

Workout



Ročníková práce

Dominik Patzák

4.E

2019/2020

Čestné prohlášení

Prohlašuji, že jsem jediným autorem tohoto projektu, všechny citace jsou řádně označené a všechna použitá literatura a další zdroje jsou v práci uvedené. Tímto dle zákona 121/2000 Sb. (tzv. Autorský zákon) ve znění pozdějších předpisů uděluji bezúplatně škole Gymnázium, Praha 6, Arabská 14 oprávnění k výkonu práva na rozmnožování díla (§ 13) a práva na sdělování díla veřejnosti (§ 18) na dobu časově neomezenou a bez omezení územního rozsahu.

V Praze dne:..... Podpis:.....

Anotace

Cílem ročníkového projektu bylo navrhnout a naprogramovat mobilní aplikaci určenou pro cvičení s přidanou váhou tak, aby to bylo co nejpřístupnější uživateli. Důležitá byla i implementace přehledného grafického prostředí, které je pro tento typ aplikace stěžejní. Práce obsahuje podrobné vysvětlení problematiky aplikace.

Abstract

The goal of this project was to design and develop a mobile application aimed towards weightlifting, which would be as accessible to the user as possible. The implementation of a well-arranged graphic environment was vital for this type of application. This documentation contains in-depth explanation of the problematics of this application.

Zadání práce

Tréninková aplikace, která bude sloužit jako místo pro programy na kulturistiku a silový trénink. Program v tomto smyslu znamená soustavu cviků (dřep, mrtvý tah, shyb), které budou členěny do tréninkových dní a týdnů. Bude možné přidávat a modifikovat jednotlivé tréninkové programy (tj. přidávat a odebírat cviky, různé cviky v týdnu aj.). Dale zde budou grafy, které budou zobrazovat zlepšení ve velké trojce (tj. změna váhy v čase ve dřepu, mrtvém tahu, tlaku na rovné lavici). Grafické prostředí umožní snadnou manipulaci s každým tréninkovým dnem. Aplikace bude obsahovat i nastavení pro personalizaci. Bude vyvíjena na Android.

Obsah

1	Úvod	6
2	Použité technologie	6
2.1	Android Studio	6
3	Backend	6
4	Frontend	7
4.1	Aktivita	7
4.1.1	StartActivity	7
4.1.2	ChooseProgramActivity	7
4.1.3	NavBarActivity	8
4.2	Fragmenty	8
4.2.1	ProgramFragment	9
4.2.2	MaxesFragment	9
4.2.3	ProgressFragment	10
4.2.4	CustomizationFragment	11
4.2.5	CustomizationFragment_2	11
4.3	ItemAdapters	12
5	Závěr	13
6	Zdroje	14

1 Úvod

Fitness a zdravý životní styl je v poslední době na vzestupu. Mnoho lidí jakéhokoliv věku se začíná více starat o stravu, sport a zdraví. Velmi populárním se stává mezi muži (ale i ženami) kulturistika a silový rozvoj. Tato aplikace se snaží usnadnit přístup výše zmíněným lidem formou jednoduchého prostředí. Dokumentace pojednává o tom, jak aplikace a funguje a jak ji může uživatel použít pro své potřeby. Jako hlavní kapitoly této dokumentace jsem zvolil Backend a Frontend. Backend pojednává o databázi v aplikaci, zatímco Frontend pojednává o grafickém zpracování projektu a o aktivitách, fragmentech a adaptérech. Cílem bylo vytvořit přehlednou a intuitivní aplikaci pro snadnou orientaci. Aplikace byla vyvíjena v Android studiu.

2 Použité technologie

2.1 Android Studio

Jakožto vývojové prostředí jsem si zvolil Android Studio. Důvod je jasný - mým záměrem bylo vytvořit mobilní aplikaci pro Android. Studio je velmi efektivní nástroj, neboť nabízí jednoduché propojení jak frontendu, tak i backendu díky zabudované podpoře pro SQLite databázi, kterou jsem použil. Velmi příjemnou vlastností Android Studia je i integrace VCS (Version Control System), respektive integrace s GitHubem, který značně ulehčil operaci s verzemi projektu. Velmi vítaným aspektem byla možnost graficky upravovat jednotlivé části projektu přímo.

3 Backend

Aplikace běží na SQLite databázi, která byla zvolena kvůli její integraci s Android studiem a kvůli jejímu snadnému použití. S touto aplikací komunikuje jediná třída, DBHandler. Třída obsahuje mnoho příkazů na vytvoření struktury databáze. Databáze má dohromady 8 tabulek, které jsou vzájemně propojeny. Dále databáze obsahuje i základní data, která se do databáze zapíší při prvním spuštění aplikace a umožní tak její chod.

```
@Override
public void onCreate(SQLiteDatabase db) {
    String tbl_cre = "CREATE TABLE IF NOT EXISTS ";
    String EXEC_TABLE = tbl_cre + TB_EXERCISES + "(" + COL_EXERCISES_ID + " INTEGER PRIMARY KEY, " + COL_EXERCISES_NAME + " TEXT )";
    String PROGRAMS_TABLE = tbl_cre + TB_PROGRAMS + "(" + COL_PROGRAM_ID + " INTEGER PRIMARY KEY, " + COL_PROGRAM_NAME + " TEXT, " + COL_PROGRAM_TYPE + " TEXT, " + COL_PROGRAM_WEEKS + " INTEGER, " +
    String WEEKS_TABLE = tbl_cre + TB_WEEKS + "(" + "id" + " INTEGER PRIMARY KEY, " + COL_WEEK_PRGNAME + " TEXT, " + COL_WEEK_WEEKID + " INTEGER, " + COL_WEEK_DAYID + " INTEGER, " + COL_WEEK_EXEC + "
    String MAXES_TABLE = tbl_cre + TB_MAXES + "(" + "id" + " INTEGER PRIMARY KEY, " + COL_MAXES_EXEC + " TEXT, " + COL_MAXES_WEIGHT + " FLOAT, " + "FOREIGN KEY(" + COL_MAXES_EXEC + ") REFERENCES " + T
    String CURRENT_PROGRAM_TABLE = tbl_cre + TB_CURR + "(" + "id" + " INTEGER PRIMARY KEY, " + COL_CURR_PROGRAM + " TEXT, " + COL_CURR_WEEK + " INTEGER, " + COL_CURR_DAY + " INTEGER, " + "FOREIGN KEY
    String PROGRAM_INCREASES_TABLE = tbl_cre + TB_INCREASES + "(" + "id" + " INTEGER PRIMARY KEY, " + COL_INC_NAME + " TEXT, " + COL_INC_AMOUNT + " FLOAT, " + "FOREIGN KEY(" + COL_INC_NAME + ") REFERE
    String PROGRESS_TABLE = tbl_cre + TB_PROGRESS + "(" + "id" + " INTEGER PRIMARY KEY, " + COL_PROGRESS_DATE + " TEXT, " + COL_PROGRESS_NAME + " TEXT, " + COL_PROGRESS_AMOUNT + " FLOAT, " + "FOREIGN
    String CUSTOM_TABLE = tbl_cre + TB_CUSTOM + "(" + "id" + " INTEGER PRIMARY KEY, " + COL_CUSTOM_WEEK + " INTEGER, " + COL_CUSTOM_DAY + " INTEGER )";
    db.execSQL(EXEC_TABLE);
    db.execSQL(PROGRAMS_TABLE);
    db.execSQL(WEEKS_TABLE);
    db.execSQL(MAXES_TABLE);
    db.execSQL(CURRENT_PROGRAM_TABLE);
    db.execSQL(PROGRAM_INCREASES_TABLE);
    db.execSQL(PROGRESS_TABLE);
    db.execSQL(CUSTOM_TABLE);
}
```

Obrázek 1: Příklad třídy DBHandler

4 Frontend

Frontend této aplikace se skládá převážně z aktivit a fragmentů, kde všechny z nich mají definované vlastní soubory rozvržení. Z důvodu implementace elementů jako ListView, které zobrazují seznam dat z databáze, bylo nutno i jim přidat jejich vlastní soubory, kde je definována jejich grafické zpracování. Těchto případů je v projektu hned několik. Kromě aktivit a fragmentů je zde i třída komunikující s databází.

4.1 Aktivita

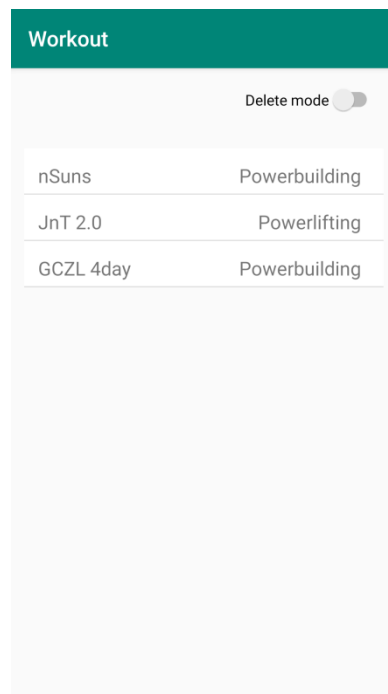
Okno, kde program vykresluje uživatelské prostředí, je poskytnuto právě aktivitami. Obecně platí, že jedna aktivita = jedno okno. Tato aplikace obsahuje více oken, což znamená, že i více aktivit. Obvykle se v aplikaci nachází jedna aktivita, takzvaná hlavní aktivita, která se objeví uživateli při spuštění aplikace. V tomto případě je touto aktivitou NavBarActivity, která slouží jako rozcestí mnoha pod-aktivit a usnadňuje manipulaci s programem. Až na jednu výjimku, tj. při prvním spuštění aplikace, je vždy první spuštěnou aktivitou. Každá aktivita v této aplikaci zahajuje jinou činnost za účelem provádění různých akcí. Přestože tyto aktivity vypadají pro uživatele jako soudržný celek, není tomu tak. Mezi aktivitami v aplikaci jsou většinou minimální závislosti.

4.1.1 StartActivity

StartActivity je prvním oknem, které se uživateli zobrazí při prvním spuštění aplikace. Je velmi jednoduchá a její jedinou funkcí je přechod na následující aktivitu pomocí stisknutí tlačítka.

4.1.2 ChooseProgramActivity

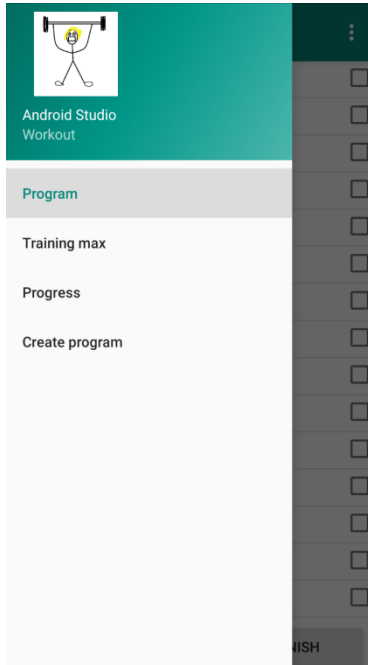
Tato aktivita slouží pro zvolení a odebrání programů. Uživateli je načten list všech stávajících programů a základní informace o nich. Chce-li uživatel program odebrat, jednoduše stačí kliknout na Switch v pravém dolním rohu. Jakmile se tak stane, programy je možné smazat. Pokud Switch je vypnutý, kliknutím na zvolený program se uživatel přenese na hlavní aktivitu.



Obrázek 2: Volení programu

4.1.3 NavBarActivity

Primární aktivita aplikace, s jejíž pomocí může uživatel snadno přepínat mezi ostatními okny. Tato aktivita se skládá z mnoha částí, mezi něž patří například Navigation bar, Toolbar a Header. V navigaci se může uživatel snadno pohybovat mezi jednotlivými fragmenty v nabídce. V Toolbaru se nachází menu, které obsahuje možnost vrátit se zpět a změnit programy.



Obrázek 3: Navigace

4.2 Fragmenty

Fragment reprezentuje chování nebo část uživatelského rozhraní ve `FragmentActivity`. V jedné aktivitě je možné mít více fragmentů, díky nimž je možné postavit UI s více okny. Fragmenty se dají znovu používat i v jiných aktivitách. Fragment je tedy jakási modulární sekce aktivity, která má svůj vlastní lifecycle, má svoje vlastní akce na sběr dat a kterou je možné přidat či odebrat během chodu aktivity. Fragment by se dal tedy jinými slovy označit za pod-aktivitu.

Fragmenty však musí být vždy částí nějaké aktivity a jejich životní cyklus je závislý na této aktivitě. Například pokud aktivitu zmrazíme, zmrazí se i všechny fragmenty v ní. V této aplikaci se nachází dohromady 5 fragmentů a všechny až na jeden jsou součástí navigace.

4.2.1 ProgramFragment

Primární fragment starající se o načtení dat o programu. Tyto data naplní ListView, který poté zobrazí výčet cviků na momentální den v týdnu. Uživatel vidí jméno cviku, váhu, počet setů a počet opakování.

Některé sety (počet setů znamená kolikrát má uživatel udělat počet opakování), které jsou jednoduše rozlišitelné od ostatních pomocí prázdného pole v počtu opakování, jsou takzvané postupové. Tyto sety znamenají, že uživatel se na dané váze snaží udělat co nejvíce opakování. Podle vykonaného počtu se zvyšuje maximální váha, kterou uživatel zadává při začátku programu. Maximální váha slouží k procentuálnímu vypočtení cviků v programu, které jsou částí velké čtyřky, o které se zmíním níže. ProgramFragment také obsahuje tlačítko v pravé dolní části, díky němuž se uživatel přenesne na další tréninkový den. Pokud ovšem nevyplnil postupový set, aplikace ho dále nepustí.

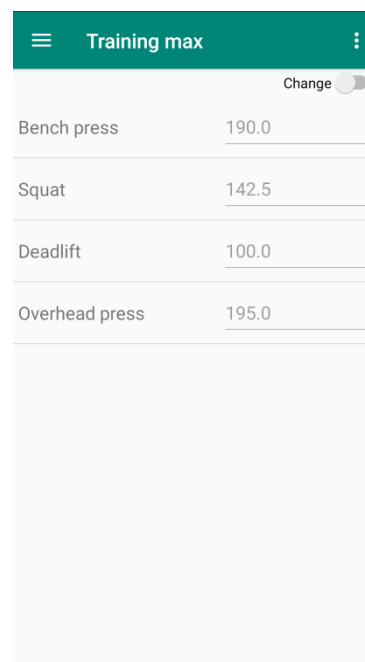


Program				
Bench press	1	8	123.5	<input type="checkbox"/>
Bench press	1	6	142.5	<input type="checkbox"/>
Bench press	1	4	161.5	<input type="checkbox"/>
Bench press	1	4	161.5	<input type="checkbox"/>
Bench press	1	4	161.5	<input type="checkbox"/>
Bench press	1	5	152.0	<input type="checkbox"/>
Bench press	1	6	142.5	<input type="checkbox"/>
Bench press	1	7	133.0	<input type="checkbox"/>
Bench press	1	8	123.5	<input type="checkbox"/>
Overhead press	1	6	50.0	<input type="checkbox"/>
Overhead press	1	5	60.0	<input type="checkbox"/>
Overhead press	1	3	70.0	<input type="checkbox"/>
Overhead press	1	5	70.0	<input type="checkbox"/>
Overhead press	1	7	70.0	<input type="checkbox"/>
Overhead press	1	4	70.0	<input type="checkbox"/>
				FINISH

Obrázek 4: Program

4.2.2 MaxesFragment

V tomto jednoduchém fragmentu jsou obsaženy políčka pro vyplnění maximální váhy pro dané cviky. Aplikace obsahuje čtyři základní cviky, s nimiž většina silových programů pracuje a na jejich bázi vypočítává procentuálně váhy v programech. Tyto čtyři cviky jsou: Tlak na rovné lavici, Mrtvý tah, Dřep a Tlak v sedě. Změna maximální váhy postupuje následovně: Klikneme na Switch a poté již jednoduše zadáme hodnotu pro požadovaný cvik.

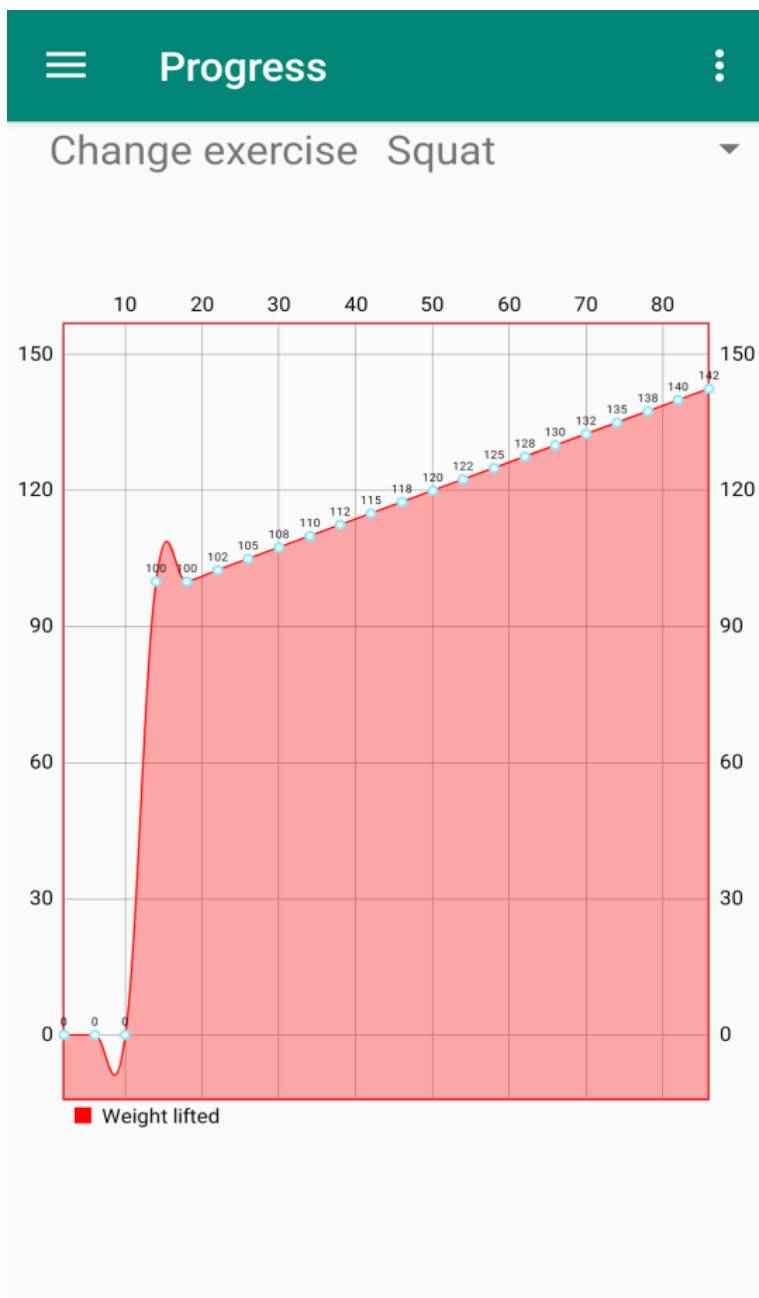


Training max	
	Change <input type="checkbox"/>
Bench press	<input type="text" value="190.0"/>
Squat	<input type="text" value="142.5"/>
Deadlift	<input type="text" value="100.0"/>
Overhead press	<input type="text" value="195.0"/>

Obrázek 5: Nastavování vah

4.2.3 ProgressFragment

Z názvu již vypovídá, že v tomto fragmentu se nachází grafy zobrazující váhový postup v čase. Pomocí MP AndroidCharts se generují grafy pro uvedené cviky v MaxesFragment. Pro každý z těchto cviků je možné zobrazit si graf postupu. Tento postup se mění dle počtu opakování v postupových setech programu.



Obrázek 6: Grafy znázorňující postup

4.2.4 CustomizationFragment

Pokud si přeje uživatel přidat vlastní program, naviguje na tento fragment. Zde se nachází mnoho polí, které slouží k definování základních vlastností programu, jako například počet dnů v týdnu, počet týdnů, jméno a typ (silový, kulturistický a mix těchto dvou). Grafické rozhraní je velmi jednoduché a snadno pochopitelné. Dvě tlačítka na samém spodku fragmentu slouží k navigaci. Červené tlačítko vrátí uživatele zpět do ProgramFragment, zelené tlačítko ho přesměruje na CustomizationFragment_2, pokud vyplní správně všechny hodnoty.

Obrázek 7: Vytváření programu

4.2.5 CustomizationFragment_2

CustomizationFragment_2 je druhou fází vytváření vlastního programu. Zde uživatel zadává cviky, které bude v jednotlivých dnech vykonávat. Každý cvik má čtyři parametry, a to jméno, sety, opakování, váhu a popřípadě zda-li se jedná o postupový set. Pro snazší zadávání cviků je zde vložen seznam nejčastějších cviků. Po klepnutí na ně se automaticky nastaví jméno cviku. Jakmile je uživatel spokojen s cviky na momentální den, tlačítkem dole se přesune na den nastávající. Po zadání všech tréninkových dnů je program přidán a uživatel s ním může začít cvičit.

Obrázek 8: Výběr cviků

4.3 ItemAdapters

ItemAdapters jsou další důležitou částí projektu. Jsou zodpovědní za správné načítání dat do seznamů v aktivitách i fragmentech. Slouží jako prostředník mezi databází a grafickým zobrazením. Těmto adaptérům jsou zprvu dodána data, která slouží k zaplnění jednotlivých elementů. Tyto elementy mohou být například tlačítka, textové pole, či pole pro editaci. Jakmile jsou tyto elementy zaplněny, pomocí `LayoutInflater` se převede rozvržení elementů na obrazovce na finální verzi `View`. Tato verze v `ListView`, seznamu v aplikaci používaném pro zobrazení dat, reprezentuje jeden řádek. Každý `ItemAdapter` v aplikaci je odlišný, neboť každý fragment či aktivita požaduje jinou formu zobrazení dat.

```
public class MaxesAdapter extends ArrayAdapter<MaxesData> {
    private ArrayList<MaxesData> dataSet;
    Context myContext;

    public static class MyViewHolder {
        TextView txtname;
        TextView txtweight;
    }

    public MaxesAdapter(ArrayList<MaxesData> data, Context context) {
        super(context, R.layout.row_maxes_item, data);
        this.dataSet = data;
        this.myContext = context;
    }

    @Override
    public View getView(int position, View convertView, ViewGroup parent) {
        MaxesData data = getItem(position);
        MyViewHolder viewHolder;

        if (convertView == null) {
            viewHolder = new MyViewHolder();
            LayoutInflater inflater = LayoutInflater.from(getContext());
            convertView = inflater.inflate(R.layout.row_maxes_item, parent, attachToRoot false);
            viewHolder.txtname = convertView.findViewById(R.id.maxesName);
            viewHolder.txtweight = convertView.findViewById(R.id.maxesWeight);
            convertView.setTag(viewHolder);
        } else {
            viewHolder = (MyViewHolder) convertView.getTag();
        }
        viewHolder.txtname.setText(data.getName());
        viewHolder.txtweight.setHint(String.valueOf(data.getWeight()));
        convertView.setOnClickListener((v) -> {
            Toast.makeText(myContext, text: "click item", Toast.LENGTH_LONG).show();
        });
        return convertView;
    }
}
```

Obrázek 9: Příklad adaptéru

5 Závěr

S finální verzí aplikace jsem vcelku spokojen, avšak mrzí mě, že v aplikaci jsou menší chyby, které zpomalují její chod či ji jinak ovlivňují. Jako příklad bych uvedl optimalizaci. Aplikace optimalizována je jen minimálně a tento fakt je viditelný při občasném delším čekání na načítání dat z databáze. Ovšem pokud se podívám z obecného hlediska, tak zadání této práce jsem splnil a vyvíjení této aplikace mi přineslo mnoho zkušeností, kterých si opravdu vážím a které zcela jistě využiji při dalším studiu na vysoké škole.

6 Zdroje

Stack Overflow: <https://stackoverflow.com/>

Android Studio dokumentace: <https://developer.android.com/docs>