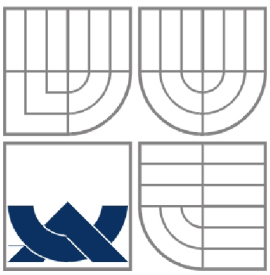
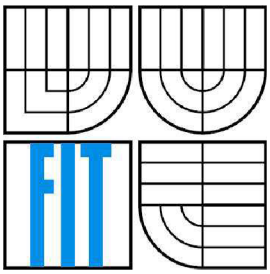


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDI

ANDROID APLIKACE PRO VÝUKU SLOVNÍ ZÁSOBY

ANDROID APPLICATION FOR LEARNING VOCABULARY

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — Jiří Černoch
AUTHOR

VEDOUCÍ PRÁCE — Ing. Horváth Zsolt
SUPERVISOR

BRNO 2013

Abstrakt

Tato bakalářská práce popisuje vývoj aplikace k výuce slovní zásoby na zařízeních s operačním systémem Android. V práci je pospána základní teorie, návrh aplikace a implementace stěžejních problémů. Celá aplikace je zaměřena na práci s SQLite databází a optimalizací na zařízení o různých velikostech displeje.

Abstract

This bachelor thesis describes the development of application for learning vocabulary on devices with the operating system Android. In thesis you can find basic common theory relevant to this problem, analyze of existing solutions, design of basic architecture and description of detail implementation. Program is based on SQLite database and is optimized for devices with different size of screen.

Klíčová slova

Android, učení slovní zásoby, výuka, metody učení, „kartičková“ metoda, vícejazyčnost, fragment, tablet, SQLite

Keywords

Android, learning vocabulary, teaching, learning methods, cards method, multilingualism, fragment, tablet, SQLite

Citace

Jiří Černoch: Android aplikace pro výuku slovní zásoby, bakalářská práce, Brno, FIT VUT v Brně, 2013

Android aplikace pro výuku slovní zásoby

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Horvátha Zsolta. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Jiří Černoch
14. 5. 2013

Poděkování

Rád bych zde poděkoval panu Ing. Horvátovi Zsoltovi, vedoucímu této bakalářské práce, za jeho odbornou pomoc při řešení, ochotu přizpůsobení zadání práce, jeho čas a připomínky k práci.

Obsah

1 Úvod

V dnešní době si člověk nevystačí pouze s mateřským jedním jazykem. Při učení nového jazyka je nejtěžší naučit se nová slovíčka. Je to však nutný základ, na kterém je znalost cizího jazyka postavena. Je to ale však i dril, nuda a nezáživná činnost. Cestou jak toto zvládnout mohou být i všudypřítomné moderní technologie jako jsou například chytré telefony a tablety, které má dnes každý při ruce. Proto jsem se rozhodl tyto dvě věci spojit a vytvořit aplikaci pro výuku slovíček pro tato zařízení. Jako cílovou platformu jsem si vybral Android, protože se jedná o nejrozšířenější platformu, která je volně šiřitelná.

1.1 Proč jsem si toto téma vybral

I já jsem hledal aplikaci, na které bych se mohl učit cizí slovíčka. I několik přátel se mě ptalo, zda bych jim nějakou aplikaci nedoporučil. Sice jich existuje již mnoho, ale žádná nevyhovovala představám. Analýze existujících řešení bude vyhrazena samostatná kapitola. Rozhodl jsem se tedy v rámci bakalářské práce takovou aplikaci vytvořit.

1.2 Pro koho je aplikace určena

Učením nových jazyků nebo opakováním již naučených, bychom se měli zabývat celý život, jelikož s jedním jazykem si člověk v dnešní době těžko vystačí. Pokud u sebe nechcete nosit neustále slovníček nebo učebnici cizího jazyka a máte k dispozici zařízení na platformě Android, pak se pomocí mého programu můžete slovíčka učit vždy a všude. Například v autobuse při cestě do práce, nebo když čekáte ve frontě a máte chvíli čas. V pohodlí domova se vám asi nebude chtít učit slovíčka na tak malém zařízení, pak můžete klidně použít třeba tablet.

1.3 Struktura práce

Celá práce je členěna do několika hlavních kapitol. Obsahem první kapitoly bylo uvedení do problému, zdůvodnění výběru tématu a vymezení cílových uživatelů.

Obsahem druhé kapitoly je teorie. Zde bude čtenář seznámen se základními teoriemi, pro řešení tohoto problému. Jako jsou základní informace o platformě Android, o vývoji na této platformě a o technologiích v aplikaci, které byly použity.

Další kapitola se zabývá podrobnou analýzou již existujících řešení. Rozborem jejich dobrých a špatných vlastností. Kapitola je doplněna názornými obrázky. Aplikace byla testována, jak na mobilním telefonu, tak na tabletu.

V další kapitole je popsán návrh uživatelského rozhraní a databáze pro ukládání dat. Posléze je popsána samotná implementace nejdůležitějších částí projektu.

Velmi důležitou kapitolou je kapitola testování. Bylo provedeno několik testování a na základě jejích výsledků byla aplikace upravena.

Závěr obsahuje zhodnocení dosažených výsledků. Také zde je nastíněn další možný vývoj.

2 Teorie

Aplikace je vyvíjena na platformě Android. Jedná se o nejrozsáhlejší volně dostupnou platformu pro mobilní zařízení. V dnešní době se vyskytuje v mnoha druzích zařízení od mobilních telefonů přes tablety a televize, ale také například i v troubě.

2.1 Historie Androidu

Společnost Android Inc. byla založena v roce 2003. V roce 2005 tuto firmu odkoupil Google Inc. a udělal z ní svoji dceřinou společnost. Tým Googlu vyvinul platformu založenou na Linuxovém jádře a v roce 2007 získal několik patentů v oblasti mobilní technologie. Poté začala veřejnost spekulovat, zda chce Google vstoupit na trh s mobilními telefony.

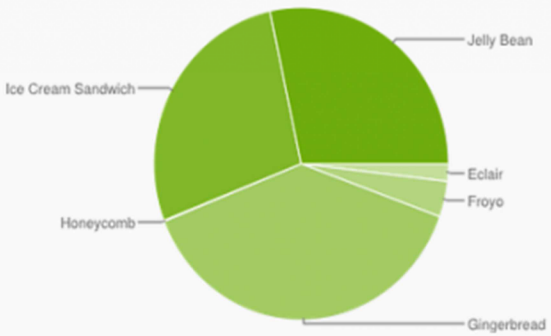
V roce 2007 bylo vytvořeno konsorcium Open Hansdet Alliance, které zahrnovalo významné firmy v oboru mobilních technologií včetně Googlu. Cílem bylo vyvinout otevřený standart pro mobilní zařízení. Výsledkem byl Android, otevřená mobilní platforma založená na linuxovém jádře. Byl také vydán první Android SDK balíček pro vývojáře.

V říjnu roku 2008 byl na trh Spojených států amerických uveden první komerční telefon od firmy HTC s operačním systémem Android. Pro rok 2011 tržní podíl platformy Android je 50,9% a je aktivován přibližně v 300 milionech zařízení.

Vyšlo již několik verzí tohoto systému. Úplně první byl Android 1.0 vydaný právě roku 2008. Aktuálně nejrozšířenější je stále Android 2.3 Gingerbread, který byl vydán roku 2010. Aktuálně nejnovější je verze 4.2.x Jelly Bean, která je zatím pouze na 2% zařízení. Na obrázku 1 můžeme vidět zastoupení jednotlivých verzí. Vývoj je takový, že procento verzí 4.0 + stoupá a starších ubývá. To je dobrá zpráva pro vývojáře, kteří mohou využívat nové technologie, které přišly s Androidem 4.0.

Tato podkapitola byla převzata z [3, 4]

Version	Codename	API	Distribution
1.6	Donut	4	0.1%
2.1	Eclair	7	1.7%
2.2	Froyo	8	3.7%
2.3 - 2.3.2	Gingerbread	9	0.1%
2.3.3 - 2.3.7		10	38.4%
3.2	Honeycomb	13	0.1%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	27.5%
4.1.x	Jelly Bean	16	26.1%
4.2.x		17	2.3%



Obrázek 1: Schéma zastoupení verzí androidu k 1. květnu 2013 podle společnosti Google

2.2 Architektura

Architektura Android platformy se skládá z pěti základních částí. Každá vrstva má svůj specifický účel a může být oddělena od vrstev ostatních.

Nejnižší vrstvu tvoří jádro. Jeho hlavním úkolem je zajišťování komunikace mezi hardwarem a softwarem. K tomuto účelu obsahuje ovladače pro konkrétní zařízení. Kromě těchto ovladačů vrstva také obsahuje správu procesů a napájení. Jádro operačního systému Android je postaveno na Linuxovém jádře verze 2.6.

Další vrstvu tvoří knihovny. Představují základní funkce systému a jsou psány v jazyce C/C++. Najdeme zde například knihovnu na podporu vykreslování 3D grafiky, databázovou knihovnu, knihovnu podporující přehrávání video a audio formátů, například. MPEG4, H.264, MP3, JPG a PNG a knihovnu pro rendering[1] bitmapových a vektorových fontů.

Nad vrstvou knihoven se nachází takzvaná Android Runtim vrstva. Ta mimo jiné obsahuje virtuální stroj zvaný Dalvik. Virtuální stroj se stará o převod kódu, ve kterém jsou aplikace napsány do nativního kódu. Kvůli licenčním problémům nebyl přímo použit virtuální stroj jazyka Java, ale musel být vytvořen právě Dalvik.

Z pohledu vývojáře nejdůležitější, vrstvou je Application Framework. Tato vrstva obsahuje knihovny napsané již v jazyce Java, které programátorovi umožňují pracovat s prvky operačního systému.

Tato podkapitola byla převzata z [3,13].

Obrázek 2: Schéma architektury [3]

[1] Rendering je tvorba obrazu na základě počítačového modelu

2.3 Prostředky pro vývoj aplikací

Všechny věci potřebné pro vývoj aplikace na platformě Android jsou volně stažitelné z internetu. Oficiálně podporovaným vývojovým prostředím je Eclipse, do kterého je potřeba si stáhnout balíček s vývojovou sadou zvaný Android SDK[2] a rozšíření Android Development Tools (ADT[3]).

SDK je balíček několika aplikací, knihoven a funkcí pro práci s Android aplikacemi. Je dostupný na všech hlavních platformách operačních systémů. Můžeme jej dělit na tři základní druhy: základní, doporučená a úplná konfigurace.

Rozšíření ADT slouží pro snadnější ovládání vývojového prostředí. Obsahuje mechanismy pro ladění aplikací, správu virtuálních zařízení a další užitečné funkce.[3,5]

2.4 Struktura Android aplikace

Základními čtyřmi stavebními bloky každé aplikace jsou tzv. komponenty aplikace. Jedná se o čtyři druhy komponent[6]:

- aktivita
- služba
- zdroj obsahu
- a všesměrový vysílač.

Aktivita představuje grafické uživatelské rozhraní pro interakci s uživatelem. Aktivita zpravidla odpovídá jedné obrazovce aplikace a je složena z několika uživatelských prvků. Jednotlivé aktivy lze mezi sebou přepínat a mohou si předávat informace. Každá aktivita má svůj životní cyklus, který je znázorněn na obrázku 5. Lze ji vyvolat příkazem `startActivity()`. Aktivita během svého života prochází různými stavy a využívá volání svých metod[7]:

- `onCreate()` - Metoda volající se po spuštění aktivity.
- `onStart()` - Metoda volající se před zobrazením aktivity uživateli.
- `onResume()` - Metoda volající se po zobrazení aktivity uživateli.
- `onPause()` - Metoda se volá, pokud se do popředí dostane jiná aktivita.
- `onStop()` - Zastavení aktivity při delším nezobrazení
- `onDestroy()` - Metoda volaná při rušení aktivity systémem.
- `onRestart()` - Metoda volána při opětovném navrácení aktivity do popředí.

Služba je komponenta, která běží na pozadí. Slouží k déle trvajícím operacím či přístupu k vzdáleným zdrojům. Jelikož běží na pozadí, tak neposkytuje uživateli žádné uživatelské rozhraní. Služby lze dělit na dva druhy a to spouštěné služby a na takzvané vázané služby. Hlavní rozdíl je, že u vázané služby může být navázáno více komponent zároveň.[3]

Zdroj obsahu je rozhraní pro sdílení dat mezi aplikacemi. Má poměrně jednoduché rozhraní, kdy nám k ovládání stačí několik základních metod.

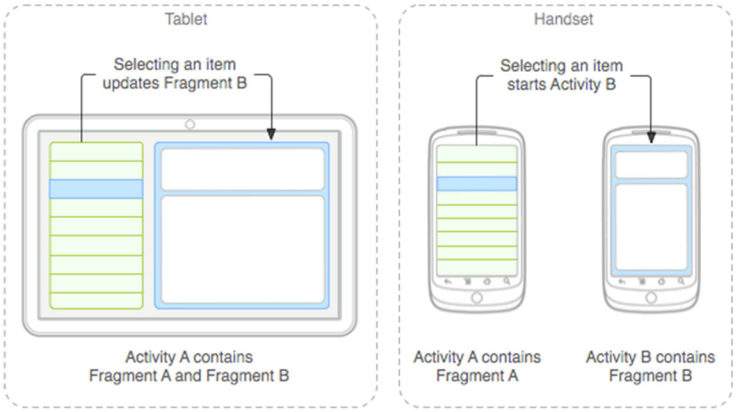
Všesměrový vysílač je opět komponenta, která nemá žádné uživatelské rozhraní, ale je velice důležitá. Umožňuje nám reagovat na události, které mohou v systému nastat.

---

[2] Android SDK balíček obsahuje knihovny nezbytné pro vývoj a testování aplikací.
[3] ADK je standartní Android plugin Eclipse.

2.5 Fragmenty

Z důvodu vzestupu poptávky po tabletech nastal problém, jak umožnit vývojářům přizpůsobovat své aplikace, jak pro malé tak pro velké displeje. Proto byly zavedeny fragmenty, které tento problém mají řešit. Fragment je nová vrstva, která byla umístěna mezi vrstvu `Activity` a `View`. Jedna aktivita může obsahovat více fragmentů. Logika fragmentů spočívá v tom, že dva fragmenty na velkém displeji mohou být zobrazeny na jedné obrazovce, naopak na menším může každý fragment představovat vlastní obrazovku. Schéma této logiky je uvedeno na obrázku 4.



Obrázek 3: Logika fragmentu

Životní cyklus fragmentu, jak můžeme vidět na obrázku 6 je podobný životnímu cyklu aktivity z obrázku 4.[7]

2.6 Velikosti obrazovek

Operační systém Android nalezneme na zařízeních, která mají různě velké obrazovky. Pokud navrhujeme aplikaci pro mobilní telefon, kde máme vytvořený hezký a přehledný vzhled, nemusí tomu tak být po nainstalování na zařízení s větší obrazovkou.

Vzhled aplikace se definuje pomocí XML souborů, které jsou uloženy ve složce `res/layout`[14]. Pokud chceme, aby se při použití aplikace na zařízeních s větší obrazovkou používali jiné zdrojové soubory můžeme si vytvořit nové složky se speciálními koncovkami. Ve složkách jsou umístěny XML soubory stejných názvů, ale odlišného obsahu a podle velikosti zobrazovací plochy se vybere, které zdrojové soubory se použijí. Pomocí těchto názvů složek zajistíme podporu pro více velikostí obrazovek.

Příklady některých názvů souborů[14]:

- `res/layout-land/` - obrazovka na šířku
- `res/layout-small/` - malá obrazovka
- `res/layout-large/` - velká obrazovka
- `res/layout-large-land/` - velká obrazovka a ještě na šířku
- `res/layout-ldpi/` - obrazovka s nízkou hustotou

- `res/layout-hdpi/` - obrazovka s vysokou hustotou

2.7 Datová uložiště

Většina aplikací si potřebuje ukládat informace o stavu aplikace, tak aby o ně uživatel nepřišel. U Android zařízení máme hned několik možností, jak toho docílit. Záleží na konkrétních potřebách, kterou možnost zvolíme.

Jednou z možností jsou sdílené předvolby, kdy jednoduchá soukromá data jsou uložena ve formátu klíč a hodnota.

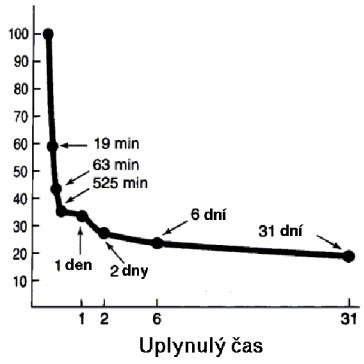
Další možností je vnitřní uložiště[8], kdy se soukromá data uloží do paměti zařízení. Data jsou uložena v soukromé paměti aplikace a ostatní aplikace k nim nemohou přistupovat. Ovšem po odinstalování budou data smazána.

Externí uložiště[8] slouží k ukládání dat, u kterých chceme, aby k nim mělo přístup více aplikací. Nevýhodou externích uložišť je, že nemusí být vždy dostupná. Proto před přístupem k datům je potřeba otestovat, zda jsou data dostupná.

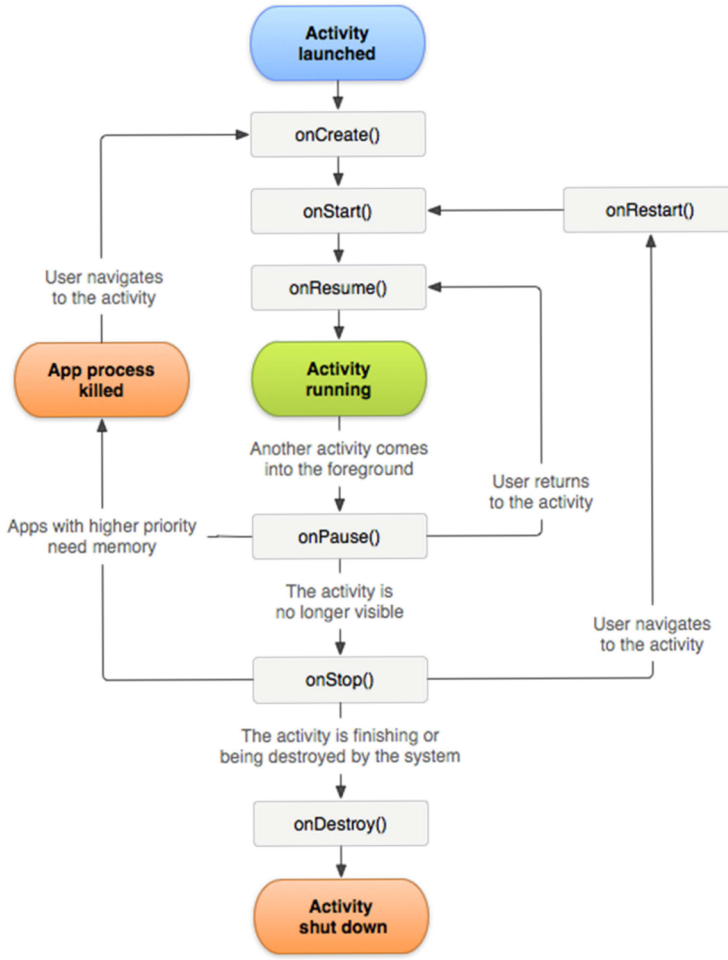
Dalším způsobem je databáze SQLite[8,15]. Jde o extrémně rychlé a spolehlivé uložiště. SQL je knihovna napsaná v jazyce C, kterou lze jednoduše připojit k programu. Pro používání databáze je pouze potřeba vytvořit třídu zděděnou ze třídy `SQLiteOpenHelper` a v ní přepsat metody `onCreate()` a `onUpgrade()`. Poté získat přístup ke čtení nebo zápisu do databáze.

2.8 Lidská paměť

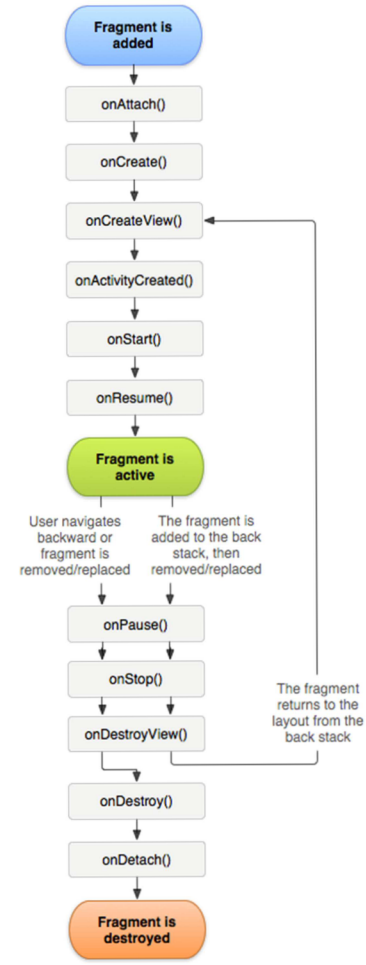
Bohužel platí, že pokud nějaké naučené slovíčko nepoužíváme, tak rychle zapomeneme. A naopak slovíčka, která používáme často si pamatujeme lépe. Proto je potřeba naši slovní zásobu opakovat. Jak můžeme vidět na obrázku, tak většinu naučených informací zapomeneme do několika hodin[1].



Obrázek 4: Ebbinghausova křivka zapomínání



Obrázek 5: živní cyklus aktivity



Obrázek 6: životní cyklus fragmentu

3 Analýza existujících řešení

Ve třetí kapitole se podíváme na již exitující řešení. Oficiálním obchodem s aplikacemi pro zařízení s Androidem je Google play. Tento obchod obsahuje velké množství aplikací. Mezi neoblíbenější v oblasti výuky slovíček patří aplikace: Slovíčka[16], TrainBrain[17], Dril[18], Vocab[19], Vocable Trainer[20], Vocablo 2[21] a VocabularyTrainer[22]. Výše zmíněné programy budeme analyzovat podle následujících důležitých parametrů:

- Design
- Slovní zásoba
- Optimalizace pro různá Android zařízení
- funkčnost

Na základě analýzy existujících řešení vytvoříme návrh vlastního řešení. Pokusíme se inspirovat dobrými vlastnostmi již existujících řešení a naopak se vyvarujeme jejich nedostatků.

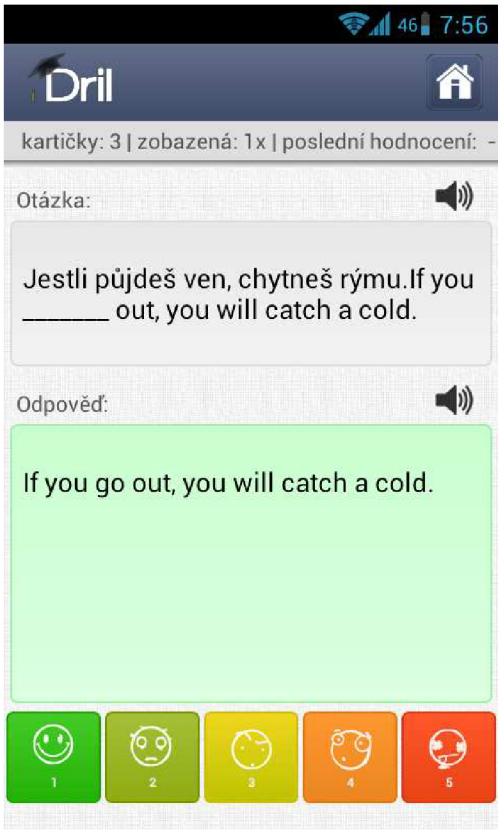
3.1 Design

Design aplikace je velice důležitý někdy dokonce důležitější než to co skutečně umí. Pokud bude aplikace sice obsahovat mnoho funkcí, ale uživatelům se nebude líbit její vzhled a nevyznají se v ní, nemá šanci uspět.

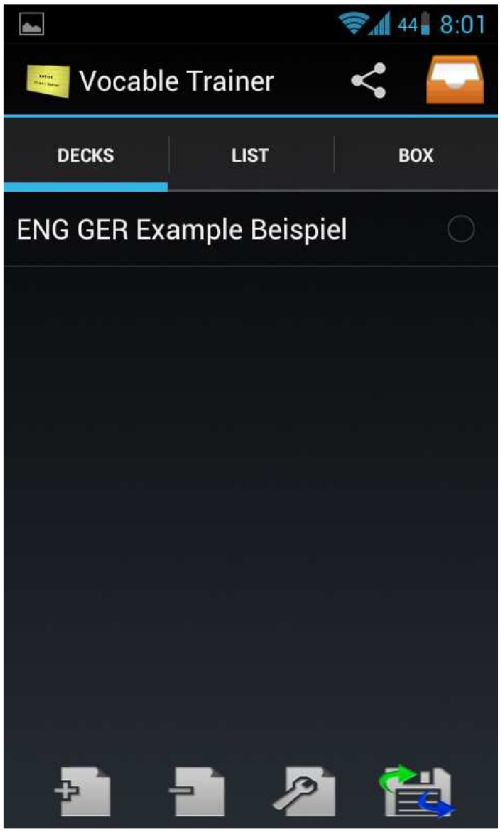
Testovaný TrainBrain a Dril, mají velmi vydařený design. Je jednoduchý a přehledný, příjemně se s ní pracuje. Mohou nám tedy sloužit, jako dobrý vzor toho, jak by taková aplikace. Ukázky z aplikace Drill můžeme vidět na obrázcích 7 a 8.

Naopak jsou tu i aplikace jako VocabularyTrainer, kterou si můžeme prohlédnout na obrázcích 9 a 10 a nebo Vocable Trainer, které dokazují, že špatný design může způsobit katastrofu. Nejsou vůbec přehledné ani intuitivní. Práce s nimi je velice náročná a u některých je téměř nemožné se dostat, až k samotnému učení a procvičování slovní zásoby.

Obrázek 7: Dril - menu



Obrázek 8: Dril - procvičování



Obrázek 9: Vocable Trainer - menu

Obrázek 10: Vocable Trainer - procvičování

3.2 Slovní zásoba

Analyzované aplikace většinou měly možnost stáhnout si lekce slovíček z internetu. Disponují většinou poměrně rozsáhlou databází slovíček. Naším úkolem není pokoušet se vytvářet tak velkou slovní zásobu.

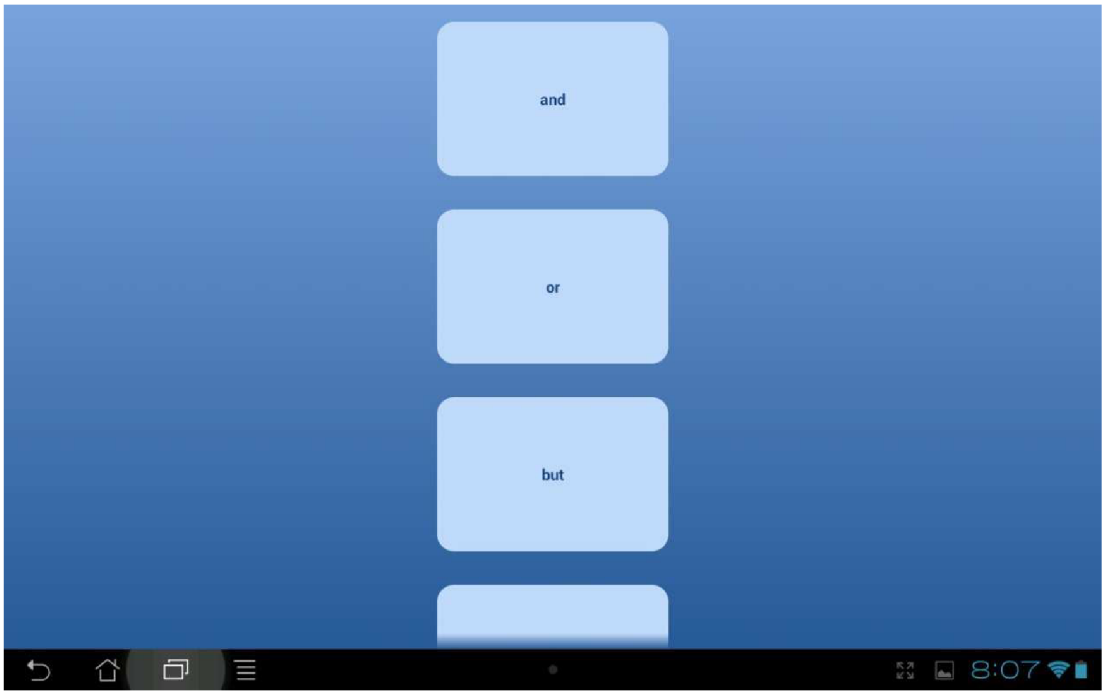
Jako důležitější vidím mít možnost si databázi doplnit vlastními slovíčky podle učebnice, kterou student právě používá.

Naše aplikace by měla obsahovat pouze základní databázi, která bude obsahovat několik stovek slovíček v nejrozšířenějších jazycích. Zbytek bude na uživateli, aby si vytvářel lekce a do nich vkládal slovíčka, která se potřebuje naučit. Navíc tím, že si slovíčka bude zadávat sám se je rychleji naučí.

Obrázek 11: TrainBrain - vytvoření nové lekce a vkládání slovíček

3.3 Optimalizace

Všechny testované aplikace byly primárně určeny pro mobilní telefony. U těch lépe provedených nebyl žádný problém a daly se na mobilních zařízeních používat. Ovšem pokud si je nainstalujeme na zařízení s větší obrazovkou, dostane se nám úplně jiného výsledku. Žádná z aplikací není optimalizována pro tablety. Dochází pouze k roztažení mobilní verze. To mělo za následek spoustu nevyužitého místa někdy dokonce nečitelnost a zhoršenou ovladatelnost. Na obrázcích 12, 13 a 14 můžeme vidět aplikace Vocat a TrainBrain na deseti palcovém tabletu.



Obrázek 12: Vocab - nevyužité místo

Obrázek 13: TrainBrain - malé písmo

Obrázek 14: TrainBrain - roztažení

3.4 Funkčnost

Testované aplikace lze rozdělil na dvě skupiny. Aplikace se základní funkčností, které sloužily výhradně k zlepšování slovní zásoby, ale neposkytovaly žádné rozšířené, pomocné funkce.

Druhá skupina aplikací již poskytovala více funkcí. Například vedení statistiky, více způsobů učení slovíček či off-line slovník. Jejich nedostatkem ale je, že jsou určené jen pro několik konkrétních jazyků Pokud chceme nějaký jiný jazyk potom si musíme stáhnout další aplikaci.

Obrázek 15: TrainBrain - slovník, statistika

3.4.1 Shrnutí analýzy

Výsledky analýzy jsou shrnuty formou tabulky. Aplikace jsem ohodnoceny v jednotlivých kategoriích pomocí značky +, kdy jedno + je nejméně a + + + jsou nejvíce.

Program	Design	Slovní zásoba	Optimalizace	Funkčnost
Slovíčka	+ +	+ + +	+	+ +
TrainBrain	+ + +	+ + +	++	+ + +
Dril	+ + +	+ + +	+	+ + +
Vocab	+	+ + +	+	+
Vocable Trainer	+	+ + +	+	+
Vocablo 2	+ +	+ + +	+	+
VocabularyTrainer	+	+ + +	+	+

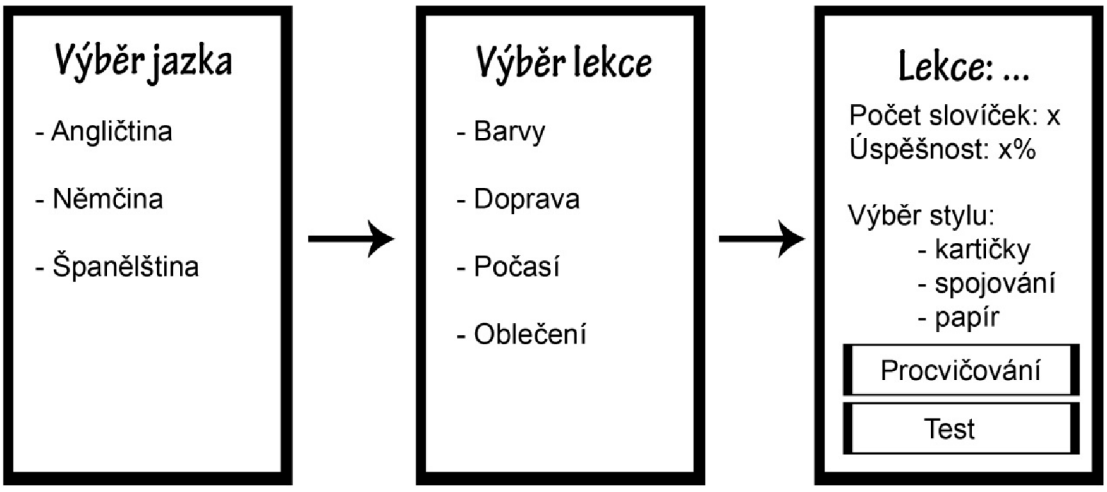
4 Návrh aplikace

V této kapitole probereme návrh aplikace. Nejdříve se podíváme na uživatelské rozhraní a popíšeme si rozdíl mezi uživatelským rozhraním pro mobilní telefon a pro tablet. Další podkapitolu tvoří popis základních grafických komponent, které budou v aplikaci použity. Třetí podkapitolu se věnuje grafickému zpracování a nakonec návrhu databáze.

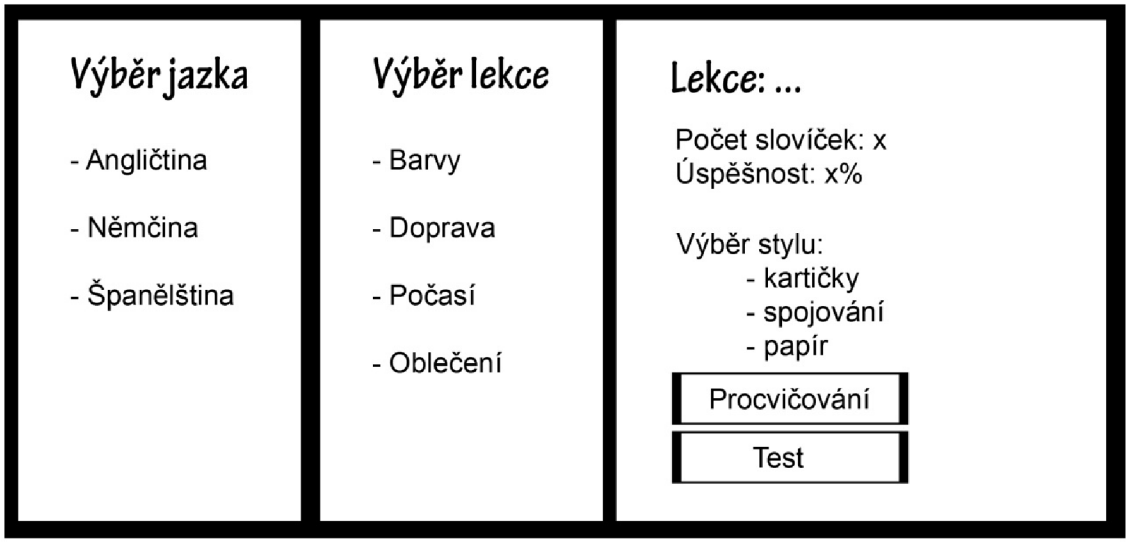
Na základě analýzy existujících řešení jsem si již udělal představu, jak by měla výsledná aplikace vypadat. Mělo by se jednat o aplikaci s jednoduchým, intuitivním designem. Aplikace by měla být optimalizována pro tablety, kterých se prodává čím dál tím více. Uživatelé by zde neměli nalézt jen základní funkce, ale měli by jim být poskytnuty i různé rozšiřující a podpůrné možnosti. Oproti existujícím řešením by mělo být více možností k zadávání nových slovíček a více způsobů k testování a procvičování slovní zásoby.

4.1 Uživatelské rozhraní

U návrhu uživatelského rozhraní bylo potřeba vzít v úvahu, že by aplikace měla být použitelná, jak pro mobilní telefony, tak pro tablety. Proto nejde vytvořit pouze jeden návrh a rozšířit ho i na tablety, nebo naopak. Hlavní ovládací prvky a styl rozhraní bude stejný. Rozdíly by měly být v detailu zobrazení informací. U mobilního telefonu bude muset uživatel projít přes aktivity výběru jazyka, lekce a poté spuštění procvičování, jak je na obrázku 16. Při zobrazení na šířku lze některé informace sloučit na jednu zobrazovací stránku, jako vidíme na obrázku 17. Dále u verze pro tablety lze využít volného místa a zároveň s procvičováním slovní zásoby se uživateli zobrazí i rozšiřující informace. Nebo mu mohou slovíčka být předkládaná ve větším počtu a v jiném seskupení.



Obrázek 16: Schéma layoutu pro mobilní telefon



Obrázek 17: Schéma layout pro tablet na šířku

4.2 Základní komponenty

Android nám poskytuje mnoho grafických komponent. Nyní se podíváme na komponenty, které budeme potřebovat v našem projektu.

4.2.1 Tlačítka

Jednou z nejdůležitějších komponent, které zajištují interakci s uživatelem jsou tlačítka. Tlačítka jsou komponenty, které reagují na klik. Mohou obsahovat text, obrázek nebo obojí. Lze je definovat v XML souboru nebo programově pomocí třídy `Button` a příkazu `new Button()`.

```
<Button
      android:id="@+id/cancel_add_lesson"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="@+string/cancel" />
```

Kód 1: XML definice tlačítka

Třída `Button` má několik potomků. Některé z nich v naší aplikaci využijeme. Třídu `RadioButton` využijeme v případě, že budeme chtít, aby si uživatel vybral jednu z možností.

4.2.2 Text

Pro práci s textem využijeme dvě základní komponenty. Pro pouhé zobrazení textového popisku využijeme třídu `TextView`. Pokud budeme potřebovat, aby nám uživatel zadal nějaký text, použijeme třídu `EditText`. Stejně jako tlačítka, lze tyto komponenty vytvořit v XML souboru nebo programově. Pomocí id můžeme dohledat již existující komponentu a měnit její vlastnosti.

```
<TextView
     android:id="@+id/text"
     android:layout_width="wrap_content"
     android:layout_height="wrap_content"
     android:layout_weight="1"
```

```
        android:textColor="#000"
        android:textSize="22sp"/>
<EditText
      android:id="@+id/editText1"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:layout_weight="1"
      android:ems="10" >
      <requestFocus />
</EditText>
```

Kód 2: XML definice komponent TextView a EditText

4.2.3 Další komponenty

Dále pro zobrazování seznamů využijeme `ListView` a pro výběr ze seznamu například `Spinner`. Pro vhodné rozložení jednotlivých komponent na obrazovce využijeme třídy `LinearLayout`. Jednotlivé komponenty můžeme do nich zabalit těchto layoutů a dosáhnout tak rozložení, jaké si představujeme.

```
<Spinner
      android:id="@+id/spinner_languages"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content" />
```

Kód 3: XML definice komponenty spinner

4.3 Grafické zpracování

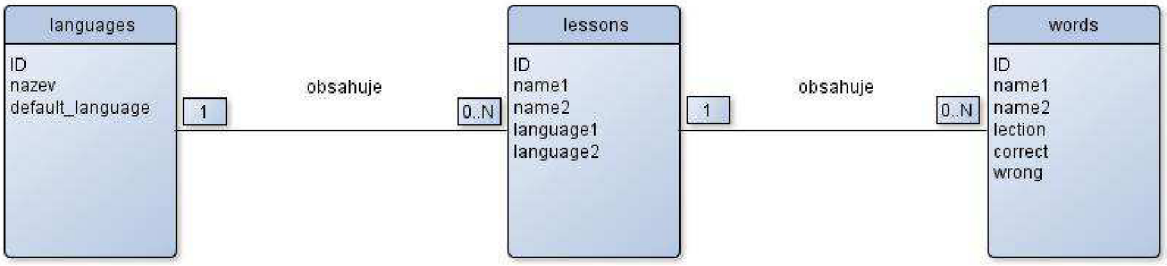
Nedílnou součástí návrhu aplikace je rozhodnutí o barevném stylu a celkovém vzhledu. První je třeba se rozhodnou, zda se zaměřit spíše na teplé nebo studené barvy. Pro naši aplikaci využijeme spíše teplé barvy, které mají působit povzbudivě, aktivně a podněcovat činnost[2].

Konkrétně se zaměříme na hnědožlutou a žlutou. Hnědá barva představuje bezpečí a teplo domova a žlutá barva, představuje lidskou potřebu se rozvíjet, povzbuzuje a osvobozuje člověka[2].

Na základě těchto informací a nápadů dotazovaných lidí bylo rozhodnuto, že celá aplikace bude laděna do tématu korkové nástěnky, na které jsou pomocí připínáčků připevněny žluté papírky.

4.4 Databáze

Slovní zásoba a statistika vztahující se ke slovíčkům se bude ukládat do databáze. Android plně podporuje databázi SQLite, kterou také využijeme v naší aplikaci. Na obrázku 18 lze vidět ER diagram navrhnuté databáze.



Obrázek 18: ER diagram databáze

Databáze obsahuje celkem tři tabulky. První se nazývá `languages`. Obsahuje pouze dva atributy - název a označení zda se jedná o rodný jazyk uživatele.

Druhá tabulka `lessons` se skládá z názvu lekce, jejího překlad a dvou cizích klíčů, označujících do kterých jazyků lekce patří.

Třetí tabulka `words` obsahuje jednak samotná slovíčka a jejich překlad. Navíc obsahuje údaje o tom kolikrát bylo zodpovězeno správně a špatně. Z těchto údajů za pomocí dotazů nad databází lze získat informace o úspěšnosti uživatele. Jak pro dané slovo, tak pro lekci či celý jazyk. Také obsahuje jeden cizí klíč, který označuje do které lekce slovíčko patří.

4.5 Výuka

Cílem je učení slovíček uživateli zjednodušit a zpříjemnit. Měl by se tedy jen pomocí pár kliknutí se dostat k samotné výuce a nezdržovat se zdlouhavým nastavováním. Jak jsme si již řekli v teorii o křivce zapomínání, Je důležité vybrat vhodné metody opakování slovní zásoby, abychom naučená slovíčka nezapomínali.

4.5.1 Metody opakování slovíček

Existuje spousta metod učení se slovíček. Každý, kdo se někdy učil nějaký cizí jazyk, ví, že bez slovní zásoby to nejde. Každý preferuje jinou metodu učení. V naší aplikaci jsme pro učení a opakování slovní zásoby zvolili metody, které nejvíce upřednostňovali dotazovaní lidé.

Uživatel si bude moci zobrazit seznam slovíček jako ve slovníku. Zde si může procvičit slovíčka, než se pustí do nějaké z testovacích metod. Bylo by bezúčelné se pustit okamžitě do testování, když by vůbec ta slovíčka neznal. Pro první naučení je ideální slovníkový seznam.

Prvním ze způsobů testování bude ruční psaní odpovědí. Tato metoda je sice časově náročnější a psaní na dotykovém displeji nemusí vyhovovat všem, ale za to se člověk naučí, jak se slovo píše. Většina respondentů odpovídala, že tuto metodu využívá, pokud má dostatek času.

Další dva způsoby jsou časově mnohem méně náročné, protože uživatel pouze kliká a nemusí nic psát. Jedním z těchto způsobů je, že si student řekne překlad slovíčka sám pro sebe a po kliknutí se mu zobrazí správná odpověď. Poté se sám ohodnotí, zda to věděl nebo ne. Poslední metodou je, že dostane na výběr ze tří možností a musí vybrat, která z nabízených možností je správně.

5 Implementace

V této kapitole je popsána implementace aplikace. Nejdříve se zaměříme na nastavení aplikace. Poté na strukturu aplikace a implementaci hlavních aktivit, fragmentů a jejich vzájemnou komunikaci. Nakonec si popíšeme implementaci databáze a způsob optimalizace pro různá zařízení.

Celá aplikace je implementována v jazyce Java a frameworku Android SDK za využití vývojového prostředí Eclipse.

5.1 Nastavení aplikace

Základní informace a nastavení aplikace se nachází v souboru `AndroidManifest.xml`[10]. Jako minimální verze systému je nastavený Android 3.0.x tedy SDK verze 11 z důvodu podpory využívání fragmentů od této verze. Cílovým systémem je Android 4.0.3 tedy SDK verze 15.

```
<uses-sdk
        android:minSdkVersion="11"
        android:targetSdkVersion="15" />
```

Kód 4: Ukázka souboru AndroidManifest.xml

V souboru jsou také definovány aktivity používané v projektu. Atribut `name` odkazuje na třídu, která danou aktivitu implementuje. K určení popisku slouží atribut `label`.

```
<activity
      android:name=".learn.LanguagesActivity"
      android:label="@string/title_activity_languages" >
      <intent-filter>
            <action android:name="android.intent.action.MAIN" />
            <category android:name="android.intent.category.LAUNCHER"
/>
      </intent-filter>
</aktivity>
```

Kód 5: Ukázka definice aktivity v manifestu projektu

V tomto souboru se také uvádí, jaká oprávnění aplikace vyžaduje. Jelikož náš projekt nepotřebuje žádná oprávnění, tak zde nebylo potřeba nic psát.

5.2 Struktura projektu

Celá práce je rozdělena do několika balíků tříd. Prvním balíkem je `project`. Obsahuje hlavní třídu `LanguagesActivity.java` spouštěnou po startu aplikace a veřejnou třídu `Globals.java`, která slouží pro uložení a manipulaci s informacemi o aktuálně vybraném jazyku a lekci v celé aplikaci.

Balík `project.learn` obsahuje všechny třídy implementující aktivity a fragmenty celé aplikace. Obsahuje například třídy `LearnActivity.java`, `PracticeActicity.java`, `LessonsFragment.java`, `DetailFragment.java` a další.

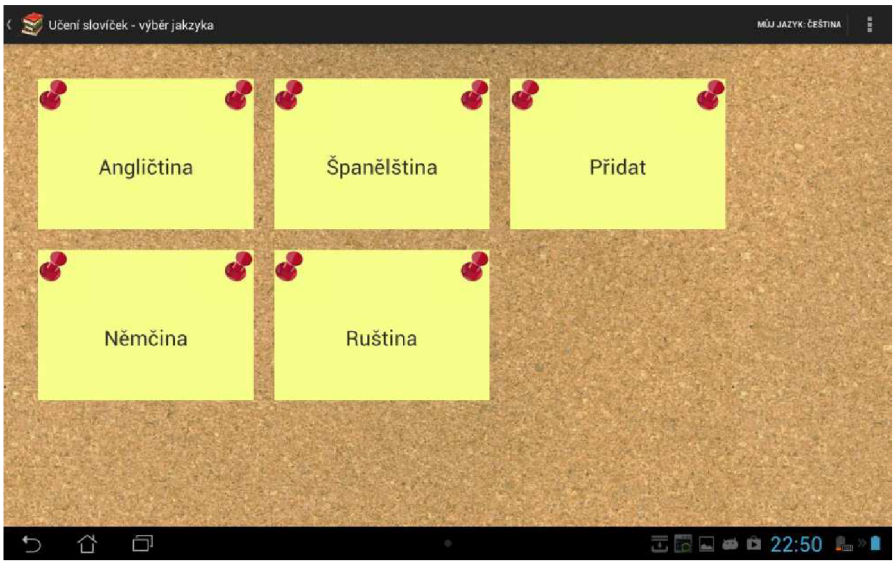
Balík `project.databse` obsahuje třídu `MySQLiteHelper.java`, která implementuje práci s SQLite databází.

5.3 Aktivity a fragmenty

Jak již bylo řečeno výše, všechny aktivity a fragmenty jsou v balíku `project.learn`. Vzhled většiny aktivit a fragmentů je definován pomocí XML souborů a programově se mění pouze obsah jednotlivých prvků. Pouze u aktivit, které obsahují proměnný počet prvků je vzhled definován programově. Celý vzhled aplikace je udělán do podoby korkové nástěnky, na které jsou připínáčky připnuté kartičky.

5.3.1 Úvodní aktivita

Jako první aktivita po spuštění aplikace je v souboru `AndroidManifest` nadefinována třída `LanguagesActivity`. Zde může uživatel v navigační liště jít do nastavení aplikace a vybrat si svůj jazyk nebo naplnit či vyprázdnit databázi. Na úvodní aktivitě se zobrazí seznam jazyků ve formě kartiček připnutých na nástěnce. Uživatel má také možnost nějaký jazyk odebrat nebo přidat nový. Při výběru možnosti odebrat jazyk se změní barva navigační lišty a klikáním na jazyky může odebírat, dokud nestiskne tlačítko hotovo v horní liště. Po výběru jazyka, který chceme procvičovat se uloží tato informace a spustí se aktivita `LearnActivity`. Na obrázku 19 můžeme vidět ukázku této aktivity.



Obrázek 19: Úvodní aktivita

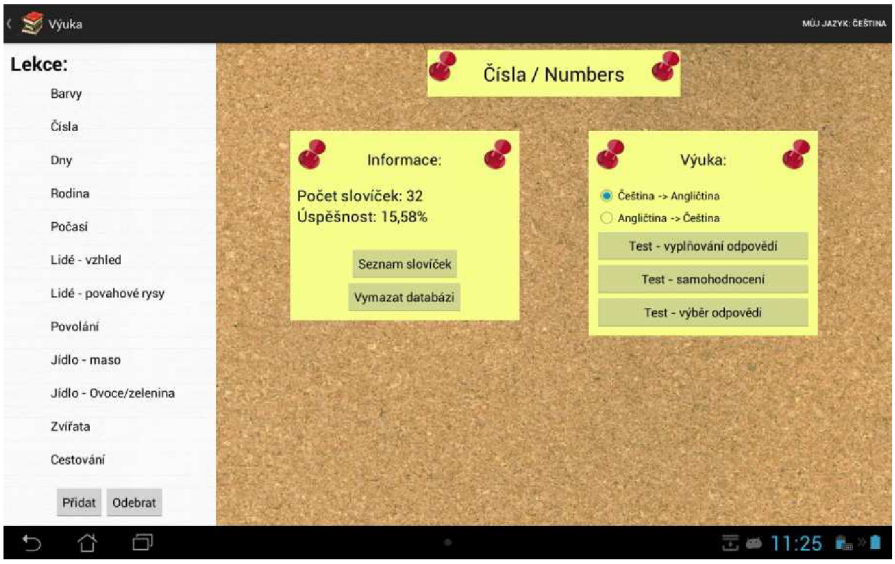
5.3.2 Seznam lekcí

Třída `LearnActivity` se skládá ze dvou fragmentů definovanými třídami `LessonsFragment` a `DetailFragment`. Třída `LessonsFragment` obsahuje seznam lekcí daného jazyka implementovaného pomocí `ListView`. Dále obsahuje dvě tlačítka na přidávání a odebírání lekcí. Při odebírání lekcí lze lekce mazat do doby, dokud se v horní liště neoznačí možnost hotovo.

V třídě `DetailFragment` se zobrazují bližší informace o vybraném jazyku nebo lekci a o akcích, které lze provádět. Při zobrazování informací o jazyku lze vyčíst celkový počet lekcí a

slovíček ve vybraném jazyku. Při výběru lekce lze zobrazit seznam slovíček, vymazat statistiky úspěšnosti dané lekce nebo pustit jeden ze tří druhů testování. Testování lze zpustit jen v případě, že vybraná lekce obsahuje dostatek slovíček. U metody vybírání správné odpovědi jsou zapotřebí alespoň tři slovíčka. U ostatních stačí jen jedno. Ukázka na obrázku 20.

V případě zobrazení na menších obrazovkách je zobrazení seznamu lekcí a informací zobrazeno pomocí dvou aktivit. První aktivita obsahuje fragment `LessonsFragment` a zobrazuje seznam lekcí. Druhá aktivita obsahuje `DetailFragment` zobrazující informace k vybrané lekci. Na obrázcích 21, 22 a 23 lze vidět zobrazení na zařízeních s menším obrazovkou.



Obrázek 23: Seznam lekcí a informace o lekci

Obrázek 20: Mobilní zobrazení seznamu lekcí

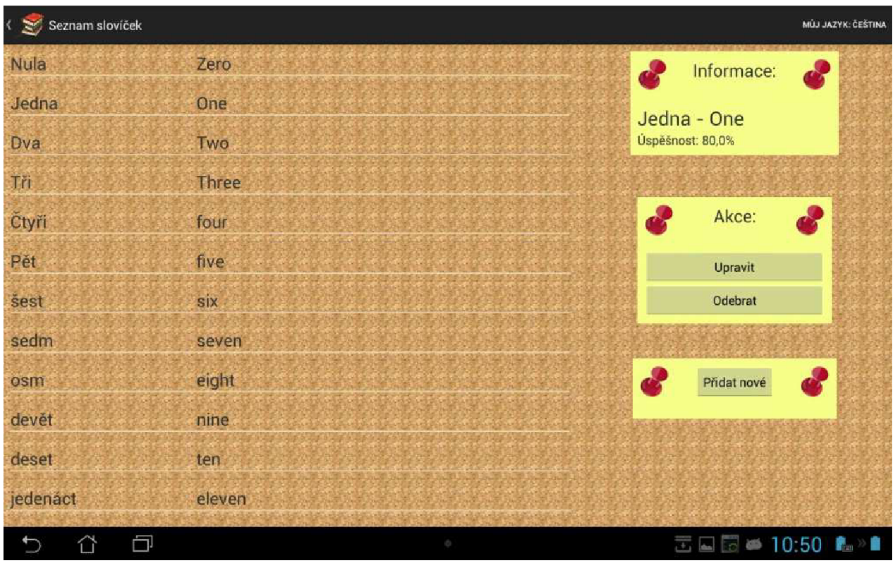
Obrázek 21: Mobilní zobrazení informací o lekcích

Obrázek 22: Mobilní zobrazení akcí se slovíčky

5.3.3 Seznam slovíček

Aktivita `PracticeActivity` se také skládá ze dvou fragmentů. Jeden z nich je definován třídou `PracticeMainFragment`. Tento fragment představuje seznam o dvou sloupcích, ve kterých jsou slovíčka dané lekce a dostupné překlady. Po kliknutí na nějaké slovíčko se o něm ve třídě `PracticeDetailFragment` zobrazí informace. Uživatel má také možnost slovíčko upravit, smazat anebo přidat úplně nové. V případě, že lekce neobsahuje žádné slovíčko, se zobrazí pouze možnost pro jeho přidání.

Pro zobrazení na zařízeních s menšími obrazovkami se opět využije dvou aktivit, kde v jedné se nachází fragment `PracticeMainFragment` a v druhé `PracticeDetailFragment`.

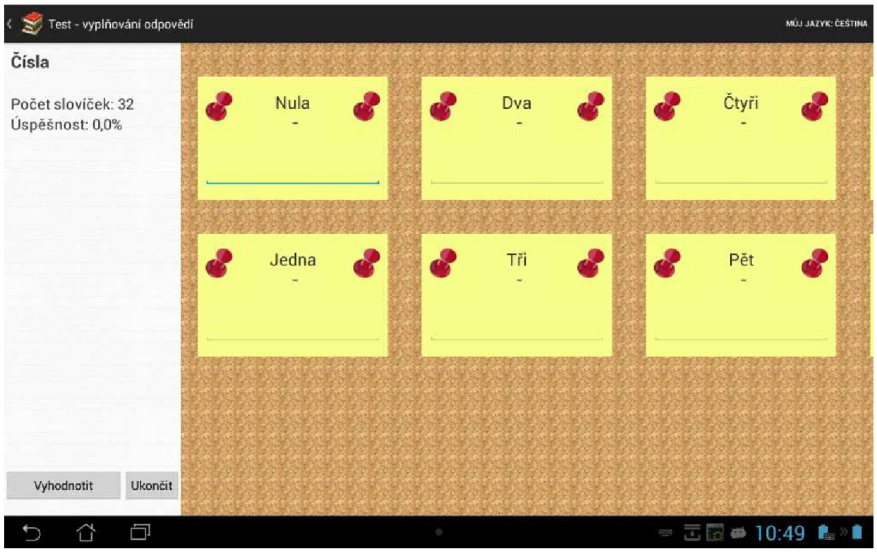


Obrázek 24: Seznam slovíček

5.3.4 Test - vyplňování odpovědí

Prvním ze způsobů testování je, že uživatel vidí slovíčko a musí napsat jeho překlad. Třída `TestActivity` obsahuje dva fragmenty. Třída `TestDetailFragment` implementující první fragment obsahuje informace o zvolené lekci a tlačítka pro vyhodnocení a ukončení testu. Třída `TestMainFragment` implementuje druhý fragment, který obsahuje kartičky se slovy. Vzhled tohoto fragmentu není definován pomocí XML souboru. Z důvodu různého počtu slovíček v lekcích je vzhled fragmentu definován programově.

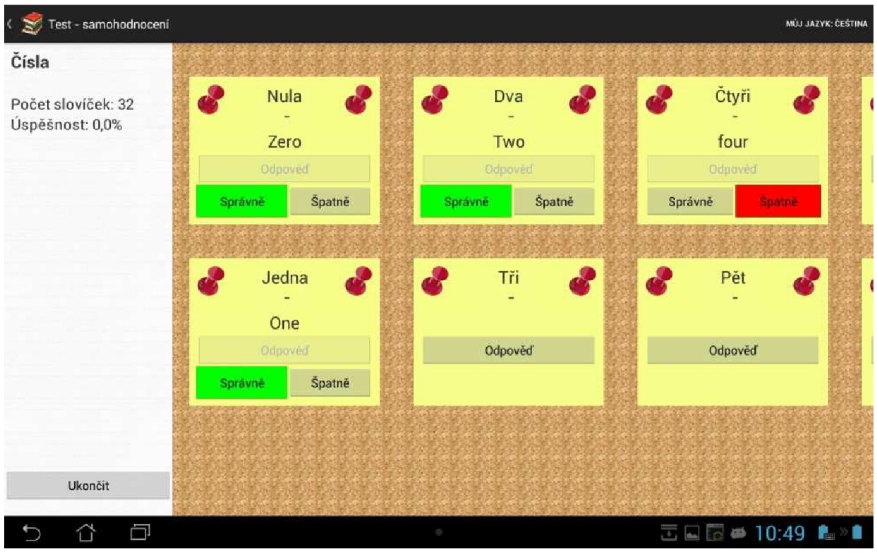
V případě menších obrazovek dochází u zobrazování testů k rozložení obrazovky na dvě.



Obrázek 25: Test - vyplňování odpovědí

5.3.5 Test - samo hodnocení

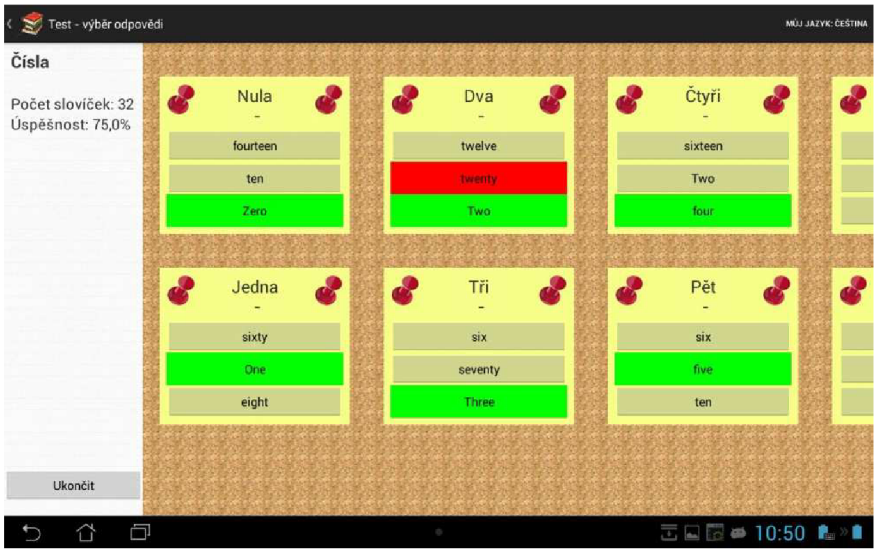
Druhým způsobem testování slovní zásoby je, kdy uživatel opět vidí napsané slovíčko, ale tentokrát nikam nepíše jeho překlad. Po kliknutí se zobrazí správný překlad slovíčka a uživatel se sám ohodnotí, zda to věděl nebo ne. Tento způsob má výhodu v tom, že je rychlejší a ne pokaždé je psaní na android zařízeních bez hardwarové klávesnice příjemné. Opět se jedná o aktivitu, která obsahuje dva fragmenty. Tyto fragmenty jsou implementovány třídami `Test1DetailFragment` a `Test2DetailFragment`.



Obrázek 26: Test - samo ohodnocování

5.3.6 Test - výběr odpovědi

Tento způsob testování se opět skládá ze dvou fragmentů a je reprezentován třídou `Test2Activity`. Informace o testované lekci a možnost ukončit test. Druhý zobrazuje testovaná slovíčka. Uživatel má na výběr ze tří odpovědí. Pokud vybere špatnou možnost, tak se graficky označí, která odpověď byla správná.



Obrázek 27: Test - výběr odpovědi

5.4 Komunikace

Aplikace by měla komunikovat s uživatelem a informovat ho o provedení akce, vyžádat si potvrzení nebo chtít zadat nějaká data. Dále mezi sebou komunikují a předávají si potřebné informace jednotlivé aktivity a fragmenty.

5.4.1 Veřejná třída

V aplikaci je potřeba uchovávat informaci o tom, který jazyk a lekce je aktuálně vybrán. Jelikož je tato informace potřeba téměř ve všech třídách, tak jsem se rozhodl vytvořit veřejnou třídu `Globals`, která je přístupná ze všech míst programu. K informacím lze přistoupit jednoduchými příkazy `Global.selectedLanguage` a `Globals.selectedLesson`.

```
public class Globals {
      public static String selectedLanguage = "none";
      public static String selectedLesson = "none";
      …
}
```

Kód 6: Ukázka souboru Globals.java

5.4.2 Rozhraní

Rozhraní (`Interface`) je využito pro komunikaci fragmentu s aktivitou, ve které se nachází. Ve speciálním souboru `Interface_learn` je definováno rozhraní a seznam funkcí, které musí být v aktivitě implementovány. Ve fragmentu je potřeba na proměnnou typu `Interface_learn` připojit aktivity pomocí funkce `onAttach(Activity aktivity)`.

```
public interface Interface_learn {
    public Menu getMenu();
    public boolean getDeleteMode();
    public void changeDeleteMode(boolean bool);
}
```

Kód 7: Třída Inrerface_lear

```
@Override
public void onAttach(Activity activity) {
    super.onAttach(activity);
        try {
            interface_learn = (Interface_learn) getActivity();
        } catch (ClassCastException e) {
        }
    }
```

Kód 8: Ukázka funkce onAttach()

5.4.3 Informační dialogy

V případě, že chce aplikace pouze oznámit uživateli, že akce úspěšně proběhla či neproběhla a není potřeba žádná zpětná vazba uživatele, tak použijeme pouze informační dialogy. Jedná se o zprávy, které se zobrazí ve spodní části obrazovky na určitou dobu. Jde o objekt `Toast[12]`, který je nejjednodušším dialogem. Ukázka tohoto dialogu je na obrázku 28.

```
Context context = this.getApplicationContext();
CharSequence text = getString(R.string.select_language);
int duration = Toast.LENGTH_SHORT;
Toast toast = Toast.makeText(context, text, duration);
toast.show();
```

Kód 9: Ukázka vytvoření informačního dialogu

5.4.4 Výstražné dialogy

Třídu `AlertDialog[11]` používáme v případě, že potřebuje od uživatele reakci na zprávu. V našem případě tyto zprávy využijeme při mazání dat z databáze, kde vyžadujeme od uživatele potvrzení jeho volby. Příklad lze vidět na obrázku 29.

```
new AlertDialog.Builder(v.getContext())
      .setMessage(R.string.delete_sure)
      .setCancelable(false)
      .setPositiveButton(R.string.yes, OnClickHandle())
      .setNegativeButton(R.string.no, null)
      .show();
```

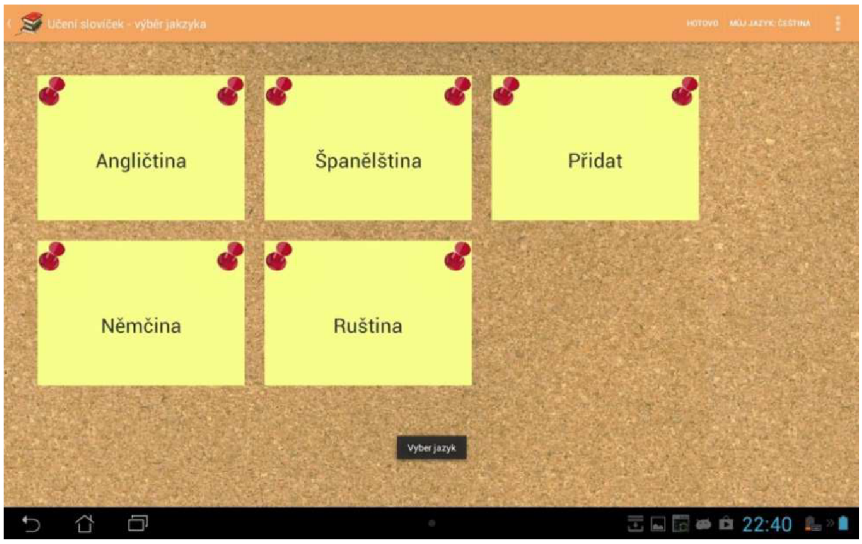
Kód 10: Ukázka vytvoření výstražného dialogu

5.4.5 Vlastní dialogy

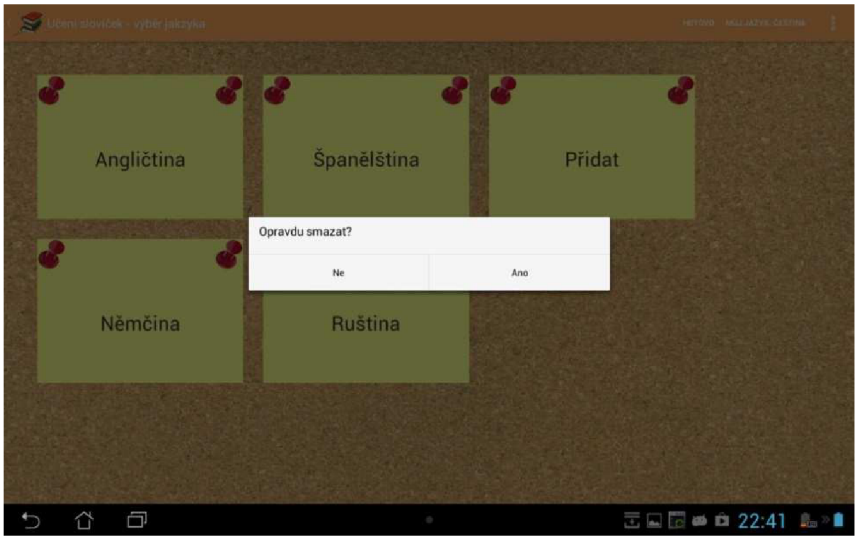
Třetí komunikace s uživatelem v naši aplikaci jsou dialogy vytvořené z objektu `Dialog[11]`. Jedná se o dialog, kterému pomocí XML souboru sami nadefinujeme jeho vzhled a obsah pomocí metody `setContentView()`. Tyto dialogy používáme při přidávání jazyka, lekce, slovíček a také při změně nastavení. Jak můžeme vidět na obrázku 30, vypadá jako malá korková nástěnka.

```
Dialog dialog = new Dialog(v.getContext());
dialog.requestWindowFeature(Window.FEATURE_NO_TITLE);
dialog.setContentView(R.layout.add_lesson);
dialog.show();
```

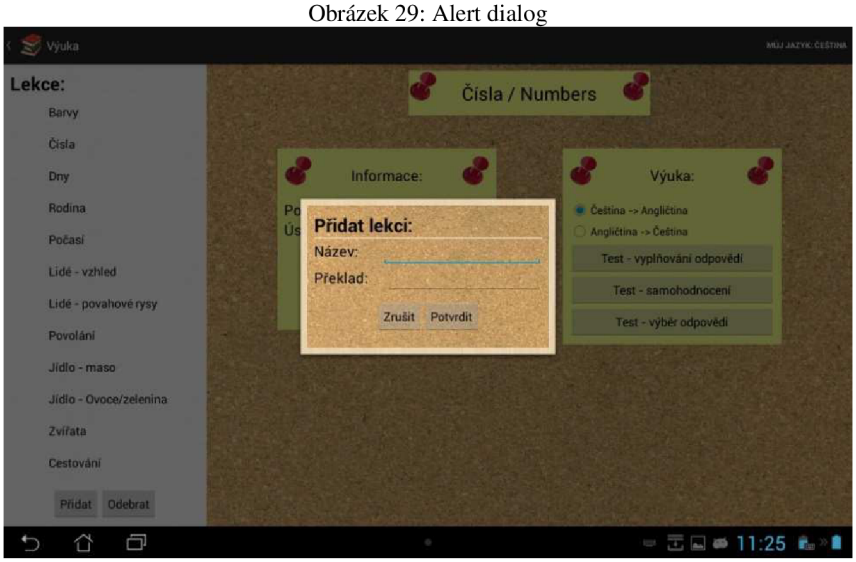
Kód 11: Ukázka vytvoření vlastního dialogu



Obrázek 28: Toast



Obrázek 29: Alert dialog



Obrázek 30: Vlastní dialog

5.5 Databáze

Jak již bylo zmíněno v kapitole 4, v aplikaci je pro ukládání dat využívána SQLite databáze. Pro vytvoření a upravování slouží třída `MySQLiteHelper.java`, která rozšiřuje třídu `SQLiteOpenHelper`. Podle ER diagramu navrženého v kapitole 4, databáze obsahuje tabulky `languages`, `lessons` a `words`.

5.5.1 Metody

V třídě `MySQLiteHelper.java` jsme implementovali metody, které jsou potřeba pro práci s databází. Pro samotné vytvoření tabulek databáze slouží metoda `OnCreate(SQLiteDatabse db)`. K jejich naplnění základním balíčkem slovíček se používáme metodu `loadDatabse(SQLiteDatabse db)`. Naopak pro vyčištění tabulek lze využít `clearDatabse(SQLiteDatabase)`.

Pro získání seznamu jazyků slouží metoda `getListOfLanguages(SQLiteDatabse db)`. Jejím parametrem je otevřená databáze a návratovou hodnotou je seznam názvů všech jazyků v databázi. K získání názvů lekcí daného jazyka složí podobná metoda, která se jmenuje `getListOfLessons(SQLiteDatabse db, String language)`. Ta navíc vyžaduje druhý parametr představující název vybraného jazyka. Posledními dvěmi metodami vracející seznam názvů jsou `getListOfWords(SQLiteDatabase db, String lesson, String language)` a `getListOfTransaltions(SQLiteDatabase db, String lesson, String language)` sloužící k získání seznamu slovíček dané lekce a jejich překladů. Tyto metody ještě navíc potřebují vědět, jaká lekce je vybraná.

Metody `getLanguageID(SQLiteDatabase db, String language)`, `getLessonID(SQLiteDatase db, String lesson, String language)` a `getWordID(SQLiteDatabse db, String word, String lesson, String language)` slouží k získání id daného jazyka, lekce či slova. Tyto metody jsou důležité pro hledání konkrétních záznamů v databázi, nebo jejich upravování.

Poslední skupinou metod jsou metody určené ke spravování statistik úspěšnosti. Pro získání počtu kolikrát bylo dané slovíčko správně či chybně zodpovězeno jsou metody `getCorrectLesson(SQLiteDatabse db, String lesson, String language)` a `getWrongLesoon(SQLiteDatabase db, String lesson, String language)`. Pro smazání všech statistik je určena metoda `deleteStatistics(SQLiteDatabase db)`. Pokud uživatel bude chtít vymazat údaje o úspěšnosti pouze u jedné konkrétní lekce, se využije metoda `clearLessonStatistics(SQLiteDatabase db, String lesson, String language)`.

5.5.2 Využití aplikace v aplikaci

Vlastní využití databáze v aplikaci se skládá z následujících tří postupných kroků:

1. Získání přístupu k databázi pomocí metody `getReadableDatabse()` nebo `getWriteableDatabase()`

2. Vlastní používání metod definovaných v předchozí podkapitole.
3. Uzavření databáze metodou `close()`.

5.6 Optimalizace

5.6.1 Řetězce

Z důvodu lepší optimalizace aplikací pro více jazyků prostředí aplikace se všechny textové řetězce v projektu píší do souboru `res/values/strings.xml`. V případě odkazování se na řetězec ze souboru v jazyce Java použijeme odkaz `R.string.string_name` a v případě odkazování se ze souboru typu XML použijeme `@string/string_name`. Syntaxe souboru strings.xml se skládá z kořenového uzlu <resources> a jednotlivých řetězců zapsaných ve značkách <string>

Naše aplikace podporuje dva jazyky. Těmi jsou čeština a angličtina. V souboru `res/values/strings.xml` jsou uvedeny řetězce v anglickém jazyce a v souboru `res/values-en/strings.xml` jsou uvedeny tytéž řetězce v českém jazyce. V případě spuštění aplikace na zařízení s jinou než českou lokalizací se tedy použijí automaticky anglické řetězce.

```
<resources>
      <string name="app_name">Učení slovíček</string>
      <string name="menu_settings">Nastavení</string>
      …
</resources>
```

Kód 12: definování řetězců

5.6.2 Zobrazení

Jak již bylo zmíněno v kapitole 1 Android aplikace může být přizpůsobená různým velikostem obrazovek, za pomocí XML souborů uložených ve speciálních složkách. To také bylo jedním z cílů, které jsme si zadali.

Naše práce je přizpůsobena zobrazení, jak mobilnímu telefonu, tak na tabletu o velikosti displeje deset palců. Zdrojové XML soubory pro zařízení s menšími obrazovkami jsou uloženy ve složce `res/layout/`. Soubory pro zobrazení na deseti palcové obrazovce a větší jsou uloženy v souboru `res/layout-sw720dp/`.

5.7 Události

Pomocí událostí může reagovat akce uživatele, jakou jsou kliknutí, přejetí, dlouhé stisknutí a další. V našem projektu často požíváme událost `setOnClockListener()`, která reaguje na kliknutí na komponentu.

V našem projektu například nastavujeme `listener` na kliknutí na `TextView`. Nejdříve si však vytvoříme obsluhu události, tak jak je ukázáno níže.

Při vytváření našeho TextView mu musíme náš `listener` přiřadit. Příkazem `textView.setOnclickListener(handleOnClick_addLanguage(dialog));`

```
View.OnClickListener handleOnClick_addLanguage(final Dialog dialog){
      return new View.OnClickListener() {
            public void onClick(View v) {
                  // co se má provéset
                  …
            }
      };
}
```

Kód 13: Ukázka obsluhy události

6 Testování

V této kapitole bude popsáno, jak byla naše aplikace testována. Nejdříve si řekneme, na jakých zařízeních byla aplikace během vývoje zkoušena. Pak se také podíváme na naši aplikaci očima nezávislých lidí, kterým jsme aplikaci předvedli.

Při vývoji byla aplikace testována na dvou zařízeních. Na mobilním telefonu Samsung Galaxy S II s operačních systémem android 4.0.4 a na tabletu Asus Transformer Pad300 s Androidem ve verzi 4.2.1 o uhlopříčce 10,1 palců. Aplikace byla optimalizována na obě zařízení a všechny objevené problémy byly také vyřešeny.

Dále proběhla dvě předvedení aplikace zkušebním uživatelům k získávání zpětné vazby. První testování proběhlo již ve fázi konceptu po návrhu aplikace na papír. Bylo sezváno několik lidí, kterým se návrh představil. Zajímal jsem se o jejich reakce a názor na návrh. Co se jim líbí a naopak nelíbí, případně jestli by udělali něco úplně jinak. Zeptali jsme se jich, co si o tom myslí a zda by je taková aplikace zajímala. Výsledkem této konzultace bylo, že je v aplikaci použito více způsobů procvičování a testování slovíček. Další připomínka, která byla zapracována bylo zpřehlednění a zjednodušení jednotlivých obrazovek. Místo jedné složité obrazovky s více informacemi jsem volil raději více jednotlivých obrazovek. Mohlo by se totiž stát, že na jedné obrazovce by bylo až moc ovládacích prvků, což může způsobit horší orientaci v aplikaci.

Po zapracování připomínek z konceptu a vytvoření první verze programu proběhlo její testování stejnými uživateli. Mezi nejvýznamnější připomínky, které byly do programu zapracovány:

- Upravení vzhled některých detailů.
- Pohyb a orientace v aplikaci nebyla vždy zcela intuitivní.
- Několik nefungujících odkazů.
- Problém při načítání databáze na některých zařízeních.

7 Závěr

Cílem práce bylo vytvořit aplikaci pro platformu Android, která by sloužila k učení opakování a slovní zásoby. Vzhledem k rozšiřování se počtů tabletu měla být aplikace přizpůsobená pro mobilní telefony i tablety. Myslím, že tento cíl byl naplněn.

Oproti návrhu bylo provedeno několik změn. Z časových důvodů bylo upuštěno od možnosti zadávání slovíček více způsoby. Slovíčka nelze importovat z jiných zdrojů, lze je zadávat jen jednotlivě. Další změna, která vyplynula z testování je již zmíněné použití více jednotlivých ovládacích obrazovek místo jedné složité pro volbu aktivit. Jedna pouze s výběrem jazyka a druhá se seznamem lekcí a dalších informací.

Práce popisuje celý vývoj a jeho nejdůležitější aspekty. Na začátku bylo potřeba si nastudovat teorii k našemu problému. Seznámit se vývojem aplikací na platformě Android a platformou Android samotnou. A také si nastudovat, jak vytvářet aplikace optimalizované pro více různých zařízení. Po nastudování teorie jsme prozkoumali již existující řešení našeho problému. Zde jsme se zaměřili na zjišťování jejich výhod a nevýhod. Po prozkoumání existujících řešení jsme konečně mohli také navrhnout vlastní řešení a probrat je s potenciálními zájemci o naši aplikaci. Ti nám poskytli zpětnou vazbu a představu o tom, co by si přáli. V kapitole 5 je popsána implementace nejdůležitějších prvků projektu. Dále se v této práci dočteme o tom, jak byla aplikace testována.

Práci shledávám pro sebe jako přínosnou a to zejména z toho pohledu, že nenavazuje na žádný jiný, a já jsem měl tedy možnost si vyzkoušet vývoj zcela nové aplikace od začátku až do konce. Za obzvlášť prospěšné považuji také získávání zkušeností s odlaďováním programu na zařízeních s různě velkými displeji.

Budoucí vývoj práce by se mohl zaměřit na rozšíření o další funkce. Se zaměřením na větší využití statistik učení a především umožnit uživateli pohodlněji rozšiřovat databázi slovní zásoby více způsoby.

Literatura

[1] Hermann Ebbingaus: Memory: A Contribution to Experimental Psychology, [online]. Přeložené 1913 [cit. 2013-05-10].
Dostupné na URL: <http://psychclassics.yorku.ca/Ebbinghaus/index.htm>

[2] Jitka Vysekalová, Růžena Komárková:Psychologie barev, Grada, Praha, 2002. ISBN 80-247-0402-1

[3] WIKIPEDIA, Android (operating systém) [online]. Poslední aktualizace květen 2013 [cit. 2012-12-20].
Dostupné na URL: <http://en.wikipedia.org/wiki/Android_(operating_system)>

[4] iDNES.cz: Android, iOS, Windows Phone: dva špičkový hráči a jeden outsider, [online]. 2012-05-12 [cit. 2012-12-20].
Dostupné na URL: <http://mobil.idnes.cz/statistiky-napric-mobilnimi-platformami-dva-spickovi-hraci-a-jeden-outsider-1sa-/mob_tech.aspx?c=A120506_223625_mob_tech_ham>

[5] Google Developers: Developer Tools, [online]. [cit. 2012-12-21].
Dostupné na URL: <http://developer.android.com/tools/index.html>

[6] Google Developers, Components, [online]. [cit. 2012-12-21].
Dostupné na URL: <http://developers.android.com/guide/components/fundament>

[7] Google Developers: Activity, [online]. [cit. 2012-12-21].
Dostupné na URL:
<http://developer.android.com/reference/android/app/Activity.html>

[8] Google Developers: Data storage, [online]. [cit. 2013-02-15].
Dostupné na URL: <http://developer.android.com/training/basics/data-storage/index.html>

[9] Google Developers: Fragments, [online]. [cit. 2012-12-21].
Dostupné na URL:
<http://developer.android.com/guide/components/fragments.html>

[10] Google Developers: The AnfroidManifest.xml File, [online]. [cit. 2013-04-20].
Dostupné na URL: <http://developer.android.com/guide/topics/manifest/manifest-intro.html>

[11] Google Developers: Dialogs, [online]. [cit. 2013-04-21].
Dostupné na URL: <http://developer.android.com/guide/topics/ui/dialogs.html>

[12] Google Developers: Toasts, [online]. [cit. 2013-04-21].
Dostupné na URL:
<http://developer.android.com/guide/topics/ui/notifiers/toasts.html>

[13] Google Developers: App Framework, [online]. [cit. 2012-12-20].
Dostupné na URL: <http://developer.android.com/about/versions/index.html>

[14] Google Developers: Supporting Diferent Devices, [online]. [cit. 2013-04-21].
Dostupné na URL: <http://developer.android.com/training/basics/supporting-devices/index.html>

[15] SQLite: SQLite Documentation, [online]. [cit. 2012-12-22].
Dostupné na URL: < https://play.google.com/store >

[16] Libor Svejda: Slovíčka (zdarma), [online]. Aktualizováno 2013-02-11 [cit. 2012-12-20].
Dostupné na URL: < https://play.google.com/store >
[17] Lukas Adamek: TrainBrain, [online]. Aktualizováno 2013-04-16 [cit. 2012-12-20].
Dostupné na URL: < https://play.google.com/store >
[18] Petr Jurkovič: Dril - angličtina efektivně, [online]. Aktualizováno 2012-11-16 [cit. 2012-12-20].
Dostupné na URL: < https://play.google.com/store >
[19] Unitybits: Vocab, [online]. Aktualizováno 2011-05-05[cit. 2012-12-20].
Dostupné na URL: <https://play.google.com/store >
[20] Luxdroid: Vocable Trainer, [online]. Aktualizováno 2013-04-08[cit. 2012-12-20].
Dostupné na URL: <https://play.google.com/store >
[21] QuidX: Vocablo 2, [online]. Aktualizováno 2013-04-27 [cit. 2012-12-20].
Dostupné na URL: <https://play.google.com/store >
[22] Trainer: VocabularyTrainer, [online]. Aktualizováno 2011-12-26 [cit. 2012-12-20].
Dostupné na URL: <https://play.google.com/store >

Seznam příloh

- Příloha č. 1 - Obsah CD

Příloha č. 1

Obsah CD

Na přiloženém CD se nacházejí následující složky:

- `bp_zprava/` - Tato písemná zpráva ve formátu DOC a PDF.
- `bp_aplikace/` - Zdrojové kódy vytvořeného aplikace.
- `bp_instalace/` - Instalační soubor vytvořené aplikace.
- `bp_plakat/` - Složka obsahující plakát.
- `bp_dokumentace/` - Složka obsahující dokumentaci k aplikaci.