TECHNICKÁ UNIVERZITA V KOŠICIACH

FAKULTA TECHNIKY A INFORMATIKY

INVENTÁRNY SYSTÉM PRE PLATFORMU ANDROID

Bakalárska práca

2012 **Vlastimil Brečka**

TECHNICKÁ UNIVERZITA V KOŠICIACH

FAKULTA ELEKTROTECHNIKY A INFORMATIKY

INVENTÁRNY SYSTÉM PRE PLATFORMU ANDROID

Bakalárska práca

Študijný program:	Informatika
Študijný odbor:	9.2.1 Informatika
Školiace pracovisko:	Katedra počítačov a informatiky (KPI)
Školiteľ:	Ing. Peter Feciľak, PhD.
Konzultant:	Ing. Peter Feciľak, PhD.

2012 Košice **Vlastimil Brečka**

Abstrakt v SJ

Cieľom bakalárskej práce bol návrh a implementácia aplikácie inventárneho systému pre platformu Android. Aplikácia je schopná prevziať aktuálny inventárny zoznam zo servera, ktorý je možno členiť do osobitých miestností. Kontrola inventáru prebieha načítaním QR kódu umiestneného na predmete pomocou vstavanej kamery mobilného telefónu a jeho následným spracovaním aplikáciou. Aplikácia má následne možnosť exportovať výsledky inventúry v podobe správy vo formáte HTML na server alebo lokálne na SD-kartu a rozoslať e-mailové notifikácie.

Kľúčové slová

Inventárny systém, inventúra, Android, QR kód

Abstrakt v AJ

The goal of bachelor‘s work was to design and implement inventory system application for Android platform. Application is able to download current inventory list from server and divide it into separate rooms. The actual inventory lookup is performed by scanning the QR code, placed on the item, via mobile phone‘s embedded camera and processing its content by the application. Upon completion, the application is able to export results as a HTML report to server or locally to phone’s SD-card and send out e-mail notifications.

Kľúčové slová v AJ

Inventory system, inventory lookup, Android, QR code

Zadanie práce

Čestné vyhlásenie

Vyhlasujem, že som celú diplomovú prácu vypracoval/a samostatne s použitím uvedenej odbornej literatúry.

Košice, 15. máj 2012 .....

vlastnoručný podpis

Poďakovanie

Chcel by som vyjadriť poďakovanie vedúcemu práce Ing. Petrovi Feciľakovi, PhD. za pripomienky pri vypracovaní práce a Branislavovi Dlugošovi za zapožičanie zariadenia mobilného telefónu.

Predhovor

Dlhotrvajúci trend vzrastajúcej popularity smartfónov v dnešnej dobe dáva možnosť a priestor softvérovým vývojárom ako ovplyvniť ďalšiu sféru ľudského života.

Hlavný dôvod, ktorý ma viedol k výberu témy je aktuálnosť modernej problematiky a predchádzajúci záujem o vytváranie aplikácií pre platformu Android. Značné rozšírenie platformy Android zväčšuje možný dosah používateľov pre danú aplikáciu. Znalosť platformy dáva možnosť vývojárom osloviť týchto potenciálnych zákazníkov poskytnutými riešeniami, rozšíriť svoje portfólio prác a zlepšiť takto svoju pozíciu na trhu práce. Bakalársku prácu som preto považoval za vhodnú príležitosť ako si plne osvojiť túto inak veľmi zaujímavú tému aj z praktických dôvodov.

Cieľom tejto práce je návrh a implementácia aplikácie inventárneho systému pre mobilnú platformu Android, ktorá zjednoduší a zefektívni procesy potrebné pri inventúre na katedre. Použité moderné postupy vykonávania inventúry sprehľadňujú a urýchľujú inak zdĺhavý úkon inventúry. Poskytnuté riešenia umožňujú taktiež jednoduchú a nenáročnú archiváciu.

V práci sa taktiež venujem stavbe operačného systému Android a teórií vývoja a základnej štruktúre vytvárania aplikácií pre platformu Google Android.

Obsah

Zoznam obrázkov

Zoznam tabuliek

Zoznam symbolov a skratiek

QR	**Q**uick **R**esponse – formát kódu rýchlej odozvy
RISC	**R**educed **I**nstruction **S**et **C**omputer – architektúra procesorov
ARM	**A**dvanced **R**ISC **M**achine – počítačová architektúra
API	**A**pplication **P**rogramming Interface – rozhranie pre programovanie aplikácií
GUI	**G**raphical **U**ser **I**nterface – grafické používateľské rozhranie
DVM	**D**alvik **V**irtual **M**achine – virtuálny stroj Dalvik
GNU	**G**NU's **n**ot **U**nix – operačný systém
SDK	**S**ofware **D**evelopment **K**it – softvérový vývojový balíček
ADT	**A**ndroid **D**evelopment **T**ools – vývojové nástroje pre Android
AVD	**A**ndroid **V**irtual **D**evice – virtuálne zariadenie Android
ADB	**A**ndroid **D**ebug **B**ridge – debugovací program pre Android
IDE	**I**ntegrated **D**evelopment **E**nviroment – integrované vývojové prostredie
LIFO	**L**ast **I**n, **F**irst **O**ut – algoritmus radenia
HTTP	**H**ypertext **T**ransfer **P**rotocol – internetový protokol
XML	e**X**tensible **M**arkup **L**anguage – rozšíriteľný značkovací jazyk
HTML	**H**yper**t**ext **M**arkup **L**anguage – hypertextový značkovací jazyk
DOM	**D**ocument **O**bjekt **M**odel – objektový model dokumentu
SAX	**S**imple **A**PI for **X**ML – API pre sekvenčný udalostne založený parser
NFC	**N**ear **F**ield **C**ommunication – technológia bezdrôtovej komunikácie

Slovník termínov

Operačný systém je softvér, ktorý spravuje zdroje počítača a poskytuje programátorom rozhranie pre prístup k týmto zdrojom. Operačný systém tiež spracúva dáta a vstupy od používateľov a odpovedá alokovaním a spravovaním úloh a interných zdrojov počítača ako služby pre používateľa.

Používateľské rozhranie sú programy a zariadenia, ktoré sú k dispozícií používateľovi systému na spracovanie dát.

Java je objektovo orientovaný programovací jazyk. Je vyvíjaný spoločnosťou Oracle. Jeho syntax vychádza z jazykov C a C++. Zdrojové programy sa nekompilujú do strojového kódu, ale do medzistupňa, tzv. bajt kódu, ktorý nie je závislý na konkrétnej platforme.

Softvérový vývojový balíček je súhrn nástrojov pre vývoj softvéru, ktorý umožňuje vytváranie aplikácií pre určitú platformu, framework, operačný systém.

Debugovanie je metodologický proces hľadania a redukovania počtu chýb a defektov v počítačovom programe s cieľom zabezpečenia korektného chovania aplikácie.

Virtuálny stroj je softvér, ktorý vytvára virtualizované prostredie medzi platformou počítača a operačným systémom, v ktorom koncový používateľ môže používať softvér na abstraktnom stroji.

Framework je softvérová štruktúra, ktorá slúži na podporu programovania a vývoj iných softvérových projektov. Môže obsahovať podporné programy, knižnice API alebo podporu pre návrhové vzory.

Úvod

Rozmach smartfónov a tabletov v poslednej dobe zmenil tieto zariadenia na neoddeliteľnú súčasť ľudského života a predmety dennej potreby. Mobilný telefón je jediné zariadenie, ktoré skutočne ľudia nosia neustále so sebou, čo má veľký dopad na potenciál a možnosti možných programov, pre nich určené. Tento poznatok, spolu s fenoménom internetu a dotykovým displejom, ktorý predstavuje skutočne prirodzenú formu interakcie so zariadením, z nich tvorí ideálny prostriedok komunikácie rôzneho druhu.

Neustály vývoj v hardvérovej oblasti podmieňuje vývoj v oblasti softvérovej do takej miery, že poskytnutý výkon umožňuje využívať takmer všetky z moderných technológií, známych zo sféry stolných počítačov. To spôsobuje možnosť jednoduchého prechodu programátorov na mobilné platformy, respektíve nevyžaduje potrebu osvojovať si úplne nový programovací jazyk, ktorý by bol špeciálne navrhnutý pre prácu v predsa stále hardvérovo ohraničených podmienkach.

Vývoj na poli dvoj a štvorjadrových ARM procesorov predpovedá blízke, zatiaľ aspoň čiastočné, zmazávanie rozdielov medzi mobilnými a stolnými verziami operačných systémov, kde jasným cieľom tohto snaženia je vytvorenie jedného rozsiahleho ekosystému pre počítače, tablety a smartfóny, a poskytovať tak celistvý používateľský zážitok naprieč všetkými systémami.

Jedným z dominantných prvkov pôsobiacich na trhu mobilných operačných systémov je firma Google so svojou platformou Android. Otvorenosť a mnohé iné kvality platformy sa pričinili o rozšírenie smartfónov a rozmach systému do takej miery, kde každý druhý smartfón je ovládaný práve týmto operačným systémom.

Spôsob vytvárania aplikácií pre platformu Android, ktorý sa opiera o všeobecne rozšírené schopnosti dnešných programátorov a hardvérové vymoženosti telefónov, boli dôvodom voľby tejto platformy pre realizáciu cieľa bakalárskej práce.

Cieľom práce bolo vytvoriť aplikáciu pre platformu Android, ktorá bude svojou funkcionalitou schopná vykonávať inventúru. Inventúra sa bude vykonávať zoskenovaním QR kódu pomocou kamery mobilného telefónu, kde bude následne obraz rozpoznaný knižnicou pre dekódovanie QR kódov a obsah kódu bude vrátený aplikácií na spracovanie. Spracované údaje z kódu pomôžu aplikácií zaznačiť tovar ako

prítomný, respektíve chýbajúci. Program taktiež ponúka riešenia pre ukladanie a archiváciu výsledkov jednotlivých inventúr.

Informačné zdroje boli čerpané hlavne z návodov a príručiek z oficiálnej web stránky pre vývojárov pre Android http://developer.android.com. Jednotlivé bežné problémy pri vývoji aplikácie boli riešené a konzultované na diskusnom fóre http://stackoverflow.com, na ktoré prispievajú aj členovia z oficiálneho Android tímu.

Na začiatku sa práca venuje teórií QR kódov, ich možnostiam, štruktúre a zloženiu. Nasleduje opis vzniku a teória architektúry platformy Android. Neskoršie kapitoly sú venované možnostiam vytvárania aplikácií a nástrojom, ktoré toto umožňujú a prácu uľahčujú. Ďalej sa pojednáva o zložení a štruktúre samotnej aplikácie, hlavných komponentocha iných prvkoch skladby. Nasledujúce kapitoly sa venujú analýze a samotnému riešeniu zadanej úlohy, jeho popisu a problémoch pri procese vývoja. V závere práca obsahuje zhrnutie problematiky, zistených problémov a vyskúmaných riešení na tieto problémy.

1 Formulácia úlohy

Úlohou bakalárskej práce je vytvoriť aplikáciu inventárneho systému pre platformu Google Android.

Je potrebné naštudovať možnosti vytvárania aplikácií, nainštalovať vývojové prostredie Eclipse a vývojové prostriedky pre platformu Android.

Aplikácia má mať schopnosť stiahnuť zoznam miestností a ich obsah zo strany servera. Musí poskytnúť voľbu miestnosti a následne zobraziť zoznam inventára, ktorý sa počas inventúry identifikuje.

V zozname označí aplikácia všetok majetok za chýbajúci. Aplikácia musí byť schopná načítať QR kód prostredníctvom kamery a rozlíšiť jeho obsah, ktorý je v pevne definovanom formáte. Po identifikácií objektu (majetku) ho zvýrazní symbolom OK.

Riešenie by malo poskytovať možnosť filtrovania výpisu na základe kritérií: všetko, iba chýbajúce, iba OK.

Aplikácia by mala poskytnúť možnosť exportovania inventárneho zoznamu z jeho stavom prostredníctvom e-mailu.

2 QR kód

QR (ang. Quick Response – rýchla odpoveď) kód je druh dvojrozmerného kódu vyvinutého japonskou firmou Denso Wave v roku 1994 za účelom rozšírenia použiteľnosti čiarových kódov a ľahkej interpretácie skenovacími zariadeniami.

Obr. 1 Ukážka QR kódu

2.1 Možnosti QR kódu

Klasický jednorozmerný čiarový kód obsahuje na rozdiel od QR kódu, dáta len v jednom smere, kde na rozdiel QR kód obsahuje dáta v smere vertikálnom aj horizontálnom a umožňuje tým uchovávať omnoho viacej informácií.

Obr. 2 Porovnanie kódov[1]

Kým klasické čiarové kódu umožňujú uchovávať maximálne 20 číslic, QR kód je schopný uchovávať až 7089 numerických znakov. Taktiež podporuje rôzne dátové typy ako čísel, znaky abecedy, Kanji symboly atď.[1]

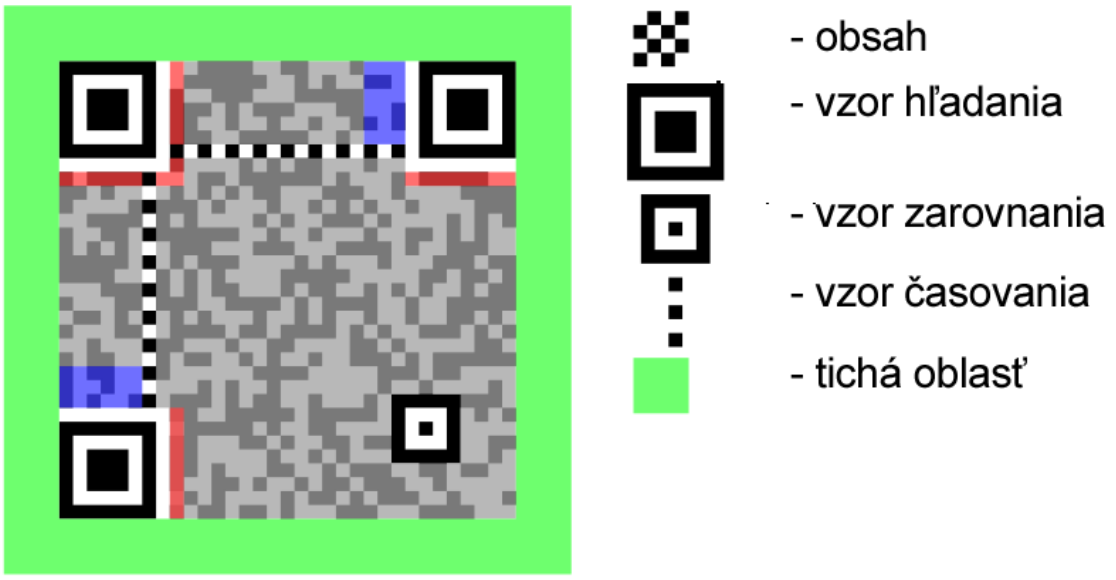
Iba numerické	Max. 7,089 znakov
Alfanumerické	Max. 4,296 znakov
Kanji	Max. 1,817 znakov

Tab. 1 Kapacita QR kódov[2]

2.2 Štruktúra

QR kódu je štvorcového tvaru a skladá sa z piatich hlavných častí.

- **Vzor hľadania** – tvoria ho štvorce nachádzajúce sa v troch zo štyroch rohov. Používa sa na uľahčenie zistenia pozície, veľkosti a uhlu otočenia QR kódu.
- **Vzor zarovnania** – využíva sa pre obnovu poškodenia. Môže byť použitý na identifikáciu či kódu obsahuje nelineárne chyby. Pre tento účel je umiestnená nezávislá čierna bunka v strede vzoru.
- **Vzor časovania** – skladá sa z bielych a čiernych buniek vo vertikálnych a horizontálnych líniách, ktoré spájajú tri vzory hľadania.
- **Tichá oblasť** – prázdna oblasť okolo kódu, nevyhnutná pre čítanie.
- **Oblasť dát** – Dáta uložené v kóde sú uschované v tejto oblasti v binárnej podobe.[3]



Obr. 3 Štruktúra QR kódu

2.3 Korekcia chýb

Kód ma schopnosť samokorekcie chýb. Poškodený kód je možné správne prečítať aj pri 30% poškodení. Korekcia je realizovaná Reed-Solomonovým algoritmom korekcie chýb. Kvôli jeho návrhu a používaniu 8-bitových slov, kód nesmie byť dlhší ako 255 slov. Samozrejme QR kódy obsahujú omnoho väčšie množstvo dát a preto je nutné obsah deliť do blokov.[3]

Obr. 4 Poškodený QR kód[2]

Vďaka korekcii chýb je možné taktiež vytvárať umelecké verzie QR kódov, ktoré budú stále plne funkčné, ale omnoho príťažlivejšie pre používateľa.

Obr. 5 Umelecké stvárnenie QR kódu[5]

3 Operačný systém Google Android

Operačný systém Google Android je open-source operačný systém založený na Linuxovom jadre. Systém poskytuje riešenia pre mobilné telefóny, tablety, netbooky, televízory a iné zariadenia. Platforma je tvorená konzorciom firiem Open Handset Alliance, ktoré je tvorené z 84 veľkých softvérových, hardvérových a telekomunikačných firiem, vedených firmou Google.[6]

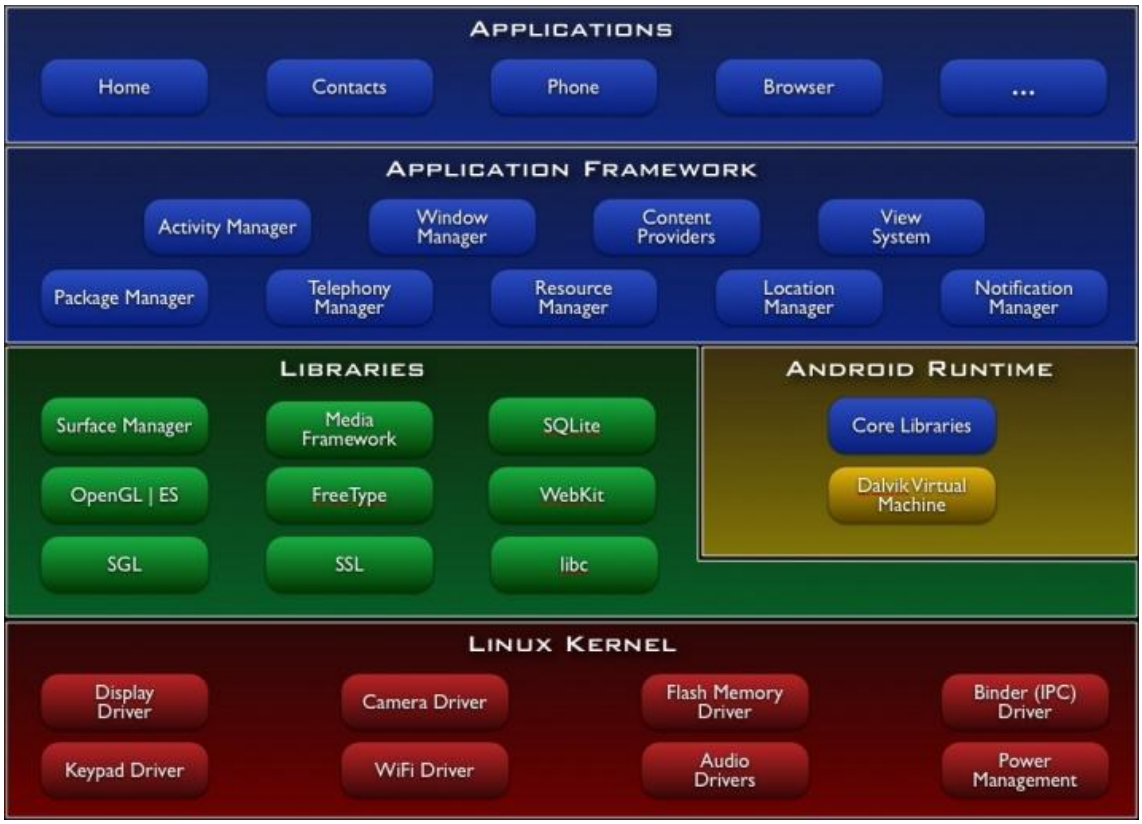
Operačný systém bol spočiatku vyvíjaný firmou Android, ktorá bola v roku 2005 odkúpená firmou Google. Následne bol zriadený Android Open Source Project, kde tento tým bol poverený vývojom a údržbou operačného systému.[7]

Obr. 6 Logo operačného systému[8]

Samotná platforma Android dáva k dispozícii nielen operačný systém s používateľským prostredím pre koncových používateľov, ale aj kompletné riešenie nasadenia operačného systému (špecifikácia ovládačov a pod.) pre mobilných operátorov a výrobcov zariadení a v neposlednom rade pre vývojárov aplikácií poskytuje efektívne nástroje pre ich vývoj - Software Development Kit.

3.1 Architektúra

Systém je rozdelený do piatich vrstiev. Každá vrstva má svoj účel a nemusí byť priamo oddelená od ostatných vrstiev.



Obr. 7 Architektúra operačného systému Android[9]

3.1.1 Linuxové jadro

Najnižšou vrstvou architektúry je jadro operačného systému, ktoré tvorí abstraktnú vrstvu medzi používaným hardvérom a zvyškom softvéru vo vyšších vrstvách. Jadro systému Androidu je postavené na Linuxe. Využíva mnoho jeho vlastností, ako sú podpora správy pamäte, správa sietí, zabudované ovládače, správa procesov alebo súbežný beh aplikácií, ktoré bežia ako samostatné procesy s prioritou stanovenou systémom.

Táto vlastnosť prispieva k stabilite a tiež ochrane systému. Naopak systém nepodporuje grafické používateľské rozhranie X Window System a ani úplnú sadu GNU knižníc. Dôvodom použitia jadra Linux bola tiež vlastnosť pomerne jednoduchej kompilácie na rôznych zariadeniach a tým zaručená prenositeľnosť.

3.1.2 Knižnice

Knižnice sú písané v jazyku C a C++, a využívajú ich rôzne časti systému. Tieto funkcie sú vývojárom poskytnuté prostredníctvom Android Application Framework. Tu sú uvedené iba niektoré príklady knižníc:

- Media Libraries - knižnica podporuje prehrávanie video a audio formátov, obrazových súborov.
- LibWebCore - knižnica webového prehliadača, ktorý podporuje aj vložené náhľady webových stránok.
- Libc - odvodená BSD štandardná knižnica systému C vyladená pre embedované zariadenia.
- SQLite - odľahčená knižnica pre prístup k relačným databázam
- OpenSSL - secure socket layer
- FreeType - knižnica pre vykresľovanie bitmapových a vektorových fontov.
- OpenGL - knižnica na vykresľovanie 3D grafiky.

3.1.3 Android Runtime

Vrstva Android Runtime obsahuje aplikačný virtuálny stroj zvaný Dalvik, ktorý bol vyvíjaný od roku 2005 špeciálne pre Android.

Virtuálny stroj Dalvik je registrovo orientovaná architektúra, využíva základné vlastnosti linuxového jadra, ako je správa pamäte alebo práca s vláknami. Vznik nového virtuálneho stroja bol iniciovaný z dvoch dôvodov. Prvým dôvodom boli licenčné práva, pretože jazyk Java a jeho knižnice sú voľne šíriteľné, zatiaľ čo virtuálny stroj Java Virtual Machine nie je. Ďalším dôvodom bola optimalizácia virtuálneho stroja pre mobilné zariadenia a to predovšetkým v oblasti pomeru úspory energie a výkonu.

V tejto vrstve sú tiež obsiahnuté základné knižnice programovacieho jazyka Java. Knižnice sa svojím obsahom blížia platforme Java Standard Edition. Hlavný rozdiel je v neprítomnosti knižníc pre užívateľské rozhranie (AWT a Swing), ktoré boli nahradené knižnicami užívateľského rozhrania pre Android alebo pridanie knižnice Apache pre prácu so sieťou. Preklad aplikácie vyvinutej pre Android prebieha skompilovaním zdrojového Java kódu do Java byte kódu pomocou rovnakého kompilátora, ako sa používa v prípade prekladu Java aplikácií. Potom sa prekompiluje Java byte kód pomocou Dalvik kompilátora a výsledný Dalvik bajt kód je spustený na DVM. Každá spustená Android aplikácie má svoj vlastný proces s vlastnou inštanciou DVM.

Android aplikácie sú spúšťané v sandboxe, izolovanom mieste v operačnom systéme, kde aplikácie nemajú prístup k zvyšku zdrojov systému, ak im vhodnými povoleniami udelenými používateľom nebolo toto pridelené pri inštalácií.

3.1.4 Aplikačný framework

Vrstva application framework je pre vývojárov najdôležitejšia. Poskytuje prístup k veľkému počtu služieb, ktoré môžu byť použité priamo v aplikáciách. Tieto služby môžu sprístupňovať dáta v iných aplikáciách, prvky používateľského rozhrania, upozorňovací stavový riadok, aplikácie bežiace na pozadí, hardvér používaného zariadenia a mnoho ďalších služieb a funkcií. Základná sada služieb zahŕňa predovšetkým:

- **Pohľady** – (z angl. *Views*) tieto prvky sú použité pre zostavenie používateľského rozhrania ako zoznamy, textové pole, tlačidlá a iné.
- **Správca zdrojov** - poskytuje prístup k zdrojom aplikácie, ako sú reťazce, grafika, pridané súbory.
- **Správca notifikácií** - umožňuje všetkým aplikáciám zobraziť vlastné upozornenie v stavovom riadku.
- **Správca aktivít** - riadi životný cyklus aplikácií a poskytuje orientáciu v zásobníku s aplikáciami.

3.1.5 Aplikácie

Najvyššiu vrstvu systému tvoria základné aplikácie, ktoré využívajú bežný používatelia. Môže ísť o aplikácie predinštalované alebo o aplikácie tretích strán, dodatočne stiahnutých z oficiálneho obchodu Google Play.[10]

4 Analýza možností vytvárania aplikácií pre platformu Android

Vývoj aplikácií pre platformu Android je realizovaný pomocou Android softvérového vývojového balíčka (z angl. software development kit) v objektovo-orientovanom jazyku Java, alebo v jazykoch C a C++ pomocou Android natívneho vývojového balíčka (z angl. native development kit).

4.1 Android softvérový vývojový balíček

Softvérový vývojový balíček pre platformu Android je súhrn nástrojov, ktoré umožňujú vývoj aplikácií pre tento operačný systém. Nástroje obsahujú debugger, knižnice, emulátor, dokumentáciu, vzorové ukážky kódov atď.

Všeobecne sú nástroje delené do dvoch skupín:

- **Nástroje balíčka** – (z angl. *SDK tools*) nástroje, ktoré sú platformovo nezávislé a sú potrebné pre každý verziu platformy.
- **Nástroje platformy** – (z angl. *platform tools*) nástroje, ktoré sú špecializované pre danú verziu platformy.[11]

Tab. 2 Tabuľka stupňov aplikačných rozhraní[12]

Verzia platformy	API level	VERSION_CODE
Android 4.0	14	ICE_CREAM_SANDWICH
Android 3.2	13	HONEYCOMB_MR2
Android 3.1.x	12	HONEYCOMB_MR1
Android 3.0.x	11	HONEYCOMB
Android 2.3.4 Android 2.3.3	10	GINGERBREAD_MR1
Android 2.3.2 Android 2.3.1 Android 2.3	9	GINGERBREAD
Android 2.2.x	8	FROYO
Android 2.1.x	7	ECLAIR_MR1
Android 2.0.1	6	ECLAIR_0_1
Android 2.0	5	ECLAIR
Android 1.6	4	DONUT
Android 1.5	3	CUPCAKE
Android 1.1	2	BASE_1_1
Android 1.0	1	BASE

Platí pravidlo spätnej kompatibility (API level). Android 2.x boli verzie určené pre mobilné telefóny, 3.x pre tablety. S príchodom novej verzie 4.0 Ice Cream Sandwich sa toto rozdelenie zjednocuje, maže rozdiely medzi API pre tablety a telefóny a uľahčuje prácu vývojárom.

4.1.1 Vývojové nástroje

Vývojové nástroje pre Android (z angl. *Android Development Tools*) je zásuvný modul pre vývojové prostredie Eclipse, predstavujúci integráciu SDK do tohto rozšíreného vývojového prostredia. Modul umožňuje jednoduchý a pohodlný vývoj Android aplikácii. Rozširuje schopnosti prostredia Eclipse a umožňuje rýchlo vytvárať Android projekty, aplikačné rozhranie, pridávať a používať komponenty z Android Framework API, debugovať aplikáciu pomocou Android SDK nástrojov.

Vývoj Android aplikácií vo vývojovom prostredí Eclipse je firmou Google silne doporučený, je to najrýchlejšia možnosť ako začať s vývojom, predovšetkým kvôli mnohým existujúcim návodom, vlastnému XML editoru pre tvorbu používateľských rozhraní.[13]

4.1.2 Debugovanie

Debugovanie je realizované pomocou Android Debug Bridge, versatilného klient – server programu príkazového riadku umožňujúceho komunikáciu s emulátorom alebo Android zariadením. Program umožňuje napríklad inštalovať aplikáciu na zariadenie, smerovať porty alebo kopírovať súbory do alebo zo zariadenia.[14]

4.1.3 Virtuálne zariadenie

Správca Android Virtual Device je správca virtuálneho zariadenia, ktorý umožňuje modelovať hardvér a softvér emulovaného zariadenia.

Obr. 8 Android virtuálne zariadenie[15]

Android virtuálne zariadenie pozostáva z:

- **hardvérového profilu** – umožňuje definovať či zariadenie má kameru, qwerty klávesnicu, koľko pamäte používa a podobne
- **softvérového profilu** – možnosť definovať, ktorá verzia platformy Android bude spustená na emulátore
- **iné nastavenia** – umožňuje meniť rozmery, orientáciu, vzhľad, vytvoriť virtuálnu SD kartu a podobne[16]

5 Analýza štruktúry Android aplikácie

5.1 Komponenty

Android aplikácia sa skladá zo štyroch základných častí:

- Aktivita (z angl. Activity)
- Služba (z angl. Service)
- Poskytovateľa obsahu (z angl. Content Provider)
- Prijímače vysielania (z angl. Broadcast Receivers)

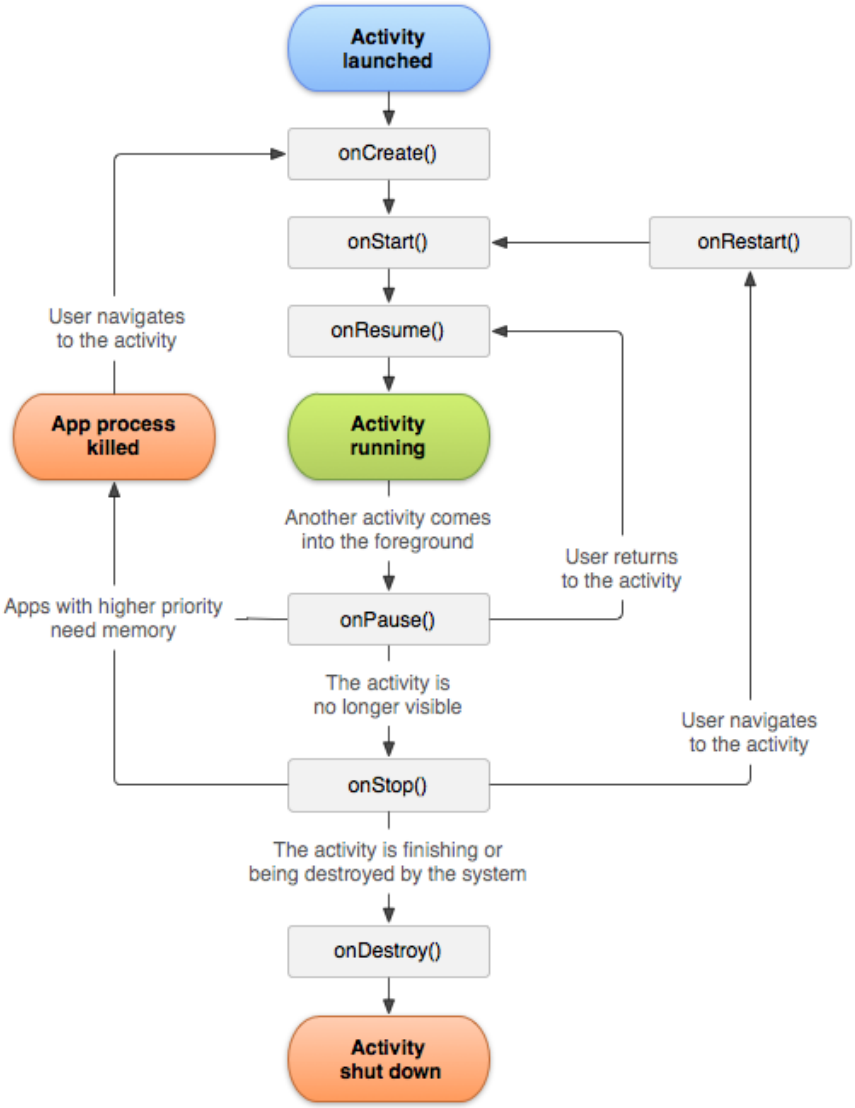
Nie každá aplikácia obsahuje všetky štyri komponenty, ale určite bude pozostávať z nejakej ich kombinácie. Použitie týchto prvkov musí byť zaznamenané v Android manifeste.**Chyba! Nenašiel sa žiaden zdroj odkazov.**

5.1.1 Aktivity

Aktivita je komponent aplikácie, ktorá poskytuje obrazovku s ktorou používatelia komunikujú. Každá aplikácia má svoj životný cyklus a pridelené okno do ktorého vykreslí používateľské rozhranie.

Aplikácia zvyčajne pozostáva z viacerých voľne previazaných aktivít. Obvykle je jedna z aktivít označená ako hlavná, a je zobrazená používateľovi pri štarte aplikácie. Každá aktivita môže spustiť inú aktivitu, pre vykonanie nejakého špecifického úkonu. Vždy, keď je spustená nová aktivita, predchádzajúca aktivita je pozastavená, ale je stále uschovaná v zásobníku aktivít a nová aktivita preberá fókus. Zásobník aktivít podlieha klasickému LIFO algoritmu, z vplýva že, ak používateľ ukončil prácu v aktuálnej aktivite a stlačí tlačidlo Naspäť, bude aktivita zo zásobníka odstránená a predošlá aktivita bude obnovená.

Keď je aktivita prerušená kvôli štartu inej aktivity, je jej o tejto udalosti oznámené pomocou zmeny jej stavu cez návratové volania metód jej životného cyklu.[18]



Obr. 9 Životný cyklus aktivity[18]

5.1.2 Služby

Služby sú komponenty aplikácie ktoré vykonávajú dlhotrvajúce operácie na pozadí a neposkytujú používateľské rozhranie. Odlišný komponent môže spustiť službu a tá bude pokračovať vo vykonávaní činnosti aj po uzavretí aplikácie. Službou sú napríklad prehrávanie hudby, sieťové transakcie, vstupné a výstupné operácie pri práci so súbormi. Služba má dve podoby:

- **Neviazaná** – služba je neviazaná, ak aplikačný komponent, napríklad aktivita, spustí službu volaním `startService()`. Spustená služba môže nekonečne vykonávať svoju činnosť aj po zničení aktivity, ktorá ju spustila a nevracia výsledok alebo návratové volanie volajúcemu.

- **Viazaná** – služba je viazaná, ak ju volajúci komponent so sebou previaže volaním `bindService()`. Viazaná služba poskytuje klient-server rozhranie, ktoré umožňuje interakciu so službou, posielať požiadavky alebo obdržiavať výsledky. Takáto služba je spustená iba pokiaľ je spustený komponent na ktorý je previazaná.[19]

5.1.3 Poskytovatelia obsahu

Poskytovateľ obsahu spravuje zdieľané dáta. Dáta môžu byť uložené v súborovom systéme, databáze alebo na internete. Pomocou poskytovateľa obsahu, aplikácie môžu dopytovať alebo meniť dáta. Napríklad, Android poskytuje poskytovateľ obsahu pre kontakty používateľa. Každá aplikácia s potrebnými právami, môže čítať a zapisovať informácie o danej osobe z aplikácie kontaktov.

5.1.4 Príjmače vysielania

Príjmače vysielania sú komponenty, ktoré reagujú na vysielacie oznámenia na celej šírke systému. Mnoho vysielaní pochádza zo samotného systému, ako napríklad vysielanie oznamujúce vypnutie obrazovky alebo nízkeho stavu batérie. Neposkytujú používateľské rozhranie, ale umožňujú vytvoriť notifikáciu v správcovi notifikácií. Vysielanie sa realizuje pomocou správ typu Intent.[20]

5.1.5 Správy typu Intent

Špeciálne správy typu Intent predstavujú médium spúšťania komponentov aplikácie. Umožňujú previazanie vlastných komponentov aplikácie alebo komponentov aplikácie externej počas behu programu. Samotná správa je objekt typu Intent a je pasívnou dátovou štruktúrou držiacou abstraktný popis operácie ktorá sa má vykonať, alebo v prípade vysielaní (z angl. broadcast), popis niečoho čo sa udialo a je oznamované. Existujú rôzne mechanizmy pre doručovanie správ k jednotlivým druhom komponentov:

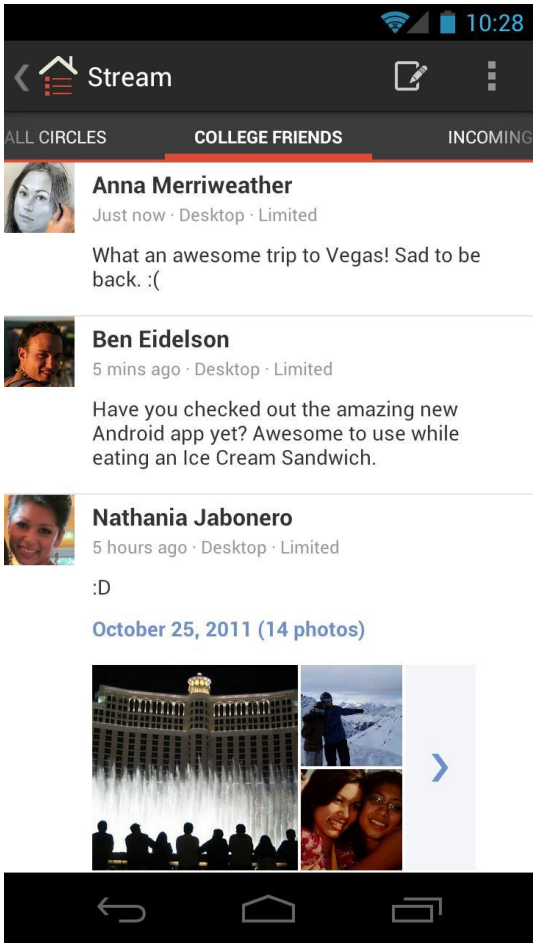
- **aktivita** – pre spustenie aktivity pre objekt typu Intent použitý ako argument volania `Context.startActivity()`
- **služba** – objekt správy je používaný pre spustenie novej služby alebo pre doručenie inštrukcií už spustenej službe. Pre naviazanie spojenia medzi volajúcim komponentom a volanou službou je použité volanie `Context.startService()`

- **vysielanie** -správa je doručená pomocou `Context.sendBroadcast()` všetkým načúvajúcim prijímačom vysielania.[21]

5.2 Používateľské rozhranie

Používateľské rozhranie Android aplikácie je tvorené z pohľadov (z angl. Views) a skupín pohľadov (z angl. ViewGroups). Existuje mnoho druhov pohľadov a skupín pohľadov, ktoré rozširujú práve triedu View.

Rozhrania sú tvorené pomocou súboru rozmiestnenia vo formáte XML alebo z prostredia jazyka Java.



Obr. 10 Ukážka používateľského rozhrania[22]

5.2.1 Pohľady

Pohľady sú základná jednotka používateľského rozhrania platformy Android. Trieda View predstavuje bázovú triedu pre prvky zvané widgety, ktoré predstavujú plne implementované objekty používateľského rozhrania, ako napríklad tlačidlá a textové polia.

Objekty pohľadov sú dátovými štruktúrami, ktorých vlastnosti sú uchovávané v ich parametroch rozmiestnenia a obsahu, pre špecifickú obdĺžnikovú oblasť, ktorú zaberajú na obrazovke. Pohľad sám spravuje svoje rozmery, rozmiestnenie, vykresľovanie, zmeny fókusu, skrolovanie a interakciu pomocou gest na obdĺžnikovej oblasti, ktorá mu prislúcha. Pohľad je taktiež prijímateľom a prostriedkom interakcie s používateľom.

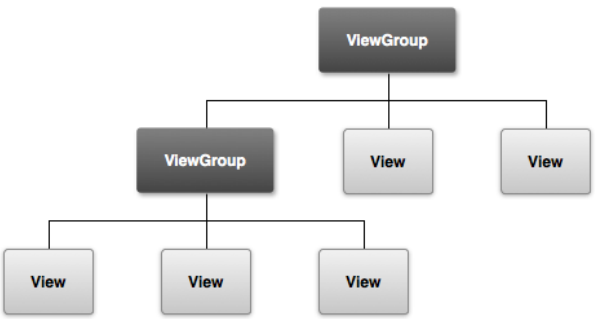
5.2.2 Skupiny pohľadov

Trieda `ViewGroup` zase predstavuje triedu od ktorej sú odvodené podtriedy zvané rozmiestnenia (z angl. layouts), ktoré poskytujú rôzne druhy rozloženia pohľadov, ako napríklad lineárne rozmiestnenie alebo rozmiestnenie relatívne.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
              android:layout_width="fill_parent"
              android:layout_height="fill_parent"
              android:orientation="vertical" >
    <TextView android:id="@+id/text"
              android:layout_width="wrap_content"
              android:layout_height="wrap_content"
              android:text="Hello, I am a TextView" />
    <Button android:id="@+id/button"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Hello, I am a Button" />
</LinearLayout>
```

Obr. 11 Príklad definície rozmiestnenia pomocou XML[23]

Najčastejšou formou tvorby rozmiestnenia rozhrania je pomocou jeho XML formou zápisu. XML poskytuje štruktúru pre rozmiestnenie, ktorá je ľahko čitateľná pre človeka. Každý element v XML, je práve pohľad alebo skupina pohľadov, respektíve ich rozšírenia. V stromovej štruktúre predstavujú pohľady listy a skupiny pohľadov reprezentujú uzly stromu. Názov elementu v XML korešponduje s názvom Java triedy, ktorú reprezentuje.[23]



Obr. 12 Schematický náhľad na hierarchiu[23]

5.3 Manifest

Pred štartom aplikačného komponentu, systém musí vedieť o jeho existencií v manifestovom súbore `AndroidManifest.xml`. V manifeste musia byť deklarované všetky komponenty aplikácie a ten sa musí nachádzať v koreňovom adresári projektu. Okrem deklarácie komponentov, manifest obsahuje okrem iného:[20]

- Práva udelené používateľom pri inštalácií aplikácie
- Deklarácie minimálneho stupňa aplikačného rozhrania
- Deklarácie hardvérových a softvérových prostriedkov požadovaných aplikáciou

```
<?xml version="1.0" encoding="utf-8"?>
<manifest ... >
    <application android:icon="@drawable/app_icon.png" ... >
        <activity android:name="com.example.project.ExampleActivity"
                  android:label="@string/example_label" ... >
        </activity>
        ...
    </application>
</manifest>
```

Obr. 13 Ukážka manifestu[20]

6 Analýza návrhu riešenia inventárneho systému

Úlohou bakalárskej práce bolo vytvoriť aplikáciu inventárneho systému pre platformu Google Android.

Aplikácia má umožňovať vykonávať inventúru v určených miestnostiach, ktorých zoznam a obsah je v preddefinovanom tvare a formáte XML uložený na strane servera. Tento súbor je aplikáciou prevzatý cez HTTP protokol a jeho dáta sú načítané do pamäte.

Identifikácia tovaru je riešená pomocou QR kódov fyzicky umiestnených na danom predmete. Načítanie tohto kódu sa zabezpečuje pomocou funkcie kamery, kde je následne obraz rozpoznaný a samotná zakódovaná informácia je interpretovaná aplikáciou pomocou knižnice ZXing, do formy reťazca znakov.

Následne je načítaný predmet porovnaný s predmetmi v miestnosti podľa identifikačného čísla. Ak sa predmet v aktuálnej miestnosti nachádza a bol doteraz označený ako chýbajúci, bude označený ako identifikovaný.

Po ukončení procesu inventúry, je možné výsledky inventúry exportovať pomocou skriptu vo formáte PHP na stranu servera, kde budú prenesené výstupy spracované do graficky prehľadného dokumentu vo formáte .html a uložené pre budúcu archiváciu. Jednotlivé výsledky budú triedené podľa dátumu inventúry a miestnosti v ktorej inventúra prebiehala. Takto archivované výstupné správy bude možné pohodlne a jednoducho prezerať v prostredí internetového prehliadača. Získané výsledky je možné uložiť taktiež aj lokálne do pamäte telefónu.

O skutočnosti vykonanej inventúry a jej výsledkov bude možné upozorniť určené osoby pomocou e-mailovej notifikácie odosielanej na preddefinované e-mailové adresy.

7 Návrh riešenia inventárneho systému

Aplikácia inventárneho systému je vytvorená pre verziu platformy 2.2 Froyo a vyššie.

Obr. 14 Ikona aplikácie

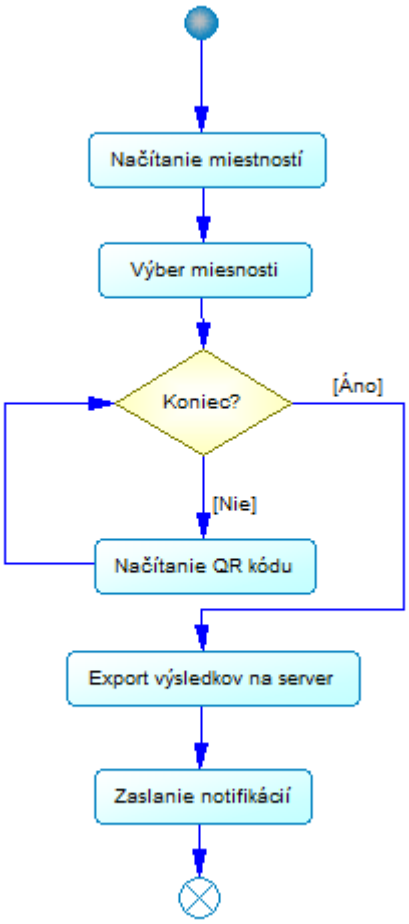
Používateľské rozhranie na mnohých miestach implementuje gestá posunutia a tým uľahčuje často používané úkony ako spúšťanie kamery alebo ak sa vo vrchnej časti obrazovky nachádza indikátor stránkovateľnosti, je takto možné stránkovať jednotlivé obrazovky.

Obr. 15 Ukážka hlavného menu

Podľa použitého návrhového vzoru odporúčaného pre aplikácie pre platformu Android platí, že ak existujú prídavné funkcie pre aktuálnu aktivitu, je možné túto funkcionalitu vyvolať stlačením Menu tlačidla. To znamená, že v používateľskom rozhraní sa nebude nachádzať žiadny odkaz na tento fakt a očakáva sa, že používateľ

bude automaticky hľadať a vyvolávať funkcionalitu v menu, ktoré sa zobrazí stlačením práve hardvérového Menu tlačidla.

Vykonávanie aktivít je lineárne sekvenčne za sebou, aj keď aplikácia poskytuje možnosť paralelnej inventúry vo viacerých miestnostiach.



Obr. 16 Zjednodušený náčrt diagramu aktivít

7.1 Načítanie miestností

Na uschovanie obsahu inventára bol použitý zdrojový súbor vo formáte XML. Stromová povaha XML súborov je ideálna pre danú úlohu, pretože celý inventár potrebujeme členiť do miestností obsahujúce samotné položky.

Progres inventúry sa udržiava v pamäti pokiaľ je aplikácia spustená a tým umožňuje vykonávať paralelne viacej inventúr. Aplikácia poskytuje možnosť znova načítať inventárny zoznam a tým zresetovať progres inventúr vo všetkých miestnostiach.

Obr. 17 Ukážka aktivity výberu miestnosti

Kvôli existencii už hotových a predtým používaných QR kódov na katedre, bol formát vlastností položiek prevzatý. Presné dodržanie nasledovnej štruktúry zdrojového súboru je nevyhnutné pre správnu funkciu parsera. Zdrojový súbor musí byť v nasledujúcom tvare:

```
<inventory>
     <room name="<meno miestnosti>">
          <item
               EVID.C.="<nové identifikačné číslo>"
               Stare_C.="staré identifikačné číslo>"
               Opis="<opis>"
               Kusov="<počet kusov>"
               Miestnost="<meno miestnosti>" />
     </room>
</inventory>
```

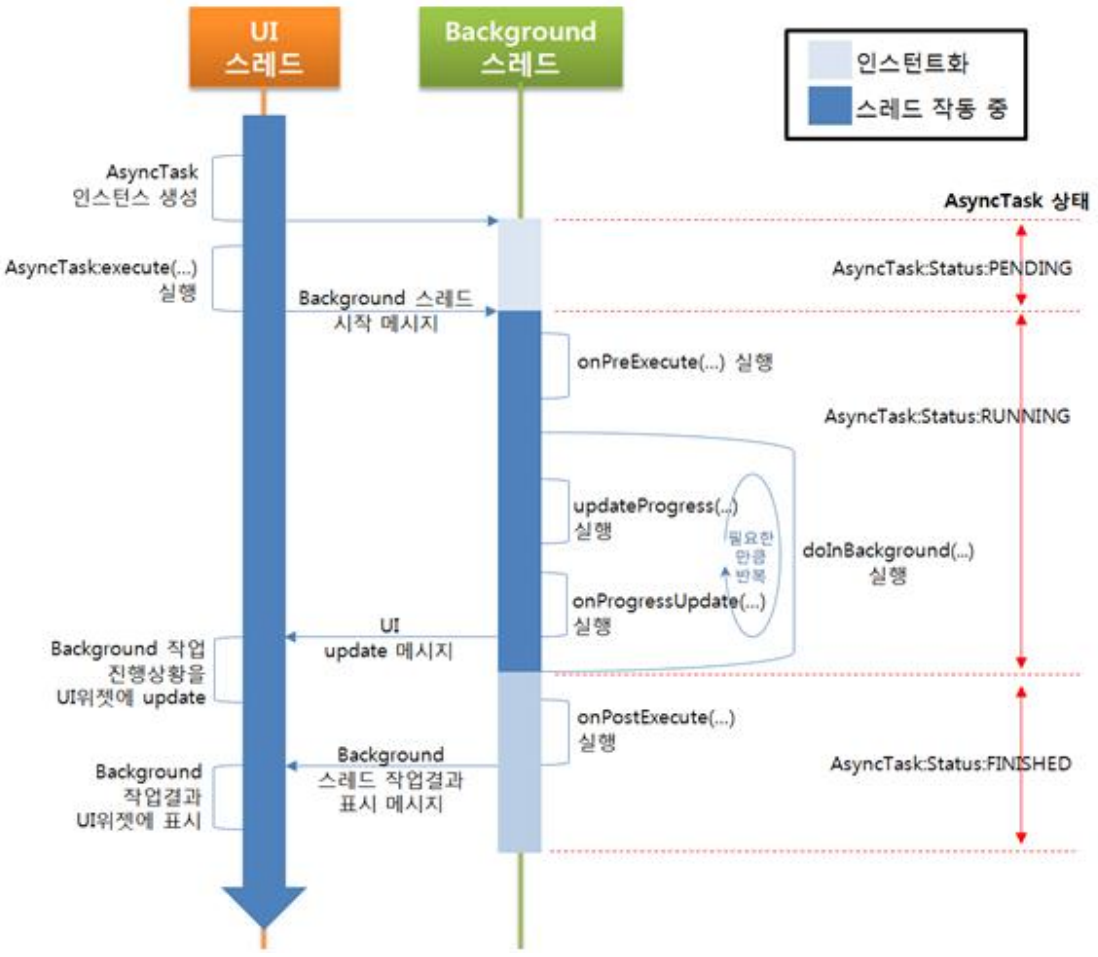
Takouto štruktúrou vieme definovať miestnosti, v miestnostiach položky, ktoré do nich patria a priradiť položkám ich potrebné dáta. Zdrojový súbor bude uložený na strane servera. Následne po spustení inventúry bude tento súbor vo formáte XML aplikáciou prevzatý a parserom ďalej spracovaný.

7.1.1 Asynchrónna úloha

V priebehu vývoja nastal klasický problém pri začiatkoch s vývojom na platforme Android. Problém spočíva v tom, že úkon prevzatia súboru z internetu, respektíve hocijakej náročnej operácie sa vykonával na hlavnom vlákne, ktorého hlavná úloha spočíva v staraní sa o používateľské rozhranie, z toho aj plynie jej názov vlákno používateľského rozhrania (z angl. user-interface thread).

Z toho vyplýva, že budú všetky úkony na hlavnom vlákne pozastavené, kým daná náročná operácia neskončí. Toto má za následok, že aplikácia sa javí ako neresponzívna, pretože hlavné vlákno vykonáva spomínanú sieťovú úlohu a nespracúvava kliknutia používateľa. V tomto prípade to nebolo až tak kritické, pretože zdrojový súbor je veľmi malý, ale je dobrým zvykom do budúcnosti to riešiť korektne.

Takýto náročný úkon má byť riešený na novom vlákne, teda asynchrónne, aby vlákno používateľského rozhrania ostalo neblokované. Pre prácu s asynchrónnymi úkonmi sa používa trieda `AsyncTask`. Asynchrónna úloha bude bežať na pozadí a po svojom ukončení zavolá príslušné návratové volanie.



Obr. 18 Diagram práce asynchrónnej úlohy[24]

7.2 Parsovanie prevzatého zdrojového súboru

Po prevzatí zdrojového XML súboru je potrebné ho preparsovať, to znamená vytiahnuť z XML potrebné údaje o položke a s týmito dátami vytvoriť v Jave objekt položky triedou `Item`, obdobne budú vytvorené objekty miestností pomocou inštancií triedy `Room`, ktorým budú priradené zoznamy položiek ktoré obsahujú.

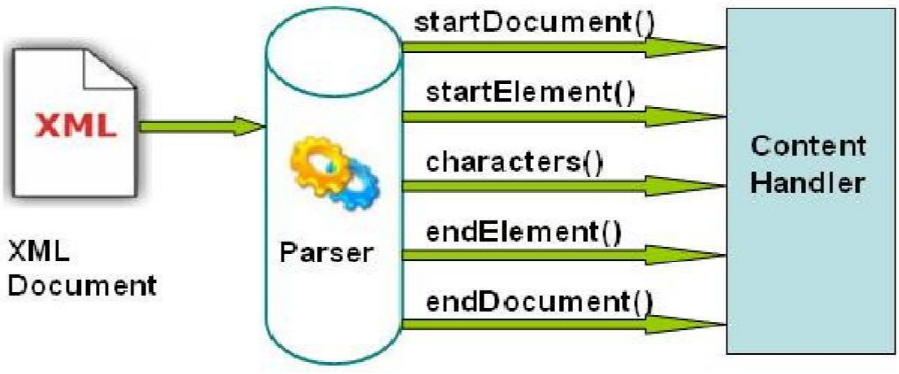
Na parsovanie bola spočiatku použitý regulárny výraz, ktorý sa javil ako postačujúci pre jednoduchú a nemeniacu sa štruktúru zdrojového súboru ale v závere vývoja sa pri testovaní ukázal ako nedostatočný pri odchytávaní chýb v XML súbore.

Kvôli väčšej robustnosti a jednoduchosti pri odchytávaní chýb bola nakoniec použitá implementácia SAX parsera zabezpečujúca trieda `Parser`.

7.2.1 Implementácia parsera

Implementovaný SAX parser je aplikačné rozhranie parsera so sekvenčným prístupom, založeného na udalostiach. Poskytuje mechanizmus pre čítanie dát z XML dokumentu, ktorý je alternatíva ktorú poskytuje objektový model dokumentu (z angl. Document Object Model). DOM pracuje ako celok, kde SAX parser pracuje s každým elementom XML dokumentu sekvenčne.

SAX parser poskytuje oproti DOM parserom určité výhody. SAX parser oznamuje len aktuálnu parsovaciu udalosť zavolaním príslušného návratového volania a neuschováva takmer žiadne informácie predtým získané. Z tohto dôvodu je minimum pamäte potrebnej pre SAX parser priamo úmerná hĺbke stromu XML súboru a predstavuje len zlomok z pamäte potrebnej pre DOM parser, ktorý najprv v pamäti vyskladá celý strom.[25]



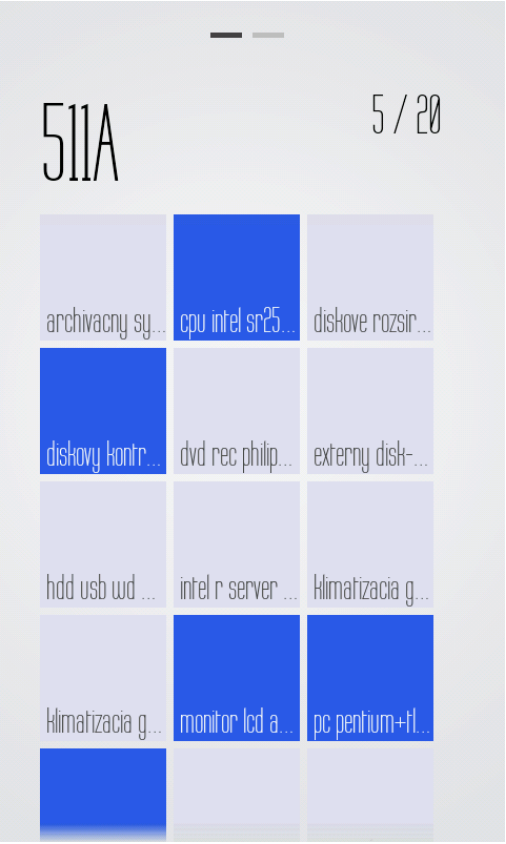
Obr. 19 Fungovanie SAX parsera[26]

7.3 Čítanie QR kódov

Po vytvorení objektov miestností a položiek môže samotná inventúra začať. Samotná kontrola stavu inventára bude vykonaná načítaním QR kódov umiestnených fyzicky na predmetoch. Načítavanie obrazu kódov je realizované pomocou funkcie kamery fotoaparátu mobilného telefónu, ktorý po spustení kamery stačí namieriť na kód a ten bude automaticky rozpoznaný a načítaný. Rozpoznanie a načítanie obsahu QR kódu je realizované pomocou knižnice ZXing.

Používateľské rozhranie miestnosti v ktorej prebieha inventúra, z dôvodu najlepšieho využitia plochy obrazovky bol použitý štýl zobrazenia položiek do formy matice. Dôležitým dizajnovým prvkom je farba samotnej položky, ktorých význam je nasledujúci:

- **Tmavomodrá farba** – označuje položku, ktorá je prítomná na sklade
- **Svetlomodrá farba** – označuje položku, ktorá na sklade prítomná nie je



Obr. 20 Ukážka používateľského rozhrania miestnosti počas inventúry

7.3.1 Knižnica ZXing

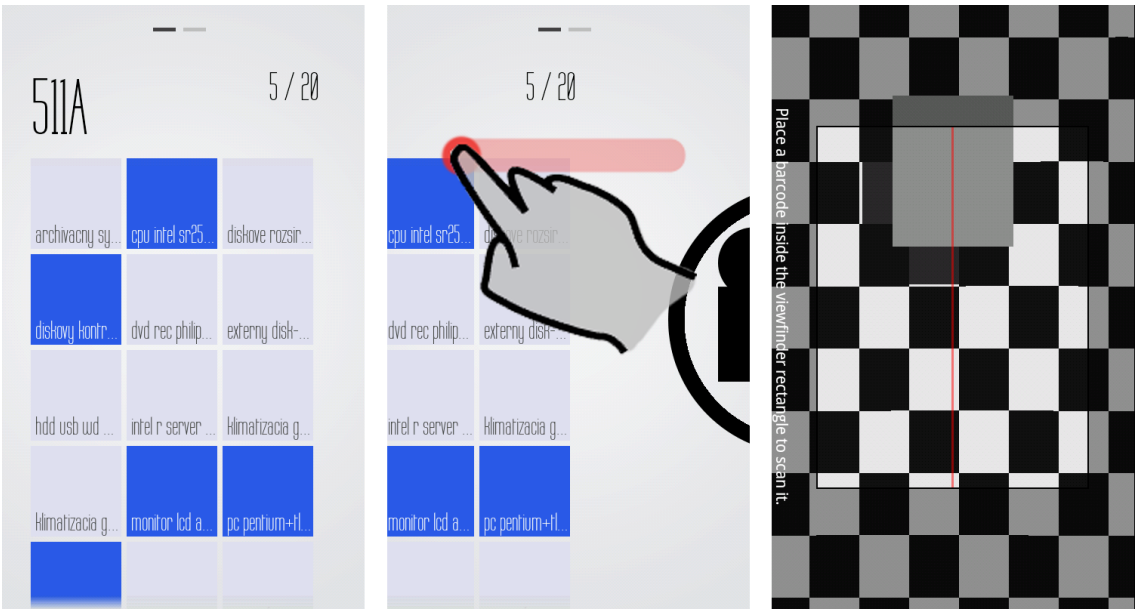
ZXing je open-source knižnica podporujúca spracovanie obrazu multiformátových kódov. Knižnica je implementovaná v jazyku Java a je portovaná do iných jazykov. Jej zámerom je využívať zabudované kamery v mobilných telefónoch pre zoskenovanie a dekódovanie čiarových kódov priamo na zariadení, bez komunikácie so serverom. V súčasnosti podporuje nasledujúce formáty kódov:[27]

- UPC-A a UPC-E
- EAN-8 a EAN-13
- Kód 39, 93 a 128
- ITF
- Codabar
- RSS-14
- QR kód
- Dátovú maticu (z angl. Data Matrix)
- Aztec
- PDF 417

7.3.2 Spustenie kamery

Spustenie kamerového modulu knižnice ZXing je realizované štartom novej aktivity skenera pomocou volania metódy `startActivityForResult()`, ktorého parametrom bude správa typu Intent s argumentom `com.google.zxing.client.android.SCAN`, ktorý predstavuje názov balíčka ZXing knižnice a príslušnej triedy, ktorá v ňom zabezpečuje skenovanie.. Následne bude spustená aktivita skenera, ktorý zosníma a dekóduje daný QR kód.

Dekódovaný obsah QR kódu bude volajúcej aktivite RoomInventory predaný v metóde `onActivityResult()`. Celý obsah QR kódu bude vrátený v podobe reťazca a predaný ďalej triede `Parser`.



Obr. 21 Ukážka spustenia skenera

7.3.3 Formát obsahu QR kódu

Pre správnu interpretáciu programom, musí byť QR kód, respektíve jeho obsah v nasledujúcom tvare:

EVID.C.: *<nové identifikačné číslo>*
Stare_C.: *<staré identifikačné číslo>*
Opis: *<opis>*
Kusov: *<počet kusov>*
Miestnost: *<meno miestnosti>*

Trieda `Parser` kontroluje avšak kvôli rýchlosti len prvý riadok pomocou regulárneho výrazu. Ak je daný vstup v korektnom tvare, je z neho extrahované jeho nové identifikačné číslo. Toto číslo je následne v cykle porovnávané s novými identifikačnými číslami položiek v načítanej miestnosti. Ak nastala zhoda, je o tomto používateľ patrične upovedomený a nájdená položka, ktorá nebola predtým ešte nájdená, bude v používateľskom rozhraní príslušne označená ako nájdená.

Obr. 22 Príklad QR kódu položky z miestnosti 511A

7.4 Výsledok inventúry

Po stlačení tlačidla pre ukončenie inventúry, poskytuje aplikácia viacero možností ako exportovať jej výsledky. Aplikácia vytvorí triedou `ResultReport` výslednú správu vo formáte HTML a ktorá bude pripravená na jej zaslanie na stranu servera.

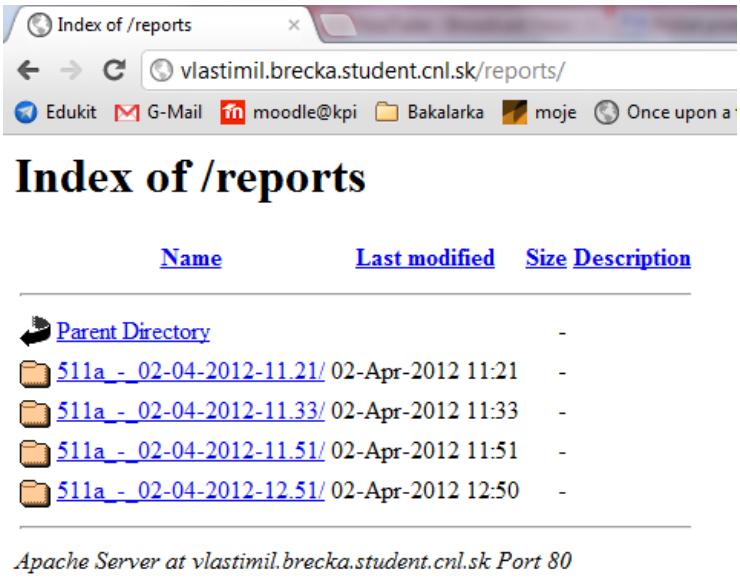
Obr. 23 Ukážka používateľského rozhrania ukončenia inventúry

7.4.1 Archivácia výsledkov

Zaslanie správy o výsledkoch inventúry je realizované asynchrónne pomocou rozšírenia triedy `AsyncTask`, cez protokol HTTP. Na strane servera, danú správu prijme skript vo formáte PHP, ktorý danú správu uloží na disk do adresáru `reports` pre ďalšiu archiváciu. Názov súboru správy je v tvare:

<názov miestnosti>_-_<deň>-<mesiac>-<rok>-<hodina>.<minúta>

Značí prehľadne kde a kedy bola vykonaná daná inventúra o ktorej správa hovorí.



Obr. 24 Ukážka štruktúry adresáru Reports

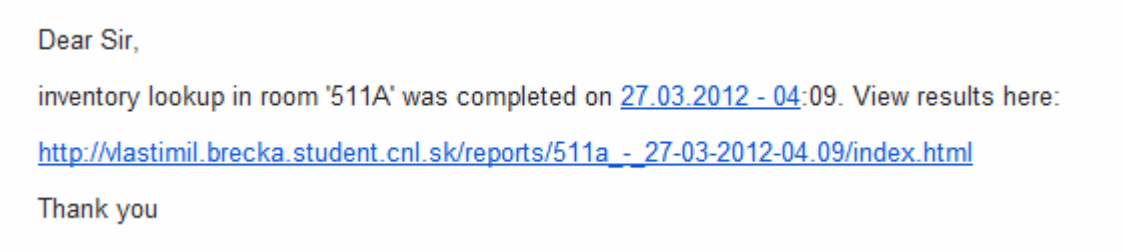
Ak nie je prítomné v danom okamihu pripojenie na internet alebo pre dôkladnejšie zálohovanie, je možné výslednú správu uložiť lokálne na SD-kartu.

Správa obsahuje graficky prehľadne rozlíšené položky, ktoré sú a nie sú na sklade, ich počet a dátum vykonanej inventúry.

Obr. 25 Ukážka HTML správy v prehliadači

7.4.2 Notifikácia

O skutočnosti ukončenia inventúry je možné notifikovať vybrané osoby pomocou e-mailu. Stlačením tlačidla notifikácií budú rozoslané notifikácie na prednastavené e-mailové adresy. Text notifikácie obsahuje dátum inventúry a odkaz na správu uloženú na serveri vo formáte HTML.


Dear Sir,

inventory lookup in room '511A' was completed on 27.03.2012 - 04:09. View results here:

http://vlastimil.brecka.student.cnl.sk/reports/511a_-_27-03-2012-04.09/index.html

Thank you

Obr. 26 Text e-mailovej notifikácie

8 Záver

Úlohou bakalárskej práce bolo oboznámiť sa so štruktúrou a fungovaním platformy Android. Naštudovaním možností vytvárania aplikácií pre túto platformu bol vybraný vhodný postup riešenia. Použitím týchto znalostí bol vytvorený a implementovaný návrh aplikácie inventárneho systému.

Analýzou návrhu bolo zvolené vývojové prostredie Eclipse. Poskytnutá aplikácia plne spĺňa požiadavky zadania na funkcionalitu a poskytuje riešenia pre archiváciu výsledkov inventúr.

Vývoj aplikácie bol zväčša bezproblémový, kde vyskytnuté problémy boli časté a bežné pri začiatkoch s vývojom pre platformu Android, plynúce z nedostatočného naštudovania dokumentácií.

Testovanie riešenia bolo nazačiatku vykonávané na virtuálnom zariadení, ktoré je ale veľmi pomalé a nie je na ňom možné testovať funkciu kamery a dekódovania obrazu. Neskôr bolo zapožičané fyzické zariadenie a s ním projekt výrazne pokročil a urýchlil sa vývoj. Toto je jedným so zistených záverov práce, ktoré hovorí o nutnosti testovania vyvíjanej aplikácie na fyzickom zariadení. Emulátor je stále vhodným prvkom pri vývoji, keďže je na ňom možné simulovať chovanie aplikácie pri rôznych rozlíšeniach displeja a rôznych iných podmienkach. Aktuálne bola pridaná možnosť grafickej akcelerácie virtuálneho zariadenia, ktorá maže výkonnostné rozdiely ale stále sa považuje za vhodné vyskúšať aspoň finálnu verziu na reálnom telefóne.

Vhodným rozšírením, ktoré prekračuje rozsah tejto práce by bolo vytvorenie centrálneho autentifikačného systému, ktorý by umožnil vytváranie používateľských kont a obslužnej stolnej aplikácie, ktorá by umožňovala spravovať inventárne zoznamy jednotlivých používateľov na vzdialenom serveri, respektíve použitie databázy, aby sa predišlo prístupu používateľa k samotným XML súborom a predišlo sa tak parsovacím chybám.

Nadviazanie na túto prácu v budúcnosti by mohlo byť použitie technológie NFC, ktorá sa pomaly stáva štandardom a jej nenáročnosť a iné vlastnosti, ako napríklad nezávislosť od svetelných podmienok a kvality kamery mobilného telefónu. Toto riešenie však nie je v súčasnosti možné, keďže NFC anténa je zabudovaná len v niekoľkých modeloch.

Zoznam použitej literatúry

[1] Denso Wave Inc.: O 2D QR kóde. [online]. [cit. 2012-5-11]. Dostupné na internete: <http://www.denso-wave.com/qrcode/aboutqr-e.html>.

[2] Denso Wave Inc.: Vlastnosti QR kódu. [online]. [cit. 2012-5-11]. Dostupné na internete: <http://www.denso-wave.com/qrcode/qrfeature-e.html>.

[3] FUHRT, Borko: Handbook of Augmented Reality. Florida: Springer Science, 2011. 746s. ISBN 978-1-4614-0063-9.

[4] Korekcia chýb QR kódu. [online]. Aktualizované 2012-5-9 [cit. 2012-5-11]. Dostupné na internete: <http://en.wikipedia.org/wiki/QR_code>.

[5] SHARABY, Orli: Form Meets Function – Extreme Makeover QR Code Edition. [online]. [cit. 2012-5-11]. Dostupné na internete: <http://blog.360i.com/emerging-media/creative-qr-codes>.

[6] Open Handset Alliance: FAQ. [online]. Aktualizované 2007-11 [cit. 2012-5-11]. Dostupné na internete: <http://www.openhandsetalliance.com/oha_faq.html>.

[7] Open Handset Alliance: Industry Leaders Announce Open Platform for Mobile Devices. [online]. Aktualizované 2007-11-5 [cit. 2012-5-11]. Dostupné na internete: <http://www.openhandsetalliance.com/press_110507.html>.

[8] Android Logo. [online]. [cit. 2012-5-11]. Dostupné na internete: <http://www.talkandroid.com/9717-android-dominating-2010/android-logo-white/>.

[9] Architektúra systému Android. [online]. [cit. 2012-5-9]. Dostupné na internete: <http://developer.android.com/images/system-architecture.jpg>.

[10] Architektúra. [online]. Aktualizované 2012-5-3 [cit. 2012-5-9]. Dostupné na internete: <http://sk.wikipedia.org/wiki/Android_(opera%C4%8Dn%C3%BD_syst%C3%A9m)>.

[11] Tools. [online]. Aktualizované 2012-9-5 [cit. 2012-5-9]. Dostupné na internete: <http://developer.android.com/guide/developing/tools/index.html>.

[12] Android API levels. [online]. Aktualizované 2012-5-9 [cit. 2012-5-11]. Dostupné na internete: <http://developer.android.com/guide/appendix/api-levels.html>.

[13] Android Developer Tools. [online]. Aktualizované 2012-5-9 [cit. 2012-5-11]. Dostupné na internete: <http://developer.android.com/guide/developing/tools/adt.html>.

[14] Android Debug Bridge. [online]. Aktualizované 2012-5-9 [cit. 2012-5-11]. Dostupné na internete: <http://developer.android.com/guide/developing/tools/adb.html>.

[15] HIMANSHU: How to Run Android Froyo on Windows 7 Computer. [online]. Aktualizované 2011 [cit. 2012-5-11]. Dostupné na internete:

<http://www.blogtechnika.com/how-to-run-android-froyo-on-windows-7-computer>.

[16] Managing Virtual Devices. [online]. Aktualizované 2012-5-9 [cit. 2012-5-11]. Dostupné na internete: <http://developer.android.com/guide/developing/devices/index.html>.

[17] GRAMLICH, Nicolas: Andbook: Android Programming. [online]. [cit. 2012-5-11]. Dostupné na internete: <http://andbook.anddev.org/>.

[18] Activies. [online]. Aktualizované 2012-5-9 [cit, 2012-5-11]. Dostupné na internete: <http://developer.android.com/guide/topics/fundamentals/activities.html>.

[19] Services. [online]. Aktualizované 2012-5-9 [cit. 2012-5-11]. Dostupné na internete: <http://developer.android.com/guide/topics/fundamentals/services.html>.

[20] Application Fundamentals. [online]. Aktualizované 2012-5-9 [cit. 2012-5-11]. Dostupné na internete: <http://developer.android.com/guide/topics/fundamentals.html>.

[21] Intents and Intent Filters. [online]. Aktualizované 2012-5-9 [cit. 2012-5-11]. Dostupné na internete: http://developer.android.com/guide/topics/intents/intents-filters.html

[22] Google+. [online]. Aktualizované 2012 [cit. 2012-5-11]. Dostupné na internete: <https://play.google.com/store/apps/details?id=com.google.android.apps.plus>.

[23] User Interface. [online]. Aktualizované 2012-5-9 [cit. 2012-5-11]. Dostupné na internete: <http://developer.android.com/guide/topics/ui/index.html>

[24] Thread with Async Task & Progress Bar. [online]. Aktualizované 2010-3-11 [cit. 2012-5-11]. Dostupné na internete: <http://tigerwoods.tistory.com/28>.

[25] About SAX [online]. Aktualizované 2004-4-27 [cit. 2012-5-7]. Dostupné na internete: <http://www.saxproject.org/about.html>.

[26] Java Mapping Concepts (DOM and SAX). [online]. Aktualizované 2009-2-11 [cit. 2012-5-11]. Dostupné na internete: <http://wiki.sdn.sap.com/wiki/display/XI/Java+Mapping+Concepts+(DOM+and+SAX)>.

[27] ZXing („Zebra Crossing“). [online]. [cit. 2012-5-7]. Dostupné na internete: <http://code.google.com/p/zxing/>.

Prílohy

Príloha A: CD médium.

Príloha B: Používateľská príručka

Príloha C: Systémová príručka