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 MULTIMEDIA

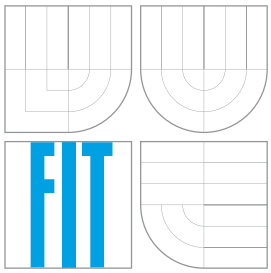
PŘEDNÁŠKY NA CESTY: ANDROID APP PRO OFFLINE PROHLÍŽENÍ PŘEDNÁŠEK NA FIT

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE JIŘÍ KOUDELKA
AUTHOR

BRNO 2015

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 MULTIMEDIA

PŘEDNÁŠKY NA CESTY: ANDROID APP PRO OFFLINE PROHLÍŽENÍ PŘEDNÁŠEK NA FIT

POCKET LECTURES: ANDROID APP FOR OFFLINE VIEWING OF FIT LECTURES

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — JIŘÍ KOUDELKA
AUTHOR

VEDOUCÍ PRÁCE — Ing. IGOR SZŐKE, Ph.D.
SUPERVISOR

BRNO 2015

Abstrakt

Bakalářská práce se zabývá návrhem, tvorbou a implementací aplikace pro operační systém Android. Tato aplikace má plnit funkci stahování a offline přehrávání záznamů přednášek. Během samotného přehrávání je k dispozici vyhledávání na základě zadaného textu nebo na základě výběru slidu probíraného v dané přednášce. Aplikace umožňuje stahovat přednášky v různé kvalitě na základě výběru uživatele.

Abstract

This thesis is concerned with the design, development and implementation of an application for the Android operating system. This application fulfills the function of downloading and offline playback of recordings of lectures. During the playback of the recordings, it is possible to search based on the entered text or on the basis of the selection of a specific slide discussed in the class. The application allows you to download the lectures in various quality, based on the user's selection.

Klíčová slova

Android, Java, mobilní aplikace, server, XML.

Keywords

Android, Java, mobile application, server, XML.

Citace

Jiří Koudelka: Přednášky na cesty: android app pro
offline prohlížení přednášek na FIT, bakalářská práce, Brno, FIT VUT v Brně, 2015

Přednášky na cesty: android app pro offline prohlížení přednášek na FIT

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Igora Szőkeho, Ph.D.

.....
Jiří Koudelka
20. května 2015

Poděkování

Děkuji svému vedoucímu práce Ing. Igorovi Szőkemu, Ph.D. za odborné vedení a rady, které mi poskytl a pomohl tak s řešením práce. Dále bych chtěl poděkovat panu Bc. Ivu Hlavničkovi za rady z oblasti programování mobilních aplikací.

Obsah

1 Úvod **3**

2 Analýza a návrh řešení **5**
2.1 Získání požadavků na aplikaci 5
2.2 Koncept řešení 5
2.3 Návrh testování 6

3 Operační systém Android **7**
3.1 Architektura operačního systému android 7
3.2 Základní prvky Android aplikace 8
3.3 Úroveň Android API 9
3.4 Oprávnění v Android aplikacích 9

4 Využité technologie **11**
4.1 Vývojové prostředí Android Studio 11
4.2 Knihovny využité v aplikaci 11
4.3 Datové struktury využité v aplikaci 12
4.4 Balíčky aplikace 14

5 Implementace aplikace **16**
5.1 Zpracování XML v Javě 16
5.2 Adresářová struktura aplikace 17
5.3 Spuštění aplikace 17
5.4 Konfigurace aplikace 17
5.5 Výběr stažených přednášek 19
5.6 Přehrávání záznamů 19
5.7 Vyhledávání na základě slidů přednášky 20
5.8 Vyhledávání podle přepisu titulek 21
5.9 Technika stemming 22
5.10 Vyhledávání v textu 23
5.11 Návrh komunikace se serverem 23
5.12 Stažení přednášky 24
5.13 Průběh stahování 24

6 Testování aplikace **26**
6.1 Testování jednotlivých úrovní API 26
6.2 Automatické testy uživatelského rozhraní 26
6.3 UI/Application Exerciser Monkey 27

6.4 Testování uživateli 27
6.5 Shrnutí a výsledky testování 27

7 Závěr **29**

A Obsah CD **32**

B getCourses.xml **33**

C Plakát **34**

Kapitola 1

Úvod

V současné době neustále roste zájem studentů na vysokých školách o nahrávání a posléze zveřejňování záznamů z přednášek probíhajících na vysokých školách. Tuto možnost již nyní několik škol v České republice svým studentům nabízí (mezi nimi je například i Fakulta informačních technologií Vysokého učení technického v Brně), nicméně přístup k těmto záznamům je pro studenty mnohdy komplikovaný nebo nepohodlný. Dalším v dnešní době se rozvíjejícím trendem je vývoj mobilních aplikací (Android, iOS, Windows phone atd.). Konkrétně operační systém Android je, jak ukazuje následující průzkum [1], nejvíce užívaným operačním systémem v mobilních technologiích.

Tato bakalářská práce se zabývá spojením těchto dvou současných trendů. Jejím předmětem je návrh, tvorba, implementace a v neposlední řadě i testování Android aplikace pojmenované Přednášky na cesty pro stahování a offline prohlížení přednášek nahraných na Fakultě informačních technologií Vysokého učení technického v Brně.

Druhá kapitola je věnována stručné analýze a návrhu řešení problémů spojených s vývojem této mobilní aplikace. Je zde rozebrán postup návrhu a získání požadavků na aplikaci od potencionálních budoucích uživatelů, popis základní koncepce aplikace a návrh implementace. Závěr kapitoly pojednává o návrhu testování aplikace v průběhu jejího vývoje, ale i po jeho dokončení.

Ke správnému pochopení, jak tato aplikace funguje, se ve třetí kapitole zabývá práce základním popisem operačního systému android a základy programování aplikací pro operační systém android. Kapitola obsahuje vysvětlení, jak vůbec Android aplikace vzniká, jaké jsou její základní prvky nezbytné pro její chod, co je a k čemu slouží úroveň Android API, k čemu slouží a proč jsou důležité oprávnění v Android aplikacích a které konkrétní oprávnění byly v aplikaci použity a další využité principy jako například aktivity nebo dialogy.

Kapitola číslo čtyři se zabývá technologiemi, které byly při implementaci dané aplikace použity a důvody, proč byly zrovna tyto technologie vybrány. Je zde popsáno, jaké vývojové prostředí bylo pro vývoj použito, jaké výhody přineslo, jaké nestandardní knihovny byly využity a k jakému účelu. V této kapitole jsou také popsány jednotlivé datové struktury, které byly použity, a popis jejich využití v aplikaci. Na závěr kapitoly jsou ještě zmíněny jednotlivé balíčky, ze kterých se aplikace skládá, jejich stručný popis a krátký popis jednotlivých tříd v balíčcích.

Další, v pořadí pátá, kapitola popisuje vlastní implementaci aplikace. Jsou zde podrobněji popsány jednotlivé principy fungování aplikace, jak aplikace zpracovává XML soubory, jak na sebe jednotlivé aktivity navazují a jak spolu komunikují. Dále je zde popsána například adresářová struktura aplikace, možnosti a způsob konfigurace aplikace, popis přehrá-

vání přednášky a vyhledávání v přednášce. V závěru kapitoly je představen návrh, jak bude aplikace v budoucnu komunikovat se serverem, jak se bude vůči němu ověřovat a současný stav jak aplikace se serverem komunikuje nyní.

I když tomu v dnešní době ne všichni vývojáři přikládají tak velkou důležitost, jedním z klíčových témat v oblasti vývoje mobilních aplikací je i samotné testování aplikace, a to jak v průběhu samotné implementace, tak především i testování výsledné aplikace. Tímto tématem se zabývá poslední kapitola. Popisuje jednotlivé metody testování, které byly využity, jejich principy a na závěr kapitoly i shrnutí včetně poznatků z testování.

V závěru publikace jsou shrnuty cíle a výsledky bakalářské práce. Dále pak hlavní problémy, které při jejím vývoji nastaly. Jsou zde uvedeny také návrhy na vylepšení aplikace.

Na CD, které je přiloženo k publikaci, jsou k dispozici kompletní zdrojové kódy aplikace, elektronická verze publikace, instalační soubor aplikace spustitelný na operačním systému Android, jednoduchý uživatelský manuál, prezentační plakát formátu A2 a krátké prezentační video.

Publikace je určena převážně odborně zaměřeným čtenářům, kteří jsou obeznámeni s problematikou programování Android aplikací.

Kapitola 2

Analýza a návrh řešení

Aplikace Přednášky na cesty byla navržena jako aplikace pro operační systém Android. Dle zadání se má jednat o aplikaci, která umožní studentům sledování záznamů přednášek bez přístupu k internetu. Uživatel aplikace si v době, kdy má přístup k internetu stáhne do mobilního zařízení vybrané přednášky, které si může později kdykoliv přehrát. Aplikace by neměla umožňovat pouze stažené záznamy přehrávat, ale uživatel by měl mít k dispozici i vyhledávání na základě jednotlivých slidů dané přednášky nebo vyhledávání v přepise jednotlivých titulků přednášky.

2.1 Získání požadavků na aplikaci

Primárními koncovými uživateli aplikace Přednášky na cesty budou především studenti. Proto byl proveden krátký osobní průzkum mezi studenty Vysokého učení technického Fakulty informačních technologií za účelem získání požadavků, které potencionální budoucí uživatelé na aplikaci mají. Po vysvětlení studentům co jim bude aplikace nabízet, většina studentů reagovala s nadšením, že by popisovanou aplikaci v budoucnu rozhodně využili.

Oproti plánovaným funkcím aplikace měli dotazovaní studenti ještě několik návrhů na vylepšení funkcionality. Jedním z návrhů byl například požadavek, aby aplikace umožňovala stahovat pouze zvukovou stopu přednášky, která by v zařízení zabrala méně paměti a v některých případech by jako studijní materiál postačovala. Dalším návrhem bylo potom například umožnění stažení kompletních slidů jednotlivých přednášek ve formátu PDF nebo studijní opory k danému předmětu. Aplikace se zapracováním těchto požadavků by potom mohla sloužit jako vhodný doplněk studentům při přípravě na zkoušky.

Z návrhů a poznámek dotazovaných studentů bylo několik poznámek a myšlenek zahrnuto do návrhu aplikace, případně zahrnuto do plánů na budoucí rozvoj aplikace. Celkově byli dotazovaní s původním návrhem však spokojeni a nebyl tedy důvod se nějak výrazně odchylovat od původního záměru.

2.2 Koncept řešení

Po provedení průzkumu mezi studenty vznikl finální návrh aplikace a s ním i návrhy na řešení jednotlivých problémů v aplikaci. Aplikace je, jak již bylo zmíněno, určena pro operační systém Android. Logicky by se dala rozdělit do několika základních částí.

První částí je stažení konkrétní přednášky do mobilního zařízení ze serveru. Aplikace se podle původního návrhu měla připojovat k serveru, který vznikal v rámci diplomové

práce za účelem přehrávání přednášek ve webovém prohlížeči. Tento server ovšem nebyl v době vývoje aplikace k dispozici, proto bylo nutné pozměnit původní záměr a aplikace se připojuje k serveru `superlectures.com`. Oproti nově zvolenému serveru se prozatím nebude potřeba ověřovat přihlašovacími údaji a uživatel bude mít k dispozici ke stažení pouze veřejně dostupné přednášky. V nastavení aplikace si bude moci uživatel zvolit zda preferuje stahování záznamů ve vyšší kvalitě nebo mu postačí záznamy v kvalitě standardní.

Další část se zabývá samotným přehráváním přednášek, které jsou již v mobilním zařízení staženy. Toto přehrávání bude probíhat bez nutnosti přístupu k internetu a uživatel si tedy bude moci přednášky přehrávat například při cestování v městské hromadné dopravě, případně na jiných místech, kde není internet k dispozici. Samotné přehrávání má podporu vyhledávání formou hledání textových řetězců v přepise titulků dané přednášky nebo vyhledávání v přehrávaném záznamu na základě jednotlivých slidů přednášky.

Poslední větší částí aplikace je již zmíněné vyhledávání. Vyhledávání v přehrávané přednášce je možno dvěma způsoby. Prvním způsobem je vyhledávání textu v přepise titulků přednášky. Přepis titulků bude stažen na pozadí při stahování konkrétní přednášky a uložen do paměti mobilního zařízení. Pro podporu vyhledávání bude využita technika `stemming`. Při inicializaci přehrávání staženého záznamu bude celý přepis titulků zpracován touto technikou a předpřipraven pro samotné vyhledávání. Po zadání hledaného textu se potom tato technika aplikuje jen na hledaný textový řetězec a hledá se v předpřipravených částech titulků. Druhou metodou je vyhledávání na základě jednotlivých slidů přednášky. Slidy jsou, stejně jako přepis titulků, staženy na pozadí při stahování záznamu přednášky. Během inicializace přehrávání staženého záznamu jsou potom předchystány s konkrétním časovým údajem do výběru, který bude uživateli poskytnut prostřednictvím seznamu slidů. Uživatel si vybere konkrétní slide a přehrávaný záznam je posunut do času, ve kterém se daný slide probírá.

2.3 Návrh testování

Za účelem testování aplikace během vývoje, i po jejím dokončení, bylo navrženo několik způsobů testování. Pro testování aplikace během jejího vývoje bylo naplánováno testování pomocí automatických testů a zátěžové testování pomocí tzv. "monkey"testů. Testování během vývoje by mělo být prováděno pravidelně, vždy po implementaci či změně větší části aplikace. Výsledky průběžných testů aplikace během jejího vývoje pomohou vývojáři odhalit závažnější chyby zavčas a je možné se tak vyhnout pozdější složité opravě.

Testování aplikace po jejím dokončení je prováděno formou uživatelských testů. Z cílové skupiny uživatelů je vybráno deset uživatelů, kteří obdrží předem předchystaný seznam úkonů k testování. Uživatelé testují aplikaci sami bez přítomnosti autora aplikace a výsledky jednotlivých testů jsou s autorem testů konzultovány později. Tato metoda testování zajistí nezávislost testerů, kteří nemohou být autorem aplikace v průběhu testování nijak ovlivněni.

Výsledky uživatelských testů poslouží autorovi aplikace jako důležitá zpětná vazba od potencionálních uživatelů a autor aplikace může později poznatky z tohoto testování zahrnout do budoucího vývoje aplikace, případně pokud by došlo k nalezení závažnějších nedostatků upravit ještě stávající verzi aplikace.

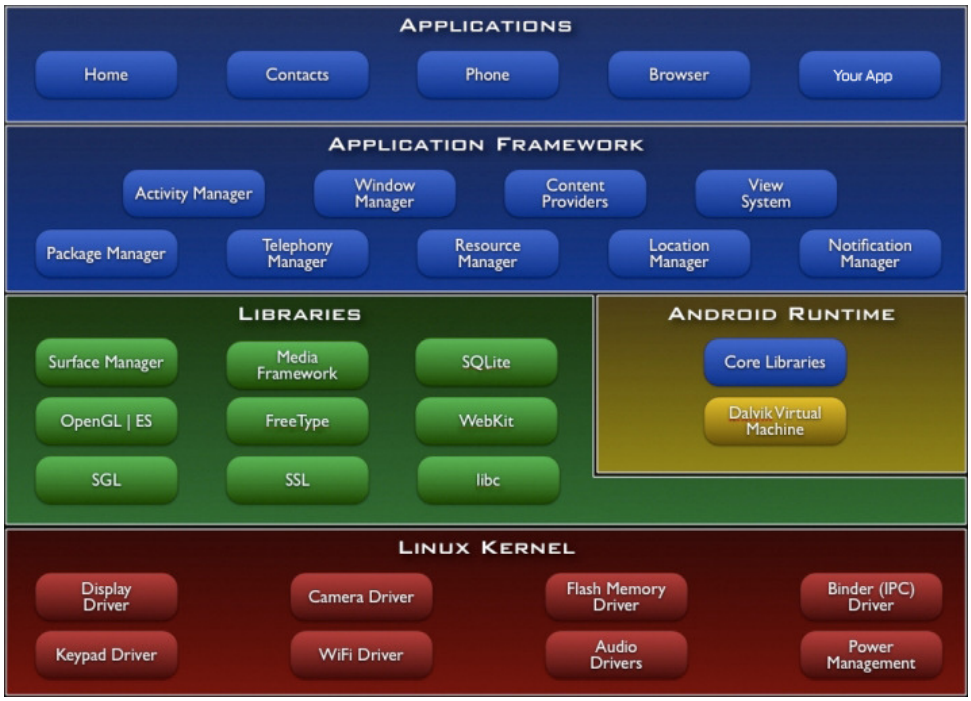
Kapitola 3

Operační systém Android

Podle průzkumů[1] patří operační systém Android k dnes nejrozšířenějším operačním systémům využívaným v mobilních technologiích. Je obecně známo, že v tomto odvětví se informační technologie vyvíjí velmi rychlým tempem a z toho důvodu se zde nebudeme zabývat detaily spojenými s tímto operačním systémem, ale popíšeme si základní obecné principy, jak tento systém funguje.

3.1 Architektura operačního systému android

Architituktura operačního systému Android je rozdělena na pět vrstev, které spolu vzájemně komunikují a tvoří spolu ucelený systém. Popis struktury jednotlivých vrstev operačního systému je znázorněn na obrázku 3.1.



Obrázek 3.1: Architektura operačního systému android (převzato z [4])

Přehled jednotlivých vrstev architektury operačního systému android [3]:

- **Linux Kernel** - Jádro operačního systému. Tvoří abstrakci mezi použitým hardwarem a softwarem ve vyšších vrstvách.
- **Libraries** - Vrstva zpřístupňující knihovny a rozhraní API pro vývoj aplikací.
- **Android Runtime** - Vrstva obsahující virtuální stroj DVM[1] a základní Java knihovny.
- **Application Framework** - Vrstva umožňující vývojáři přistupovat ke službám, které operační systém android poskytuje a využívat je ve svých aplikacích.
- **Applications** - Nejvyšší vrstva představující samotnou aplikaci využívanou uživatelem.

3.2 Základní prvky Android aplikace

Pro správnou funkčnost aplikace Přednášky na cesty je využito několik základních prvků. V této kapitole jsou popsány ty, které jsou v aplikaci nejčastěji použity a jsou pro správnou funkci aplikace Přednášky na cesty nepostradatelné.

Activity

Jeden z nejdůležitějších prvků pro tvorbu Android aplikace je tzv. activity (dále aktivita). Aktivita představuje prezentační vrstvu Android aplikace, přes kterou může uživatel s aplikací komunikovat, a to jak formou výstupů (aktivita zobrazuje uživateli data na display), tak i formou vstupů (uživatel může zadávat vstupní data aplikaci). Aktivitu musí bezprostředně obsahovat každá aplikace pro operační systém Android. Většinou je aktivita zobrazena přes celý display, ale může mít i podobu dialogového okna. Každá aktivita je potomkem třídy `Activity`[2].

View

Dalším důležitým prvkem při vývoji Android aplikací jsou tzv. view. View jsou využívány aktivitou pro jejich naplnění obsahem, který bude prezentován uživateli. Většinou se view vyskytuje ve formě XML souboru, kde jsou uloženy veškeré informace, které potřebuje aktivita pro správné zobrazení dat uživateli. Ten potom může tyto data měnit, mazat atd. Každé view je potomkem třídy `View`[3].

ListView

ListView slouží k zobrazování většího počtu dat formou jakéhosi seznamu. K propojení jednotlivých položek seznamu s konkrétními daty je zapotřebí tvz. adaptéru. ListView jsou v aplikaci Přednášky na cesty využity na místech, kde uživatel vybírá z nabídky možností. Například výběr přednášek ke stažení, výběr již stažených přednášek nebo i při vyhledávání na základě slidů. Každé ListView je potomkem třídy `ListView`[4].

[1] http://source.android.com/devices/tech/dalvik/
[2] http://developer.android.com/reference/android/app/Activity.html
[3] http://developer.android.com/reference/android/view/View.html
[4] http://developer.android.com/reference/android/widget/ListView.html

Android Manifest

Pro správný chod android aplikace je zapotřebí také Android Manifest [3]. Jedná se o XML soubor, ve kterém jsou uloženy konfigurační informace. Mezi nejdůležitější informace, které jsou uloženy v Android Manifestu patří například:

- Údaje o tom, jaké komponenty jsou aplikaci k dispozici.
- Seznam knihoven, využitých v android aplikaci.
- Deklarace oprávnění aplikace určující přístup k chráněným částem API.
- Minimální úroveň Android API.
- Maximální úroveň Android API.
- Atd.

3.3 Úroveň Android API

Úroveň Android API hraje důležitou roli v zajištění kompatibility mezi naprogramovanými aplikacemi a zařízeními, do kterých jsou nainstalovány. API úroveň je reprezentována prostřednictvím celého čísla v závislosti na dané verzi Android platformy. Platforma Android poskytuje rámec API, který mohou aplikace využívat pro komunikaci se základním systémem Android [3].

Rámec API je složen z následujících komponent:

- Sada balíčků a tříd.
- Sada elementů a atributů pro deklaraci v souboru `AndroidManifest.xml`.
- Sada elementů a atributů pro deklaraci a přístup k prostředkům (zdrojům).
- Soubor záměrů.
- Sada oprávnění, která aplikace mohou požadovat.

Konkrétní číslo úrovně API udává konkrétní verzi Android platformy, na které daná aplikace poběží. Pro každou aplikaci musí být v `Android manifestu` 3.2 uvedena minimální `android:minSdkVersion` a maximální `android:targetSdkVersion` hodnota úrovně API. Podle těchto uvedených hodnot je potom aplikace dostupná či nikoliv pro konkrétní verzi operačního systému.

Aplikace přednášky na cesty má stanovenu minimální úroveň API na číslo 13, což odpovídá verzi Android 3.2, která nese jméno `HONEYCOMB`. Maximální úroveň je potom stanovena na API úroveň 21 odpovídající verzi Android 5.0 nesoucí jméno `LOLLIPOP`.

3.4 Oprávnění v Android aplikacích

Jak již bylo naznačeno, pokud chce Android aplikace přistupovat k chráněným částem API, jako je například přístup k ukládání souborů do externí paměti nebo využití internetu, musí být aplikaci uděleno v Android Manifestu patřičné oprávnění. Za předpokladu, že by toto oprávnění uděleno nebylo, aplikace by při pokusu o přístup ke chráněnému zdroji

byla ukončena s chybovým hlášením. O oprávněních, která aplikace vyžaduje, je uživatel informován při její instalaci a musí tato oprávnění potvrdit.

Aplikace vyvíjená za účelem offline přehrávání přednášek pro svůj běh vyžaduje a obsahuje následující oprávnění.

```
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission android:name="android.permission.READ_PHONE_STATE" />
<uses-permission android:name="android.permission.DOWNLOAD_WITHOUT_NOTIFICATION" />
```

Jednotlivá oprávnění v aplikaci

V aplikaci Přednášky na cesty bylo použito několik nezbytně nutných oprávnění:

- **WRITE_EXTERNAL_STORAGE** - oprávnění, které umožňuje aplikaci ukládat soubory na externí úložiště (například SD karta mobilního zařízení).
- **INTERNET** - oprávnění, které umožňuje aplikaci přistupovat k internetu (v případě aplikace Přednášky na cesty je toto oprávnění nutné pro stahování přednášek).
- **ACCESS_NETWORK_STATE** - oprávnění, které umožňuje aplikaci zjistit, zda je v mobilním zařízení aktuálně přístup k internetu.
- **READ_PHONE_STATE** - oprávnění pro zjištění stavu telefonu (například za účelem zjištění, zda zrovna neprobíhá nějaké stahování).
- **DOWNLOAD_WITHOUT_NOTIFICATION** - oprávnění pro umožnění stahování souborů do mobilního zařízení bez toho, aby o tom byl uživatel informován (aplikace tento druh stahování využívá například pro stahování jednotlivých slidů přednášky).

Kapitola 4

Využité technologie

Při programování aplikací pro operační systém Android se využívá převážně programovací jazyk Java, a ani aplikace Přednášky na cesty není výjimkou. Mimo standardních knihoven programovacího jazyka Java jsou při vývoji aplikací pro operační systém Android užity specializované knihovny, které přispívají k jednoduššímu programování aplikací. Pro analýzu a zpracování XML souboru, ve kterém přichází data od serveru, ze kterého se bude přednáška stahovat, je využita knihovna SAXParser, která je pro účel vývoje mobilních aplikací nejvhodnější. Více informací o zpracování XML souborů a důvody, proč je právě tato knihovna nejvhodnějším řešením se nachází ve páté kapitole 5.1.

4.1 Vývojové prostředí Android Studio

Pro vývoj aplikace Přednášky na cesty bylo využito vývojového prostředí Android Studio, které je založeno na technologii `IntelliJ IDEA` a je vyvíjeno společností Google. Součástí android studia jsou i debugovací nástroje pro vývoj mobilních aplikací a emulátor mobilních zařízení, který umožňuje vývojáři testovat vyvíjenou aplikaci.

4.2 Knihovny využité v aplikaci

V aplikaci se využívá několik knihoven pro podporu některých funkcionalit, případně pro testování aplikace. Mezi knihovny využité v této aplikaci se řadí například:

Lucene

Lucene[1] je engine poskytující fulltextové vyhledávání a indexační funkce pro aplikace. Je vyvíjen pod open source licencí a je tedy volně přístupný a není problém s jeho využitím ve vlastních pracích.

Původně byl Lucene vyvíjen v programovacím jazyce Java. Dnes je však již rozšířen a lze jej použít i při programování v ostatních programovacích jazycích jako například C++, Python, PHP, Perl a dalších.

V aplikaci Přednášky na cesty je Lucene využit jako podpora při vyhledávání v textu.

[1] https://lucene.apache.org/core/

Android Support

V aplikaci je využívána mimo klasické knihovny pro vývoj android aplikací i tvz. `Android Support` knihovna[2]. Tato knihovna nám zpřístupňuje funkcionalitu vyšších úrovní API v úrovních nižších.

Využitím této knihovny je možné aplikaci Přednášky na cesty využívat i na starších verzích operačního systému Android než by tomu bylo, kdyby tato knihovna použita nebyla. To umožňuje dostupnout aplikace většímu počtu uživatelů.

Espresso

Espresso[3] je framework umožňující vývojáři psát automatické testy uživatelského rozhraní v rámci aplikace. Poskytuje sadu API rozhraní, pomocí kterých vývojář testy sestaví. K dispozici nabízí například simulaci kliknutí, gesta, zadávání textu a další.

Pro svoje spuštění využívá spouštěč JUnit testů `AndroidJUnitRunner` 4.2.

AndroidJUnitRunner

AndroidJUnitRunner[4] umožnuje načítat, spouštět a vyhodnocovat jednotlivé testovací sady připravené frameworkem `Espresso` 4.2.

4.3 Datové struktury využité v aplikaci

Pro aplikaci Přednášky na cesty bylo navrhnuto několik základních objektů, které reprezentují základní prvky, které aplikace využívá. Tyto datové struktury se nachází v balíčku `cz.pnc.datastructures`.

Objekt Lecture

Objekt Lecture reprezentuje jednu přednášku a uchovává o ní důležité informace. Využití objektu Lecture je i při stahování nové přednášky, konkrétně slouží pro reprezentaci jednoho řádku v `ListView` při výběru nové přednášky. Tento objekt se skládá z několika atributů:

- **author** - proměnná datového typu `String`, která obsahuje jméno autora přednášky
- **speaker** - proměnná datového typu `String`, která obsahuje jméno přednášejícího
- **recorded** - proměnná datového typu `String`, která obsahuje datum, kdy byla přednáška pořízena
- **length** - proměnná datového typu `String`, která obsahuje informace o tom, jaká je celková délka přednášky
- **slides** - kolekce objektů `Slide` 4.3, která obsahuje objekty nesoucí informace o jednotlivých slidech přednášky
- **downloadLink** - proměnná datového typu `String`, která obsahuje odkaz pro stažení přednášky

[2] http://developer.android.com/tools/support-library/index.html
[3] http://developer.android.com/tools/testing-support-library/index.html#Espresso
[4] http://developer.android.com/tools/testing-support-library/index.html#AndroidJUnitRunner

- **numberOfSlides** - proměnná datového typu `Integer`, která obsahuje počet slidů přednášky
- **pdfLink** - proměnná datového typu `String`, která obsahuje odkaz na stažení PDF souboru s texty k přednášce
- **name** - proměnná datového typu `String`, která obsahuje název přednášky
- **lectureId** - proměnná datového typu `Long`, která obsahuje identifikátor dané přednášky
- **subtitlesLink** - proměnná datového typu `String`, která obsahuje odkaz ke stažení titulků
- **course** - proměnná datového typu `String`, která obsahuje jméno předmětu dané přednášky

Objekt Slide

Objekt Slide reprezentuje jeden slide k dané přednášce a bývá součástí kolekce objektů tohoto typu, která je uložena v objektu Lecture. Tento objekt se skládá z několika atributů:

- **time** - proměnná datového typu `Integer`, která obsahuje údaj o čase, od kterého je v přednášce daný slide probírán
- **downloadLink** - proměnná datového typu `String`, která obsahuje odkaz na stažení konkrétního slidu přednášky
- **slideName** - proměnná datového typu `String`, která obsahuje jméno konkrétního slidu přednášky

Objekt TextTimer

Objekt TextTimer reprezentuje část textu převzatou z přepisu titulků, které obdrží aplikace od serveru. Dále pak tento objekt obsahuje údaje o časování této části textu v celkovém videu a originální znění textu před úpravou za účelem lepšího vyhledávání. Objekt je složen z několika atributů:

- **text** - proměnná datového typu `String`, která obsahuje část textu z přepisu titulků po převedení do tvaru pro vhodnější vyhledávání
- **startTime** - proměnná datového typu `String`, která obsahuje čas, ve kterém se tato část textu ve videu začíná vyskytovat
- **endTime** - proměnná datového typu `String`, která obsahuje čas, ve kterém se tato část textu ve videu přestává vyskytovat
- **originalText** - proměnná datového typu `String`, která obsahuje část textu z přepisu titulků v originálním tvaru

Objekt RowItem

Objekt RowItem reprezentuje jeden řádek v `ListView`, pomocí kterého se vyhledává v konkrétní přednášce na základě jednotlivých slidů. Tento objekt se skládá z několika atributů:

- **imagePath** - proměnná datového typu `String`, která obsahuje URL cestu k jednomu konkrétnímu slidu
- **title** - proměnná datového typu `String`, která obsahuje název konkrétního slidu
- **startTime** - proměnná datového typu `String`, která obsahuje informace o tom, kdy se konkrétní slide v přednášce vyskytuje; čas je ve formátu, který je čitelný pro uživatele
- **videoPosition** - proměnná datového typu `String`, která obsahuje informace o tom, kdy se konkrétní slide v přednášce vyskytuje; časový údaj je v této proměnné uložen v sekundách
- **categoryId** - proměnná datového typu `Long`, která obsahuje identifikátor předmětu
- **numberOfLectures** - proměnná datového typu `Long`, která obsahuje počet přednášek daného předmětu

Objekt Course

Objekt Course reprezentuje jeden řádek v `ListView`, pomocí kterého se vybírá, pro který předmět bude uživatel stahovat přednášku. Objekt `Course` se skládá z následujících atributů:

- **name** - proměnná datového typu `String`, která obsahuje jméno předmětu
- **shortcut** - proměnná datového typu `String`, která obsahuje zkratku daného předmětu
- **downloadLink** - proměnná datového typu `String`, která obsahuje odkaz na stažení XML souboru s informaceni o dané přednášce

4.4 Balíčky aplikace

Celá aplikace je rozložena do několika logických celků a na základě těchto celků i do jednotlivých balíčků. Prvním je již zmiňovaný balíček `Datastructures` 4.3, který obsahuje základní objekty využívané v aplikaci. Tento balíček byl již detailněji popsán v předchozí kapitole, a tak nebude popisován znovu. Ostatní balíčky si popíšeme v následujících bodech.

connection

Connection je balíček, obsahující třídu pro vytváření, posílání a příjímíná `HTTP requestů`, pomocí kterých aplikace komunikuje se serverem.

Jméno třídy umístěné v tomto balíčku je `HttpConnection` a pro svou komunikaci se serverem využívá standardních knihoven programovacího jakzyka Java. Konkrétně `java.net`.

downloader

Downloader obsahuje třídy, které uživateli zprostředkují výběr předmětu a následně pak i konkrétní přednášky, které chce stáhnout do mobilního zařízení. Mimo výběru se zde nachází i třída pro samotné stažení dané přednášky.

Třídy `MyCourseAdapter` a `MyLectureAdapter` reprezentují adaptéry pro `ListView` zprostředkovávající samotný výběr. Výběr předmětu zajišťuje třída `CourseSelector`, výběr a následné stažení přednášky potom třída `LectureDownloader`.

imageList

Balíček imageList obsahuje třídu potřebnou k vyhledávání v jednotlivých slidech přednášky.

Třída v tomto balíčku se jmenuje `MyListViewAdapter`. Tato třída reprezentuje adaptér pro `ListView` zpřístupňující jednotlivé slidy přednášky.

lectures

Tento balíček obsahuje třídy pro základní běh aplikace a enumerace pro sestavování chybových hlášek. Enumerace `ErrorMessage` obsahuje výčet jednotlivých chybových hlášek, které mohou být aplikací vypsány a enumerace `ErrorTypes` je výčtem hlaviček těchto chybových hlášení.

Balíček lectures dále obsahuje třídu `MainActivity`, která je vnímána jako hlavní třída aplikace, třídu `SettingsActivity`, která uživateli zpřístupňuje nastavení aplikace, třídu `NavigationDrawerFragment`, která zpřístupňuje navigační menu na úvodní obrazovce. Dále pak třídy `VideoListActivity` a `VideoListAdapter` sloužící pro výběr konkrétní stažené přednášky. Nakonec potom třída `VideoSystem` starající se o přehrávání samotného záznamu přednášky.

resources

Balíček resources obsahuje soubor a třídu pro základní konfiguraci aplikace. V konfiguračním souboru `config.properties` jsou v textové podobě uloženy jednotlivé konfigurační parametry. Třída `ConfigReader` potom zpřístupňuje jednotlivé položky, které jsou v tomto konfiguračním souboru uloženy.

stemmer

Dalším balíčkem, který se v aplikaci nachází je balíček stemmer. V tomto balíčku jsou třídy sloužící k vyhledávání v textu. Konkrétně je to třída `TextConverter`, která v aplikaci slouží ke konverzi jednotlivých titulků na jejich kmenový tvar.

Mezi další třídy v tomto balíčku se řadí třída `Stemmer` a třída `TextSearch`. Činnost těchto tříd je popsána v kapitole zabývající se vyhledáváním v textu 5.8.

xmlParse

Posledním balíčkem je balíček `xmlParse`, který obsahuje pouze jedinou třídu, a to konkrétně `XmlParser`. Tato třída slouží k získávání informací z XML souborů a je detailněji popsána v kapitole zabývající se touto problematikou 5.1.

Kapitola 5

Implementace aplikace

Aplikace je implementována a navržena pro operační systém Android. Celá aplikace je implementována v programovacím jazyce Java za využití technologií pro vývoj Android aplikací.

5.1 Zpracování XML v Javě

V programovacím jazyce Java se zpracování XML souboru provádí dvěma způsoby. Zpracováním XML pomocí stromové reprezentace dokumentu, nebo pomocí proudového zpracování XML dokumentu. Na výhody a nevýhody, případně důvody, proč je vhodné či nevhodné využití konkrétní metody v android aplikacích se zaměřují následující body.

Zpracování XML pomocí stromové reprezentace dokumentu

Zpracování dokumentu pomocí této metody probíhá načtením celého dokumentu do paměti a následným získáváním potřebných informací za pomocí knihovny DOM (Document Object Model). Tato varianta ovšem logicky zabírá více místa v paměti, a tak využití této varianty v mobilních aplikacích není nejvhodnější volbou.

Proudové zpracování XML dokumentu

Zpracování XML dokumentu za pomoci této metody probíhá postupně. Při každém výskytu uceleného prvku je vyvolána událost, kterou si sami, vlastně implementovanými metodami, zpracujeme. V programovacím jazyce Java je tento druh zpracování implementován pomocí knihovny SAX[1] (Simple API for XML). Tato varianta je oproti metodě zpracování XML pomocí stromové reprezentace dokumentu rychlejší a paměťově méně náročná. Na základě těchto dvou hlavních výhod je tato varianta vhodná pro použití v mobilních aplikacích, kde jsou na paměť kladeny větší požadavky než je tomu například u desktop aplikací.

Zpracování XML v aplikaci Přednášky na cesty

Pro zpracování XML souborů v aplikaci Přednášky na cesty slouží třída `XmlParser`. Jak již bylo zmíněno, tato třída využívá knihovnu `javax.xml.parsers.SAXParser` a využívá některé její metody. Tato třída je v aplikaci využita na více místech vždy v případě, že je potřeba zpracovat nějaký XML soubor.

[1] http://sax.sourceforge.net/

Při prvním průchodu funkcí pro zpracování XML souborů se zpracovává soubor, který aplikace obdrží ze strany serveru. Během tohoto zpracování se zjistí informace o přednášce, a tyto informace se uloží do objektu typu `Lecture`. Mezi jednotlivé informace, které se ze zpracovávaného souboru zjistí, patří odkaz pro stažení dané přednášky, informace o jednotlivých slidech přednášky, odkaz na stažení PDF souboru se slidy přednášky a obecné informace o přednášce (jméno, délka, autor přednášky, ...).

Druhý průchod funkcí slouží ke zpracování XML souboru, který obsahuje přepis titulků s časováním. Jednotlivé části textu jsou i s jednotlivými časy začátku a konce výskytu ukládány do objektů typu `TextTimer`, které se později využívají při vyhledávání během přehrávání přednášky.

5.2 Adresářová struktura aplikace

Primární adresář aplikace Přednášky na cesty je pojmenován `PNC`. Tento adrešář potom obsahuje adresář `prednasky`, do kterého jsou ukládány jednotlivé přednášky. Dále je zde uložen konfigurační soubor `config.properties` a jsou zde ukládány dočasné XML soubory, které aplikace obdrží od serveru za účelem získání konkrétních přednášek. Tyto dočasné soubory (getLect.xml a getCourses.xml) jsou při využívání aplikace přepisovány. Více o těchto dočasných souborech bude popsáno v kapitole, která se zabývá komunikací se serverem 5.11.

Jednotlivé přednášky se ukládají do adresáře `prednasky` následujícím způsobem. Každá přednáška má vlastní adresář, který je pojmenován jménem dané přednášky. V tomto adresáři je uložen samotný videozáznam přednášky ve formátu `mp4`, soubor ve formátu `tt` s přepisem titulků k dané přednášce, soubor `course.txt`, který obsahuje informace o tom, do kterého předmětu daná přednáška spadá a `XML` soubor s informacemi o dané přednášce. Nachází se zde také adresář slides, kde jsou uloženy jednotlivé slidy k dané přednášce.

Celá adresářová struktuje je znázorněna na obrázku 5.1.

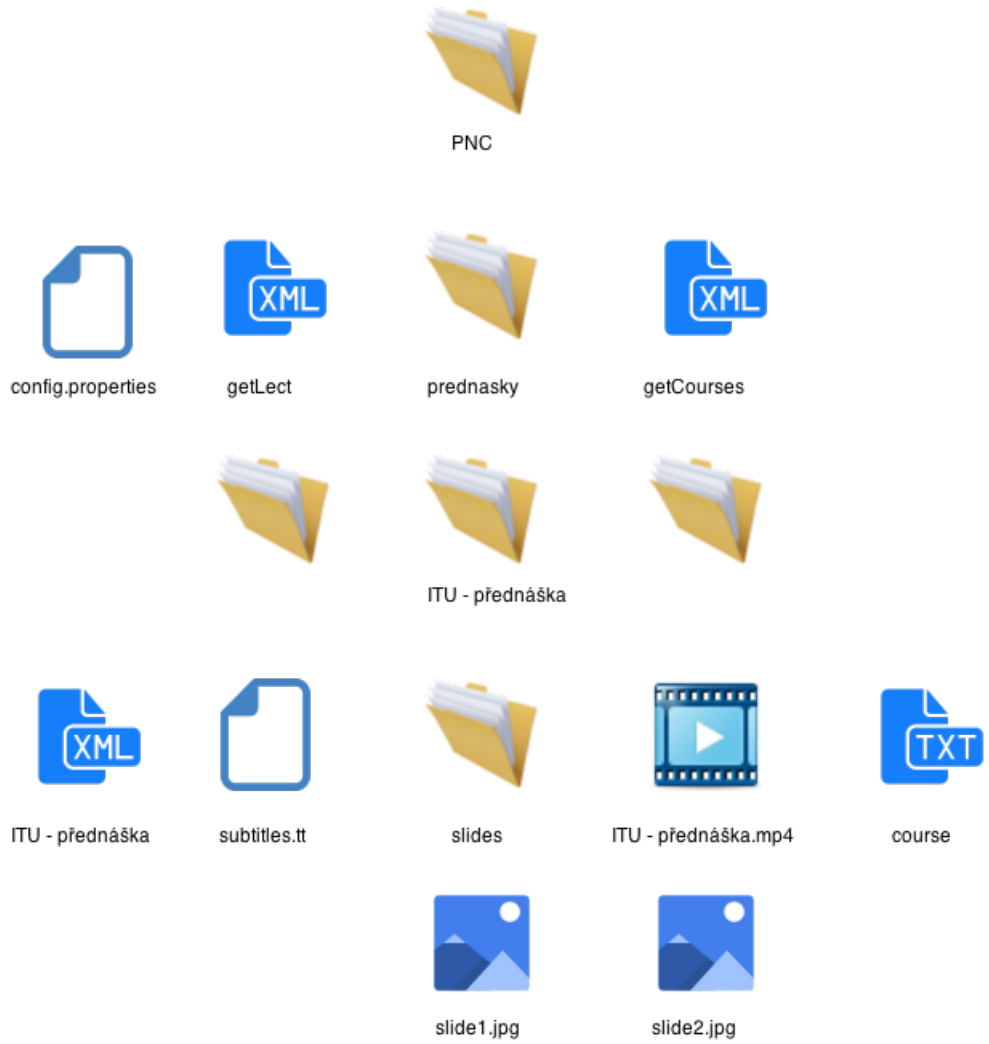
5.3 Spuštění aplikace

Při prvním spuštění aplikace dochází k vytvoření adresářové struktury aplikace a následně potom zobrazení úvodní aktivity aplikace. Zda se jedná o první spuštění nebo nikoliv se aplikace dozví z konfiguračního souboru. Při každém dalším spuštění aplikace se již jen testuje, zda nedošlo k poškození adresářové struktury aplikace. Úvodní activita aplikace je reprezentována třídou `MainActivity`, která rozšiřuje třídu `Activity`.

Úvodní aktivita aplikace je tvořena rozložením `RelativeLayout`. Uprostřed activity se nachází menu pro přístup k jednotlivým funkcím aplikace. Z menu je tedy možno dostat se na výběr již stažených přednášek, do části aplikace určené pro stahování nových přednášek, do nastavení celé aplikace nebo případně aplikaci ukončit.

5.4 Konfigurace aplikace

Aplikace je konfigurovatelná prostřednictvím GUI v menu aplikace. Nastavit se dá například kvalita stahovaného záznamu přednášky nebo například zda se má po zvolení slidu či zvolení nalezené shody v titulcích uzavřít výběr a zobrazit přehrávaný záznam. Konfigurační údaje jsou uloženy v konfiguračním souboru `config.properties` a je tedy možné editovat je i ručně. Tato možnost však není doporučena nezkušeným uživatelům z důvodu možného poškození dat uložených v tomto konfiguračním souboru.



Obrázek 5.1: adresářová struktura aplikace.

Aplikace přistupuje k datům v konfiguračním souboru prostřednictím třídy `ConfigReader`. Pro přístup k jednotlivým položkám v souboru `config.properties` je využita především knihovna `java.util.Properties`.

Příklad konfiguračního souboru je uveden v následující ukázce.

```
#Config file
#Fri May 08 21:59:27 CEST 2015
firstStart=1
highResolution=0
closeImage=0
closeText=0
```

5.5 Výběr stažených přednášek

Výběr přednášek, které jsou již v mobilním zařízení staženy, zprostředkovává třída `VideoListActivity`, která rozšiřuje třídu `Activity`. Jednotlivé přednášky jsou nabízeny prostřednictvím rozložení `GridView`, které pro práci s jednotlivými položkami využívá adaptér. Tento adaptér je reprezentován třídou `VideoListAdapter`, která rozšiřuje třídu `ArrayAdapter<Lecture>`.

Stažené přednášky se dají filtrovat podle jednotlivých předmětů. Výběr je zprostředkován pomocí tzv. `Spinneru`, kde jsou na výběr jednotlivé předměty.

Po zvolení konkrétní přednášky z nabídky se zavolá metoda `playVideo`, která si sestaví URL adresu k video souboru, cestu k adresáři s uloženými slidy, cestu ke XML souboru s informacemi o přednášce a cestu k souboru s přepisem titulků. Tyto cesty se předají prostřednictvím `Intentu` do aktivity, která přehrává jednotlivé záznamy.

V tomto výběru přednášek má uživatel možnost také odstraňovat přednášky z mobilního zařízení. Samotné odstranění se provádí gestem přidržení na konkrétní přednášce. Uživatel je poté prostřednictvím dialogového okna vyzván k potvrzení odstranění záznamu.

5.6 Přehrávání záznamů

O samotné přehrávání záznamů přednášek se stará aktivita, která je reprezentována třídou `VideoSystem` jež rozšiřuje třídu `Activity` a implementuje rozhraní `AdapterView.OnItemClickListener`. Přehrávání zvolené přednášky se odehrává ve středu aktivity. Po levé i pravé straně aktivity je možnost vyvolat `DrawerLayout`, což můžeme popsat jako nějaký kontejner nejvyšší úrovně obsahu okna, který může být vytažen z okraje okna. Levý `DrawerLayout` slouží k vyhledávání v přepisech titulků. Pravý potom k vyhledávání v jednotlivých slidech přednášky. Ukázka, jak vypadá přehrávání záznamu přednášky, je znázorněna na obrázku 5.2.

Obrázek 5.2: ukázka samotného přehrávače záznamů přednášek.

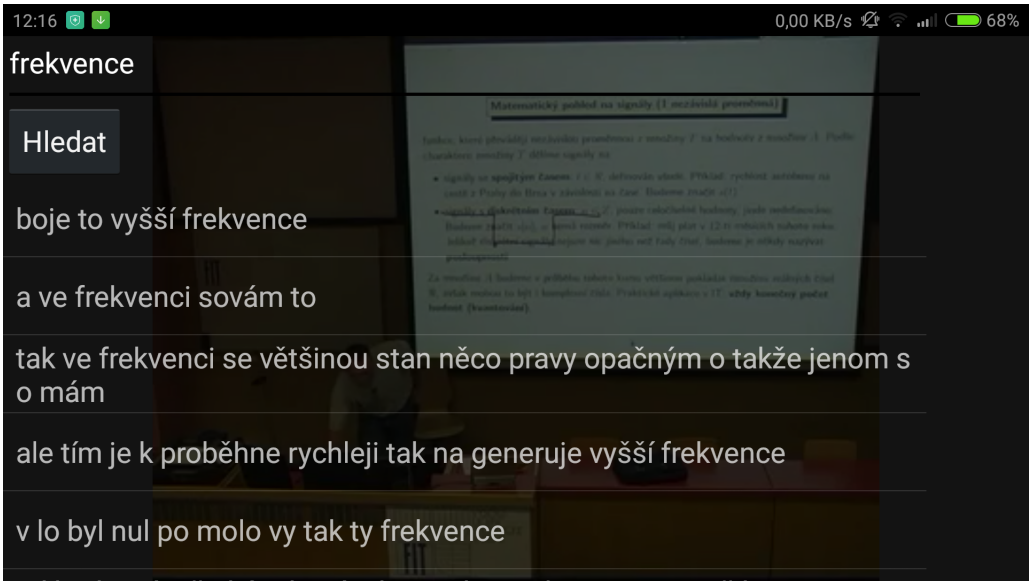
Při startu této aktivity se zavolá inicializační funkce `init`. V této funkci se uloží cesty

k potřebným souborům, které byly předány z aktivity `VideoListActivity` prostřednictvím intentu. Následně se provede zpracování XML souboru s informacemi o přednášce a na základě získaných informací se vytvoří seznam slidů.

K vytvoření seznamu slidů slouží funkce `createImageList`, která obdrží jako vstupní parametr list objektů typu `Slide`. Tato funkce potom bere jeden slide po druhém a konvertuje jej na objekt datového typu `RowItem`, který reprezentuje jeden řádek v `ListView`, ze kterého si uživatel může jednolivé slidy vybírat. Během samotné konverze se musí do nově vznikajícího objektu typu `RowItem` nastavit následující hodnoty: cesta k souboru datového typu `JPG`, který reprezentuje jeden slide přednášky, jméno konkrétního slidu a řetězec, který se bude u slidu zobrazovat. V případě aplikace Přednášky na cesty bude zobrazovaný řetězec udávat v jakém konkrétním čase se daný slide v přednášce probírá.

Dále pak proběhne nastavení odpovídajících `ListView` do levého a pravého `DrawerLayoutu` a samotné spuštění přehrávání zvoleného záznamu přednášky.

Při gestu tažením z levého okraje displaye směrem doprava se objeví levý `DrawerLayout`, ve kterém je možno vyhledávat na základě zadaného textového řetězce v přepise titulek záznamu přednášky. Jak vypadá takto vyvolaný `DrawerLayout` je možno vidět na obrázku 5.3.



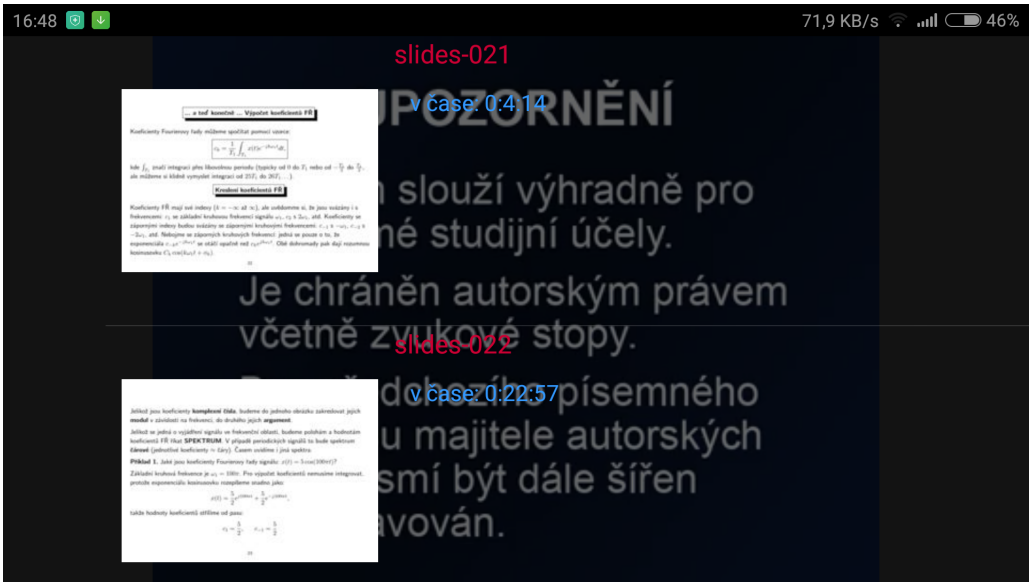
Obrázek 5.3: ukázka vyhledávání v přepise titulek konkrétní přednášky.

Při gestu opačném, tedy tažením z pravého okraje displaye směrem doleva, se objeví pravý `DrawerLayout`, ve kterém může uživatel vyhledávat v právě přehrávaném záznamu přednášky na základě jednotlivých slidů přednášky. Jak jednotlivé slidy vypadají je znázorněno na obrázku 5.4.

Jednotlivým způsobům vyhledávání se věnují následující kapitoly.

5.7 Vyhledávání na základě slidů přednášky

Prvním způsobem vyhledávání je vyhledávání na základě slidů přednášky. Jednotlivé slidy, jak již bylo řečeno, jsou zobrazovány v pravém `DrawerLayoutu` a to za pomoci `ListView`,



Obrázek 5.4: ukázka vyhledávání prostřednictvím slidů konkrétní přednášky.

který využívá vlastní adaptér. Tento adaptér je reprezentován třídou `MyListViewAdapter`, která rozšiřuje třídu `ArrayAdapter<>`.

Jednotlivé slidy jsou reprezentovány objektem typu `RowItem` 4.3. Tento objekt v sobě nese informace o daném slidu. Po výběru jednoho slidu je zavolána metoda `setVideoFromSlide`, která bere jako vstupní argument objekt typu `RowItem`. Z atributu `videoPosition` tohoto objektu se získá čas, ve kterém se daný slide vyskytuje v přednášce a po převedení na milisekundy se nastaví čas přehrávaného videa do výsledné hodnoty.

V závislosti na nastavení v konfiguraci aplikace se po vybrání jednoho ze slidů zobrazený `DrawerLayout` schová a pro opětovnou možnost výběru slidu je potřeba opět pravý `DrawerLayout` vyvolat příslušným gestem.

5.8 Vyhledávání podle přepisu titulek

Dalším způsobem, kterým je možno, v právě přehrávaném záznamu přednášky vyhledávat, je vyhledávání v přepisu titulek. Toto vyhledávání je zprostředkováno prostřednictvím levého `DrawerLayoutu`, který se dá vyvolat gestem tažením z levého okraje displaye směrem doprava. Zobrazení jednotlivých řádků se provádí prostřednictvím `ListView` a jako adaptér se využívá klasický `ArrayAdapter<String>`.

Po vytažení levého `DrawerLayoutu`, zadání hledaného řetězce do pole, určeného pro vyhledávání, a stisknutí příslušného tlačítka pro vyhledávání se zavolá funkce `search`. Funkce search využívá pro své vyhledávání třídu `XmlParser`, a to hned ve dvou průchodech. Během prvního průchodu jsou získávány obecné informace o přednášce z XML souboru. V druhém průchodu potom tedy zpracovává soubor s titulky dané přednášky.

Titulky, které byly zpracovány během druhého průchodu xml parserem jsou uloženy do listu objektů typu `TextTimer`. O konverzi mezi titulky a objektem typu `TextTimer` se stará třída `TextConverter`. Tento list je potom předán jako jeden z argumentů metodě `search`, která se nachází ve třídě `TextSearch`. Podrobněji se vyhledáváním v přepisech z titulků

bude zabývat následující kapitola.

Po vyhledání hledaného řetězce a zvolením jednoho z možných nalezených textů je zavolána metoda `setVideo`, která nastaví video, běžící na pozadí, na čas, ve kterém se hledaný text v přednášce nachází.

V závislosti na nastavení v konfiguraci aplikace se po výběru konkrétního hledaného textu levý `DrawerLayout` zasune a na displayi je pouze aktuálně přehrávaná přednáška.

5.9 Technika stemming

Pro vyhledávání v textu je použita technika, která se nazývá `stemming`. Jedná se o techniku, kde je zdrojový text, ve kterém hledáme, slovo po slovu zbavován přípon a koncovek. Každé slovo je tedy z původního znění předěláno na kmen tohoto slova (například slovo frekvence má kmen frekvenc). Stejným způsobem jsou převáděna i slova zadaná uživatelem, podle kterých se má vyhledávat. Tato technika je výhodná zejména z toho hlediska, že nám odstraní problémy s časováním nebo skloňováním hledaných slov.

Stemmer pro získávání kmene slov je reprezentován třídou `Stemmer` a pro svou funkčnost využívá knihovny `Lucene` 4.2. Hlavní funkce této třídy, která se volá, pokud chci získat délku daného slova po normalizaci, je metoda `stem`. Metoda stem bere jako vstupní parametry posloupnost znaků daného slova a jeho původní délku.

Základní algoritmus stemmingu byl převzat z publikace Indexing and stemming approaches for the Czech language [2].

Postup normalizace slova

První část normalizace se skládá z odstranění koncovky daného slova. O tento úkol se stará metoda `removeCase`, která bere jako vstupní parametr posloupnost znaků slova a délku daného slova. Postup odstranění koncovky probíhá následujícím způsobem. Dané slovo je testováno na délku a pomocí knihovní metody `endsWith` na koncovku. Pokud projde slovo danou kombinací podmínek je jeho výsledná velikost zmenšena o délku testované koncovky a tato nová velikost je vrácena do funkce stem jako nová velikost daného slova. Krátký příklad je uveden v následující ukázce zdrojového kódu.

```
if(len > 4 &&
    (endsWith(s, len, "em") ||
      endsWith(s, len, "es") ||
      endsWith(s, len, "ém") ||
      endsWith(s, len, "ím") ||
      endsWith(s, len, "ům") ||
      endsWith(s, len, "at") ||
      endsWith(s, len, "ám") ||
      endsWith(s, len, "os") ||
      endsWith(s, len, "us") ||
      endsWith(s, len, "ým") ||
      endsWith(s, len, "mi") ||
      endsWith(s, len, "ou")))
return len -2;
```

Další částí normalizace je odstranění koncovek slov v přivlastňovacích tvarech za pomocí metody `removePossessives`. Jedná se o odstranění koncovek "ov", "in"a "ův"u slov, které jsou delší jak pět znaků.

Poslední fází je zavolání metody `normalize`, která převádí určitou posloupnost znaků na konci kmene slova na znaky jiné a případně provede zkrácení délky daného slova. Příklad změny takovéhoto slova je uveden v následující ukázce zdrojového kódu.

```
switch(s[len - 1]){
    case 'c':
    case 'č':
        s[len - 1] = 'k';
        return len;
    case 'z':
    case 'ž':
        s[len - 1] = 'h';
        return len;
}
```

5.10 Vyhledávání v textu

Jak již bylo zmíněno pro vyhledávání v textu se využívá techniky `stemming`. Třídou, která tuto techniku využívá a zajišťuje tak tedy samotné vyhledávání v aplikaci, je třída `TextSearch`. Tato třída je využívána vždy, pokud aplikace potřebuje vyhledávat v textu. Na základě metod této třídy je aplikace schopna rozhodnout, zda se hledaný text ve zdrojovém textu nachází či nikoliv. Druhou funkcí této třídy je také samotné vyhledání výskytů hledaného textu.

Návrat jednotlivých výskytů hledaného textu je proveden prostřednictvím seznamu objektů typu `TextTimer` 4.3.

5.11 Návrh komunikace se serverem

Aplikace Přednášky na cesty se bude v budoucnu připojovat k serveru, který je vytvářen v rámci Diplomové práce. Uživatel odešle prostřednictvím aplikace serveru přihlašovací údaje a server je následně ověří oproti školním serverům Vysokého učení technického. V případě, že byly přihlašovací údaje zadány korektně, uživatel si může stáhnout nabízené přednášky pro pozdější offline prohlížení.

Připojení k serveru

Pokud chce uživatel stáhnout novou přednášku, musí se nejdříve připojit k serveru. K tomu je vyzván, pokud v hlavním menu aplikace zvolí možnost stáhnout přednášku. Po zvolení této možnosti je ověřeno, zda má mobilní zařízení přístup k internetu. V přápadě, že připojení k internetu není k dispozici, je uživatel upozorněnen prostřednictvím `Toastu`[2] a není mu umožněno dále pokračovat v přihlašování. V opačném případě je vyzván prostřednictvím dialogového okna k zadání uživatelských údajů, které jsou odeslány serveru k ověření.

V současné době není přihlašování k serveru ověřováno a uživateli stačí zadat libovolné, neprázdné přihlašovací údaje.

Při úspěšném ověření přihlašovacích údajů obdrží aplikace od serveru token pomocí kterého bude později přistupovat k serveru s požadavky o zaslání příslušných souborů.

Výběr předmětu

Po úspěšném přihlášení se odešle na server `HTTP GET request`, který požádá server o seznam všech předmětů, které mají na serveru k dispozici nějaké přednášky ke stažení. Aplikace obdrží od serveru odkaz ke stažení XML souboru, který tyto informace obsahuje.

[2] http://developer.android.com/reference/android/widget/Toast.html

Takto stažený soubor je poté dočasně umístěn v kořenové složce aplikace a pojmenován `getCourses`. Příklad takového souboru je uveden v příloze B.

Po zpracování staženého souboru s přehledem daných předmětů je uživateli sestavena nabídka prostřednictvím `ListView`, kde si zvolí, z kterého předmětu chce stahovat přednášku.

Výběr přednášky

Pokud uživatel zvolil konkrétní předmět, je serveru opět odeslán `HTTP GET request` s požadavkem na získání XML souboru s jednotlivými přednáškami ke stažení. Aplikace následně obdrží od serveru soubor s informacemi o jednotlivých přednáškách. Tento soubor je také dočasně uložen v kořenovém adresáři aplikace a je pojmenován `getLectures.xml`.

Pro zpracování a sestavení `ListView` s nabídkou jednotlivých přenášek ke stažení je využito objektů `Lecture` 4.3, kde je ukládáno pouze jméno přednášky, délka a odkaz pro stažení XML souboru s informacemi o konkrétní přednášce. Jakmile je XML soubor zpracován, jsou uživateli nabídnuty ke stažení jednotlivé přednášky daného předmětu.

5.12 Stažení přednášky

Princip popsaný v předcházející kapitole je pouze teoretický model, který je v aplikaci částečně implementován. Z důvodu nedostupnosti serveru, který zatím není k dispozici, se pro stažení přednášek využívá API serveru `superlectures.com`. Jednotlivé přednášky jsou nabízeny uživateli stejným způsobem, jaký je popsán v předchozí kapitole s tím rozdílem, že se uživatel nemusí vůči serveru přihlašovat a má k dispozici pouze veřejně dostupné přednášky. Aplikace nezískává seznam předmětů a jednotlivých přednášek pomocí GET requestů, ale využívá API serveru `superlectures.com`.

Samotné stažení přednášky je prováděno po jejím výběru z nabídky sestavené z XML souboru `getLectures.xml`.

Nejdříve je ověřeno, zda se již v úložišti zařízení zvolená přednáška nevyskytuje. Následně je vytvořen adresář ve složce `prednasky`, který má stejné jméno jako název stahované přednášky. V nově vytvořeném adresáři je vytvořen další adresář se jménem `slides`, do kterého budou ukládány jednotlivé stažené slidy.

Jakmile jsou vytvořeny všechny potřebné adresáře, je stažen XML soubor nesoucí informace o dané přednášce a uložen do adresáře, který je pojmenován stejně jako přednáška. Stažený XML je zpracován a je z něj získán odkaz pro stažení samotné přednášky (Podle nastavení aplikace se bude stahovat video ve vyšší kvalitě nebo v kvalitě normální), odkaz pro stažení titulků a odkazy pro stažení jednotlivých slidů.

5.13 Průběh stahování

O stažení konkrétní zvolené přednášky se stará metoda `downloadLecture`, která se nachází ve třídě `LectureDownloader`. Aplikace k tomuto účelu využívá `android.app.downloadManager` a jeho možnosti. Pro stažení XML souboru s informacemi o dané přednášce, přepisu titulek a jednotlivých slidů, se využívá nastavení viditelnosti stahování v notifikační liště na `VISIBILITY_HIDDEN`. Toto nastavení potřebuje oprávnění `DOWNLOAD_WITHOUT_NOTIFICATION` v android manifestu, avšak umožní nám, že pro každý slide, XML soubor a přepis titulků nebude v notifikační liště zobrazena notifikace o průběhu stahování.

Notifikace o průběhu stahování bude zobrazena pouze při stahování samotné přednášky, a to jak při samotném průběhu stahování, tak i při dokončení stahování. Za tímto účelem se nastavuje viditelnost notifikací pro stahování na `VISIBILITY_VISIBLE_NOTIFY_COMPLETED`.

Stahování XML souboru s informacemi o zvolené přednášce se musí provést jako první. Teprve až po jeho stažení aplikace získá odkazy pro stahování dalších souborů. Je potřeba tedy detekovat, kdy je XML soubor stažen. K tomuto účelu je využit `android.content.BroadcastReceiver`. Po stažení XML souboru je zavolána metoda `onComplete`, která provede stažení všech dalších souborů.

Kapitola 6

Testování aplikace

Součástí každého vývoje aplikací, ať už se jedná o aplikace mobilní či například desktopové, je bezpochyby i testování. Ne každý vývojář se však tímto řídí, a tak se často dostávájí na trh, především s mobilními aplikacemi, aplikace, které nejsou vždy plně funkční, případně kompatibilní s verzemi OS Android uvedenými ve své specifikaci prostřednictvím úrovně API 3.3.

Aplikace Přednášky na cesty byla testována několika metodami, které jsou uvedeny v následujících kapitolách.

6.1 Testování jednotlivých úrovní API

Úplně první a nejzákladnější test, který byl potřeba provést, je otestování kompatibility aplikace Přednášky na cesty s jednotlivými úrovněmi API, pro které je aplikace naimplementována. Tyto úrovně jsou uvedeny v android manifestu 3.2 a jejich minimální a maximální hodnota je 13 a 21.

Pro tyto testy byl zvolen emulátor mobilních android zařízení, který poskytuje vývojové prostředí `Android Studio`. Pro testování bylo využito emulace zařízení Nexus 4, 7 a 10, Nexus Galaxy a Nexus One. Pro jednotlivá emulovaná zařízení byla postupně volena konkrétní úroveň API pro otestování všech přípustných kombinací.

Aplikace se na testovaných zařízeních zobrazovala vždy korektně a jednotlivé úrovně API neměly žádný vliv na její funkcionalitu.

6.2 Automatické testy uživatelského rozhraní

Automatické testy uživatelského rozhraní umožňují vývojáři napsat si vlastní sadu testů uživatelského rozhraní, které se později jenom spustí a automaticky se provádí. Případný tester tedy nemusí jednotlivé úkony testovacích scénářů provádět ručně, ale stačí pouze spustit předpřipravené testy.

K tomuto účelu byla využita knihovna `Espresso` 4.2, která využívá pro spuštění svých testů služeb `AndroidJUnitRunner` 4.2. Sada testů uživatelského rozhraní se postupně spouštěla a ověřovalo se, zda aplikace reaguje korektně na zadávané podněty.

Během těchto testů se podařilo odhalit a vyřešit několik chyb spojených s uživatelským rozhraním.

6.3 UI/Application Exerciser Monkey

Tento tzv. "monkey"test se používá pro otestování, zda je aplikace schopna čelit velkému počtu doteků, gest a zadávaných znaků v krátkém časovém úseku. Jedná se o nástroj příkazové řádky, který se spouští společně při začátku debugování aplikace. Tento typ testu je také někdy nazýván jako stresový test aplikace.

Testování touto metodou neodhalilo v aplikaci žádné chyby a lze tedy říci, že aplikace je odolná vůči velkému množství uživatelských vstupů v krátkém časovém intervalu.

6.4 Testování uživateli

Poslední fáze testování spočívala v uvolnění aplikace mezi několik konkrétních subjektů z potencionální cílové skupiny uživatelů. Tato fáze testování má většinou pro vývojáře největší přínos, protože si nepíší testy sami, ale aplikaci testují reální uživatelé, kteří ji budou používat. Uživatelé mohou nad ovládáním aplikace uvažovat jiným způsobem a odhalí tak nedostatky či chyby, které by vývojář odhaloval těžko. Uživatelé testovali aplikaci na svých mobilních zařízeních s různým rozlišením obrazovky a s různými verzemi operačního systému Android. Ve všech testovaných případech se aplikace zobrazovala korektně a byla plně funkční.

Samotné uživatelské testování probíhalo následujícím způsobem. Uživatelé obdrželi od autora seznam úkonů, které mají provést, otestovat tím, zda je aplikace schopna splnit požadované úkony a zda je intuitivní do takové míry, že je uživatel, který aplikaci vidí poprvé v životě, schopen tyto úkony splnit. Seznam úkonů byl následující. Uživatel měl za úkol spustit aplikaci, přihlásit se k serveru a vybrat si libovolnou dostupnou přednášku. Vybranou přednášku měl uživatel za úkol stáhnout do mobilního zařízení. Po úspěšném stažení přednášky měl uživatel za úkol stažený záznam přednášky spustit a vyhledat v něm na základě libovolného slidu přednášky. Součástí uživatelského testování bylo i vyhledávání v přepisech titulků. Uživatel měl za úkol najít v přehrávaném záznamu, kde se v přednášce mluví o frekvenci a zvolit si jeden takový výskyt v nabídce nalezených shod. Na závěr testů měl uživatel za úkol odstranit staženou přednášku z mobilního zařízení.

Uživatelské testy dopadly převážně podle očekávání, a to tak, že uživatelé byli schopni tyto úkony zvládnout bez problémů. Občasný problém, který u některých uživatelů nastal, byl ten, že při přehrávání záznamu přednášky nebyli schopni najít, kde se v právě přehrávané přednášce vyhledává. U jednoho uživatele se objevil i problém s odstraněním přednášky, kde podle něj nebylo dostatečně intuitivní, že je možno vybraný záznam odstranit po dlouhém stisknutí při výběru přednášky.

6.5 Shrnutí a výsledky testování

Jak již bylo řečeno, na testování mobilních aplikací by měl vývojář klást velký důraz. Není to vždy ovšem úplně nejjednodušší záležitost, protože existuje velké mnozstvní mobilních zařízení, které se liší jak rozlišením displaye, tak i verzí operačního systému Android.

V rámci testů, které proběhly na emulovaných zařízeních, byly otestovány veškeré kombinace, které nám tato metoda poskytla. Výsledky dopadly pozitivně a nebyl odhalen žádný problém v zobrazení či funkčnosti.

Testování automatickými testy za pomoci předpřipravených testovacích scénářů sloužilo spíše pro ověření, zda nebyla do aplikace v průběhu jejího vývoje zanesena nějaká chyba.

Stejně tak tomu bylo u testování tzv. "monkey"metodou, kde byla snaha ověřit spíše schopnost aplikace reagovat na velký počet vstupních podnětů v krátkém časovém úseku.

Asi nejdůležitější částí testování je testování na samotných uživatelích. Co se funkcionality týče, nebyla při této fázi testování odhalena žádná závažná chyba. S čím však několik uživatelů spokojeno nebylo je vzhled uživatelského rozhraní. Tento nedostatek již z důvodu pozdního uvolnění aplikace k testování nebylo možné zavčas opravit.

Kapitola 7

Závěr

Bakalářská práce se zabývá návrhem a implementací Android aplikace Přednášky na cesty. Téma tvorby této aplikace jsem si zvolil především z důvodu, že by se mohlo jednat o aplikaci, která bude do budoucna využívána a bude někomu pomáhat.

Vzhledem k tomu, že se jednalo o mnou první vyvíjenou Android aplikaci, značná část přípravy byla využita na učení se principů programování mobilních aplikací. Po osvojení těchto základů následoval logický návrh aplikace. Navrhl jsem základní struktury a třídy, které by aplikace měla využívat a seznamoval se s rozhraním serveru superlecture.com, na který je aplikace nyní napojena.

Původním záměrem aplikace bylo, že se student přihlásí k serveru, který vznikal v rámci diplomové práce za účelem přehrávání záznamů přednášek. Z tohoto serveru se měly po ověření, že k nim má uživatel přístup, stahovat jednotlivé přednášky do mobilního zařízení. Tato část aplikace bohužel nebyla zcela implementována, protože server nebyl v době implementace k dispozici a neměl na své straně podporu pro přihlašování aplikací třetích stran. V současné době je tedy zvolena varianta, že se uživatel připojuje k serveru superlectures.com a má k dispozici pouze veřejně dostupné přednášky.

Aplikace umožňuje uživateli takto stažené přednášky později přehrávat a vyhledávat v nich na základě textu případně jednotlivých slidů dané přednášky.

Z uživatelského testování vyplynul výsledek, že by aplikace do budoucna měla být vylepšena především po straně grafického uživatelského rozhraní. Dalším vhodným rozšířením by mohla být možnost stahování pouze zvukových záznamů přednášek, kompletních slidů přednášky ve formátu PDF, přidání identifikátoru, zda již uživatel danou přednášku shlédl či nikoliv nebo například volba mezi denním a nočním režimem grafického rozhraní.

Aplikace je v současné době publikována a veřejně dostupná prostřednictvím obchodu Google store. Aplikace je zcela zdarma a je volně k dispozici všem uživatelům.

V obchodě Google store je možné aplikaci najít pod názvem `Pocket Lectures`, případně je možnost využít přiloženého QR kódu.

Literatura

[1] Kilián, K.: Android dominuje trhu mobilních telefonů - obsadil 85 [online]. `http://www.svetandroida.cz/android-trh-dominance-201408`, 2014-08-18 [cit. 2015-01-15].

[2] Ljiljana Dolamic and Jacques Savoy: Indexing and stemming approaches for the Czech language. *Information Processing and Management: an International Journal*, ročník 45, 2009: s. 714–720.

[3] Ujbányai, M.: *Programujeme pro Android*. Grada, 2012, iSBN 978-80-247-3995-3.

[4] WWW stránky: Android architecture - The key concepts of android OS [online]. `http://www.android-app-market.com/android-architecture.html`, 2012-02-17 [cit. 2015-01-16].

Příloha A

Obsah CD

Přiložené CD obsahuje

- `src` adresář s kompletními zdrojovými kódy aplikace
- `manual.txt` soubor se základním popisem ovládání aplikace
- `bachelor_thesis.pdf` elektronická verze textu bakalářské práce
- `pnc.apk` instalační soubor aplikace Přednášky na cesty
- `poster.pdf` propagační plakát aplikace
- `video.mp4` prezentační video k bakalářské práci

Příloha B

getCourses.xml

```
<?xml version="1.0" encoding="utf8"?>
<data>
  <category id="1" parent_id="0" name="ISS␣Signály␣a␣systémy" number_of_lectures="0"
      >
    <description></description>
    <text></text>
  </category>
  <category id="2" parent_id="1" name="2013" number_of_lectures="13">
    <description></description>
    <text></text>
  </category>
</data>
```

Příloha C

Plakát