UNIVERZITA KOMENSKÉHO V BRATISLAVE

FAKULTA MATEMATIKY, FYZIKY A INFORMATIKY

Mobilný sprievodca FMFI

Bakalárska práca

2016 **Eduard Princ**

UNIVERZITA KOMENSKÉHO V BRATISLAVE

FAKULTA MATEMATIKY, FYZIKY A INFORMATIKY

Mobilný sprievodca FMFI

Bakalárska práca

Študijný program: Aplikovaná informatika

Študijný odbor: Aplikovaná informatika

Školiace pracovisko: Katedra aplikovanej informatiky

Školiteľ: Ing. Alexander Šimko, Phd.

2016 **Eduard Princ**

Zadanie záverečnej práce

doplniť…

Čestné vyhlásenie

…

Abstrakt

Autor…

Abstract

Author…

Obsah

Úvod

Dnes už asi naozaj každý človek vlastní mobilný telefón, dokonca sa do popredia stále viac a viac dostávajú “smart” zariadenia. Nie len mobily a notebooky, ale aj tablety a niekoľko ďalších. Bežnou súčasťou takýchto zariadení sa stala navigácia. Spočiatku prerazila v automobiloch ako tzv. GPS-ka, potom sa dostala do “smart” zariadení pre lepšie orientovanie sa a urýchlenie cestovania v nie celkom známich miestach.

V mojej bakalárskej práci sa budem venovať navigácií vnútri v budovách, konkrétne v budove našej školy FMFI UK. Tento typ navigácie nieje až taký bežný oproti vonkajšej. Tú takmer úplne ovládol monopol Google so svojou Google Maps. No tá, ako aj ostatné “outdoor-ové” navigácie bývaju vnútri v budovách zbytočné, nielen preto že ,pochopiteľne, nemajú plán budóv a sú nepresné, ale aj preto že nie sú schopné správne rozoznať poschodia budovy. A tak som sa rozhodol spraviť android aplikáciu pre smartfóny, kt. by uľahčila pohyb po našej budove príležitostným navštevníkom, prvákom, alebo každému, kto by potreboval. Osoba by po prihlásení videla, kde v budove sa nachádza, následne by zadala miesto kam sa chce dostať a na základe jednoduchých pokynov by tam bola odnavigovaná. Taktiež sa pri jednotlivých dverách, resp. miestnostiach bude nachádzať info alebo link na web. stránku. Ak pôjde o kabinet tak popis s osobami, ktoré sa tu nachádzajú, alebo ak sa bude jednať o miestnosť, kde sa vybavujú veci tak vskratke popis, čo sa tu dá zistiť.

Táto aplikácia by mohla byť prípadne využiteľná aj pre iné veľké firmy, ktoré majú mnoho zamestnancov a veľké priestory ako napr. automobilky, rôzne fabriky pre výrobu alebo aj verejné štátne budovy v ktorých ľudia bežné hľadajú dvere s nejakým konkrétnym číslom. Dôležité by bolo, aby daná firma alebo inštitúcia dodala plány budovy, nech môžem upraviť a načítať konkrétnu mapu a prispôsobit ďalšie nastavenia.

1. Východiská

1.1 Teoretická časť

V tejto časti sa zameriam na teoretickú časť mojej bakalárskej práce. Hlavnú časť bude tvoriť teória, na základe ktorej budem ďalej vychádzať v implementácií samotnej aplikácií. Na úvod ešte spomeniem a vysvetlím pojmy, ktorých význam je dôležitý pre lepšie orientovanie sa v texte a celkové porozumenie .

platforma - pracovné prostredie, spoločne hardware i software, ktoré umožňuje bezproblémovú činnosť programov. V mojej problematike budem avšak zväčša myslieť software, čiže určitý operačný systém, knižnice, alebo aj použité programovacie jazyky napr. Java

Java - jeden z najpoužívanejších programovacích jazykov, ktorý je multiplatformný, t. j. môže sa používať na viacerých OS od čipových kariet, cez mobilné telefóny až po aplikácie pre desktopové počítače. Od roku 2007 je vyvíjaná ako open source.

open source - je označenie pre počítačový software s voľne prístupným zdrojovým kódom

API -

1.2 GPS, Wifi, Bluetooth

Táto časť bude opisovať ďalšie možnosti, ako by som mohol riešit navigáciu v mojej android aplikácií. Všetky tri majú svoje plusy, no spomeniem mínusy, pre ktoré som sa rozhodol si ich nezvoliť. GPS nie je presná (cca 5-10 m) čize v budovách to je neprijateľné. Mohla by nesprávne rozlíšiť dvere, chodbu alebo tiež aj poschodie, na ktorom sa dotyčná osoba nachádza. Navigácia pomocou Bluetooth snímačov by bola presná, no takéto bohužiaľ nemáme k dispozícií po celej budove. Keby som mal robiť aplikáciu pre nejakú väčšiu firmu určite by som im túto možnosť navrhol. No a napokon Wifi, nad touto možnosťou som najviac rozmýšlal nakoľko wifi sieťou máme pokrytú celu budovu, no obával som sa jednak slepými bodmi, kde by bol signál príliš slabý a taktiež aj problematikou spojenou s wifi ako prechody na druhý wifi route atď. Ako najideálnejšia a najlepšie využiteľná možnosť pre FMFI je QR kód, ktorým dosiahneme úplnú presnosť, malú spotrebu energie aplikácie a mohla by fungovať bez pripojenia na internet.

GPS

Wifi

Bluetooth

1.3 QR kód

QR kód je čiarový kód, v ktorom sú zakódované textové informácie. Najčastejšie sa používajú pre uloženie internetovej adresy alebo kontaktných údajov. Dnes sa už pomocou aplikácií dá vytvoriť vlastný QR kód i prečítať pomocou použitia mobilu s fotoaparátom. Pre porovnanie, na rozdiel od čiarových kódov na potravinách, ktoré majú zakódované 12 cifier, QR kód je schopný uložiť vo svojej najväčšej veľkosti až 4300 znakov. Najmenší QR kód ma veľkosť 21x21 bodov a najväčší 177x177 bodov.

obr. 1

1.3.1 Využitie v mojej bakalárskej práci

QR kód budem používať ako najpresnejšiu a najúčinejšiu metódu pre zistenie polohy vnútri budovy FMFI, aby som maximalizoval použiteľnosť mojej aplikácie. Používateľ aplikácie nasníma QR kód, v ktorom sa bude ukrývať informácia o jeho presnej polohe. Okrem toho, že aplikácia dostane polohu klienta, zobrazia sa mu informácie o daných dveriach, kto sa za nimi nachádza resp. čo tam môže zistiť. Čiže bude slúžiť aj na presnejšie informácie o mieste , kde práve stojí. Ak sa počas chôdze, resp. navigacií k bodu, kde si praje klient ísť, stratí, stačí mu načítať najbližší QR kód a poloha sa opäť update-ne a prepočíta sa najbližšia trasa ešte raz ku cieľu.

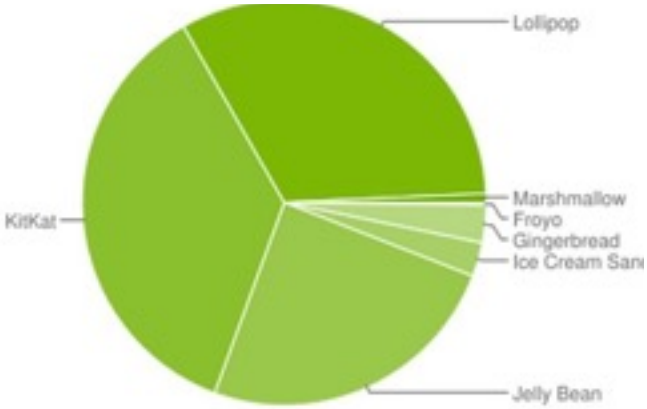
1.4 Android

V roku 2003 vznikla malá spoločnosť Android Inc., ktorú v roku 2005 kúpil Google Android je rozsiahla open source platforma, ktorá zahŕňa v sebe operačný systém založený na Linuxovom jadre, ktorý sa stará o systémové služby ako: bezpečnosť, sieť, ovládače, správa pamäti a procesov. Bola vytvorená hlavne pre mobilné zariadenia ako smartphone, tablet, navigácie, PDA. Pri vývoji tohoto prostredia sa dbalo na rozdielnu veľkosť obrazovky, menší výkon batérie, menej výkonnosti a dostupnej pamäti, a tiež na rôznorodosť hardvéru. Táto flexibilná platforma je stavaná nielen pre

koncových užívateľov ale aj pre vývojarov aplikácií, ktorým poskytuje nástroje pre ich vývoj - SDK (Software Development Kit).

Java Byte Code

Version	Codename	API	Distribution
2.2	Froyo	8	0.2%
2.3.3 - 2.3.7	Gingerbread	10	3.0%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	2.7%
4.1.x	Jelly Bean	16	9.0%
4.2.x		17	12.2%
4.3		18	3.5%
4.4	KitKat	19	36.1%
5.0	Lollipop	21	16.9%
5.1		22	15.7%
6.0	Marshmallow	23	0.7%



obr. 2 - prehľad rozdelenia androidu z januára 2016

1.4.1 Knižnice, architektúra - Dalvik

V systéme android je mnoho **knižníc** jazyka C a C++. Vývojári k nim majú prístup pomocou aplikačných frameworkov. Medzi niektoré základné knižnice patria knižnice multimédií (pre prehrávanie zvuku a videa), knižnice systému C, displeja, 3D knižnice, SGL knižnice (zakládne knižnice 2D gradfiky), SQ Lite (databáza) a mnoho ďalších.

Android používa vlastný virtuálny stroj **Dalvik** VM. Tento vytvára runtime prostredie pre aplikácie v Jave a slúži ako prekladač z Java byte code do Dalvik byte code. Dovôdom prekladu je ten, že v časoch vývinu Dalviku sa jazyk Java často menil, zatiaľ čo Java Byte code ostával viac menej rovnaký. Z tohoto vyplýva, že Android je možné programovať v ľubovoľnom jazyku, ktorý podporuje preklad do Java byte code. Avšak napriek tomu, že má Dalvik základy z Javy, nedodržuje niektoré štandardy Java SE a ani Java ME, čiže nie je 100% kompatibilný s Javou. Preto aj bol v roku 2014 v Android 5.0 “Lollipop” bol Dalvik nahradený Android Runtime (ART).

obr.3 Vrstvy operačného systému Android

1.4.2 Aktivity

Aplikácia nemá len jeden vstupný bod ako je *main()* metóda, ale je rozdelená do tzv. aktivít, ktoré sú na sebe nezávislé a môžu existovať samostatne. Pre užívateľa sú to najviac viditeľné časti programu. Každá aktivita väčšinou predstavuje jednu obrazovku aplikácie s konkrétnou funkcionalitou. Tieto slúžia ako hlavný prostriedok pre komunikáciu s uživateľom. Aplikácia pri spustení zobrazí jednu aktivitu, ktorá bola vybraná ako hlavná. Medzi ostatnými si môže používateľ ďalej nezávisle prepínať. Riadia sa tzv. životným cyklom, kde dochádza k prepínaniu mezi aktívnymi na popredíu a pasívnymi na pozadí, prip. spúšťaním či ukončovaním aktivity.

Životný cyklus aktivít

V prípade, že aktuálnu aktivitu nahradí iná, súčasná sa neukončí, ale iba za pozastaví. Prejde do pozadia, kde sa v prípade, že by ju chcel používateľ znovu použiť naspäť aktivuje. Systém automaticky ukončuje dlho nepoužívané aktivity z dôvodu uvoľnenia pamäti. Tie jednotlivé zmeny riadi Activity Manager. Vývojár pomocou implementovania príslušných metód reaguje na tieto zmeny. Ďalej vskratke opíšem jednotlivé stavy životného cyklu:

Spustenie aktivity

Ak aktivita ešte nebeží, alebo nie je uložená v pamäti, je nanovo vytvorená a zavolaná metodou *onCreate()*, ktorá je pomerne náročná na systém. Po štarte a zadeklarovaní ďalších príkazov metódy *onCreate()* aktivita prejde zo spúšťajúceho stavu do aktívneho stavu.

Aktívny stav

Do aktívneho stavu prechádzajú aktivity buď automaticky zo spúšťajúceho stavu, alebo prechodom zo stavu pozastavenia prip. zastavenia. V závislosti na predchádzajúcom stave aktivity, sú postupne zavolané metódy *onStart()* a *onResume()*. Aktívny stav je ten, pri ktorom je aktivita zobrazená na displaji a dochádza k interakcií s užívateľom, t. j. reakcia na stlačenia tlačidiel, zobrazenie informácií…V tomto stave ma aktivita prioritný prístup k systémovým zdrojom aby fungovala čo najplynulejšie. Aktivita ostáva v tomto stave pokiaľ nie je nahradená inou aktivitou.

Stav pozastavenia

Pokiaľ aktivita nie je aktívna, ale je stále viditeľná je zavolaná metóda *onPause()*. Tento stav nie je veľmi obvyklý, nakoľko je väčšina aktivít zobrazená po celý čas na displeji. Táto situácia väčšinou nastane pri zobrazení dialógového okna. Metóda *onPause()* je zavolaná aj v prípade, že aktivita prechádza do stavu zastavenia. Aktivita má stále prioritný prístup k systémovým zdrojom, keďže je zobrazená. Z toho stavu prechádza ďalej do stavu zastavenia ak nie je ďalej zobrazená, alebo prechádza do aktívneho stavu ak sa dostane do popredia.

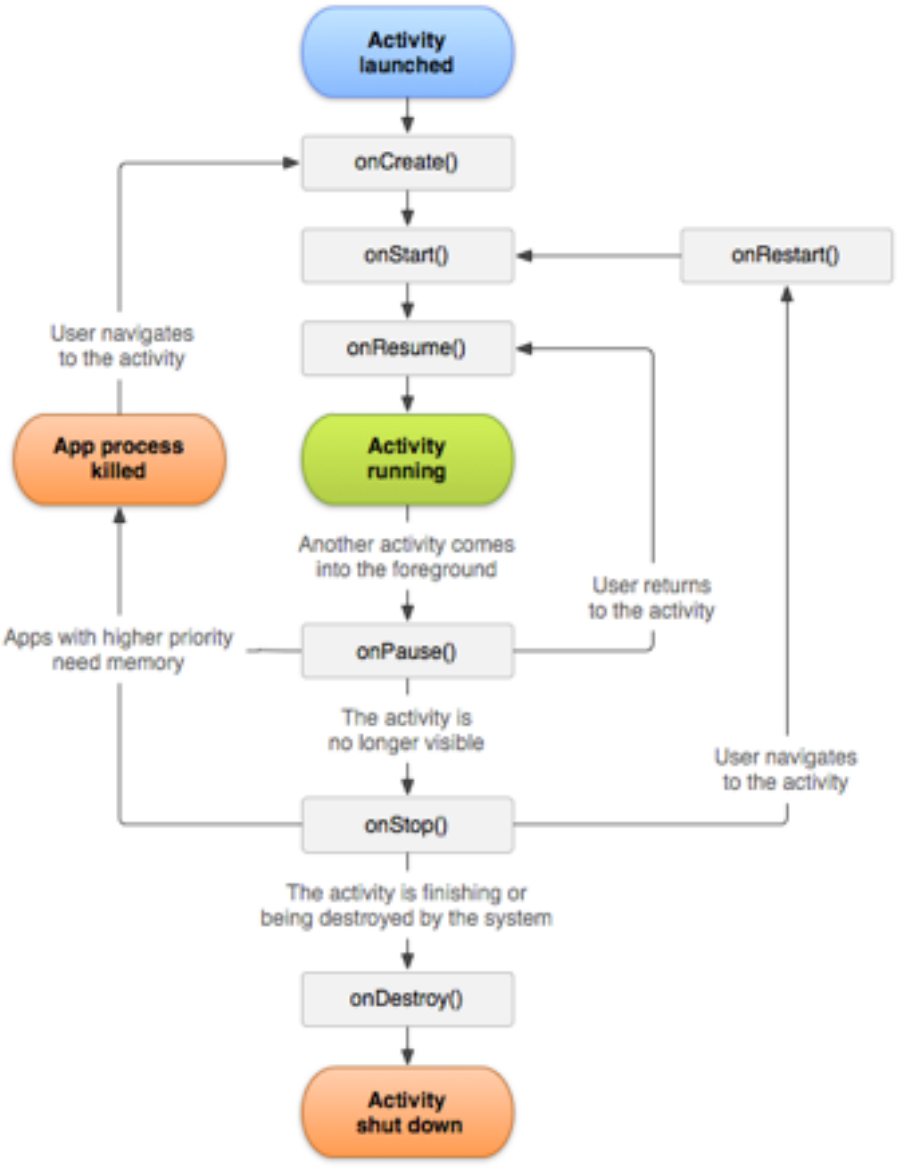
Stav zastavenia

V prípade, keď aktivita prestáva byť zobrazená na displeji (zväčša nahradená inou aplikáciou), prechádza do stavu zastavenia a je zavolaná metóda *onStop()*. Aktivita je uložená v pamäti pre prípad, že by ju chcel užívateľ znovu zobraziť, vtedy by sa zavolala metóda *onRestart()* a aktivita

by prešla do aktívneho stavu. Pri ukončeniu aktivita pokračuje prechodom do ukončovacieho stavu. Ak by nastala požiadavka na uvoľnenie miesta z pamäti, aktivita je zmazaná a pri jej opätovnom spustení beží od začiatku

Ukončovací stav

Ak má nastať ukončenie aktivity je zavolaná metóda *onDestroy()*. V tejto metóde môže aktivita vykonať ukončovacie akcie ako je napríklad uloženie, no vtedy nie je zaručené že bude riadne ukončená a zavolaná metóda *onDestroy()*. Preto sa odporúča robiť tieto akcie v stave pozastavenia.



obr. č 5 Stavový diagram životného cyklu Android Aktivity

1.5 Java

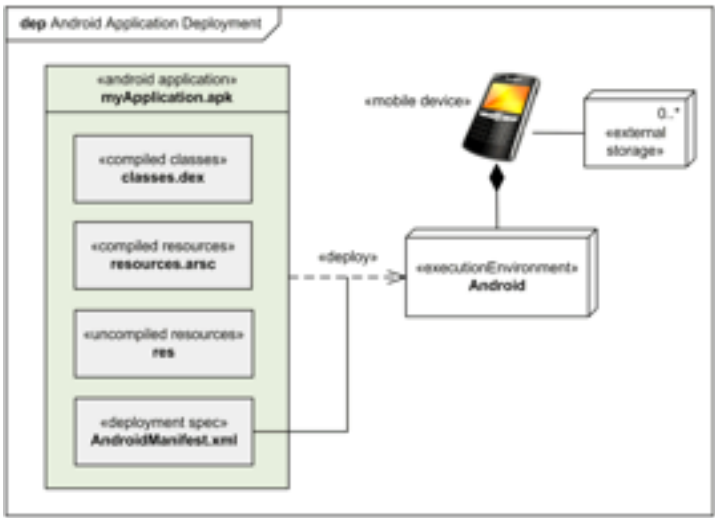
Je objektovo-orientovaný programovací jazyk, ktorý vychádza zo syntaxu jazykov C a C++. Ako som už spomenul, tento jazyk nie je závislý od konkrétnej platformy vďaka zdrojovým programom, ktoré sa nekompilujú do strojového kódu, ale do medzistupňa, tzv. “byte-code”. Tento jazyk bol vyvynutý aby bol bezpečný, odolný, výkonný a zároveň tak jednoduchý a ľahko naučiteľný.

2. Použité technológie

2.1. Android studio

2.1.1 Úvod

Kedže robím prvý krát svoju android aplikáciu, ako vývojovú platformu som sa rozhodol pre rýchlo rozvíjajúcu sa novinku od spoločnosti Google. Vývojové prostredie s názvom Android studio má podľa mňa veľkú perspektívu, a nakoľko jediná a zároveň aj najviac používaná konkurencia Eclipse mi prišla viac komplexnejšia a so širším záberom na možnosti, bolo to pre mňa zložitejšie ako Android Studio, ktoré sa specializuje len na projekty pre android.



obr. č. 6 vývoj android aplikácie

2.1.2 Java SE Development Kit

Tak ako aj Java samotná je produktom spoločnosti Oracle, ktorá do tohto balíčka zakomponovala skupinu základných nástrojov pre vývoj aplikácie. Rokmi ako plynul vývoj bol označovaný

viacerými skratkami, od JDK cez J2SE až po dnešný Java SE 8. Označením SE (Standard Edition) sa myslí Java, kt. bola postupne rozširovaná a vývijaná od prvej verzie. Súčasťou Java SE Development Kit je Java Runtime Enviroment, kt. obsahuje virtuálny stroj spúšťaný príkazov *java* a sadu knižníc- API knižnice základných fukncií. Okrem toho sa vo vývojovom balíčku nachádza napr. prekladač, debuger, skupina tried a mnoho ďalších príkazov potrebných pre vývoj. JDK predtavuje časť verzie SDK (software development kit), ktorá zodpovedá za písanie, vývoj a funkčnosť Java programov. Pre rôzne platformy sú k dispozícií aj rôzne JDK.

2.1.3 SDK tools

2.2 ZXing

ZXing je knižnica, ktorá podporuje dekódovanie a generovanie čiarových kódov ako napríklad QR kód, PDF 417, UPC, EAN, CodaBar,Code 93, 128 a 39…Je to opec-source *multiplatforma* od spoločnosti Google pre 1D a 2D "čiarové obráky" implementovaná v Jave s portami pre ostatné jazyky. Jadro obrázka dekóduje knižnica a testuje kód.

2.3 ZBar

ZBar je podobná multiplatforma ako ZXing. Je to tak isto open source software pre čítanie viacerých variácií čiarových kódov ako video strem, obrázkové súbory…Podporuje mnoho populárnych typov čiarových kódov vrátane EAN, UPC, QR Code, Code 128 a 39. Zbar je pod licenciou GNU LGPL, umožňujúca vývoj aj open source aj pre comerčné projekty. Vlastnotsi: vysoká rýchlosť (real time skenovať priamo z videa), malé požiadavky na veľkosť pamäti a veľkosť kódu, nie je limitovaná obrázkami, presná a vhodná pre vývoj aplikácií používajúce nenákladné procesory/hardware. Komponenty môžu byť použité spolu alebo oddelene.

Použiteľnosť **ZXingu** a **ZBaru** sú až na niekoľko odlišností na približne rovnakej úrovni. Z recenzií používateľov najväčšie rozdiely boli napríklad, že ZBar nedporuje iné 2D čiarové kódy okrem QR kódu, čo pre moju prácu nie je dôležité. Je perfektne rýchly a bezproblémový. Naopak ZXing má problém pri čítaní zložitejších, resp. väčších QR kódov. Rozhodol som sa preto práve ZBar.

*V mojej práci je výhodnejší ZBaru nakoľko nie je potrebný čítať neznáme kódy, resp. nepotrebujem širšie spektrum kódov ale stačí mi čítať predpripravené, v tomto prípade je ZBar rýchlejší a teda lepšia možnosť…

3. Zdroje

http://www.codeproject.com/Article

http://dotekomanie.cz/2013/05/prvni-dojmy-android-studia-13/

http://magazin.stahuj.centrum.cz/qr-kody-na-co-jsou-jak-je-vytvaret-cist-a-pouzivat/

http://developer.android.com/index.html

http://www.tutorialspoint.com/android/

https://zxingnet.codeplex.com

http://stackoverflow.com/questions/

+ ! odcitovat z knihy napr. !

[1] [2] [3] [4]

…aspon 1/2 na 1s popriradzovat…