VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV INFORMAČNÍCH SYSTÉMŮ
DEPARTMENT OF INFORMATION SYSTEMS

ANDROID APLIKACE PRO GIT S PODPOROU GIT-LFS A GIT-ANNEX

THESIS TITLE

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — **PETR MAREK**
AUTHOR

VEDOUCÍ PRÁCE — **RNDr. MAREK RYCHLÝ, Ph.D.**
SUPERVISOR

BRNO 2020

Vysoké učení technické v Brně
Fakulta informačních technologií

Ústav informačních systémů (UIFS) Akademický rok 2019/2020

Zadání bakalářské práce

Student: **Marek Petr**
Program: Informační technologie
Název: **Android aplikace pro Git s podporou git-lfs a git-annex**
Android Application for Git with git-lfs and git-annex Support
Kategorie: Uživatelská rozhraní

Zadání:

1. Seznamte se s Git a jeho rozšířeními git-lfs a git-annex. Prozkoumejte existující aplikace (nejen pro operační systém Android) pro ovládání repositářů Git, git-lfs a git-annex, soustřeďte se zejména na aplikace s grafickým uživatelským rozhraním.
2. Navrhněte aplikaci pro operační systém Android, která umožní ovládat Git repositáře s podporou git-lfs a git-annex. Zaměřte se na uživatelskou přívětivost a minimalizaci velikosti repositářů ("shallow clone", ignorování a odstraňování nepotřebných souborů, aj.). Řešte také problémy kompatibility s úložištěm (např. podpora symbolických odkazů).
3. Po konzultaci s vedoucím aplikaci pro operační systém Android implementujte.
4. Řešení otestujte, vyhodnoťte a diskutujte výsledky. Výsledný software publikujte jako open-source.

Literatura:

- Ľuboslav Lacko. Vývoj aplikací pro Android. Computer Press, Brno, 2015. ISBN 978-80-251-4347-6.
- Scott Chacon. Pro Git. CZ.NIC, Praha, 2009. ISBN 978-80-904248-1-4

Pro udělení zápočtu za první semestr je požadováno:

- Body 1, 2 a započatá práce na bodu 3.

Podrobné závazné pokyny pro vypracování práce viz https://www.fit.vut.cz/study/theses/

Vedoucí práce: **Rychlý Marek, RNDr., Ph.D.**
Vedoucí ústavu: Kolář Dušan, doc. Dr. Ing.
Datum zadání: 1. listopadu 2019
Datum odevzdání: 14. května 2020
Datum schválení: 21. října 2019

Abstrakt

Cílem práce je návrh a vývoj aplikace pro zařízení systému android. Tato aplikace umožňuje použití programu Git a jeho rozšíření Git LFS a Git annex, s podporou uživatelského rozhraní. Poslouží zejména vývojářům k usnadnění práce s GIT a velkými soubory. Aplikace tedy předpokládá, že ji budou používat uživatelé obeznámení s tímto verzovacím systémem. Uživatelské rozhraní je tak maximálně transparentní za účelem efektivního řešení problémů vznikajících při použití Git.

Abstract

This thesis aims to design and develop an android application. The application's purpose is to serve Git and its extensions Git LFS and Git annex with the aid of user interface. Its target audience is mainly developers looking for easier work with Git and large files on an android system. Its user interface is therefore designed to provide a transparent environment, which makes collision resolving easier.

Klíčová slova

android, android-studio, git, git-lfs, git-annex, lfs, annex, asynchronní programování, vlákno, proces, room, databáze, křížová kompilace

Keywords

android, android-studio, git, git-lfs, git-annex, lfs, annex, async-task, thread, process, room, database, cross-compilation

Citace

MAREK, Petr. *Android aplikace pro Git s podporou*
git-lfs a git-annex. Brno, 2020. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce RNDr. Marek Rychlý, Ph.D.

Android aplikace pro Git s podporou git-lfs a git-annex

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana X... Další informace mi poskytli... Uvedl jsem všechny literární prameny, publikace a další zdroje, ze kterých jsem čerpal.

.....
Petr Marek
10. března 2020

Poděkování

V této sekci je možno uvést poděkování vedoucímu práce a těm, kteří poskytli odbornou pomoc (externí zadavatel, konzultant apod.).

Obsah

Kapitola 1

Úvod

Trend poslední doby byl a stále je neustálé zvětšování obrazovek zařízení i jejich výkonu. Dostali jsme se již do takové fáze, že je tyto zařízení možné využívat obdobně jako klasické počítače a tak je jejich využití pro synchronizaci souborů na snadě. Vyvíjená aplikace poskytuje systém, který může každý uživatel využít pro svůj účel a svým způsobem. Nejčastěji jet Git využíván programátory pro verzování souborů vyvíjených programů. Rozšíření Git LFS a Git Annex pak pro přidání velkých souborů do těchto repozitářů. Jejich využitím se dosáhne efektivity využití prostoru zařízení a současně plného využití systému Git. Tyto rozšíření lze ale využívat i samostatně. Například pro ukládání videí nebo i jiných velkých souborů na externí úložiště pro pozdější synchronizaci mezi různými zařízeními různých systémů.

Cílem práce je navrhnout, implementovat a otestovat aplikaci určenou pro operační systém Android. Tato aplikace bude uživateli zprostředkovávat Git pro zařízení systému Android formou přívětivého grafického rozhraní. Dále bude implementovat rozšíření *Git LFS* a *Git Annex* za účelem synchronizace velkých souborů a snížení velikosti jejich repozitářů. Aplikace je určena zejména vývojářům a jiným pokročilým uživatelům. Je tedy navržena jako maximálně transparentní při zachování prvků jednoduchého ovládání mobilního zařízení.

1.0.1 Git

Git slouží zejména programátorům k verzování jejich práce, popřípadě jejího sdílení s ostatními členy týmu. Nicméně jeho využití je široké a to zejména při využití rozšíření Git LFS nebo Git Annex, která se zaměřují na práci s velkými soubory.

1.0.2 Git LFS

Git Large File Storage (LFS) nahrazuje velké soubory v repozitářích ukazateli. Samotné soubory jsou pak uloženy na vzdáleném serveru.

1.0.3 Git Annex

Git annex řeší stejný problém jako Git LFS, ale mírně odlišným způsobem. V repozitáři je uložen symbolický odkaz na klíč, který je hash daného souboru. Samotný soubor je pak uložen v adresáři *.git/annex/*. Při změně souboru se mění jen jeho hash a aktualizuje symbolický odkaz. Tímto způsobem je zajištěno šetření místa, jelikož samotný soubor je v repozitáři uložen maximálně jednou.

Kapitola 2

Specifikace řešení

Hlavním cilem aplikace je nabídnout uživatelům řešení pro verzování a synchronizaci souborů jejich Git repozitářů na zařízení systému Android.

2.1 Funkce aplikace

Hlavním účelem aplikace je co nejjednodušší správa Git repozitářů. K tomuto účelu bude mít aplikace dvě základní funkce - správa repozitářů a funkce Gitu.

2.1.1 správa repozitářů

Aplikace bude především umět základní správu repozitářů na zařízení. Nejprve uživatel přidá repozitář. Bude moci přidat již existující lokální repozitář, inicializovat nový nebo klonovat vzdálený. Při otevření repozitáře nad ním může provádět základní funkce Gitu a také některé vybrané funkce již zmíněných rozšíření. V případě úplného ukončení práce s repozitářem ho může přímo z aplikace ze zařízení smazat.

2.2 Cílová skupina

2.3 Průzkum existujících řešení

Během průzkumu již existující aplikací jsem se zaměřil jak na aplikace operačního systému Android, tak na desktopové operační systémy Linux a Windows.

2.3.1 Android

Pro operační systémy Android je trh s řešeními Gitu velice omezený. Existují zde několik málo aplikací s podporou pouze pro čtení repozitáře ale i takové, které zvládají i ostatní základní příkazy Gitu. Jejich popis a mé postřehy z nich se dočtete na následujících řádkách.

MGit

Za zmínku z nich stojí *MGit*. Bohužel neposkytuje podporu pro *Git LFS* ani *Git Annex*. K implementaci funkcí Gitu využívá knihovnu *JGit*. Ta sice v aktuální verzi podporuje *Git LFS*, ale v té, kterou aplikace využívá ji ještě nemá. K jejím přednostem patří otevřený kód a velice intuitivní ovládání. Úvodní obrazovka aplikace se seznam repozitářů. Po kliknutí

na některý se zobrazí obrazovka s jeho detaily. Nalezneme zde prohlížeč jeho souborů, log a status repozitáře. Na této obrazovce se také nachází základní ovládací prvek Gitu aplikace. Jím je drawer, který se vysunuje z pravé strany obrazovky. V něm jsou obsaženy všechny poskytované funkce Gitu. Tedy jeho užívání není při porozumění obecného užívání Gitu nijak náročné. Tato aplikace má integrovaný prohlížeč souborů i jejich editování. Ovšem tento editor není dokonalý. Špatně se v něm posouvá kurzor a navíc nemaže konce řádků. Práce s ním je tedy spíše na obtíž. Naštěstí zde autoři přidali i možnost zvolení vlastního editoru z nainstalovaných aplikací.

Pocket Git

Dále existuje například aplikace *Pocket Git*. Ta je placená a její kód není veřejně přístupný. Využívá integrovaného správce souborů, ale editor již nechává plně na jiných aplikacích. *Pocket Git* má na první pohled přehlednější uživatelské rozhraní. Jednotlivé funkce gitu rozděluje do různých kategorií a vedle souborů přidává ikonku o jeho stavu. Nicméně *Add* a *Commit* jsou natolik integrované do prohlížeče souborů, že jejich správné použití není vůbec intuitivní. Navíc při práci s tou aplikací často narazíte na nejednoznačná chybová hlášení, která neobsahují bližší popis chyby.

Termux

Pro vývojáře upřednostňující příkazový řádek je možnost instalace aplikace *Termux* a nainstalování Gitu do prostředí jeho terminálu. Tam je i možné doinstalovat rozšíření *Git LFS* a *Git Annex*. *Git LFS* lde doinstalovat přímo jako balíček. *Git Annex* je možné stáhnout z `https://GitAnnex.branchable.com/` a dle návodu uvést do provozu. Obě tato rozšíření lze ovládat z příkazové řádky, přičemž *Git Annex* i přes uživatelské rozhraní. To je implementováno v prohlížeči. Tato webová aplikace je přehledná i pro mobilní zařízení a umožňuje synchronizaci souborů mezi repozitáři různých zařízení.

2.3.2 Desktop

GitKraken

Na Linux i Windows existuje mnoho aplikací, které práci s repozitáři zvládají velice dobře. Nicméně prostředí Androidu je od toho desktopového natolik rozdílné, že prostor pro inspiraci je značně omezený. Dobré zkušenosti mám například s aplikací *GitKraken*. Ta zobrazuje repozitář přehledně ve stromové struktuře. V ní lze přímo najetím myši na uzel provádět změny. Funkce Gitu má přehledně zobrazené v horním panelu. Navíc jsou zde dobře řešeny konflikty v souborech. Na jedné straně obrazovky vidíte jednu verzi a na druhé straně druhou. Ve spodní části obrazovky se generuje nová verze. Tu vytváříte postupným procházením obou současných verzí a vybíráním vyhovující varianty. *GitKraken* umí pracovat i s *Git LFS*. K ovládání takto sledovaných souborů používá zvláštní vysouvací nabídku s funkcemi *Git LFS*. Ta se v případě práce s repozitářem podporující toto rozšíření zobrazí vedle základních funkcí. Které soubory takto sleduje lze měnit v nastavení repozitáře nebo při přidávání souborů do stage.

Ungit

Na první pohled dobrým dojmem působí i aplikace *Ungit*. Ta vás při každé akci naviguje krok po kroku a usnadňuje tak používání Gitu pro méně zkušené uživatele. Jedná se o

webovou aplikaci založenou na *node.js*. Pro její instalaci je třeba příkazová řádka, pro spuštění pak webový prohlížeč. Její hlavní výhoda je tedy nezávislost na platformě. Její ovládání je rychlé, jelikož aplikace zjednodušuje určité procedury Gitu. Například sama nabízí *Commit* bez nutnosti přidávát soubory do *Stage*. Nicméně aplikace tím zapouzdřuje většinu funkcí. Na základní obrazovce kromě stromu změn repozitáře není další ovládací prvek a aplikace se tak v konečném důsledku jeví až příliš uzavřeně.

2.3.3 Zhodnocení průzkumu

Z testování aplikací vyplynulo, že nejjednodušší způsob práce s Gitem je tehdy, když aplikace transparentně zobrazuje funkce Gitu a jejich použití nechá na uživateli. Předejde se tím chybám, jejichž hlášení nejsou vždy dostačující k vyřešení problému. Pokud je funkce dobře zpracována, není třeba vést uživatele krok po kroku. Ovládání se tak urychlí a je stále přehledné.

Testované aplikace často využívají vlastní textový editor a správce souborů. V obou případech tyto aplikace integrují velice jednoduché verze a jejich použitelnost je tak značně omezená.

Dalším bodem jsou chybová hlášení. Těm by měla aplikace pokud možno předcházet. Pokud chybě již není vyhnutí, alespoň by měla mít dobrý popis a nebo i návrh jejího řešení.

Kapitola 3

Vývoj aplikací pro systém Android

Kapitola 4

Návrh aplikace

Dle zhodnocení průzkumu a vlastních zkušeností jsem usoudil, že aplikace bude

1. přehledná, ale nebude příliš zapouzdřovat funkce Gitu.
2. uživatele přehledně informovat o tom co právě dělá, co očekává a jaký je výstup.
3. využívat externí správce souborů i textový editor.
4. mít co nejmenší počet aktivit a přechodů mezi nimi.

4.1 Funkce Aplikace

Git je velice komplexní systém a jeho plná implementace by na zařízeních android byla velice nepřehledná. Proto jsem se rozhodl nejprve implementovat pouze základní funkce Gitu tak, aby aplikace byla funkční a bylo možné další funkce postupně přidávat.

4.1.1 Funkce Gitu

Jelikož žádné aplikace pro Android kromě *Termux* neimplementují rozšíření *Git Annex* a nenalezl jsem knihovnu, která by toto dokázala, rozhodl jsem se pro ně využít zkompilované binární soubory. Ty bude aplikace volat pro provedení funkcí Gitu. V případě repozitáře podporujícího *Git LFS* nebo *Git Annex* k nim přibudou i příkazy starající se o soubory sledované těmito rozšířeními. Tyto příkazy budou dostupné při otevření repozitáře v bočním výsuvném panelu aplikace.

4.2 Architektura aplikace

4.2.1 Úvod

Aplikace je psána v jazyce *Java*. Dále využívá návrhového vzoru *Model–view–viewmodel* (dále jen MVVM) a *SQLite* databáze. K přístupu k databázi používá *Room Persistence Library* (dále jen Room). Tato knihovna poskytuje abstraktní vrstvu nad databází *SQLite*, zajišťující robustní přístup při plném využití potenciálu tohoto systému řízení báze dat.

4.2.2 Kotlin vs. Java

Od 7. května 2019 se *Kotlin* stal preferovaným jazykem vývoje pro Android. Proto jsem od začátku plánoval programovat aplikaci právě v něm. Nicméně během programování aplikace jsem zjistil, že naprostá většina zdrojů na internetu pro řešení problémů pro tuto platformu je psána v Javě. Android studio sice umožňuje zkonvertovat kód do Kotlinu, nicméně ani to není to vždy dokonalé. Užití Kotlinu má tu výhodu, že dovoluje programátorovi vynechat určité části kódu, které jsou nutné pro běh aplikace, ale přímo neřeší daný problém. V angličtině se pro ně vžil výraz boiler-plate code. Ovšem tento kód je přesto třeba vygenerovat, ale o to se již stará Kotlin. To je také jeden z důvodů, proč Kotlin trvá déle zkompilovat. Pokročilým Android developerům jistě přijde rychlejí práce vhod, ale jako začínají programátor na této platformě více ocením transparetnost Javy.

4.2.3 Databáze

Databáze je využívána pro získání přehledu o repozitářích Gitu, které chce uživatel aplikací sledovat. Každý takový repozitář je reprezentován tabulkou databáze *repo*. Tato tabulka obsahuje absolutní cestu ke složce repozitáře, vzdálené URL, status repozitáře. Dále uživatelské jméno a heslo ke Git profilu a případně další dodatečné informace. Všechny tyto položky je třeba při provádění příkazů Gitu aktualizovat tak, aby stav tabulky v okamžitém časem odpovídal stavu repozitáře.

4.2.4 Návrhový vzor

Model–view–viewmodel je v době psaní této práce doporučovaným návrhovým vzorem Android aplikací. K volbě tohoto návrhového vzoru dopomohlo také využití knihovny *Room*. Tato knihovna totiž spoléhá na využití *MVVM* vzoru alespoň pro účely funkčnosti databáze. Je tomu tak proto, že data, která zavisí na databázi se ukládají do proměnné datového typu *LiveData*. Hodnotu této proměnné lze sledovat a na jejím základě řídit běh aplikace. Aby byla hodnota této proměnné perzistentní při běhu aplikace, uchovává se její hodnota ve *viewmodelu*. Hlavní takovou proměnnou je v této aplikaci seznam všech Git repozitářů. Ta se nachází ve třídě *RepoRepository*, nicméně její instanciace se provádí pouze v části *view_model* MVVM.

4.3 Aktivity

Aplikace bude mít dvě základní obrazovky a tedy i dvě základní aktivity. První, domovská obrazovka je seznam Git repozitářů. Ten se bude dynamicky měnit podle počtu přidaných repozitářů do aplikace.

4.3.1 Balíčky

Podle zaměření tříd je aplikace dělena do třech základních balíčků. Jsou jimi *java*, *assets*, a *res*. V balíčku *Java* je specifikováno chování aplikace. Balíček *assets* obsahuje soubory nutné pro běh aplikace. V této aplikaci to budou binární spustitelné soubory Gitu. Posledním balíčkem je *res*. Ten obsahuje všechny layouty, ikony a další grafické i textové prvky, které aplikace používá pro grafické rozhraní.

activities

Nejdůležitější součást balíčku *Java* je balíček *activities*. Ten aplikaci dělí na obrazovky a o každou z nich se stará jedna třída. Tedy v případě, že aplikace potřebuje určitou obrazovku, je zavolána příslušná aktivita s jejím chováním. Všechny aktivity aplikace rozšiřují základní aktivitu *BasicAbstractActivity*. Ta implementuje společné prvky rozhraní aktivit. Například získávání oprávnění, zobrazení různých oznámení a dialogů.

adapters

Tento balíček obsahuje třídy, které slouží k zobrazení položek stejného typu. Tato aplikace je využívá k zobrazení seznamu repozitářů základní obrazovky a funkcí Gitu v bočním výsuvném panelu.

database

Tento balíček obsahuje balíček *model*, ve kterém se nachází třída Repo. Ta implementuje tabulku databáze uchovávající všechny potřebné informace o repozitáři. Instance databáze se uchovává ve třidě *RepoDatabase*. K přístupu k ní se využívá třída *RepoDao*. Tato třída obsahuje metody volající *SQLite* dotazy databáze. Aplikaci je databáze zprostředkována třídou *RepoRepository*, která odpovídá *Repository* modelu MVVM.

fragments

Fragmenty, třídy které dynamicky rozšiřují nebo mění obsah aktivity. Aplikace používá fragmenty například k instalaci a nastavení.

installer

O instalaci binárních souborů se stará třída *AssetInstaller* v balíčku *installer*. Ta při prvním spuštění aplikace zkopíruje potřebné soubory ze složky *assets* do interní paměti zařízení. Systém Android podporuje symbolické odkazy. Ovšem při kopírování souboru symbolického odkazu dochází ke kopírování souboru, na který odkaz odkazuje, nikoliv samotného odkazu. Proto je nutné tyto odkazy vytvářet během instalace zvlášť.

tasks

Základní funkce Gitu i jeho rozšíření *Git Annex* a *Git LFS*, jsou implementovány v balíčku *tasks*. Tyto třídy rozšiřují základní třídu *AbstractExecutor* využívající *ProcessBuilder*. Ta spustí binární soubor, který vykoná danou funkci Gitu na zařízení.

utilities

Jedná se o statické metody a proměnné, které jsou použitelné kdekoliv v aplikaci.

view_models

Třídy obsahující perzistentní data a implementující logiku nad nimi. Tento balíček odpovídá části *view_model* MVVM.

4.4 Manipulace se soubory

Průzkumník souborů je možné implementovat různými způsoby s různým stupněm jeho komplexnosti. Proto je pro prohlížení souborů repozitářů využíváno externího aplikace. Uživateli je tak ponechána volnost při jeho volbě a předejde se tak hledání kompromisů pro jeho implementaci.

Jelikož úprava souborů probíhá za použití průzkumníku souborů, je logické, že textový editor je rovněž využit z externích aplikací.

4.5 Grafické uživatelské rozhraní

Jelikož jsem již tušil jakým směrem se bude funkční část aplikace odebírat, rozhodl jsem se vytvořit i základní snímky uživatelského rozhraní. Rozhraní se nejvíce inspiruje aplikací *MGit* a přidává prvky vzniklé z požadavků na aplikaci.

Dále je zde navrhnuto zpracování rozlišení repozitářů, které podporují jedno z rozšíření. V případě, že repozitář bude podporovat *Git LFS* přidají se dané funkce do draweru k ostaním funkcím Gitu. Dále se do nastavení repozitáře *Edit repository* přidá možnost změnit sledované soubory tímto rozšířením podle jejich přípony.

Pokud uživatel bude pracovat s rozšířením *Git Annex*, do možností ovládání repozitáře se přidá tlačítko *Annex Web app*. To spustí webovou aplikaci *git-annexu* a umožní sdílení souborů i tímto způsobem.

Kapitola 5

Implementace

Kapitola 6

Testování

Kapitola 7

Závěr

Literatura