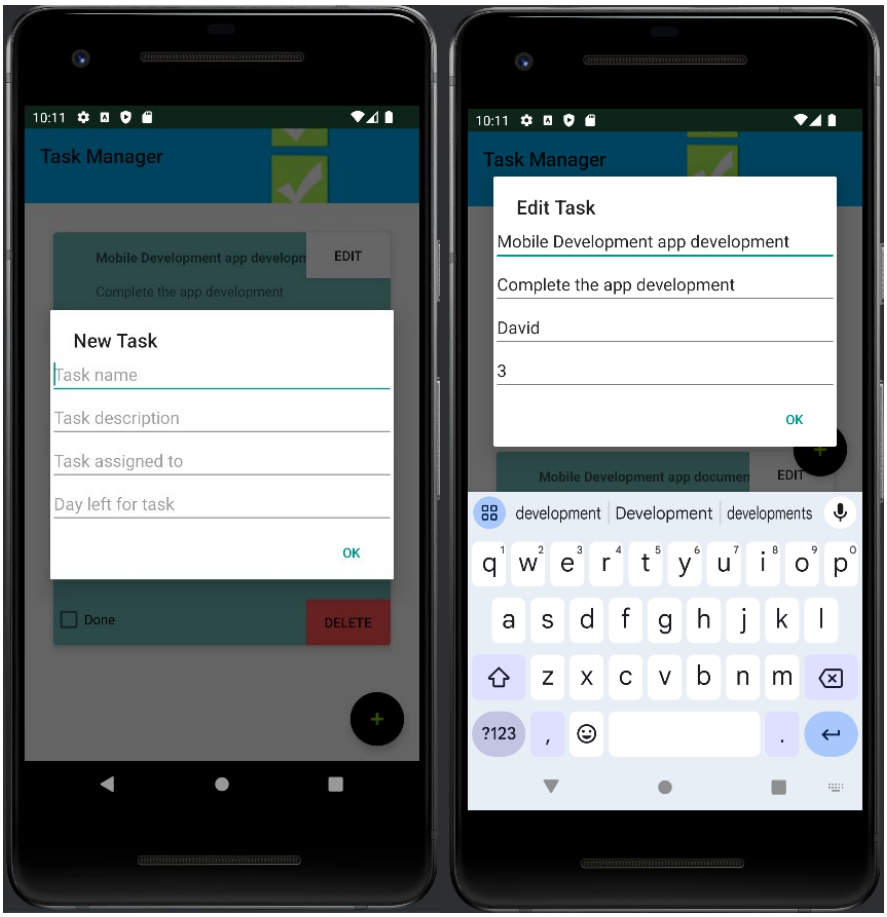
1.1. A feladatok listázása



A jobb alsó sarokban megjelenő zöld pluszjel segítségével adhatunk hozzá új feladatokat az alkalmazásunkhoz. A feladatok megadásánál kötelező a név megadása a felhasználó számára könnyű identifikáció érdekében, azonban ez módosítható később.

A leírás, a feladat megoldására beosztott személy és a határidő nem kötelezően megadandó adat. A leírásba a feladat bővebb leírására van lehetőség. A megoldásra beosztott személy megadásával nem csak a saját, hanem másoknak kiadandó vagy mások segítségével megoldandó feladatokat is felvehetünk alkalmazásunkba. A hátralevő napok száma a határidő tartásához szükséges. Ez a feladat közeledtével csökkenni fog, naponta, a határidő letelte után pedig mínuszba is megy, jelezve, hogy ez egy lejárt feladat.

1.2. A feladatok felvitele és módosítása



Az egyes feladatok egy listában jelennek meg kártyák formájában. Az egyes feladatok a kártyákon megjelenő gombok segítségével módosíthatóak és törölhetőek, valamint a Done felirat melletti kipipálható mezővel jelezhető a befejezett állapot.

A feladatok módosításakor a létrehozásnál is megjelenő felugró ablakban módosíthatunk minden megadott értéket.

2. A programkód

Az alkalmazás az Android Studio Bumblebee verziójában Kotlin nyelven történt leprogramozásra. Az alkalmazás forrásfile-jai az alábbi github repositoryban elérhetőek:

Redd-15/Mobile-Development-IRE699: Mobile Development hand-in IRE699

2.1. Az adatbázis

Az alkalmazás alapját adó adatbáziskapcsolatot az *AppDatabase* osztály menedzseli, melynek egyetlen példánya a *getInstance()* függvény meghívásával kérhető le, illetve ennek legelső meghívásakor jön létre. Az adatok egy lokális *tasks.db* nevű file-ba kerülnek mentésre.

```
@Database(entities = arrayOf(Task::class), version = 2)
abstract class AppDatabase : RoomDatabase() {

    abstract fun TaskItemDao(): TaskItemDAO

    companion object {
        private var INSTANCE: AppDatabase? = null

        fun getInstance(context: Context): AppDatabase {
            if (INSTANCE == null) {
                INSTANCE = Room.databaseBuilder(context.applicationContext,
                        AppDatabase::class.java, name: "tasks.db")
                        .build()
            }
            return INSTANCE!!
        }

        fun destroyInstance() {
            INSTANCE = null
        }
    }
}
```

2.2. A *Task* egyed

Az egyedek definíciója a *Task* modell osztályban került implementálásra. Az egyed a felhasználói felületen is megjelenő adatokat tartalmazza *String*-ként triviális módon. Ez alól kivételt képez az elsődleges kulcs, mely az adatbázis által generált egyedi azonosító. Ezt az adatbázis az egyed mentésekor generálja. További kivételek még a *day_left* és a *done* változók. A *day_left* név az adatbázis oldaláról félrevezető név, mivel az adatbázisban a dátum kerül letárolásra, mely napon le fog járni a feladat határideje. Azonban amint az adatbázisból beolvasásra kerül az egyed, akkor ez átváltásra kerül a hátra levő napok számává, ezzel átláthatóbbá téve a sürgős feladatok fontosságát. A *done* változó nem string, hanem boolean típusként kerül letárolásra, mivel az csak a feladat kész állapotát jelzi.

```
@Entity(tableName = "tasklist")
data class Task(@PrimaryKey(autoGenerate = true) var itemId: Long?,
                @ColumnInfo(name = "name") var name: String,
                @ColumnInfo(name = "description") var description: String,
                @ColumnInfo(name = "assigned_to") var assigned: String,
                @ColumnInfo(name = "day_left") var day_left: String,
                @ColumnInfo(name = "done") var done: Boolean
) : Serializable
```

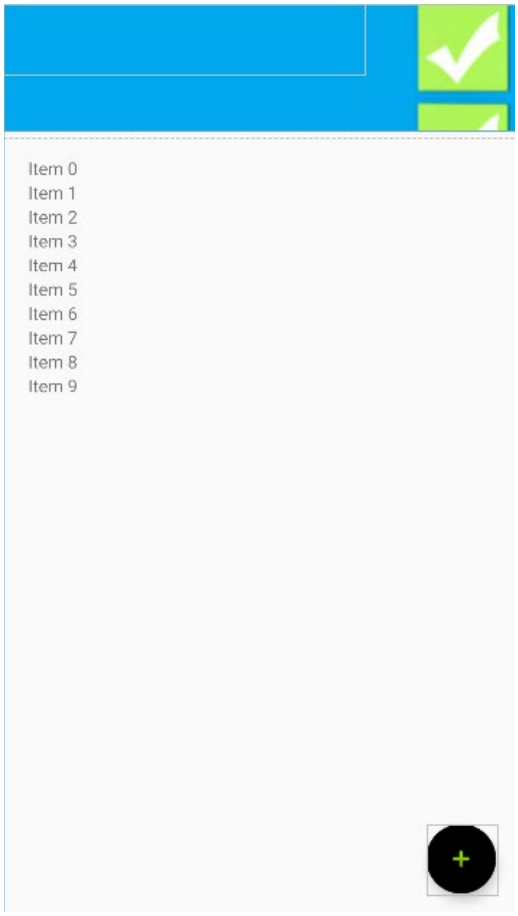
2.3. TaskItemDAO

A *TaskItemDAO* interface felelős az adatokon végezhető adatbázisműveletek definiálására. Itt az összes feladat lekérdezése, új létrehozása, illetve adott feladat módosítása és törlése szerepel.

```
@Dao
interface TaskItemDAO {

    //Az összes listázása
    @Query( value: "SELECT * FROM tasklist")
    fun findAllItems(): List<Task>

    //Egy elem beszúrása
    @Insert
    fun insertItem(item: Task): Long
    //Egy törlése
    @Delete
    fun deleteItem(item: Task)
    //Egy módosítása
    @Update
    fun updateItem(item: Task)

}
```

2.4. Main Activity

A Main Activity file-ban található függvények vezérlik az alkalmazás fő futását. Itt kerültek implementálásra az alkalmazás indításakor és leállításakor kezelendő akciók, valamint a feladatok listájának létrehozása és azok kezelése. Egy feladat módosítása vagy egy új létrehozása esetén a TaskDialog osztály kerül meghívásra



2.5. TaskDialog

Ezen osztály kezeli azon dialógust mely a létrehozáskor és módosításkor ugrik fel. Itt kerülnek az értékek módosításra vagy megadásra.

Ezen osztály felel továbbá a számításokért is a feladat lejártáig maradó napok számát illetően. Módosítás és létrehozás esetén az aznapi dátum alapján kerül kiszámításra, hogy mely nap lesz a pontos határidő és ezen dátum kerül végül lementésre String formátumban az adatbázisba. Később a megjelenítéskor ez visszaszámolásra kerül.

```
private fun handleItemCreate() {

    var today = LocalDate.now()
    var day_left = Period.of( years: 0, months: 0,etDay_left.text.toString().toInt())
    var expireDate = today.plus(day_left)
    TaskItemHandler.TaskItemCreated(Task(
            itemId: null,
            etName.text.toString(),
            etDescription.text.toString(),
            etAssigned.text.toString(),
            expireDate.toString(),
            done: false
    ))
}
```

2.6. TaskAdapter

Végül a *TaskAdapter* osztály felel a feladatokat listázó *RecyclerView* tartalmának kezeléséért. Az *items* lista tartalmazza az alkalmazás használata közben folyamatosan változó *task* objektumokat, melyeket a létrehozás szerint rendez sorba. Az osztály szorosan együttműködik a *row_item.xml* file-al mivel ennek segítségével hozza létre és menedzseli a feladat kártyákat az alkalmazás.