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

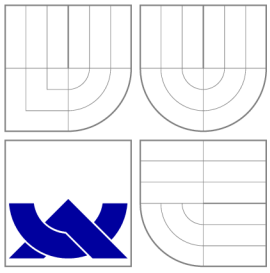
MOBILNÍ APLIKACE PRO MAPOVÁNÍ OPENSTREETMAP V TERÉNU

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

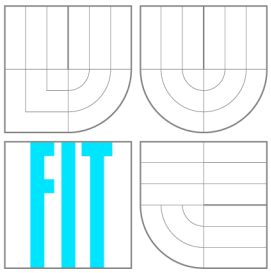
AUTOR PRÁCE
AUTHOR

Bc. MIROSLAV TESAŘ

BRNO 2012



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

MOBILNÍ APLIKACE PRO MAPOVÁNÍ OPENSTREETMAP V TERÉNU

OPENSTREETMAP TERRAIN MAPPING MOBILE APPLICATION

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

Bc. MIROSLAV TESAŘ

VEDOUCÍ PRÁCE
SUPERVISOR

ING. RADEK BARTOŇ

BRNO 2012

Abstrakt

Tato diplomová práce se zabývá tvorbou aplikace pro mobilní zařízení určené k mapování OpenStreetMap v terénu. V rámci teoretického úvodu je popsán projekt OpenStreetMap a způsoby sběru dat. Jsou zde popsána specifika vývoje pro vybrané mobilní platformy. Po důkladnějším seznámení s platformou Android a projektem OSMTracker je popsán návrh, podle kterého probíhá následná implementace rozšíření pro tuto aplikaci.

Abstract

This master's thesis deals with creating OpenStreetMap terrain mapping mobile application. The introduction describes project OpenStreetMap and the methods of collecting data. After that mobile platforms specifics are described. Android and OSMTracker are introduced in the next part of this work. Analysis is followed by description of implementation of OSMTracker extension.

Klíčová slova

OSM, OpenStreetMap, Android, Mapy, GPS, OSMTracker.

Keywords

OSM, OpenStreetMap, Android, Maps, GPS, OSMTracker.

Citace

Miroslav Tesař: Mobilní aplikace pro mapování OpenStreetMap v terénu, diplomová práce, Brno, FIT VUT v Brně, 2012

Mobilní aplikace pro mapování OpenStreetMap v terénu

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně pod vedením Ing. Radka Bartoně.

.....
Miroslav Tesař
23. května 2012

Poděkování

Děkuji panu Ing. Bartoňovi za odbornou pomoc a připomínky při zpracování práce a také děkuji panu Nicolasi Guillauminu za cenné rady při studiu a vývoji rozšíření pro aplikaci OSMTracker.

Obsah

Kapitola 1

Úvod

Dopisy, noviny i knihy v papírové podobě dnes již stále více ustupují jejich elektronické formě, není tedy překvapením, že i mapy lze dnes prohlížet v elektronickém formátu. Podrobné informace o místě, letecké snímky z různých období a pohledů, možnost virtuální procházky po ulicích nebo plánování přesunů během výletu nebylo nikdy jednodušší. Přístup k internetu je dnes narozdíl od doby před rokem 2000 brán jako nedílná součást denního života, téměř každý dnes během dne komunikuje prostřednictvím internetu, vyřizuje pracovní nebo osobní korespondenci, objednává zboží či jen hledá potřebné informace.

Nedílnou součástí lidského života je nabyté informace předávat dál, dělit se o ně. Napříč internetem proto vznikají kanály, jak informace ukládat a sdílet s ostatními, ať formou novinových článků, příspěvky na stále oblíbenějších sociálních sítích, vlastním blogu nebo úpravou internetových encyklopedií. Myšlenka sdílení dat byla základem projektu uživateli vytvářených a spravovaných map sdružené projektem OpenStreetMap.Velkou mírou k vytvoření těchto map přispívají sami uživatelé sběrem informací přímo v terénu pomocí zařízení s vestavěným GPS příjmačem.

Technologie postupuje kupředu mílovými kroky a zatímco ještě před deseti lety se do mobilních telefonů pomalu začaly vkrádat funkce jako fotoaparát či přehrávač multimediálních souborů, v dnešní době jsou tyto funkce zcela běžnou součástí většiny zařízení. Mezi již téměř standardní vybavení mobilních telefonů a stále oblíbenějších tabletů patří příjimač GPS. Využití těchto zařízení se přímo nabízí ke sběru georeferencovaných dat.

Tato práce pojednává o studiu způsobů sběru informací k editaci map projektu OpenStreetMap a vytvoření aplikace pro mobilní zařízení umožňující pohodlný sběr dat. V první části nabídne čtenáři postupné seznámení s projektem OpenStreetMap, jeho historií a možností práce se sesbíranými daty. Představí tři vybrané mobilní platformy, pro které by bylo možné nástroj vytvořit. Porovná jejich slabé a silné stránky a u jedné z nich vybere existující řešení ve formě aplikace OSMTracker, kterou přiblíží a pokusí se identifikovat její silné a slabé stránky.

Následně jsou specifikovány požadavky kladené na aplikaci pro sběr georeferencovaných dat a navrženo rozšíření aplikace odstraňující slabé stránky a podtrhující ty silné. Následně je popsána implementace rozšíření.

V závěru práce je shrnut průběh práce, poznatky během práce nabyté a nastíněno budoucí pokračování projektu aplikace OSMTracker.

Kapitola 2

OpenStreetMap

Projekt OpenStreetMap [2] je iniciativa, jejíž úkolem je vytvoření volně dostupných geografických dat a jejich následná vizualizace do formy silniční mapy, uličních map měst, atd. Tato data jsou tvořena uživateli pro uživatele, projekt je tedy založen na kolektivní spolupráci a na koncepci otevřeného zdrojového kódu, tedy každý z uživatelů může přidávat nebo editovat geografická data. Stručnou historií, způsobem získávání dat a práce s nimi se zabývá tato kapitola.

2.1 Historie

V roce 2004 byl založen projekt OpenStreetMap[1], jeho koncept je velice podobný projektu otevřené encyklopedie wikipedia.org. V roce 2006 byla založena nezisková organizace stejného jména, jejímž cílem bylo a nadále je podporovat projekt a dále jej rozšiřovat. Organizace již od svého vzniku získávala podpůrné prostředky pro tvorbu geografických dat jako např. v roce 2006 společnost Yahoo poskytla své letecké snímky, v roce 2007 byly poskytnuty silniční mapy Nizozemska, Indie, Číny a USA. První z významných institucí, které aktivně využívaly data na svých webových stránkách byla Oxford University.

V roce 2008 byl projekt rozšířen o možnost stahovat data do GPS[2] příjmače a následné používání pro navigaci. V té době firma obdržela finanční podporu od firmy CloudMade, která se zabývala komerčním využitím mapových podkladů a jejich zpřístupňování přes vlastní API[3].

2.2 Sběr dat

Původní data jsou sbírána dobrovolníky pomocí ručních GPS příjmačů pěšky, na kole, autem nebo lodí, zaznamenána a následně před uložením do projektu zpracována v jednom z dostupných editorů na počítači a poté nahrána do databáze OpenStreetMap. Pro snadnější a rychlejší úpravu dat, jsou dostupné např. výše zmíněné letecké snímky nebo mapy cest. Dostupnost těchto dat urychlí zpracování nasbíraných dat a mohou práci výrazně ulehčit. Nejlépe jsou zpracované mapy západní Evropy, kde idea projektu vznikla, a USA.

Datové podklady jsou uvolňovány vládními organizacemi jako veřejná data, např. mezi zdroje za Českou republiku byla zpřístupněna katastrální mapa ČR, ale i komerčními fir-

[1] http://www.openstreetmap.org
[2] Global Positioning System
[3] z angl. Aplication Interface - Aplikační rozhraní

mami, v České republice poskytla úplnou síť silnic 1. a 2. třídy firma HELP SERVICE - REMOTE SENSING.

2.3 Nástroje pro vykreslování a editaci map

Nejjednodušším způsobem pro vykreslení map je mapový server OpenStreetMap využívající aplikaci OpenLayers založenou na konceptu AJAX[4]. Tento server obsahuje dva pohledy na mapu vykreslovanou pomocí systémů, které na požádání nebo opakovaně překreslují mozaiku mapových listů.

Zobrazit mapy umí nejen desktopové aplikace, např. Maperitive[5], ale i aplikace pro mobilní telefony, kde pro každý operační systém existují komerční nebo volně dostupné aplikace, jejichž funkčnost se liší, od prostého zobrazování map až po složitou navigaci terénem.

V dnešní době sběr dat může kromě GPS přístrojů se záznamem trasy provádět i mobilní telefon s GPS příjmačem, operačním systémem a příslušnou aplikací určenou pro sběr dat, kdy jsou následně data exportována.

Základními stavebními prvky, s kterými při zaznamenávání pracujeme jsou:

- body,
- cesty,
- plochy.

Cesty jsou spojnice dvou bodů a plochy se skládají z uzavřených cest. Každému z těchto stavebních prvků je nutné přiřadit vlastnosti(*tagy*), v podobě klíče a jeho hodnoty, kdy prvky s určitou vlastností reprezentují určitý objekt. Bodu je tedy možné přiřadit vlastnost např. `amenity = post_office`, asfaltové dvouproudé silnici první třídy např. `higway = secondary; surface = asphlat; lanes = 2` a ploše lesa například `natural = wood`. Každý z objektů může mít více atributů, které blíže specifikují vlastnosti objektu.

Sběrné aplikace poskytují pouze omezené možnosti, specifikovat bližší vlastnosti, stejně jako provádět úpravu, je nutné již ve specializovaném editoru.

Potlach

Nejjednodušším a zároveň nejpřístupnějším způsobem editace je využití vestavěného webového editoru Potlach přímo na portálu OpenStreetMap. Pro zpřístupnění možností editování je nutná registrace a přihlášení. Poté již přes záložku *Edit* se zpřístupní editovací rozhraní založené na technologii Flash, jehož výhodou je intuitivní ovládání.

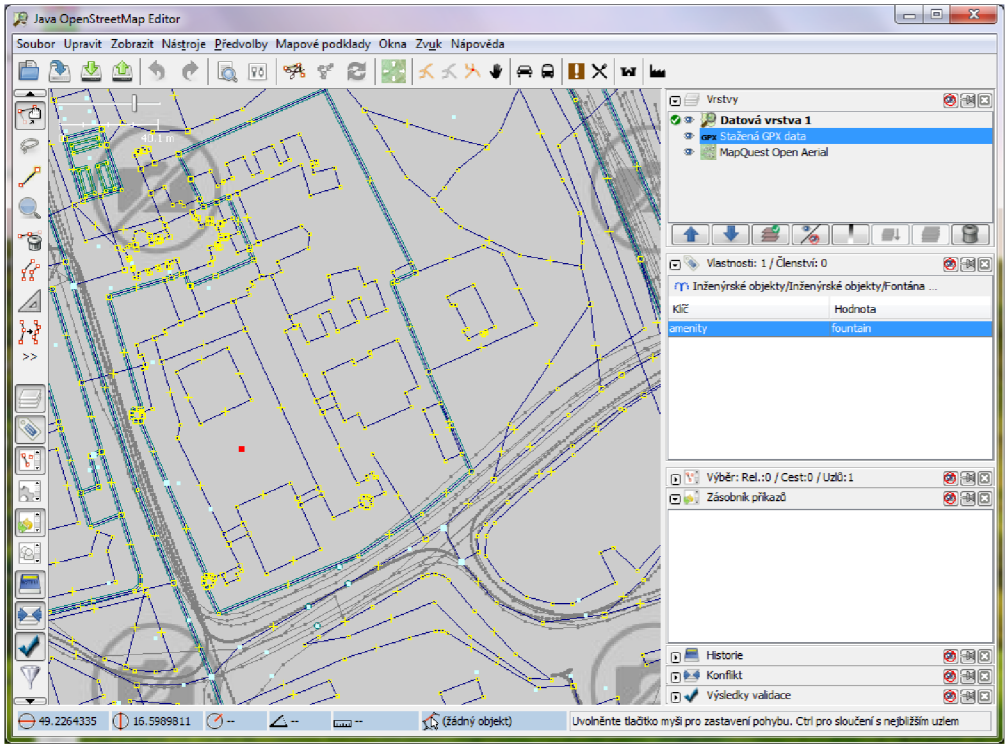
JOSM

Nejpoužívanějším desktopovým editorem je v Javě napsaný a v podobě JAR souboru dodávaný editor JOSM[6]. Jedná se o malý (na disku zabírá řádově MB) offline editor. Což

[4] Asynchronous JavaScript and XML - umožňuje, aby stránka kontatkovala server a obdržela od něj libovolná data v XML bez toho, aby se musela celá znovu nahrávat

[5] http://maperitive.net/

[6] http://josm.openstreetmap.de/



Obrázek 2.1: Editace bodů okolí fakulty.

znamená, že než dojde k editaci je nutné stáhnout data ze serveru. S daty se zde nepracuje online, ale vše, co uživatel upraví, je uloženo lokálně, poté je nutné zeditovaná data nahrát na server. Pokud by docházelo ke konfliktům, editor na toto upozorní a nabídne možnost výběru, které z konfliktních dat uložit. Na obrázku 2.1 z aplikace JOSM je při editaci červeně zvýrazněný bod reprezentující fontánu. Vlastnosti bodu je možné vidět vpravo v prostředním okně.

Kapitola 3

Možnosti mobilních zařízení

Při vývoji aplikací pro mobilní zařízení je prvním krokem prozkoumat, jaké možnosti nabízí. Dříve byl vývoj aplikací zcela v rukou výrobce. S postupem času byly telefony navrhovány tak, aby bylo možné aplikovat plnohodnotný operační systém a umožnit uživatelům vyvíjet vlastní aplikace bez ohledu na výrobce zařízení. V současné době převažují mobilní zařízení (mobilní telefony a stále populárnější tablety) s operačním systémem.

Vybavení mobilních zařízení se také změnilo, zatímco dříve bylo nutné mít např. pro fotografování, navigaci, nebo jen přehrávání hudby několik různých přístrojů různých značek a rozměrů, dnes máme vše v jednom malém zařízení. Pro účely této práce se v této kapitole seznámíme se způsobem získání polohy mobilního zařízení a jakým způsobem lze tyto informace přenést, např. do počítače.

3.1 Přehled mobilních platforem

Prvním krokem je tedy seznámit se s platformami, pro které je možné vyvíjet vlastní aplikace. Při popisu platformy je potřeba zaměřit se na určitá kritéria, např. rozšířenost zařízení s jednotlivými platformami mezi zákazníky, nejčastější vybavení zařízení používající danou platformu, dostupnost vývojářských prostředků, dostupnou dokumentaci, šířku vývojářské základny atd.

Tabulka 3.1 ukazuje rozložení trhu ve třetím čtvrtletí roku 2011 na poli mobilních telefonů zpracována firmou Gatner [9] zabývající se výzkumem.

Pro účely této práce si popíšeme tři platformy, od kterých je v budoucnosti očekáváno, že obsadí, nebo již obsadily první příčky v tabulkách prodejnosti a vývoj pro tyto platformy má určitý potenciál. Platformy popíšeme v následujícím pořadí, firmou Microsoft vyvíjený Windows Phone 7, firmou Apple vyvíjený iOS a Googlem vyvíjený operační systém Android, kterému se budeme záměrně věnovat více, než zbávajícím dvěma.

3.2 Windows Phone 7

První z platforem je Windows Phone 7 vyvinutý firmou Microsoft. Byl vydán 21.října 2010 a je nástupcem operačního systému Windows Mobile. Zaměřuje se více na masy než na podnikovou sféru a přichází s novým uživatelským rozhraním Metro.

Operační systém	Prodaných jednotek	Podíl na trhu %
Android	60 490,4	52,5
Symbian	19 500,1	16,9
iOS	17 295,3	15,0
Research in Motion	12 701,1	11,0
Bada	2 478,5	2,2
Microsoft	1 701,9	1,5
Ostatní	1 018,1	0,9
Celkově	115 185,4	100

Tabulka 3.1: Rozdělení trhu ve třetím čtvrtletí roku 2011 (jednotky v řádech tisíců).

Vývoj aplikací

Pro Windows Phone 7 je možnost vyvýjet aplikace prostřednictvím moderní platformy pro tvorbu bohatých webových aplikací *Silverlight* nebo *XNA frameworku* [11], který se používá např. i pro tvorbu jednodušších her pro Xbox 360 nebo aplikací pro přehrávač Zune. Oproti Windows Mobile, kde bylo potřeba mít placenou edici Visual Studia alespoň ve verzi Professional, jsou vývojové nástroje dostupné zdarma. Obsahují nejen vývojové prostředí Visual Studio Express s nástroji a emulátorem, ale i program pro tvorbu grafiky.

Silverlight

Vychází z desktopové technologie Windows Presentation Foundation, známý především jako plugin do internetových prohlížečů. Podporuje technologie známé z velkého Silverlightu jako například IIS Smooth Shading (kvalita přehrávaného videa se automaticky určí podle rychlosti připojení) atd. Také má přístup k různým součástem telefonu (akcelerometr, fotoaparát, mikrofon, multidotykový displej a další) a hardwarové akceleraci telefonu.

Silverlight se hodí pro psaní především různých pomocných programů nebo aplikací napojených na internet, připojující se ke vzdáleným službám, využívajících lokační služby a napojení na mapy, přehrávající video atd. Při předpokladu, že v budoucnu bude běžnější a hojněji využívané na telefonech neomezenené připojení k internetu, bude tato platforma dobře připravena.

XNA

XNA je vlastně nadstavba DirectX[1], která byla doposud určená k programování pro hudební přehrávače Zune a Zune HD. Nyní od verze 4.0, která se od předchozích částečně liší je možné programovat 2D a 3D hry akcelerované grafickým hardwarem pro Windows Phone 7. Hlavní myšlenkou je zjednodušit práci vývojářům, např. poskytnutí nástrojů pro načítání obrázků, starání se o správné udržování dat v paměti, přehrávání hudby, komunikace po síti nebo počítání s vektory a maticemi. Programuje se pomocí řízených jazyků C# nebo VB.NET. S XNA vytvářet zajímavé hry v krátké době, ale nejedná se o kompletní herní engine a pro větší projekty je nutné znát základy 3D grafiky a např. fyziku, kolize nebo práci s kamerami je již nutné si napsat sami.

[1] sada knihoven poskytujících rozhraní pro ovládání moderního hardwaru

Nabídka aplikací

Aplikace pro Windows Phone 7 jsou nabízeny na Marketplace[2], což zaručuje, že aplikace byly schváleny firmou Microsoft. V současné době jsou nabízeny tisíce různých aplikací nejrůznějších skupin, buď za poplatek nebo zcela zdarma.

Hardwarové vybavení

Firma Microsoft již při vývoji tohoto operačního systému stanovila minimální požadavky na hardwarové vybavení přístroje tak, aby budoucí uživatel měl zajištěno, že při používání zařízení s tímto systémem nebude omezován výkonem nebo funkčností. Mezi těmito požadavky byla například specifikována přítomnost kapacitního multidotikového displeje, senzorů (A-GPS, kompas, zrychlení, přiblížení), alespoň pětimegapixelový fotoaparát, minimální vestavěná a uživatelská pamět a další.

3.3 iOS

Mobilní operační systém iOS byl určený původně pouze pro mobilní telefony iPhone. Následně se rozšířil do dalších zařízení jako iPod Touch, iPad je vytvořený firmou Apple Inc. původně pod názvem iPhone OS. Název iOS se používá nově až od čtvrté verze systému a je v souladu s pojmenováním dalších produktů společnosti Apple Inc. (iPod, iPad, ...)

Jedná se o systém UNIXového typu. Vychází ze systému používaného v počítačích společnosti Apple Mac OS X, kterému byla pro použití v mobilním zařízení upravena fukncionalita odebráním některých částí a poskytnutí podpory dotykového ovládání.

Vývoj aplikací

Vývoj pro iOS probíhá v aplikaci XCode[8], vývojovém prostředí poskytnutém přímo firmou Apple, které je nabízené sice zdarma, ale dostupné pouze pro operační systém společnosti Mac OS X, takže vývoj v Linuxu či Windows nebyl v minulosti možný. Vznikaly proto projekty, které se pokusily tento projekt řešit kompilováním programů napsaných v jiném jazyce do nativního kódu Objective-C nebo C, v nichž napsané aplikace lze v iOS spoustět, ale byly zakázány v licenčním ujednání a po velké nevoli vývojářů opět povoleny.

Architektura

Systém iOS se skládá ze čtyř základních vrstev, které uživateli poskytují základní funkčnost a vývojářům poskytují API a frameworky[3] nezbytné k vývoji aplikací.

Cocoa Touch

Tato vrstva obsahuje nejdůležitější frameworky při vývoji aplikací. Poskytují nástroje pro implementaci interakce s uživatelem a grafické rozhraní. Je vhodné touto vrstvou začínat pří vývoji aplikací a ostatní používat pouze v případě potřeby. Obsahuje např. frameworky pro multitasking (činnost více procesů současně), který je přístupný až od verze 4.0, šifrování dat, rozpoznávání gest, sdílení souborů, zobrazování map, práci s událostmi, práci se zprávami a další.

[2] http://www.windowsphone.com/cs-CZ/marketplace

[3] softwarová struktura sloužící jako podpora při programování a vývoji

Media layer

Tato vrstva slouží k vývoji aplikací s propracovanou grafikou a zvukem, umožňující přehrávání animací, videí a zvuků.

Poskytuje knihovny pro práci s grafikou, jako jsou technologie pro kreslení 2D vektorů a hardwarově akcelerované vykreslování 2D/3D objektů, pokročilou podporu animací, přístup k obrázkové knihovně uživatele a čtení a zápis většiny rozšířených grafických formátů. Dále poskytuje technologie pro práci se zvukem, přehrávání kvalitních audiozáznamů, používání vibrací, přístup k iTunes a dalším.

Core Services layer

Tato vrstva poskytuje vývojáři nástroje pro práci např. s databází kontaktů uživatele, ukládání strukturovaných dat, práci s řetězci, přístup k audiovideo na nízké úrovni, přístup k informacím o mobilní síti, dostupnosti připojení k internetu a jeho nastavení, událostem v kalendáři, zjištění aktuální polohy uživatele a další.

Core OS

Poslední a nejnižší vrstva poskytuje nízkoúrovňové funkce ostatním technologiím na ní postaveným. Je to rozhraní pro práci s matematickými funkcemi, podpora pro komunikaci s externími zařízeními přes Bluetooth nebo konektor zařízení a navazování komunikace s dostupným příslušenstvím a zaručení bezpečnosti citlivých dat, certifikáty, klíče apod.

Nabídka aplikací

Aplikace pro mobilní zařízení společnosti Apple jsou nabízeny prostřednictvím App Store[4], která nabízí statisíce otestovaných a schválených aplikací, nabízených zdarma nebo za poplatky v řádech jednotek dolarů.

Hardwarové vybavení

Narozdíl od zbylých dvou systémů, iOS je nabízen pouze v zařízeních vyrobených firmou Apple, odpadá tím specifikace hardwarových požadavků na ostatní výrobce. Tento operační systém je navržen přímo na míru zařízení, které firma vyrábí.

3.4 Google Android

V dnešní době (první čtvtletí roku 2012) je více než každý druhý prodaný mobilní telefon s operačním systémem vybaven systémem Android[4]. Jedná se o nejdynamičtěji se rozvíjející operační systém na trhu mobilních zařízení a proto se mu budeme věnovat více. Tato část popisuje historii vývoje, architekturu a postupy při vytváření aplikací pro operační systém Android.

Historie

Na přelomu tisíciletí začala společnost Android Inc. sídlící v Kaliforinii v USA vyvíjet nový operační systém pro mobilní telefony Android. V roce 2005 se společnost Google, bez jakékoliv mediální pozornosti, rozhodla odkoupit firmu Android Inc., což je považováno za

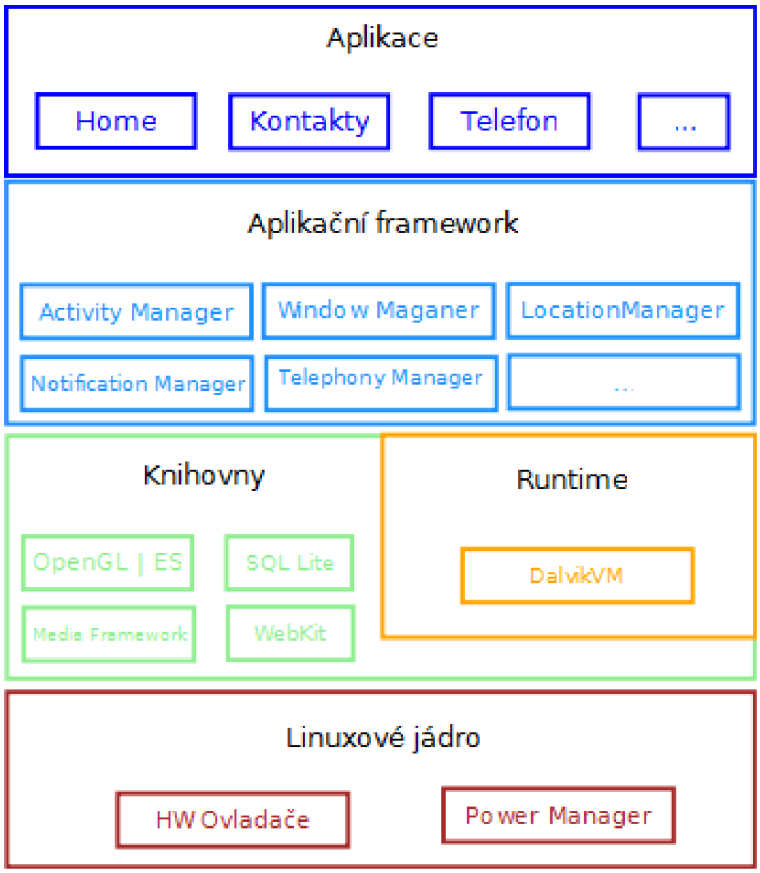
[4] http://www.apple.com/iphone/from-the-app-store/

definitivní rozhodnutí vstoupit na trh s mobilními zařízeními. Vývoj platformy, postavené na stabilním Linuxu, s cílem vytvořit snadno udržovatelný systém, již probíhal pod záštitou společnosti Google. Společnosti jako LG, HTC, Motorola, Samsung, Google, Texas Instruments, NVidia, Intel, Qualcomm a další 5. prosince 2007 ustanovili konsorcium s názvem *Open Handset Alliance* s cílem podílet se na vývoji otevřeného operačního systému pro mobilní telefony. Zároveň představily první společný produkt, systém Android postavený na Linuxovém jádře verze 2.6.

Systém Android (ve verzi 1.0) byl představen 23. září 2008 a poprvé nasazen do telefonu dostupného široké veřejnosti *T-Mobile G1* (HTC Dream). Pod Apache licencí jako *open source* byly v říjnu stejného roku uvolněny zdrojové kódy Androidu.

Architektura

Na obrázku 3.1 je možné vidět schéma struktury platformy Android [3]. Jednotlivé vrstvy si nyní blíže popíšeme.



Obrázek 3.1: Android: schéma architektury.

Aplikace

Systém je dodáván včetně základního jádra s aplikacemi jako jsou např. aplikace pro práci s kontakty, SMS, emailový klient, mapy, prohlížeč, kalendář, aj.

Aplikační Framework

Pokud programátorům nevyhovují standardní aplikace, umožňuje Android psát si vlastní aplikace, které nahradí původně dodávané. Tento proces probíhá na vrstvě aplikačního frameworku, programátor má plný přístup k adresáři kontaktů, sms zprávám, informacím o pozicích, událostem v systému, atd.

Na této vrstvě probíhá také komunikace mezi jednotlivými aplikacemi pomocí systému `Intents`, díky kterému mohou aplikace navzájem spolupracovat. Rozhodne-li se uživatel nahradit výchozí aplikaci vlastní, je tato událost obsloužena zde.

Knihovny

Android obsahuje sadu C/C++ knihoven používajících různé komponenty systému. Některé z nich můžeme vidět níže.

- **Media libraries** - tyto knihovny umožnují práci s multimédii, např s formáty MP3, MPEG4, AAC, AMR, JPG nebo PNG,
- **LibWebCore** - engine prohlížeče webových stránek,
- **SQLLite** - engine pro jednoduchou práci s databázemi.

Runtime

Tato vrstva obsahuje sadu knihoven poskytujících většinu funkcionality programovacího jazyka Java.

Každá spuštěná aplikace na Androidu běží ve vlastním procesu jako instance *Dalvik VM*. Tento virtuální stroj je vytvořen na míru potřebám mobilního operačního systému. Oproti JVM běžícímu na stolních počítačích, je Dalvik VM méně náročný na paměť i výkon. Dalvik VM běží nad linuxovým jádrem, které mu poskytuje nízkoúrovňovou funkcionalitu jako např. správu procesů atd.

Linuxové jádro

Základem Androidu je Linux verze 2.6, který poskytuje systémové služby jako správa paměti, správa procesů, síťová komunikace, bezpečnost a přístup k ovladačům. Jádro lze chápat jako abstraktní vrstvu mezi softwarem a hardwarem zařízení.

Vývoj aplikací

Pro vývoj aplikací je potřeba mít nainstalovaný *Android Software Development Kit (SDK)*. Tento kit obsahuje:

- **emulátor telefonu** - umožňuje uživateli vytvořit virtuální zařízení (AVD) specifikované tak, aby přesně napodobovala zařízení používající operační systém Android,
- **ukázkové zdrojové kódy** - slouží jako demonstrace práce s komponenty systému,

- **různé vývojářské nástroje** - např. nástroj `ddms` sloužící k ladění aplikací, simulování práce s GPS, přístupu k souborům zařízení aj.,
- **knihovny pro překlad aplikace**.

Základním programovacím jazykem pro tvorbu aplikací je zde Java doplněná o určitá specifika. Pro tvorbu většiny aplikací stačí pouze nainstalovat toto SDK, nakonfigurovat své oblíbené IDE (např. plugin pro podporu programování aplikací pro Android v prostředí Eclipse je vyvíjen přímo Googlem) a programátor může začít psát první aplikaci, např. `Hello world`.

Komponenty aplikace

Při tvorbě aplikací pro stolní počitač, kde máme tvorbu domény pod vlastní kontrolou se nestaráme o souběžně puštěné aplikace. Pokud s nimi chceme komunikovat, používáme obvykle nějaké rozhraní.

Systém Android využívá podobné koncepce, ale jinak zabalené a strukturované. Sestává se tedy z těchto komponentů [10]:

- **aktivity** *(Activities)* - stavební kamámen pro tvorbu uživatelského rozhraní, můžeme chápat jako analogii oken či dialogů aplikace pro stolní počítač,
- **dodavatelé obsahu** *(Content Providers)* - zajišťuje úroveň abstrakce dat uložených v zařízení zpřístupněná více aplikacím,
- **služby** *(Services)* - oproti aktivitám a dodavatelům obsahu jsou služby navržené tak, aby pokračovaly ve své práce nezávisle na aktivitě,
- **záměry** *(Intents)* - systémové zprávy upozorňující aplikace na výstkyt událostí, např. události aplikace.

Návrh rozhraní

I když je možné vytvářet grafické uživatelské rozhraní k aktivitě prostřednictvím zdrojového kódu Java, častější je pro definici rozhraní používat na XML[5] založený soubor s návrhem uživatelského rozhraní. Dynamicky vytvářet instance grafického rozhraní je třeba použít v případě, kdy není podoba známa při kompilaci.

V ostatních případech se aktivity navrhují specifikací vlastností a vztahů prvků rozhraní v souboru založeném na XML. Každý XML soubor obsahuje strom elementů prvků grafického rozhraní, které udávají, jak má prvek vypadat nebo jak se má chovat. Součástí SDK systému Android je nástroj `aapt`, který tyto návrhy používá. Nástroj vygeneruje v projektu z XML specifikace rozhraní soubor, který umožní přistupovat k prvkům v návrhu přímo ze zdrojového kódu.

Ačkoliv lze generovat veškeré rozhraní pomocí zdrojového kódu, použití XML návrhů je z důvodu využití nástrojů pro definování rozhraní výhodnější. Existují nástroje, které za pomoci uživatelsky přívětivějšího grafického rozhraní automaticky vygenerují XML soubor s návrhem. Důvodem generování XML kódu je možnost následné úpravy, které je mnohem jednodušší provádět v XML než ve zdrojovém kódu jazyka Java. Současně je výhodou skládání (nebo chcete-li vrstvení) více prvků do komplexnějšího řešení. Pokud požadovaný

[5] eXtensible Markup Language - rozšiřitelný značkovací jazyk

vzhled nelze dosáhnout pomocí primitivních prvků, lze z více jednoduchých prkvů dosáhnout složitějšího a tedy i cíleného.

V XML souboru navrhneme tedy rozhraní, přiřadíme prvkům atributy jako např. písmo, pozice, a unikátní id, pomocí kterého lze potom přímo v kódu přistupovat k vlastnostem prvku a pracovat s nimi. Návrh rozhraní pro jednoduchou aplikaci lze vidět v tabulce 3.2.

```
<?xml version="1.0 "encoding="utf-8 "?>
<TextView xmlns:android="http://schemas.android.com/apk/res/android "
   android:layout_width="fill_parent"
   android:layout_height="wrap_content"
   android:text="Hello World"
   />
```

Tabulka 3.2: Příklad definice zobrazení textu pomocí XML souboru

Nabídka aplikací

Nabídka aplikací pro zařízení vybavené systémem Android jsou uživatelům dostupné přes službu Google Play[6] dříve známou jako Android Market. Vývojáři se jednoduše zaregistrují a zveřejní svojí aplikaci, buď jako zcela zdarma nebo formou placeného obsahu. Uživatelé poté aplikaci mohou po stažení aplikaci hodnotit a komentovat.

Hardwarové vybavení

Od představení první verze operačního systému se mění nároky na hardwarové vybavení. Operační systém nemá předem definované požadavky na hardware a napříč verzemi systému, ale i v závislosti na ceně zařízení, je možné, že se vybavení bude lišit a díky tomuto nelze zclea zajistit funkčnost aplikací na všech zařízeních.

3.5 Způsoby získávání georeferencovaných dat

GPS

Global Positioning System [12], zkráceně GPS je americkou armádou vyvinutý jediný plně funkční globální družicový polohový systém (GNSS)[13] s celosvětovým pokrytím. Konkurenční systém GLONASS vyvíjený společně evropskými a čínskými středisky zabývajícími se družicovými polohovými systémy je plánován uvést do plného provozu po roce 2012.

Určení polohy

V současné době ve výšce více než 20 km obíhá 32 satelitů s dobou oběhu necelých 12 hodin. GPS příjmač získá informace od viditelných satelitů (satelitů v tu chvíli nad obzorem a je na ně přímý výhled z místa, kde se GPS přijímač nachází) ve formě času a identifikační značky satelitu. Na základě znalostí polohy satelitů lze zjistit vzdálenost ke konkrétnímu satelitu. Tím poté okolo každého satelitu vznikne virtuální koule, na jejímž povrchu se někde uživatel nachází. S platným signálem alespoň ze tří satelitů lze určit polohu, pro

[6] https://play.google.com/store

data o nadmořské výšce je nutné mít o satelit více. Obecně tedy platí, čím více satelitů, tím přesnější poloha. O výpočty se uživatel (programátor, aplikace) nemusí starat, veškeré výpočty provádí GPS chip přijímače a předává pouze informace o poloze, nadmořské výšce, rychlosti atd.

Souřadnice

Souřadnice na Zemi se v souřadném systému GPS udávají jako dvě čísla - zeměpisná délka a zeměpisná šířka.

Zeměpisná délka určuje polohu na povrchu směrem na východ nebo na západ od Greenwichského poledníku. Body leží na východ nebo na západ v rozmezí hodnot 0° - 180° na obě strany. Body na západní polokouli mají zeměpisnou délku zápornou, body ležící na východ, tedy na východní polokouli mají zeměpisnou délku kladnou.

Zeměpisnou šířku určujeme v rozsahu 0° - 90° severně nebo jižně od rovníku, kde kladné hodnoty určují body severně od rovnínku a záporné body na jižní polokouli.

Pro zápis souřadnic lze použít několika způsobů.

- *49°13'35.503"N, 16°35'48.124"E* je jeden z nich, kdy N a E určují severní šířku od poledníku a východní délku od Londýnské hvězdárny, a číslo je reprezentováno stupni, minutami a vteřinami,
- *49.2265286N, 16.5967011E* tato hodnota je ekvivalentem výše zmíněného formátu, zde se ovšem používá pouze desetinného čísla,

Formát GPX[1] používá zápis ve formě: *lat = 49.2265286, lon = 16.5967011*, kde „lat“ určuje zeměpisnou šířku a „lon“ zeměpisnou délku. V případě, že se bude jednat o jižní šířku, resp. západní délku budou tyto hodnoty záporné.

Přesnost přijímačů GPS signálu v mobilních zařízeních civilního sektoru je udávána na desítky metrů. S touto přesností je již možné využít mobilní telefony nebo tablety k osobní navigaci, místo automobilové navigace a zobrazování míst na různých typech map, např. Google Maps.

Kapitola 4

Neformální specifikace požadavků

Cílem projektu je umožnit uživatelům sbírání georeferencovaných dat v terénu s využitím mobilního telefonu s GPS příjmačem ve formátu určeném k dalšímu zpracování v dostupných editorech určených k přípravě dat pro aktualizaci nebo vytvoření geografických dat a jejich umístění do volně přístupné databáze OpenStreetMap. S požadavky na aplikaci nás seznámí tato kapitola.

4.1 Požadavky na mobilní telefon

Pro zaznamenávání a sbírání dat je nutné vlastnit mobilní telefon nebo tablet s operačním systémem Google Android ve verzi 1.6 a vyšší, vybavený GPS příjímačem, mobilním připojením k internetu, fotoaparátem, a možností uložit nasbíraná data do vnitřního uložiště přístroje, většinou paměťová karta. Běžně dostupné přístroje prodáváné od roku 2010 tyto požadavky splňují bez výjimky. Mobilní připojení je u telefonů standardem, nicméně ne každý uživatel vlastní datový tarif a s tímto faktem se musí také počítat.

4.2 Požadavky na aplikaci

Předpokladem každé aplikace je zručnost uživatele nainstalovat, spustit a v případě nutnosti s pomocí manuálu aplikaci ovládat aplikaci. Ovládání musí být navrženo tak, aby uživatel i v případě zhoršených podmínek, např. jízda na kole, byl schopen bezpečně a s jistotou aplikaci ovládat.

4.3 Požadavky na funkčnost

Uživatel v mobilním telefonu spustí GPS modul, pokud ne, po spuštění aplikace mu bude tato skutečnost připomenuta a umožněno spuštění GPS modulu. Bez tohoto nebude aplikace schopna dále fungovat a bude uživateli znemožněno další používání některých částí aplikace. Uživatel bude mít zpřístupněno několik částí aplikace. Zásadní čási je sbírání dat. Jak bylo zmíněno v kapitole 2. Jedná se tedy o tyto tři základní prvky:

- body,
- cesty,
- plochy.

Kromě těchto prvků musí být uživateli možno uložit k aktuální pozici fotografii, textovou nebo hlasovou poznámku.

Ovládání aplikace bude prováděno tlačítky na obrazovce nebo hlasem. Uživatel musí také mít přehled o již provedeném záznamu cesty, nejen po jejím dokončení ale i během ní. Uživateli by mělo být umožněno alespoň částečně definovat rozvržení ovládacích prvků tak, aby odpovídalo jeho požadavkům.

Kapitola 5

Analýza specifikovaných požadavků

Pro splnění všech požadavků je nutné prozkoumat všechy možnosti, které se nám nabízí. O vyhodnocování těchto možností a výsledném výběru informuje tato kapitola.

5.1 Výběr platformy

V kapitole 3 jsme se seznámili se třemi mobilními platformami a mezi těmito platformami bylo rozhodováno. V této části bude popsáno, proč byla nakonec vybrána platforma Android.

Systém společnosti Apple má silnou uživatelskou základnu, bohužel, s vývojem pro tuto platformu je to složitější. Pokud chcete vyvíjet pro iOS, je v první řadě nutná registrace zdarma u společnosti Apple, po registraci obdržíte přístup k SDK, vývojovému prostředí XCODE a dalším. Nicméně vývoj je možný pouze na přístrojích firmy. U konvenčních Windows nebo na některé z linuxových distribucí oficiální podpora neexistuje. Problém nastává, pokud bychom chtěli výslednou aplikaci nahrát do zařízení či uvolnit na App Store. Zde je již nutný poplatek pro běžného nefiremního vývojáře $99 na rok. Nutnost dalších investic mluví výrazně v neprospěch iOS.

Windows Phone 7 zatím nenabízí dostatečnou uživatelskou základnu. Nicméně nabízí vysoký potenciál, protože platforma je na trhu zatím krátce a stojí za ní silná a stabilní společnost. Bohužel jedním z cílů práce je nabídnout co nejvíce uživatelům nástroj pro mapování OpenStreetMap, pokud by výsledná aplikace byla úspěšná, tvorba její obdoby pro WP7 bude dalším logickým krokem.

Pro tento projekt byla vybrána platforma Android. K jejímu výběru přispěly stále sílící prodeje mobilních zařízení (telefon nebo tablet), kdy tato platforma je volbou i pro cenově dostupnější telefony a tudíž dosažitelná i pro běžnější uživatele. I tato levnější zařízení splňují požadavky na hardware, a je tedy i částeně výzvou vytvořit dostupné řešení takové, které nabídne univerzální prostředek pro práci s OpenStreetMap. Uživatelé zařízení netvoří pouze malou specifickou komunitu, jako by to bylo např. u WP7 zařízení a jsou otevření novým přístupům. Dalšími faktory bylo programování v jazyce Java a přímá integrace do vývojového prostředí Eclipse.

Zbývající platformy nabízí dle přehledu možnosti k realizaci také, ale při rozhodování byla rozhodujícím kritériem dostupnost, uživatelská rozšířenost a možnosti vývoje.

5.2 Dostupné aplikace

Pro platformy iOS, Android a WP7 existuje nespočet aplikací využívajících OpenStreetMap. Jednotlivé aplikace se dělí podle základních funkcí, které:

- zobrazují mapy,
- navigují,
- zaznamenávájí,
- monitorují,
- editují.

Pří studiu konkurenčních aplikací, byly vybírány aplikace navržené a realizované pro operační systém Android.

Přehled od jednoduchých aplikací, zajišťujících jednotlivé z výše zmíněných funkcí, po složitější s kombinací dvou a více podporovaných vlastností nalezneme na stránkách projektu OpenStreetMap[1], díky kterému je možné si udělat vlastní obraz např. o funkcích jednolivých aplikací.

Při studiu možností dostupných aplikací, bylo zjištěno, že částečnou projektem požadovanou funkčnost nabízí hned několik aplikací a bylo tedy zvažováno, zda se nepřipojit k týmu vývojářů a nabídnout uživatelům pokročilejší funkce.

Připojení se k již probíhajícímu projektu sebou přináší nesčetné množství výhod, jakými jsou například již hotové základní funkce, ale také překážky a omezení, kterými mohou být například návrh aplikace nepočítající s rozšířeními nebo jednoúčelovost celkového řešení, nebo naopak zbytečná robustnost a tím i náročnost systému.

Navzdory i těmto záporům byla zvolena cesta připojením se k vývoji výběrem aplikace OSMTracker, která je podrobně rozebrána v následující části.

5.3 OSMTracker

Aplikace OSMTracker pro Android[7], je inspirována úspěšnou aplikací pro dnes již málo používaný operační systém Windows Mobile. Tato aplikace poskytuje vhodný základ pro vyvíjenou aplikaci. Aplikace je pod licencí GNU (General Public Licence) a je tedy možné upravovat a rožšiřovat funkčnosti aplikace. Než vytvářet vlastní základ a dále jej rozšiřovat, bylo rozhodnuto připojit se k vývojářskému týmu aplikace OSMTracker, navrhout a implementovat rozšíření tak, aby finální podoba co nejvíce splňovala původní požadavky.

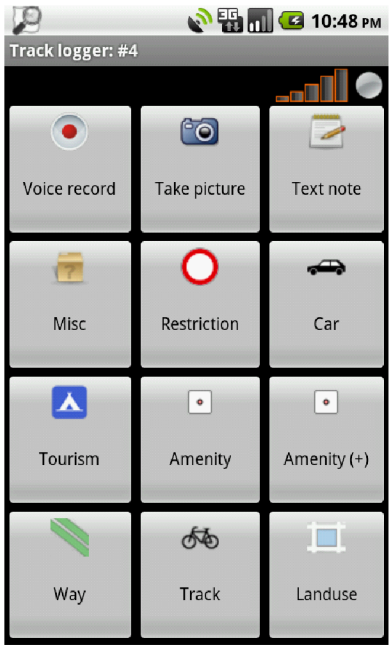
Základní funkce

Aplikace je nástrojem k zaznamenávání cesty a označení bodů na cestě z předem definované množiny tzv. *tagů*. Po spuštění aplikace je uživateli nabídnut správce již zaznamenaných tras, kde je možné s těmito trasami pracovat. Uživatel v této části aplikace má možnost:

- vytvořit nový záznam trasy,
- pokračovat v již existujícím záznamenávání,

[1] http://wiki.openstreetmap.org/wiki/Android

- zobrazit zaznamenanou trasu (bez nebo s mapovými podklady na pozadí),
- exportovat záznamy do GPX,
- zobrazit podrobnosti a smazat.



Obrázek 5.1: OSMTracker: Ukázka aplikace.

Záznam trasy

Zaznamanávání trasy probíhá následovně. Po vytvoření nebo při návratu k záznamu trasy je uživateli zobrazeno zaznamenávací grafické rozhraní. Toto rozhraní má formu dlaždic s předem definovanými funkcemi. Tyto dlaždice jsou uživateli znepřístupněny dokud nejsou nalezeny GPS satelity a podle nich určena poloha. V případě, že je GPS v přístroji zcela vypnuta, je uživatel upozorněn, a je po něm spuštění GPS v mobilním zařízení vyžadováno.

Po nalezení polohy jsou tyto dlaždice uživateli zpřístupněny, a uživatel může pokračovat v zaznamenávání trasy.
Dlaždice mohou být následujících typů:

- `voicerec` - jedná se o povinné tlačítko na každé obrazovce (*layoutu*), slouží k uložení hlasového záznamu k aktuálnímu místu, délka hlasového záznamu je na uživateli v rozmezí 2 - 30 sekund,
- `picture` - další povinné tlačítko na každé obrazovce, které tentokrát k aktuálním souřadnicím umožní uživateli vyfotografovat nebo natočit obrazový záznam a uloží jej do uživatelem definového adresáře na SD kartě přístroje,
- `textnote` - poslední z povinných tlačítek, které ukládá k aktuálnímu bodu textovou poznámku,

- `page` - tlačítko, říkající že pod ním se skrývá nová skupina tlačítek,
- `tag` - poslední z typů, který již označuje zaznamenávání bodu.

Takto zaznamenananou cestu je následně možné exportovat do GPX formátu připraveného k dalšímu zpracování v OSM nástrojích jako například JOSM.

Vlastní definice rozhraní

Typy dlaždic jsou takto rozvrženy záměrně, protože uživateli je umožněna vlastní definice rozhraní vytvořením vlastního XML *eXtended Markup Language* souboru 5.1.

Kořenovým elementem celého grafického rozhraní je element `layouts` uvozující celý dokument. Element `layout` uvozuje jednotlivé obrazovky, tento element má povinný pouze jeden atribut `name` určující jméno.

Jednotlivé řádky určují elementy `row`, jejichž obsahem jsou pouze již elementy `button`, určené povinnými atributy `type`, jejichž hodnoty jsou vypsány výše, v případě hodnot atributu `type="tag"` nebo `type="page"` je poviným atributem `label`, který určuje popis a zároveň zaznamenávanou hodnotu. V druhém případě je navíc povinný atribut `targetlayout`, který určuje, kam se po stisku tlačítka má přejít a zobrazit novou část s dalšími dlaždicemi.

Dalším atributem je `icon`, určující ikonu, zde je v případě uživatelské definice mít ikony uložené ve stejném adresáři jako celý XML soubor. Společně s atributem `iconpos`, který určuje pozici ikony na dlaždici, je nepovinný.

```
<layouts>
   <layout name="root">
      <row>
         <button type="voicerec" icon="voice_32x32"/>
         <button type="picture" icon="camera_32x32"/>
         <button type="textnote" icon="text_32x32"/>
      </row>
      <row>
         <button type="page" label="Way" icon="btn_way" targetlayout="way"/>
         <button type="tag"  label="Bus stop" icon="btn_misc_bus_stop"/>
         ...
      </row>
      <row>
         ...
      </row>
   <layout name="way">
      ...
   </layout>
</layouts>
```

Tabulka 5.1: Příklad XML souboru

Zhodnocení stávajícího stavu aplikace

Aplikace tedy umožňuje získávat podklady k vytváření map, záznamem trasy a exportem do souboru formátu GPX, který je již možné následně zpracovat v některém z OSM nástrojů, jakým může být třeba JOSM.

Silnou stránkou aplikace je možnost definice vlastního uživatelského rozhraní, které umožňuje personalizovat ovládání zcela podle potřeb uživatelů. Současně ale nutnost vytvářet toto rozhraní v textovém editoru počítače, a následný přenos do mobilního zařízení je značně nepohodlný a uživatelé volají po vytvoření pohodlnějšího nástroje, který by nejen umožňoval vytvářet toto rozhraní přímo v mobilním zařízení, ale navíc také dovolil uživateli tato rozhraní spravovat, editovat a upravovat, ne tedy pouze vytvořit a při nutnosti drobných úprav vytvářet rozhraní znovu od začátku a nabídnout tak uživateli větší volnost, například i rozšířenou definicí zaznamenávaných hodnot.

Dalším nedostatkem je nemožnost zaznamenávat s možností zobrazení mapy na pozadí, mohou nastat situace, kdy uživatel bude pouze chtít upřesnit určitá data, a bude pro něj žádoucí, aby měl na obrazovce zařízení možnost vidět mapu, a zároveň vytvořit záznam k určitému bodu.

Kapitola 6

Návrh implementované aplikace

Základní prvky funkčnosti aplikace OSMTracker již splňuje. Tato kapitola popisuje návrh rozšíření zpříjemňujících uživateli práci s aplikací.

Ovládání bude vycházet z konvencí užívaných u zařízení s operačním systémem Android, bude tedy plně dotykové, využívající základní klávesy reprezentované buď hardwarově nebo softwarově a to *Menu* a *Zpět*.

Rožšíření lze rozdělit do dvou základních částí.

6.1 Editace a správa uživatelských rozhraní

Aplikace OSMTracker umožňuje definovat uživatelské rozhraní vytvořením XML souboru v počítači. Toto se jeví jako nešikovný krok. Uživatel je nucen zorientovat se v XML jazyce, což jej odrazuje od plného využití potenciálu aplikace. Tato část tedy popisuje návrh nástroje umožňjícího uživateli pomocí přímého editoru vytvářet a spravovat vlastní uživatelská rozraní. Uživatel nemusí vůbec vědět, že vytváří a zpracovává XML soubor. Základem tohoto rozšíření bude generování XML kodu odpovídajícího rozložení.

Úprava logovaných dat

Pro uživatele je vhodnější, aby místo zaznamenání prostého jednoslovného názvu mohl sám nadefinovat detaily podle potřeby tak, aby následná práce s daty v některém z editorů byla co nejpříjemnější a pro jeho potřeby nejpřesnější. Kvůli tomuto rozšíříme element `button` typu `tag` nebo `layout` o nový atribut `detail` jehož hodnota, bude-li vyplněna, udá, co přesně se zaznamená. Např. fragment 6.1 upřesňuje autobusovou zastávku s lavičkou. Obsah detailu bude zcela na uživateli.

```
<button type="tag" label="Bus stop"detail="highway=bus_stop; bench=yes"
icon= "btn_misc_bus_stop"/>
```

Tabulka 6.1: Nová definice tlačítka s novým atributem detail.

Správa uživatelských rozhraní

Uživateli bude umožněna správa uživatelských rozhraní a to v následujícím rozsahu:

- vytvářet,
- editovat,
- duplikovat,
- mazat.

Uživatelské rozhraní bude jednoduché, řešené pouze výčtovým seznamem aktuálních uložených uživatelem vytvořených souborů obsahující *layouty* (rozhraní). Ty jsou již nyní ukládány na pamětovou kartu zařízení do adresáře, který má uživatel možnost sám si nadefinovat. Tato rozhraní nalezneme v podadresáři `\layouts` ve formě XML souboru, jehož struktura je popsána v kapitole analýzy v tabulce 5.1. S tímto souborem se tedy bude dále pracovat.

Umístění v aplikaci

Přístup ke správě uživatelských rozhraní bude z nabídky nastavení aplikace v části správy uživatelského rozhraní. Zde je již možnost volit vlastní rozhraní, orientaci displeje, zobrazování map na pozadí. Proto se toto umístění jeví jako logické.

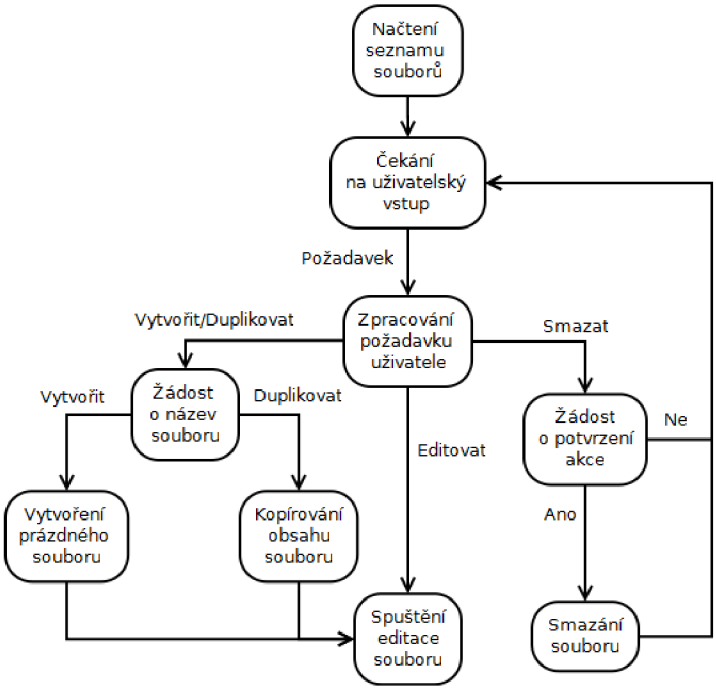
Interakce s uživatelem

Hlavní třída zajišťující interakci s uživatelem, nazvaná `CustomLayoutManager` bude rozšířením třídy `ListActivity`. Grafické rozhraní bude reprezentováno instancí třídy `ListView`, položky budou tvořit všechny XML soubory uložené v uživatelem specifikovaném adresáři.

Editaci grafického rozhraní vyvoláme stiskem položky menu, o jejíž odchycení se stará metoda `onListItemClick()`. Po delším stisku položky se vyvolá kontextové menu, které nabídne možnost vybranou položku zduplikovat, nebo smazat. Při mazání je vyžadováno po uživateli potvrzení jeho rozhodnutí. Zamezíme tím smazání souboru s grafickým rozložením omylem. O práci s kontextovým menu se starají metody `onCreateContextMenu()` volaná vždy při vyvoláním kontextového menu a metoda `onContextItemSelected()`, která zpracovává vybranou položku. Pro přidání vytvoříme položku Menu voleb vyvolaného po stisku tlačítka *Menu* o práci s těmito položkami se starají metody `onCreateOptionsMenu()` a `onOptionsItemSelected()`.

Popis činnosti správce

Vnitřní strukturu činnosti lze vidět na obrázku 6.1. Požadavek je závislý na uživatelském vstupu, způsoby zadávání jsou popsány výše, tedy editace pouhým stiskem položky a přechod do nové editační aktivity. Při vytváření nebo duplikování souboru bude po uživateli vyžadováno vložení názvu nového souboru a poté bude vytvořen prázdný soubor případně do něj nakopírován obsah původního, bude-li se jednat o duplikaci souboru. Po tomto úkonu se opět spustí aktivita editace souboru. Při mazání souboru vybráním, stejně jako v případě duplikace, z kontextového menu bude po uživateli vyžadováno potvrzení o smazání.

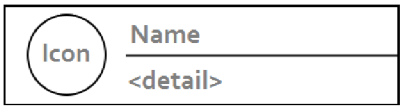


Obrázek 6.1: Popis činnosti správce.

Editor uživatelského rozhraní

Tato část editoru bude koncepčně vycházet z předchozí, umožní tedy uživateli:

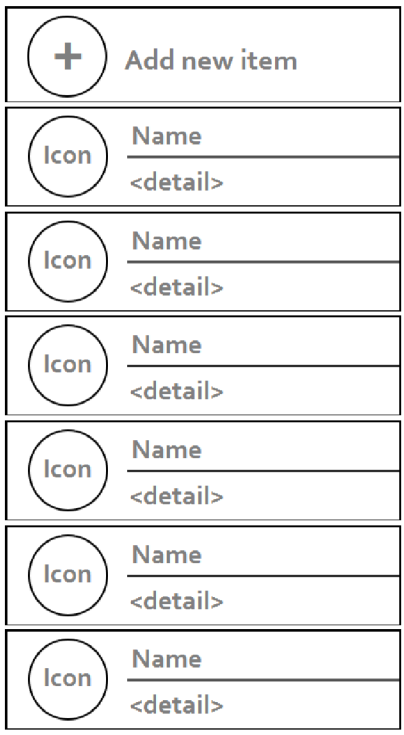
- vytvářet,
- editovat,
- mazat,
- měnit pořadí.



Obrázek 6.2: Položka seznamu.

Základ zobrazení editoru bude opět výčtový seznam, ovšem u položek bude v závislosti typu položky zobrazen kromě názvu i detail a obrázek ikony. Na obrázku 6.2 v levé části je místo pro vybranou ikonu, pod položkou jméno bude uloženo krátké výstižné jméno položky, pod položkoku detailu bude v případě dalšího uspořádání (složky) uložen popis obsahu. V případě konečného tlačítka, zde budou informace, které budou zaznamenány a následně exportovány do GPX souboru s celou cestou. Pro jednodušší přidávání položek bude v horní části, nad seznamem umístěné tlačítko. Ve správci je přidání položky pomocí

posloupnosti Menu - Přidat v pořádku, protože se bude přidávat položka méně často a bylo proto zvoleno toto řešení, v případě většího množství přidáváných položek je lepší ušetřit jeden krok umístěním tohoto ovládacího prvku přímo nad seznam těchto prvků. Výsledný návrh vzhledu lze vidět na obrázku 6.3.



Obrázek 6.3: Návrh zobrazení editoru.

Umístění v aplikaci

Editor uživatelského rozhraní přímo navazuje na správce zmíněného výše. Na obrázku 6.1 je přístup k této části vyobrazen jako blok činnosti Spuštění editace souboru. Jiná možnost dostat se k této části by byla zcela zbytečná a matoucí.

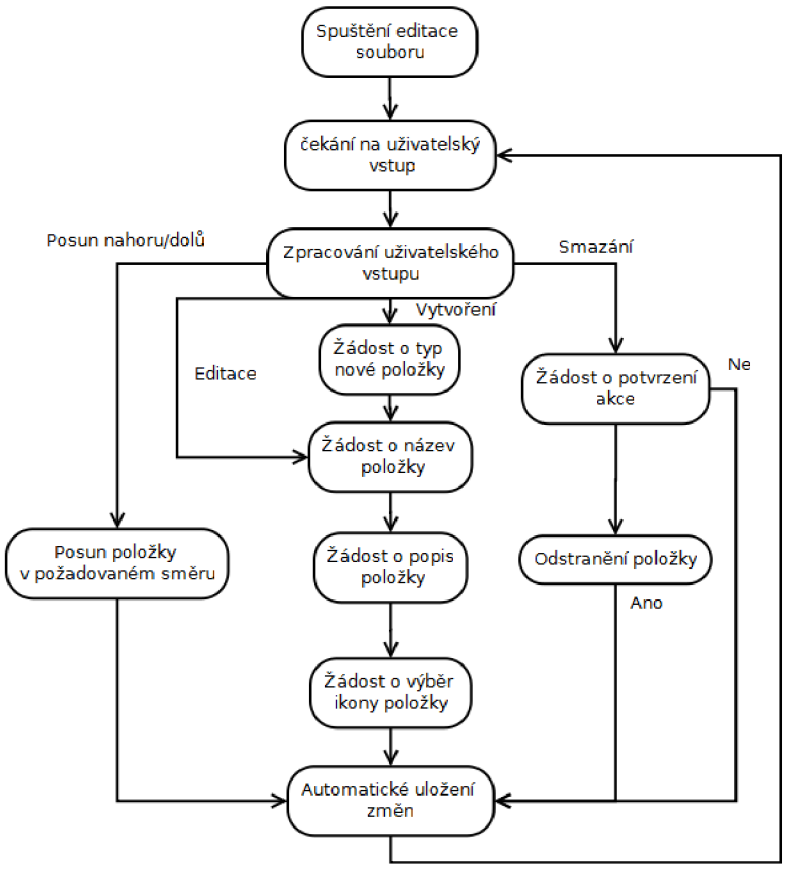
Interakce s uživatelem

Zodpovědnou třídu za editor uživatelských rozhraní nazveme `CustomLayoutEditor` bude opět rozšířením třídy `ListActivity`, seznam bude opět reprezentovat instance třídy `ListView`, pracovat zde budeme pouze s kontextovým menu a je tedy nutné implementovat metody `onCreateContextMenu()` a `onContextItemSelected()`. Tlačítko bude reprezentováno instancí třídy `Button` a pro obsloužení stistku tlačítka musí být implementována metoda `onClick()` o interakci s uživatelem se budou starat instance třídy `AlertDialog`, skrze které bude možné vkládat texty nebo vybírat položky.

Popis činnosti editoru

Strukturu činnosti editoru lze vidět na obrázku 6.4. Po spuštění této části se načtou položky v seznamu pokud již část rozložení (*layout*) položky obsahuje. Po stisku tlačítka pro přidání položky bude uživatel vyzván k výběru typu nové položky. Na výběr mu bude nabídnuto ze dvou možností, typ `tag`, který určuje, že vytvořená položka bude určená k záznamu a typ `layout`, která určuje že tato položka bude obsahovat další složky, nebo položky záznamu. Poté bude dialogy uživatel požádán o název, popis a volbu ikony nové položk.

Při stisku vybrané položky bude uživatel opět vyzván, tentokrát k úpravě výše zmíněných vlastností. Po vyvolání kontextového menu bude uživateli nabídnuto posunout položku o jedno místo nahoru nebo dolů a možnost smazat položku, která bude vyžadovat potvrzení akce smazání.



Obrázek 6.4: Popis činnosti editoru.

6.2 Záznam s mapou na pozadí

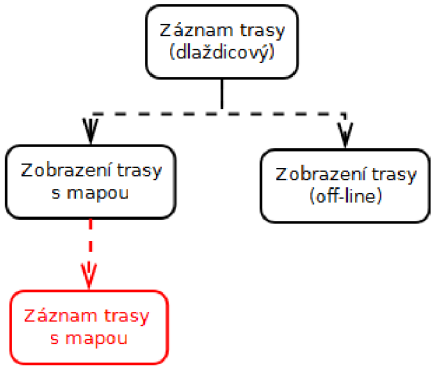
Druhou hlavní částí je zaznamenávání trasy s mapou na pozadí. Aplikace nabízí uživateli možnost zobrazit zaznamenanou trasu. Při připojení mobilního zařízení k internetu umí tuto trasu zobrazit s mapovými podklady v pozadí. Lze tedy vidět kudy jsme se pohybovali.

Dlaždicové uspořádání rozhraní je výhodné v případě, kdy uživatel potřebuje pohodlně zaznamenávat data v případech, kdy má stížené podmínky k ovládání zařízení. Nastane-li situace, že je nutné vidět i co v mapě již zaznamenané je, má uživatel možnost se přepnout k mapovému zobrazení. Poté je nutné se vrátit zpět, a zaznamenat bod. Tato část popíše návrh ovládání tak, aby uživatel nemusel tyto kroky v případě potřeby opakovat. Pouze přepne do režimu mapového zaznamenávání a z této části mu budou zpřístupněny všechny vlastnosti. Součástí této možnosti bude hlasové ovládání, které zaznamená stejná data, jako v případě, kdyby uživatel stiskl příslušné tlačítko.

Základ zobrazení tvoří vrstva mapy, do které se bude zobrazovat zaznamenaná trasa. Nad touto vrstvou bude zobrazena ovládácí vrstva. Tato vrstva bude rozdělena do dvou částí, část bude mít pevnou strukturu, kde budou umístěny prvky přístupné trvale. V druhé části uživateli umožníme nezávisle na původním dlaždicovém uspořádání nadefinovat vlastní specifické prvky ovládání. Protože tato část není známa předem, a může se během užívání aplikace měnit, bude tato část dynamická, a bude pouze na uživateli jaké prvky si zvolí používat, jak si je rozdělí, a které z nově vytvořených rozložení užije pro tento způsob ovládání.

Umístění v hierarchii aplikace

Protože se jedná zatím o doplňkovou možnost (primární zůstává dlaždicově uspořádané rozhraní), přístup k této části bude umožněn pouze z části zobrazení trasy na pozadí map. Uživatel musí mít v nastavení aplikace povolené zobrazování OpenStreetMap pozadí. Dalším předpokladem je přítomnost připojení k internetu. Mapy budou stahovány online, pokud připojení k internetu není přístupné, nebude uživateli přístup umožněn. Schéma přístupu je viditelné na obrázku 6.5.



Obrázek 6.5: Umístění ovládání s mapami na pozadí v hierarchii aplikace.

Uživatelské rozhraní

Uživatelskému rozhraní bude dominovat pozadí s mapou zobrazující zaznamenanou trasu. Rozvržení ovládácích prvků naleznemen na obrázku 6.6. V levé části nalezneme čtyři ovládací ikony reprezentující následující funkce (na obrázku pod čísly 1-4):

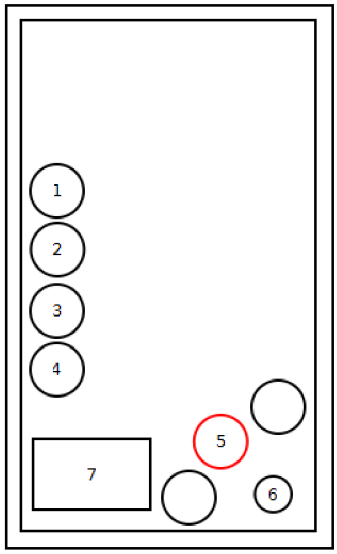
- textová poznámka,

- hlasový záznam,
- obrazový záznam,
- ovládání hlasem.

Textová poznámka, hlasový a obrazový záznam budou fungovat stejně jako v případě dlaždicového uspořádání rozhraní. Ovládání hlasem po stisku vyvolá dialog pro vložení hlasového příkazu, který bude v případě shody proveden.

V pravé spodní části bude část s ovládacími prvky záznamu uspořádaná do čtvrtkruhu. Tento prvek bude jako jediný pohyblivý, ikony se budou posouvat v závislosti pohybu prstu po obrazovce. Aktuální a vybratelný prvek bude vždy na vrcholu (na obrázku barevně odlišený s číslem 5). Vedle kruhového ovládání nalezneme ještě popis aktuálního vybraného ovládacího prvku (7), kde se budou zobrazovat hodnoty názvu a detailu. V pravém dolním rohu (ve středu kruhu) bude ikona sloužící k návratu (6).

Obsah kruhového ovládání bude vycházet z vlastní definice uživatelského rozhraní, bude tedy možné vytvořit ovládací prvky, pomocí vlastního editoru stejně jako v případě dlaždicového ovládání. Způsob ovládádání bude obdobný dlaždicovému uspořádání rozhraní.



Obrázek 6.6: Návrh rozmístění ovládacích prvků.

Prvky interakce s uživatelem

Zodpovědnou třídu nazveme `MapTrackLoggerView` která bude jakousi překrývající prvky přes původní třídu `DisplayTrackMap`. Pro statické ovládací prvky (tedy všechny mimo obsah čtvrtkruhu) budou implementovány metody `onClick()`, které se postarají o provedení příkazů. V případě kruhového ovládání, se uživatel bude pohybovat v seznamu prvků pomocí pohybu prstu nahoru a dolů po obrazovce. Délka pohybu bude úměrná vzdálenosti posunu po obrazovce.

Vzhledem ke kruhovému uspořádání bude pohyb počítán pomocí úhlové vzdálenosti, která zároveň bude určovat o jaký kus se budou pohybující ikony posouvat. Po uvol-

nění prstu z obrazovky musíme počítat se zaokrouhlením pohybu k nejbližšímu pevnému bodu, kde budou prvky zobrazeny. O výpočet nového bodu se bude starat objekt třídy `CircleUtil`. Pro potvrzení výběru bude pro prostřední (aktuální) prvek implementována metoda `onClik()` pomocí které bude zaznamenán stav obdobným způsobem jako při dlaždicovém zaznamenání.

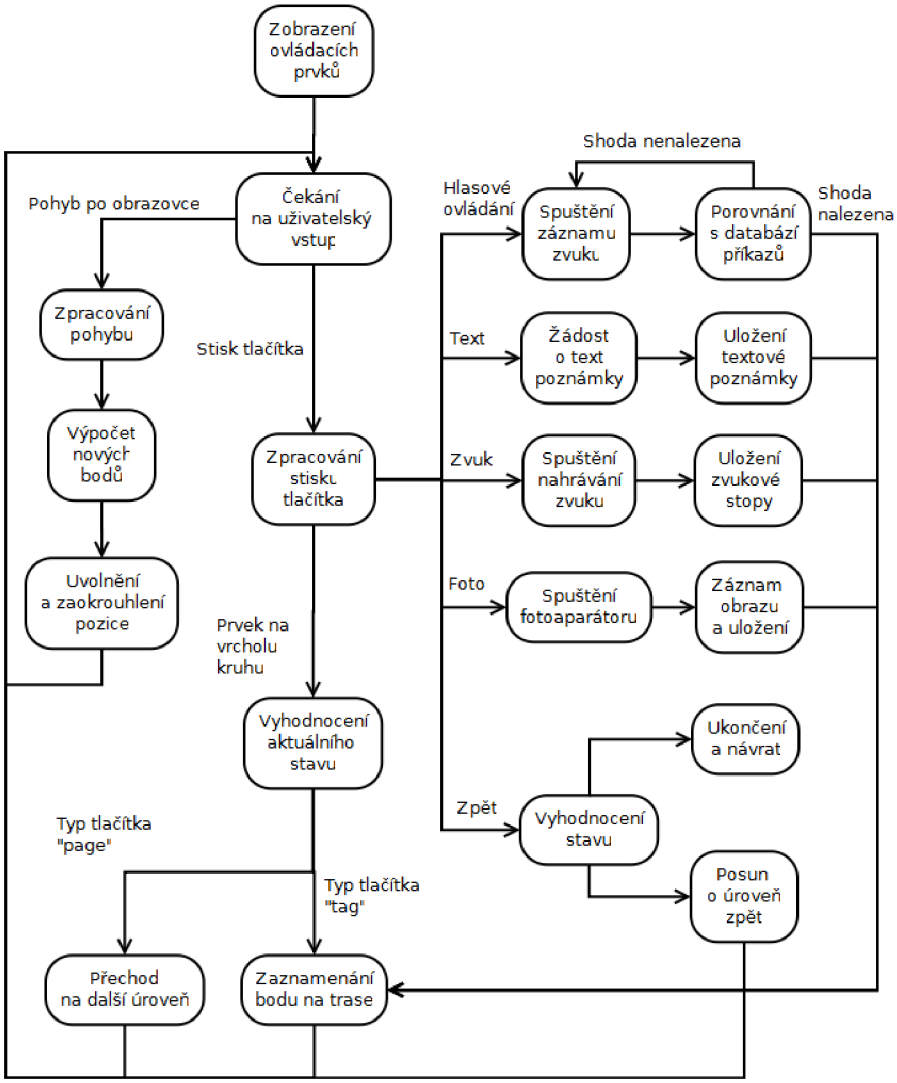
Popis činnosti

Obrázek 6.7 ilustruje činnosti ovládacích prvků na obrazovce během záznamu s mapou na obrazovce. Po stisku ikony ze zobrazení trasy na mapě, se přes aktuální zobrazení překreslí vstva s ovládacími prvky viz. obrázek 6.6. Na obrazovce se zobrazí ikony ve čtvrtkruhu reprezentující uživatelem vybrané rozhraní v nastavení. Jako první bude zobrazen prvek XML souboru `layout="root"`.

Při pohybu prstem po obrazovce se v pravém spodním rohu budou po kruhové trajektorii posouvat ikony. Při pohybu směrem dolů se posunou o úhel svírající počáteční bod (tam kde jsme poprvé položili prst na obrazovku) a koncový bod (kde se prst nachází). Po uvolnění se ikony vrátí do základního postavení v závislosti na úhlu a směru posledního pohybu. Prostřední ikona nyní bude plnit funkci tlačítka, které bude po stisku reagovat v závislosti na typu prvku. Pokud je aktuální prvek, odpovídající elementu `button` typu `page`, přejde se na stránku v XML dokumentu, na kterou odkazuje, pokud je typu `tag`, provede se záznam bodu na trase.

Činnosti po stisku ikon textové nebo zvukové poznámky kopírují stejně jako žádost o záznam obrazu postup, na který jsou uživatelé zvyklí z dlaždicového uspořádání. V případě požadavku hlasového ovládání, se spustí zaznamenávání zvuku, po jeho zaznamenání dojde k vyhodnocení. V případě shody, k provedení příkazu. V případě neshody bude provedeno nové rozpoznávání za účelem nalezení správného příkazu.

Po stisku návratové ikony, v pravém spodním rohu, je v případě základního rozložení (`"root"`) proveden příkaz ukončení zaznamenávání, a návratu pouze do zobrazení trasy. Pokdu jsou ikony v zanoření, provede se návrat o úroveň výše.



Obrázek 6.7: Popis činnosti záznamu s mapou na pozadí.

Kapitola 7

Popis implementace

Následující část se zabývá implementací navržených úprav, postupy a způsoby jejich dosažení a problémy, které vyplynuly během implementace a testování.

Pro vývoj bylo použito vývojové prostředí Eclipse[1], které poskytuje nástroje pro vývoj aplikací pro operační systém Android přímo od společnosti Google. Sdílení zdrojových kódů aplikace probíhá pomocí GIT[5] repozitáře umístěného přímo na GitHubu[2].

Po počátečním seznámení se se strukturou kódu, které probíhalo částečně během fáze analýzy, a postupně hlouběji během implementace, byla započata implementace popsaná návrhem v kapitole 6

Testování probíhalo již během návrhu spoluprací s uživateli formou rozhovorů. Na stránkách projektu jsou shrnuty požadavky uživatelů. Z těchto uživatelů bylo několik individuálně osloveno a společně s autorem projektu byly již během návrhu některé úpravy a přístupy co se týče rozhraní konzultovány. Toto testování probíhalo následně i během implementace.

Původní aplikace je implementována tak, aby ji bylo možné spustit na zařízeních s verzí operačního systému *Android 1.6* a vyšší. Při implementaci některých úprav bylo nutné k tomuto omezení přihlédnout a vyhnout se řešením možným pouze v novějších verzí. Naštěstí tento krok nebyl potřebný u podstatných funkcí ale jednalo se pouze o podpůrné a nebylo tedy nutné dosahovat větších kompromisů. Bohužel některé funkce nebylo možné otestovat na zařízeních s touto verzí operačního systému vzhledem k jeho již malému rozšíření.

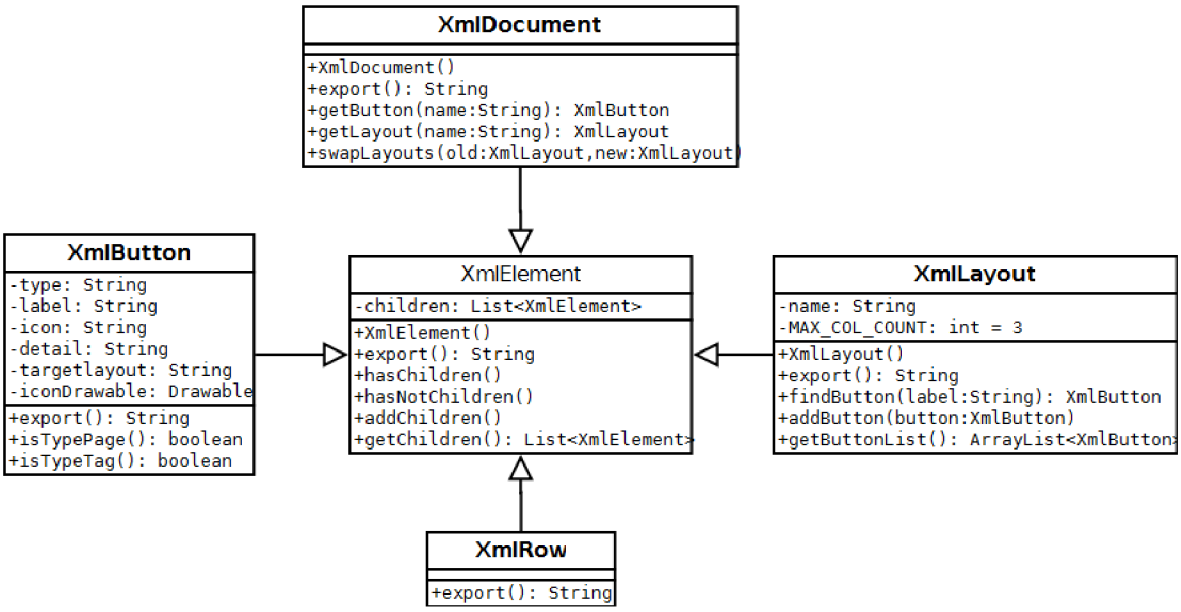
7.1 Reprezentace XML

Prvním krokem, před samotnou implementací bylo nutné vytvořit reprezentaci XML souboru uvnitř aplikace. V původním návrhu aplikace byl obsah (příklad tabulka 5.1) souboru zpracován a přímo uložen jako objekt třídy `HashMap<String, ViewGroup>`, kdy první z dvojice udával název jednotlivých ploch (hodnota atributu `name` elementu `layout`), druhý z dvojice již reprezentuje objekt třídy `DisablableTableLayout`, který má v jednotlivých řádcích tabulky již načtené koncové objekty třídy `Button`, reprezentující jednotlivá tlačítka (obrázek 5.1). V závislosti na situaci a požadavcích uživatele, je vždy zobrazena uživateli jen jedna z požadovaných ploch.

Tato reprezentace je navržena pouze s ohledem na zobrazení, nikoliv na další zpracování a bylo tedy nutné navrhnout vlastní reprezentaci dat v aplikaci. Výsledný diagram tříd může

[1] http://www.eclipse.org/
[2] https://github.com/

čtenář vidět na obrázku 7.1. Nyní si detailněji popíšeme jeho funkčnost.



Obrázek 7.1: Diagram tříd reprezentace XML uvnitř aplikace.

Abstraktní třída `XmlElement` reprezentuje element XML souboru. Jediným jejím atributem je seznam potomků. Abstraktní metodou, a tedy povinnou implementovat v ostatních třídách je metoda `export()`, která slouží při zpětném vytváření XML řetězců, k správnému seskupení potomků a atributů do požadované podoby.

Celý XML dokument reprezentuje třída `XmlDocument` s implemnentovanou metodou `export()`, v níž se doplní příslušné značky a vyvolá se požadavek na výstup potomků. Metody `getButton()` a `getLayout()` vyhledají v celém dokumentu požadované prvky. Potomka této třídy reprezentuje objekt třídy `XmlLayout` jehož atributy `name` a `MAX_COL_COUNT` reprezentují jméno (atribut *name* příslušného elementu) respektive počet sloupců v jednotlivém rozložení. Hodnota 3 sloupců je odvozena z původního návrhu aplikace. Metoda `export()` přidá na začátek a na konec uvozující element *layout* s atributem a jeho hodnotou příslušející prvku. Metody `getButton()` a `addButton()` naleznou a vrátí příslušný prvek resp. přidají nový prvek do posledního řádku. Pokud je řádek plný, vytvoří nový. Metoda `getButtonList()` vytvoří seznam všech příslušných prvků bez ohledu na to, v jakém řádku jsou zařazeny. Tato metoda je používána v případě nutnosti pracovat se seznamy bez ohledu na rozdělení do řádků. Kvůli kompatibilitě s původním návrhem je nutné zachovat i reprezentaci jednotlivých řádků třídou `XmlRow`. Třídou reprezentující koncové prvky s kterými uživatel přijde fyzicky do kontaktu je `XmlButton`. Nově zde přibyly atributy `detail`, který reprezentuje, co bude doopravdy zaznamenáváno, a `iconDrawable` reprezentující obrázek ikony, která je definována atributem `icon`.

Provést realizaci tohoto kroku ještě před samotnými výše navrženými funkcemi bylo z pohledu následujícího vývoje velice důležité, zabránilo se tak komplikacím, které by mohly následně nastat při práci s prvky.

7.2 Správa rozhraní

Na základě návrhu bylo nutné v první řadě zajistit přístup ke správci rozhraní. Na základě návrhu je tento přístup řešen přímo z aktivity nastavení. Grafické rozhraní této aktivity je definováno v XML souboru, bylo proto nutné soubor *preferences.xml* upravit, přidat nové prvky pro vstup do nové aktivity. V průběhu testování vyplynula nutnost, aby uživatel specifikoval, zda bude chtít používat vestavěné ikony pro jednotlivé zaznamenávací prvky (v současné době jich má uživatel k dispozici téměř 90), nebo bude chtít používat vlastní ikony umístěné v adresáři aplikace. Zároveň během testování a připomínek uživateli vyplynulo oddělení nastavení rozhraní pro ovládání původním způsobem a novým s mapou na pozadí.

Pro přechod do nové aktivity byl proto vytvořen element `Preference`, pro použití vlastních ikon element `CheckBoxPreference` a pro výběr nových vlastních rozhraní element `ListPreference`. Zároveň bylo ve třídě nutné ošetřit akce jednotlivýcho nových prvků. V průběhu prvních testování zároveň vyplynula nutnost zajistit existenci veškerých potřebných adresářů již při prvním spuštění nové verze aplikace, která se liší i v dalších menších detailech, které si popíšeme dále v této kapitole.

Správa souborů rozhraní

Po stisku možnosti z nastavení je uživateli nabídnut seznam s obsahem adresáře `layouts` v domovském adresáři aplikace, který uživatel definoval. O práci s touto částí aplikace se stará třída `CustomLayoutManager`, která rozšiřuje třídu `ListActivity`. V aplikacích pro systém Android je pro reprezentaci uživatelského rozhraní využíváno tzv. aktivit. Tyto aktivity je nutné, kromě vytvoření odpovídajících tříd zároveň definovat v souboru `AndroidManifest.xml`, který je povinnou součástí při tvorbě aplikací pro operační systémy Android. Tento soubor deklaruje obsah aplikace a způsoby jakými se aplikace propojí s operačním systémem.

Po nutných úpravách toho souboru je možné nové aktivity bez problémů spustit. Po vytovření aktivity je vyvolána metoda `onCreate()`, v které v našem případě se zde postará o přiřazení definice rozhraní z XML souboru a zavedení součástí jako registraci kontextového menu. Při opuštění aktivity dochází k uložení stavu a při její opětovné aplikaci je volána metoda `onResume()`, zde z nastavení aplikace získáme umístění, kde nalezneme spravované soubory s rozhraními a z tohoto umístění naplníme seznam se soubory pomocí metody `populateLayout()`, která získá z adresáře názvy všech souborů a uloží je do seznamu.

Práce s položkami

Nyní je několik možností, jak s položkami zacházet. Po stisku položky spustíme novou aktivitu editace rozhraní (komunikace mezi aktivitami bude popsaná níže). Stisk položky je obsloužen v metodě `onListItemClick()`, kdy je k výběru důležitá pozice položky, kterou uživatel stiskl.

Kontextové menu, další možnost jak s položkami pracovat obsahuje možnosti editace, duplikace a smazání souborů. Kontextové menu je vyvoláno delším stiskem položky seznamu a o obsluhu stisku jednotlivých položek tohoto menu se stará metoda `onContextItemSelected()`, která na základě identifikátoru vybrané položky spustí aktivitu s editací, v případě výběru editace. V případě výběru duplikace požádá uživatele o vložení jména souboru pomocí dialogu a následně vytvoří přesnou kopii vybraného souboru v adresáři, nedělá se ovšem jen fyzická kopie, ale vstupní soubor se nejprve rozloží a načte do objektu třídy

`XmlDocument` a následně pomocí metody `export()` uloží do nového souboru. Tento postup byl zvolen z několika důvodů. Prvním bylo zachování posloupnosti kroků při tvorbě nových souborů s rozhraním, a druhý důvodem je možnost budoucího rozšíření duplikovaného rozhraní o doplnění dalších informací před uložením do souboru (např. doplnění informací o původním rozložení, nebo o autorovi, atd.). V případě volby smazání je uživatel pouze požádán pomocí dialogu o potvrzení volby smazání.

Vytvoření nového souboru

Vytvoření nového souboru s ovládacími prvky probíhá z menu voleb, vyvolaného tlačítkem *Menu* na přístroji, reprezentovaném buď fyzickou nebo programovou klávesou. Po této volbě je uživatel požádán o název nového souboru pomocí dialogu. Je vytvořen nový soubor obsahující pouze kořenový adresář a uložen.

Předávání informací mezi aktivitami

Při volbě editace nebo vytvoření rozložení je nutné předat z jedné aktivity do druhé informace. Předávání informací mezi aktivitami probíhá vytvořením objektu třídy `Intent` (*záměr*), kdy při vytvoření je jí předán aktuální kontext a název třídy aktivity, kterou chceme vytvořit. Pomocí metody `putExtra()`, která má dva parametry jí můžeme předat další informace. První parametr určuje klíč, a druhý parametr obsahuje samotnou informaci. V tomto případě předáme název vstupního souboru a názvu rozhraní, které chceme aktuálně zobrazit. U prvnotního zobrazení je to vždy element s názvem *root*. Po vložení těchto informací již můžeme aktivitu spustit.

Definice vlastních rozhraní

O správu již vybraného rozhraní, lépe řečeno určité části rozhraní se stará třída `CustomLayoutEditor`, která je jako třída výše rozšířemím `ListActivity` poskytující nástroje k práci se seznamy.

V první řadě je nutné provést kroky k inicializaci základních prvků, nastavit zobrazení objektu třídy `View`. Veškeré definice rozhraní jsou v aplikacích pro operační systémy Android prováděny pomocí XML souborů. V tomto případě se jedná o soubor `cle_adder.xml`, který obsahuje prvky třídy `ListView` a `Button`, pro zobrazení návodu v případě prázdného seznamu je tu prvek třídy `TextView`, jeho obsahem je pouze návod, jak přidat nový prvek.

Po těchto krocích je nutné převzít předávané informace pro další práci s nimi. Následně je nutné načíst do objektů třídy `XmlDocument` obsah předaného souboru a obsah souboru s definicí ikon. O zpracování těchto souborů se stará třída `XmlReaderUtil`, jejíž metoda `parseXml()` obdrží jako parametr soubor s obsahem, nebo objekt třídy `XmlPullParser`, která poskytuje nástroje k zpracování XML souborů. Metoda již vrací objekt třídy `XmlDocument`, obsahující reprezentaci všech elementů vstupního XML souboru. Při tomto zpracovávání jsou zcela ignorovány elementy tlačítek pro záznam zvuku, videa a textové poznámky. Tyto prvky jsou ovšem při exportu dokumentu zpětně doplňovány. Protože jsou prkvy stále stejné a neměnné, je zbytečné je uchovávat v několika kopiích v paměti.

Vytvoření prvku

Po načtení všech potřebných souborů do paměti aplikace, tato aktivita ošetřuje uživatelské vstupy. Nový prvek uživatel přidá po stisku tlačítka v horní části obrazovky. Narozdíl od

správce, který byl popsán výše, přidávání prvků bude častější a proto bylo tlačítko umístěno přímo na obrazovku. U tohoto tlačítka je implementována pouze metoda `onClick()`, kdy po jejím stisku je uživateli pomocí výčtového dialogu nabídnuto, zda chce vytvořit nový prvek dalšího rozložení, nebo zda chce vytvořit už koncový prvek.

V obou případech je po uživateli vyžadováno posloupností dialogů vložení názvu, v případě jen nového prvku rozložení popisu, a v případě koncového tlačítka, detailu, který bude zaznamenán. Následně je uživatel ještě požádán o výběr ikony. Po posloupnosti těchto kroků je prvek uložen, a v případě že se jednalo o novou složku, je spuštěna nová aktivita, které je předán opět název vstupního souboru a název nového rozložení, do kterého budou všechny nové prvky vkládány.

Úprava prvku

Měnit typ prvku není možné, jakmile uživatel vytvoří složku s dalšími rozhraními, už není možné její typ změnit a naopak. Ostatní vlastnosti je možné měnit. Změny se provádí z kontextového menu. Po jeho vyvolání je uživateli nabídnuto co může změnit. Po volbě editace, je postup obdobný jako při vytváření položky, tedy nové jméno nový popis a ikona. Zatímco v případě vytvoření se nová položka zařadí na konec, nyní zůstává pořadí zachováno. Po těchto změnách je opět celý dokument uložen.

Kontextová nabídka uživateli dále umožňuje posouvat aktuální prvek směrem nahoru nebo dolů v hierarchii, posun je možný pouze o jeden krok. Během testování uživateli byla tato funkčnost považována spíše za doplňkovou možnost, a tedy není zcela nutné implementovat posun o více míst v jedom kroku.

Zároveň není z bezpečnostních důvodů uživateli umožněno smazat neprázdnou složku. Uživatel je při takovémto pokusu dialogem pouze upozorněn, že složka není prázdná. Je-li složka prázdná, nebo jedná se o koncový prvek, je dotázán dialogem, zda chce položku opravdu smazat.

Problémy během implementace první části

První otázkou, která nebyla v návrhu řešena byla, jakým způsobem upravená data ukládat. Vyvstával problém, zda nechat uložení všech změn na uživateli, a jak v případě, že se vyskytnou neočekávané problémy (např. nešikovné opuštění aplikace) postupovat. Byly proto vytipovány části, kde uživatel provádí změny a po provedení všech těchto změn, je dokument uložen (proveden export objektu a jeho vložení do souboru). Tímto krokem odpadla povinnost uživatele průběžně ručně ukládat. Je tedy zajištěno, že se uživatel vrátí i po nešikovném opuštění zpět k uloženým datům a není nucen opakovat ztracené úkony.

V první verzi aplikace, která byla předána k testování uživatelům, nebyla možná vlastní definice prvků k zaznamenání. Uživatel byl svázán pouze omezenou množinou, z které si vybíral, a určoval tedy pouze umístění prvku, a ne jeho přesnou podobu. Toto byl hlavní nedostatek, který byl uživateli nejčastěji připomínkován. V současné podobě ovšem aplikace nemůže ohlídat, v jakém formátu bude zaznamenávaný vstup, je tedy na uživateli zda zvolí popis standardními vlastnostmi popsanými v kapitole 2, nebo zda zvolí popis tak, aby byl pro něj srozumitelný (např. souvislá věta v rodném jazyce). Toto řešení se ve finále ovšem nakonec osvědčilo, protože nabízí širší využití aplikace. Jediným omezujícím prvkem aplikace je sada ikon, kterou lze ovšem rozšířit přidáním nových souborů s obrázky a rozšíření vstupního XML souboru, kde jsou ikony seskupeny, aby byl výběr mezi nimi přehlednější.

Výběr příslušné ikony sebou ovšem nesl značné komplikace. Nakonec byl tento krok řešen v souladu s ostatním výběrem dat, a bylo tedy použito klasického dialogu s výběrem ze seznamu. Problém nastal, když bylo nutné kromě názvu ikon přidat i jejich obrázek. Bylo tedy nutné vytvořit metodu, která vrátí objekt třídy `ListAdapter` a přiřadí jej dialogu. Tato metoda se postará o to, že každý prvek zobrazení seznamu je nahrazen podle definice v souboru XML, kde jsou umístěny dva prvky, `TextView`, který reprezentuje název a `ImageView` reprezentující obrázetk ikony, kterou uživatel vybere. Pro objekt třídy `ArrayAdapter`, bylo nutné přepsat metodu `getView`, která upraví zobrazení příslušných prvků. V našem případě textu a obrázku. Výsledek lze vidět na obrázku 7.2.

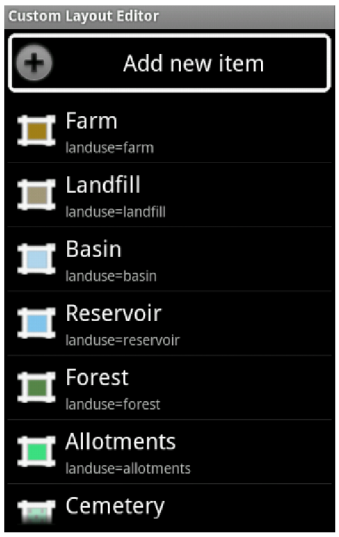
Obrázek 7.2: Nabídka výběru obrázku ikony.

Obdobný způsob řešení byl vybrán i pro zobrazení jednotlivých položek v seznamu. Na obrázku 7.3 přibyly ještě prvky pro zobrazení detailu. Na pravé straně v případě, že se nejedná o typ tagu, ale dalšího rozložení je ještě vidět rozlišení, že se jedná o položku s rozloženími, viz. obrázek 7.4

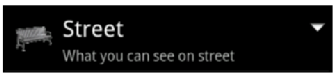
7.3 Implementace logování s mapou na pozadí

Druhou částí implementovanou v rámci této diplomové práce je ovládání s mapovými podklady na pozadí. Omezujícími faktory pro tuto část implementace je nutné připojení k internetu. Bez tohoto připojení není možné zobrazit mapu na pozadí a tato funkčnost bez zobrazené mapy zcela pozbývá na významu. Dalším prvkem, který vyžaduje přítomnost připojení k internetu je rozpoznávání slov.

Vzledem ke složitosti implementace řešení rozpoznávání slov, které by samostatné mohlo vydat na diplomovou práci, bylo zvoleno řešení, které nabízí přímo Android SDK, *Speech input*, který je využíván pro vyhledávání, vkládání textu, atd. Po stisku příslušného tlačítka vyzve uživatele, aby namluvil text, a ten poté zpracuje. Způsob integrace tohoto rozpozná-



Obrázek 7.3: Příklad obrazovky pouze s tagy.



Obrázek 7.4: Zobrazení prvku obsahujícího další obrazovku s tagy

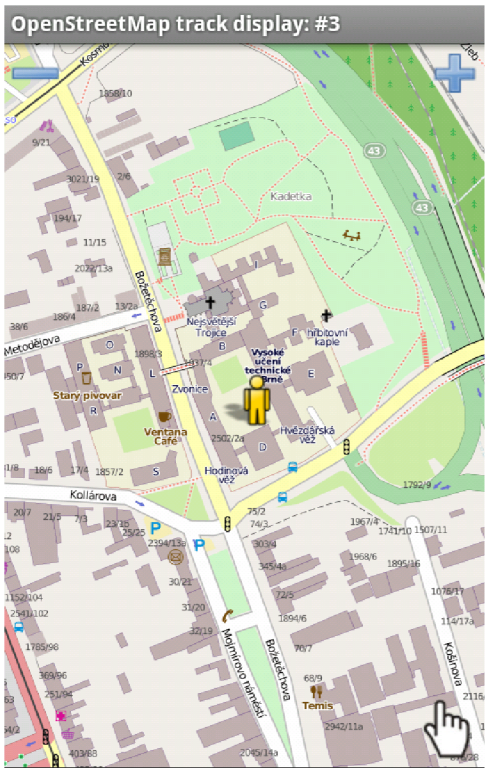
vání za účelem hlasového ovládání bude rozveden později.

Vytvoření vrstvy

O zobrazení trasy se stará aktivita `DisplayTrackMap`, která pro zobrazení trasy využívá volně dostupnou knihovnu *osmdroid*[6]. Tato knihovna poskytuje přístup k mapám, stará se o zobrazení polohy na mapě, a umožňuje uživateli měnit úroveň přiblížení, viz.obrázek 7.5. Toto je základní funkčnost. Pro případ logování s mapou na pozadí, bylo zbytečné vytvářet novou aktivitu, která by z velké části kopírovala obsah stávající. Bylo tedy vhodné zvolit možnost přidání nové vrstvy s ovládacími prvky. Novou vrstvu reprezentuje třída `MapTrackLoggerView`. Základní prvky této vrstvy je možné vidět na obrázku 6.5. Statické prvky jsou definované v soboru `displaycontrol.xml`, který je přenesen do popředí jakmile je tato vrstva pomocí nově přidaného prvku na původním zobrazení trasy vyvolána. Kvůli akcím spojeným s prvky v aktivitě musely být doimplementovány ve třídě aktivity metody `onClick()` pro všechny statické prvky. Jejich funkčnost odpovídá způsobům, které uživatel zná již z klasického dlaždicového uspořádání.

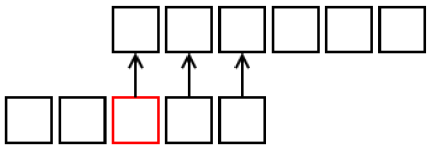
Zobrazení prvků

Základ dynamických prvků tvoří výřez kruhu zobrazující vybranou část z pole prvků aktuální zobrazené vrstvy. Výběr prvků demonstruje obrázek 7.6. Horní část obrázku zobrazuje pole vstupních prvků, Červeně označený spodní prvek určuje středový prvek, který je aktuálně zobrazován. Pokud prvky z vybíraného seznamu nemají svůj předobraz, aplikace se



Obrázek 7.5: Zobrazení místa na mapě.

chová jako kdyby tam nic nebylo. Uživatelům více vyhovovala verze, kdy byl posun tímto způsobem označen jako konec a začátek a nedocházelo k uzavření kruhu. O výběr tohoto podseznamu se stará třída `ImageArrayUtil`. Třída je koncipována tak, aby dokázala pracovat se seznamy obsahující libovolné objekty, a je tedy využitelná i v případě další potřeby.



Obrázek 7.6: Výběr omezeného počtu prvků z pole.

Pohyb mezi prkvy

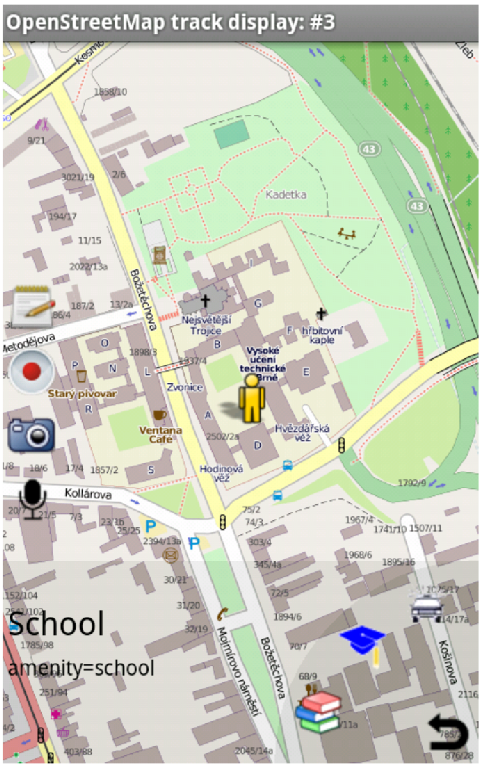
Pohyb prvků po kružnici je přímo závislý na úhlu, který svírají dva vektory tvořené společným bodem v souřadnicovém počátku a body, kde uživatel poprvé přiložil prst na obrazovku, či změnil směr pohybu, a kde se prst momentálně nachází. Úhel těmito vektory sevřený odpovídá úhlu pohybu po kružnici. K posunu aktuálně vybraného prvku dochází vždy, překročí-li úhel stanovený krok, který je poměrem mezi základním rozsahem, v případě čtvrtkruhu je to 90° a celkovým počtem zobrazovaných prvků, v tomto případě obrázků

ikon. K této změně dochází, jakmile prvky opustí svou vymezenou pozici, která je od základního rozmístění polovina poměru na každou stranu. Po uvolnění prstu se prkvy vždy vrátí do základního postavení. O počítání nových pozic na obrazovce se stará třída `CircleUtil`, jejíž metody `getDefaultPoints` a `getPointsWithAngle` počítají pozice prvků na základě počtu zobrazených prvků, kvadrantu kruhu, poloměru, souřadnicích středu a úhlu posunu od počátečních hodnot. Základní funkčnost pro práci s vektory zajišťuje třída `VectorUtil`.

O reakci na pohyb po dotykové ploše zařízení se stará metoda `onTouchEvent`, která dokáže rozpoznat události stisku, uvolnění a pohybu po ploše. O ostatní události(směr, bod) se již stará vnitřní logika, reagující na změnu směru a stisk nebo uvolnění.

Překreslení plátna, které pokrývá celou viditelnou plochu obstarává metoda `onDraw()`, která je vyvolána vždy při zavolání metody `invalidate()`, která způsobí překreslení celého plátna.

Na obrázku 7.7 je možné vidět, přichystanou obrazovku k záznamu, že na daném místě se nalézá škola. Stačí pouze stisknout místo, kde se nachází ikona reprezentující školu.



Obrázek 7.7: Výsledná obrazovka zobrazení logování s mapou na pozadí.

Hlasové ovládání

Původní návrh implementace hlasového ovládání na základě rozpoznávání řeči, musel být kvůli velké míře chybovosti upraven. K optimalizaci postačil algoritmus vyhledávání slov pomocí Hammingovy vzdálenosti dvou slov mezi všemi názvy prvků. Princip Hammingovy vzdálenosti spočívá v porovnání jednotlivých hlásek dvou řetězců, a pokud se tato písmena

liší, je hodnota Hammingovy vzdálenosti zvýšena o 1, čili, slova *pes* a *les* mají Hammingovu vzdálenost 1, zatímco slova *les* a *krb* 3.

Tato optimalizace vznikla vložením podmínky, že bude proveden příkaz, jehož Hammingova vzdálenost s vzorovým slovem je menší nebo rovna 2, tedy že se slova liší pouze ve dvou hláskách. Tento limit se během testování jevil jako plně postačující.

Kapitola 8

Závěr

8.1 Shrnutí

Cílem práce bylo seznámit se s projektem OpenStreetMap, způsoby sběru dat a editace map. Vybrat vhodnou platformu a pro ni vytvořit nástroj, který umožní uživatelům získat pohodlně podrobné podklady, buď pro jeho vlastní potřebu, nebo pro další práci s projektem OpenStreetMap. Byla vybrána již existující aplikace, která do určité míry již tyto požadavky splňovala. Bylo tedy vytvořeno vhodné rozšíření s důrazem na pohodlné ovládání a ponechání na uživateli, jakým způsobem bude s aplikací pracovat.

Tato práce je zprávou o průběhu tvorby rozšíření aplikace OSMTracker. Seznámila čtenáře se teoretickými znalostmi o projektu OpenStreetMap a základy vývoje pro vybrané mobilní platformy. V následujících částech rozebírá stávající stav aplikace, a popisuje návrh a implementaci rozšíření.

Připojit se k již běžícímu vývoji aplikace sebou během práce přineslo mnohé problémy, jako jsou například nutnost pochopení původních záměrů vývojářů, přizpůsobení se celkovému konceptu návrhu aplikace a jejich omezení, která jsou v určitých ohledech značná.

Práce se stávajícími uživateli byla velice přínosná, vytvářet rozšíření již existujícího řešení kladlo velký důraz na zodpovědnost ke stávajícím uživatelům, kteří mají sami zájem, aby aplikace byla použitelná. Jejich přínos k vývoji byl značný. Díky nim se podařilo neomezit ani nezměnit funkčnost, na kterou jsou zvyklí a kvůli které tuto aplikaci do nynějška používali a zároveň jim nabídnout to, co jim v aplikaci chybělo.

V době dokončení této práce aplikace právě prochází procesem revize původním vývojáře, protože vývoj probíhal paralelně, i když na zcela odlišných funkcích, a nastaly drobné kolize. Její uvolnění ostatním uživatelům ke stažení zdarma přes Google Play je očekávána v nejbližších dnech.

8.2 Možnosti budoucího vývoje

Aplikace OSMTracker s novými úpravami nabízí uživatelům možnost, pokud se chtějí správě map věnovat, pohodlnějšího a robustnějšího nástroje, který vznikl uživatelem pro uživatele. I přes prozatím kladné ohlasy testujícími uživateli, je zde možnost dalšího budoucího vývoje.

Implementované prvků bylo testujícími uživateli označeno jako nejvíce vyhovující. Na výběr měli z několika variant. Za úvahu ještě stojí verze pro držení v levé ruce, nicméně, uživatelé označili verzi pro pravoruké jako dostačující. Hodnoty určující rozmístění prvků jsou koncipovány tak, že v budoucnu bude možné nechat na uživateli definovat pozici,

poloměr a počet prvků ve čtvrtkruhu.

Mezi další možné úpravy lze zařadit vytvoření zaznamenávání v režimu, kdy není nutné připojení k internetu. Tedy eliminace hlasového ovládání a vytvoření řešení, kdy budou mapové podklady nahrané přímo v telefonu a přístupné tedy i offline. Tato možnost zde existuje, ale byly by nutné podstatnější zásahy do aplikace a přítomnost mapových podkladů v telefonu by byla náročná na volné místo v uložišti telefonu.

Dalším z možných rozšíření je vytvoření inteligentního rozhraní, které by se uživateli přizpůsobovalo na základě např. rychlosti, kterou se pohybuje (využití by bylo možné např. v automobilu, na kole, nebo ve vlaku), či oblasti v které se nachází (město, les).

Se znalostmi získanými při tomto projektu bude tvorba zaměřením podobné aplikace pro zbývající dvě platformy ulehčena. Další na řadě bude vytvoření aplikace pro operační systém Windows Phone 7. Nicméně stále nedošlo k vyčerpání potenciálu stávající aplikace, a uživatelé během užívání jistě odhalí nové nedostatky, které bude nutné odstranit.

Literatura

[1] GPX - The GPS Exchange Format. [online], 2004 - 2008, [cit. 26.dubna 2012]. URL http://www.topografix.com/gpx.asp

[2] OpenStreetMap. [online], 2011, [cit. 2011-12-30]. URL http://cs.wikipedia.org/wiki/Openstreetmap

[3] What is Android. [online], 2011-12-25, [cit. 2011-12-30]. URL http://developer.android.com/guide/basics/what-is-android.html

[4] Android. [online], 2011-12-28, [cit. 2011-12-30]. URL http://en.wikipedia.org/wiki/Android_%28operating_system%29

[5] Chachon, S.: *Pro Git*. CZ.NIC, 2009, iSBN 978-80-904248-1-4.

[6] Gramlich, N.: OSM Viewer. [online], 2011, [cit. 2011-12-30]. URL http://code.google.com/p/osmdroid/

[7] Guillaumin, N.: OSM Tracker for Android. [online], 2011, [cit. 2011-12-30]. URL http://code.google.com/p/osmtracker-android/

[8] Jeff LaMarche, D. M.: *iPhone SDK*. Computer Press, 2010, iSBN 970-80-251-2820-6.

[9] Gartner Says Sales of Mobile Devices Grew 5.6 Percent in Third Quarter of 2011; Smartphone Sales Increased 42 Percent. [online], 2011-11-15, [cit. 2011-12-30]. URL http://www.gartner.com/it/page.jsp?id=1848514

[10] Murphy, M. L.: *Android 2 Průvodce programováním mobilních aplikací*. Computer Press, 2011, iSBN 978-80-251-3194-7.

[11] Petzold, C.: *Programming Windows Phone 7*. Microsoft Press, 2010, iSBN 978-0-7356-4335-2.

[12] Radek Hojgr, J. S.: *GPS Praktická uživatelská příručka*. Computer Press, 2007, iSBN 80-251-1734-0.

[13] Rapant, P.: *Družicové polohové systémy*. VŠB - TU Ostrava, 2002, iSBN 80-248-0124-8.

Dodatek A

Obsah CD

Obsah přiloženého CD je následující:

- soubor *dp.pdf*
  je textová podoba diplomové práce,
- adresář *OSMTracker*
  obsahuje výslednou aplikaci určenou pro koncového uživatele,
- adresář *source\osmtracker*
  obsahuje zdrojové kódy aplikace v jazyce Java,
- adresář *source\paper*
  obsahuje zdrojové kódy k technické zprávě v jazyce (LaTeX),
- soubor *poster.pdf*
  je plakát vytvořený k aplikaci.