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

SÍŤOVÁ HRA V NOKIA QT SDK

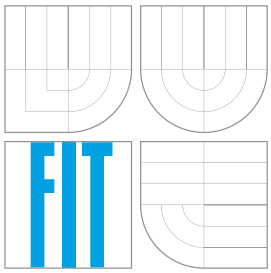
BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

TOMÁŠ PRAMUKA

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

SÍŤOVÁ HRA V NOKIA QT SDK

NETWORK GAME IN NOKIA QT SDK

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — TOMÁŠ PRAMUKA
AUTHOR

VEDOUCÍ PRÁCE — Ing. DAVID BAŘINA
SUPERVISOR

BRNO 2012

Abstrakt

Práce se zaobírá návrhem aplikace, síťové hry v Nokia Qt SDK pro mobilní zařízení Nokia N900. Aplikace umožňuje hrát hru dvěma hráčům proti sobě v reálném čase. Práce obsahuje informace o mobilním zařízení Nokia N900, o jeho hardwarové a softwarové části, informace o vývoji aplikací v Nokia Qt SDK a o jednotlivých častech a nástrojích SDK. Součástí práce je návrh aplikace.

Abstract

This work is about application design of network game, developed in Nokia Qt SDK for the mobile device Nokia N900. Application afford opportunity to play game in real time for two players. Work contains informations about mobile device Nokia N900, both hardware and software part, informations about developing applications in Nokia Qt SDK and informations about tools and features of Nokia Qt SDK. Work also contains application design.

Klíčová slova

Qt, Nokia, N900. Mobility API, Qt Simulator, síťová hra, pro více hráčů

Keywords

Qt, Nokia, N900. Mobility API, Qt Simulator, network game, multiplayer

Citace

Tomáš Pramuka: Síťová hra v Nokia Qt SDK, bakalářská práce, Brno, FIT VUT v Brně, 2012

Síťová hra v Nokia Qt SDK

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Davida Bařiny. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Tomáš Pramuka
16. května 2012

Poděkování

Ďakujem za trpezlivosť pri vypracovávaní mojej práce vedúcemu Ing. Davidovi Bařinovi.

Obsah

Kapitola 1

Úvod

Mobilné telefóny prešli od začiatku svojho vývoja obrovskými zmenami. Vďaka technologickému pokroku sa zmenšovala ich veľkosť a zároveň sa zvyšoval ich výkon a využiteľné možnosti. Mobilné telefóny prestali byť iba telefónmi, stali sa výkonnými vreckovými počítačmi a okrem vytvárania hovorov zvládajú veľké množstvo rozdielných funkcií, ako prehrávanie hudby či videa, pripojenie a práca s internetom a podobne. Medzi tieto funckie sa radia v neposlednom rade aj hry. Ako prvá hra na mobilné telefóny bola vytvorená, dnes už kultová, hra Snake. Výkonom mobilných telefónov rástla aj náročnosť hier na grafický a výpočtový výkon zariadenia. V dnešnej dobe existuje celá plejáda rôznych hier na rôznych mobilných platformách. Väčšina hier je postavená na jednoduchom princípe, často bez možnosti hrať proti protivníkovi.

Cieľom tejto bakalárske práce je navrhnúť a implementovať sieťovú hru pre dvoch hráčov na báze hry Bomberman od firmy Hudsonsoft. Výsledná aplikácia by mala poskytovať možnost prepojenia dvoch hráčov pomocou internetu. Podmienkou pre hru je mať funkčné internetové pripojenie s minimálnym, ideálne žiadnymi výpadkami pre hladký beh hry. Aplikácia je primárne navrhnutá pre zariadenie Nokia N900, ktoré som mal k dispozíci pri tvorbe tejto práce.

Obrázek 1.1: Pôvodná hra Bomberman

Čitateľ v kapitole 2 nájde popis mobilného zariadenia Nokia N900, popis hardwarovej aj softwarovej časti. V kapitole 3 sú uvedené informácie o vývoji aplikácii v Nokia Qt SDK a popis jednotlivých častí SDK. V kapitole 4 je popísaný návrh aplikácie a v záverečnej 5. kapitole je zhrnutie pracovného procesu.

Kapitola 2

Prostredie Nokia Qt SDK a Nokia N900

2.1 Nokia N900

V tejto kapitole sú uvedené podrobné informácie o hardwarovej a softwarovej vybavenosti mobilného telefónu Nokia N900. Po stručnej charakteristike prístroja je popísané vyhotovenie telefónu a displej. Ďalej v texte sú uvedené informácie o procesoroch a pamäťových kapacitách. Ovládanie a hardwarové rozšírenia sú popísané na konci hardwarovej časti. Softwarová časť obsahuje informácie o operačnom systéme Meamo 5 Fremantle.

2.2 Hardware zariadenia N900

Nokia N900 je chytrý telefón (smartphone) vyrobený firmou Nokia, ktorý vychádza z modelu Nokia N810. Nokia uviedla model N900 na trh v novembri 2009. Nokia N900 poskytuje pokročilé funkcie mobilného internetového zariadenia, z ktorého je možný jednoduchý a rýchly prístup na web, e-mail a prístup k ostatným službám vyžadujúcim internetové pripojenie. Telefón je schopný prehrávať multimédia (hudba, video, internetové rádio) pomocou obsiahnutého prehrávača médií a zobrazovať obrázky. Telefón si uložené multimédia a obrázky sám vyhľadá v súborovom systéme telefónu. N900 má zabudovaný 5 megapixelový digitálny fotoaparát pre fotografovanie a nahrávanie videa. N900 je samozrejmá prijímať a vytvárať hovory a SMS či MMS ako bežný telefón. Nokia N900 [4][3] je vyrobená v bočne výsuvnom vyhotovení s rozmermi 110,9 mm $\times$ 58,8 mm $\times$ 19,55 mm a hmotnosťou spolu s batériou 191 gramov a objemom približne 113 cm^3. Obsahuje dotykový displej, bočne výsuvnú klávesnicu a stojanček.

Nokia N900 obsahuje dotykový displej s uhlopriečkou 3,5“ a rozlíšením 800 $\times$ 480 pixelov. Režim zobrazenia sa dá prepínať medzi režimom na šírku a režimom na výšku, ktorý je využívaný pri hlasových hovoroch. Prepínanie medzi režimami na šírku a na výšku zabezpečuje 3-osový akcelerometer, ktorého funkcie sú vyžívané aj na snímanie užívateľského vstupu v niektorých aplikáciách. Pracovná plocha obsahuje 4 prispôsobiteľné zobrazenia, medzi ktorými sa dá jednoducho prepínať. Akcelerácia 3D grafiky s podporou OpenGL ES 2,0[4] je zabezpečená grafickým procesorom *PowerVR SGX 530* od firmy Imagination Technologies. Displej je schopný zobraziť 65 000 farieb. Výpočtová sila Nokie N900 je zabezpečená *OMAP 3430* [6] SoC (system on chip, teda integrovaný obvod, ktorý integruje počítačové a ostatné elektronické systémy do jediného čipu) vyrobený firmou Texas instru-

Obrázek 2.1: Nokia N900 s uvedenými rozmermi a vysunutou QWERTY klávesnicou

ments. OMAP 3430 je zostavený z troch mikroprocesorov. Mikroprocesor *ARM Cortex A8* s frekvenciou 600 MHz (pri správnom pretaktovaní, môže frekvencia dosiahnuť až 1,15 GHz) zabezpečuje beh operačného systému a ostatných aplikácií. Mikroprocesor *TMS320 C64x* s frekvenciou 430 MHz slúži na spracovanie obrazu z fotoaparátu, spracovanie zvuku a prenos dát. Hlavný dôvod používania TMS320 C64x je v odľahčení ARM Cortex A8 mikroprocesoru od spracovávania audio a video signálov. Poslednou časťou OMAP 3430 je grafický procesor PowerVR SGX 530 spomenutý v časti o displeji.

Telefón využíva 256 MB operačnej pamäte RAM mDDR(Mobile DDR) a až 768 MB ako swap spravovaných operačným systémom [4], čím nám spolu telefón ponúka až 1 GB virtuálnej pamäte. Nokia N900 obsahuje 32 GB eMMC (Embedded MultiMediaCard) ako vnútorné úložisko dát s možnosťou rozšírenia o ďalších až 16 GB pomocou microSD karty. 32 GB eMMC karta je rozdelená na 3 partície:

1. 2 GB ext3 partícia pripojená k /home
2. 768 MB ako swap
3. zvyšných cca. 27 GB voľného miesta je pripojených k /home/user/MyDocs ako VFAT

Okrem 32 GB eMMC karty telefón obsahuje 256 MB NAND neodstrániteľné úložisko, formátované ako UBIFS a obsahuje bootloader, jadro a koreňový adresár /. Nokia N900 obsahuje niekoľko spôsobov ovládania. Hlavným ovládacím prvkom je dotykový displej, na ktorom sa dá zobraziť virtuálna klávesnica, alebo plnohodnotná vysúvacia klávesnica QWERTZ. Operačný systém obsluhuje predikciu slov a môže byť nakonfigurovaný užívateľom (nakonfigurovať sa dá doplňanie slov, automatické medzery medzi slovami a automatické veľké písmena na začiatku vety). Niektoré funkcie a nastavenia telefónu sa dajú ovládať špeciálnymi tlačidlami vytvorenými práve pre jednu funkciu. Tieto vyhradené tlačidlá ovládajú zoom, hlasitosť, spúšť fotoaparátu a uzamykanie telefónu. Na panely, vľavo od dotykového dispeja, sa nachádza snímač osvetlenia, snímač priblíženia, objektív sekundárneho fotoaparátu, svetelný indikátor a slúchadlo telefónu. 5 megapixelový digitálny fotoaparát na zadnej strane prístroja slúži na zachytávanie statického obrazu alebo videa. Fotoaparát obsahuje automatické zameriavanie, dvojitý LED blesk, trojnásobný digitálny zoom a

možnosť využívať pomer strán 4:3 alebo 16:9. Nahrávanie videa je možné v rozlíšení 848 x 480 pixelov a 25 fps (frames per seconds, obrázkov za sekundu). N900 má mikrofón na nahrávanie zvuku a stereo reproduktory umiestnené na každej strane zariadenia. Na pripojenie slúchadiel, alebo externých reproduktorov obsahuje telefón 3.5 mm TRRS konektor. Zariadenie využíva frekvencie EGSM 850/900/1800/1900 alebo WCDMA 900/1700/2100. Telefón sa môže pripojiť na bezdrôtovú sieť pomocou WLAN IEEE 802.11b/g so zabezpečením WEP, WPA alebo WPA2. N900 obsahuje rozhranie Bluetooth verzie 2.1. Pomocou Bluetooth rozhrania je možné využiť stereo audio výstup s A2DP profilom (advanced audio distribution profile), teda streamovať audio výstup z jedného zariadenia na druhé, napríklad z telefónu do bezdrôtových slúchadiel. V aute sa dá využívať HFP(hands-free profile) Bluetooth profil. Prenos súborov cez FTP je podporovaný spolu s OPP profilom pre posielanie a prijímanie objektov. Bluetooth rozhranie sa používa aj ako FM prijímač, dovoľujúci počúvať FM rádia na frekvenciách od 88.1 do 107.9 MHz.

2.3 Software zariadenia N900

Maemo 5 Fremantle [5] je defaultný operačný systém pre telefón Nokia N900, ktorý vychádza z linuxovej distribúcie Debian. Maemo 5 predstavuje konzistentné užívateľské prostredie, ktoré umožňuje využívať viac prstové gestá a GUI X-server systém okien, ktorý je postavený na Xorg. Maemo je postavený na jadre linuxových operačných systémov, pretože linux podporuje veľkú škálu hardwarových platforiem a je používaný v rôznych typoch zariadení od náramkových hodiniek až po veľké serverové systémy. Meamo je postavené na operačnom systéme Linux 2.6. Linuxové jadro poskytuje abstraktnú hardwarovú vrstvu pre systémové zariadenia, pre správu pamäte, procesov a súborov, pre sieťové služby a pre mnoho ďalších. Architektúra užívateľského prostredia operačného systému Maemo 5 je postavená na frameworku GNOME a využíva hlavne GTK+ widget set. Maemo zdedil z GNOME frameworku GTK+, GStreamer multimediálny framework, GConf konfiguračný manažment a XML knižnicu. Maemo teda poskytuje rozšírenie GTK+/GNOME technológie pre mobilné pracovné plochy využívaním knižnice Hildon, ktorá je nadstavbou GNOME, práve pre mobilné zariadenia. Maemo je postavené na štandardnej GNU C knižnici. Pre sieťové pripojenia používa OpenSSL knižnicu, ktorá poskytuje sieťovú ochranu a libcurl knižnicu poskytujúcu HTTP prístup pre aplikácie. Maemo poskytuje Hardware Abstraction Layer (HAL) pre abstrakciu hardwaru. HAL poskytuje zdieľanú knižnicu, ktorá obsahuje API pre jednotlivé zariadenia. HAL okrem tejto funkcie zavádza správny ovládač nového pripojeného zariadenia, spravuje a vytvára /dev súbory atď. Komunikačný kanál medzi aplikáciami je zaistený softwarom DBUS [1], ktorý poskytuje jednoduchý spôsob ako medzi sebou komunikujú aplikácie. DBUS zaisťuje aj komunikáciu medzi systémom a aplikáciami. DBUS je naimplementovaný ako démon, ktorý sa dá spustiť vo viacerých inštanciách (kanáloch). Inštancia DBUS-u môže byť buď privátna, teda medzi dvoma aplikáciami, alebo systémová, ktorá spočíva v doručovaní signálov od HAL démona príslušným aplikáciám. Systém ponúka SQL databázu SQLite 3 na ukladanie užívateľských dát a je prístupná cez rozhranie knižnice. Maemo launcher spúšťa všetky aplikácie na platforme Maemo so snahou o zrýchlenie štartu aplikácie zdieľaním niektorých inicializačných dát. Je zložený z dvoch častí:

1. Maemo-invoker, je spúšťaný DBUS démonom, alebo skriptom a žiada Maemo-launcher o spustenie požadovanej aplikácie.

2. Maemo-launcher, server, ktorý inicializuje väčšinu dát používaných aplikáciami.

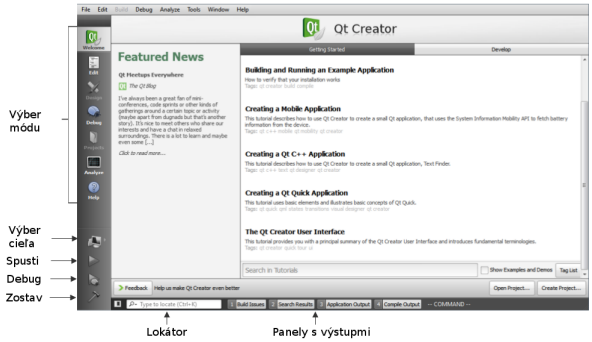
Nokia N900 má predinštalované aplikácie na prehliadanie web stránok (MicroB web browser, RSS čítačku), aplikácie pre podporu médii (fotoaparát, prezeranie fotiek a multimediálny prehrávač), systémové nástroje (súborový manažér, manažér aplikácií), rôzne utility (napríklad: hodiny, kalkulačka, kalendár, pdf reader a mnoho ďalších), hry a OVI maps. Ostatná softwarová výbava sa dá stiahnuť a nainštalovať a jedná sa hlavne o aplikácie vytvárané treťou stranou.

2.4 Nokia Qt SDK

V kapitole sú uvedené informácie o Nokia Qt SDK, ktorý je použitý pri implementácii výslednej aplikácie. Po charakteristike vývojového kitu je popísaný program *qmake* ktorý je používaný pre vytváranie aplikácií nezávislých na platforme. Nasleduje popis nástrojov Nokia Qt SDK a to nástroja Qt Simulator a knižnice Qt Mobility a jej častí. Qt je knižnica, nezávislá na platforme, ktorá je hlavne používaná na vytváranie softwarových aplikácií s užívateľským grafickým prostredím ,vyvíjaná oddelením Qt Development Frameworks vo fínskej firme Nokia od roku 2008, ktorá Qt zakúpila od pôvodného tvorcu, nórskej firmy Trolltech. Trolltech vytvoril Qt toolkit v roku 1999. Okrem GUI aplikácií sa dá použiť aj na vytváranie konzolových aplikácií. Qt je multiplatformný framework, teda sa dá využiť na vytváranie aplikácií na rôznych operačných systémoch a pre rôzne operačné systémy (Windows, Linux, MacOS). Okrem desktopových aplikácií podporuje vytváranie mobilných aplikácií pre mobilné operačné systémy Symbian, Maemo a MeeGo. Qt je knižnica pre programovací jazyk C++, ale existuje aj pre jazyky Python (PyQt), Ruby (QtRuby), C, Perl, Pascal, C#, Java (Jambi) a Haskell. Nokia Qt SDK (software development kit) obsahuje Qt framework, Qt Creator IDE (integrated development environment) a rôzne nástroje (simulátor, lokálny alebo vzdialený kompilátor a podobne).

2.5 Tvorba aplikácií v Qt Creator IDE

Qt Creator [7] 2.2 je integrované vývojové prostredie pre tvorbu aplikácií pomocou frameworku Qt. Qt Creator je multiplatformné IDE, ktoré ponúka mnoho funkcií uľahčujúcich písanie kódu.

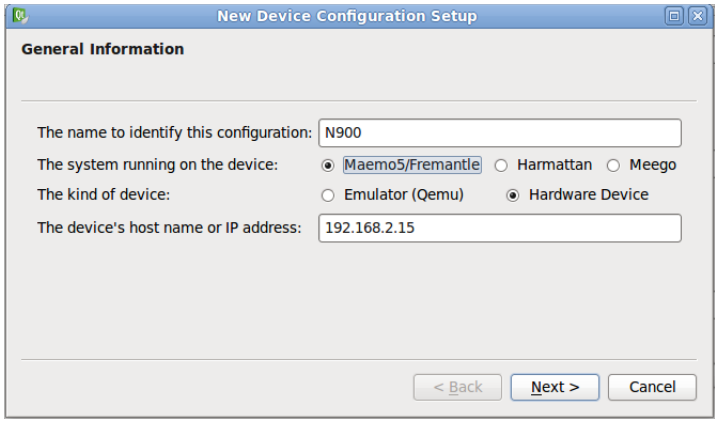


Obrázek 2.2: Screenshot s popisom IDE Qt Creator

Okrem integrovaného nástroja pre jednoduchú tvorbu schém a formulárov pre grafické užívateľské prostredie ponúka aj grafický debugger, zvýrazňovanie syntaxu zdrojového kódu pre lepšiu prehľadnosť a automatické dopĺňanie textu. Tvorba aplikácií prebieha vo vytvorených alebo importovaných projektoch, ktoré združujú všetky zdrojové kódy vytváranej

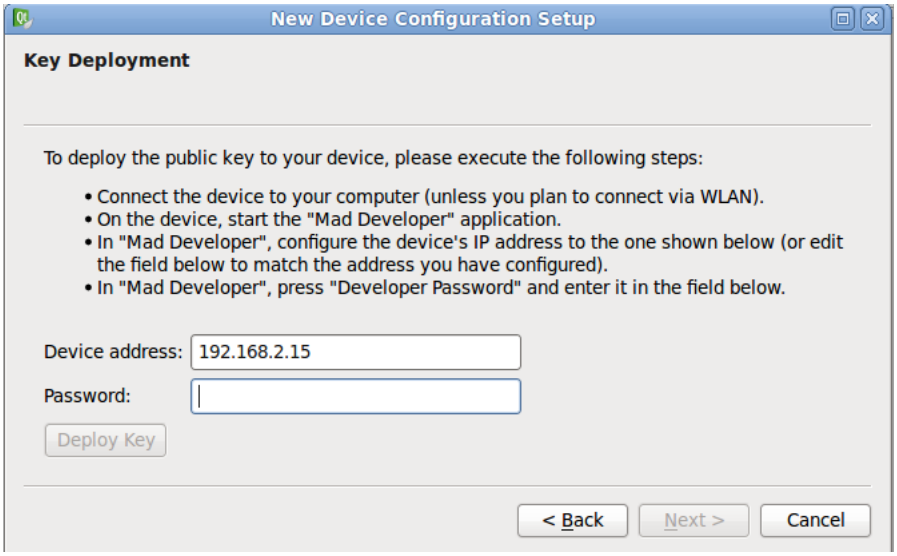
aplikácie, formuláre a ostatné potrebné súbory do prehľadného stromového zobrazenia. Pri vytváraní nového projektu QtCreator ponúka predpripravené šablóny pre rôzne typy projektov. Pomocou QtCreator-a sme schopní jednoducho a rýchlo začať konzolové projekty, projekty s užívateľským rozhraním, projekty pre mobilné telefóny, C++ knižnice, či doplnky pre samotný QtCreator. Samozrejmosťou je možnosť vytvoriť projekt z rôznych verziovacích systémov naklonovaním repozitárov zo zadanej URL adresy. Po vybratí typu projektu je nutné projekt pomenovať a určiť miesto uloženia na disku. Ďalším krokom je určenie typu zariadenia, cieľa, pre ktoré je aplikácia vyvíjaná. Okrem základného typu, desktopovej aplikácie, je možné vybrať cieľ aplikácie mobilný telefón a QtSimulator. Pokiaľ je cieľom našej aplikácie mobilný telefón, je nutné nainštalovať pomocou nástroja SDK Maintenance Tool nástroje, pre vývoj aplikácií pre rozdielne ciele. QtCreator sa dá jednoducho rozšíriť o Maemo Toolchain, slúžiaci na vývoj aplikácií pre operačný systém Maemo, o MeeGo / Harmattan a o novšie verzie Qt frameworku pre vývoj desktopových aplikácií. Po vybratí cieľov aplikácie je možnosť vybrať nástroj pre verziovanie projektu (Git, Mercurial, apod.). V dialógu sú zobrazené súbory, ktoré sa vytvoria zo šablóny pre daný typ projektu. Tieto súbory tvoria základnú kostru programu, ktorý je možné ihneď po vytvorení preložiť bez pridávania a vytvárania ďalších súborov. Výsledkom je samozrejme zatiaľ prázdny program, ktorý spustí hlavné okno programu.

Pred samotným preložením programu na mobilný telefón je nutné spárovať QtCreator so zariadením, na ktorom sa bude vyvíjať a testovať. Pre tento proces je esenciálna inštalácia programu MAD Developer na mobilné zariadenie. MAD Developer je GUI aplikácia 2.7, ktorá vytvorí na zariadení užívateľský účet *developer*, pod ktorým sa spúšťa vytváraná aplikácia, spravuje USB sieťovanie a vytváranie *developer* hesla. V QtCreator v menu Tools - Options... - Maemo potrebujeme pridať nové zariadenie. Po stlačení tlačítka Add sa zobrazí dialóg so základnými informáciami o zariadení. Po vyplnení názvu konfigurácie , vybratí správneho systému na zariadení, typu zariadenia a jeho IP adresy 2.3 importujeme súkromný kľúč na zariadenie. Je možné použiť už vytvorený kľúč, alebo vytvoriť nový. Po



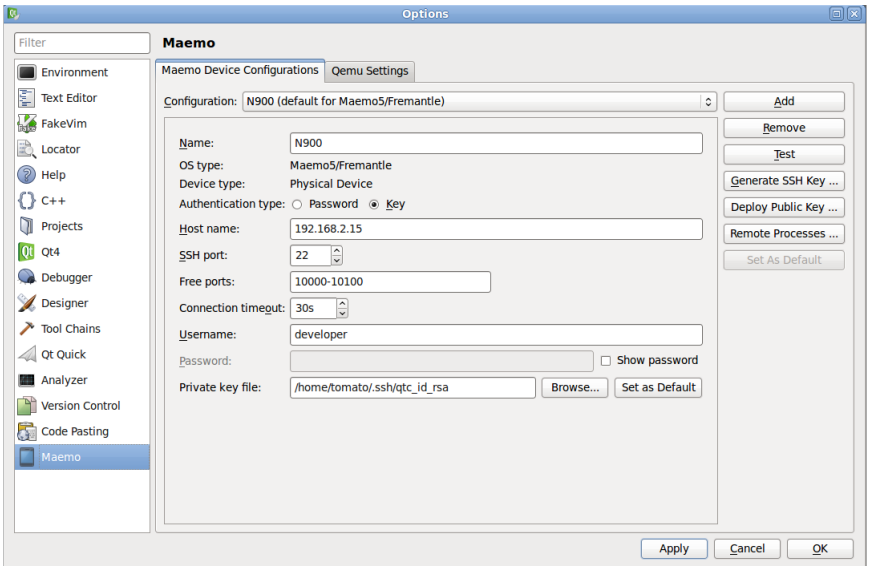
Obrázek 2.3: Zadávanie základných informácií o zariadení

zadaní kľúča sa zobrazí ďalší dialóg, ktorý od nás požaduje vývojárske heslo 2.4. Vývojárske heslo vygeneruje MAD Developer v sekcii *Developer Password*. Vygenerované heslo z MAD Developer-a zadáme do QtCreator-a a následne kliknutím na tlačítko *Deploy Key* heslo nasadíme do užívania.



Obrázek 2.4: Zadávanie Developer hesla

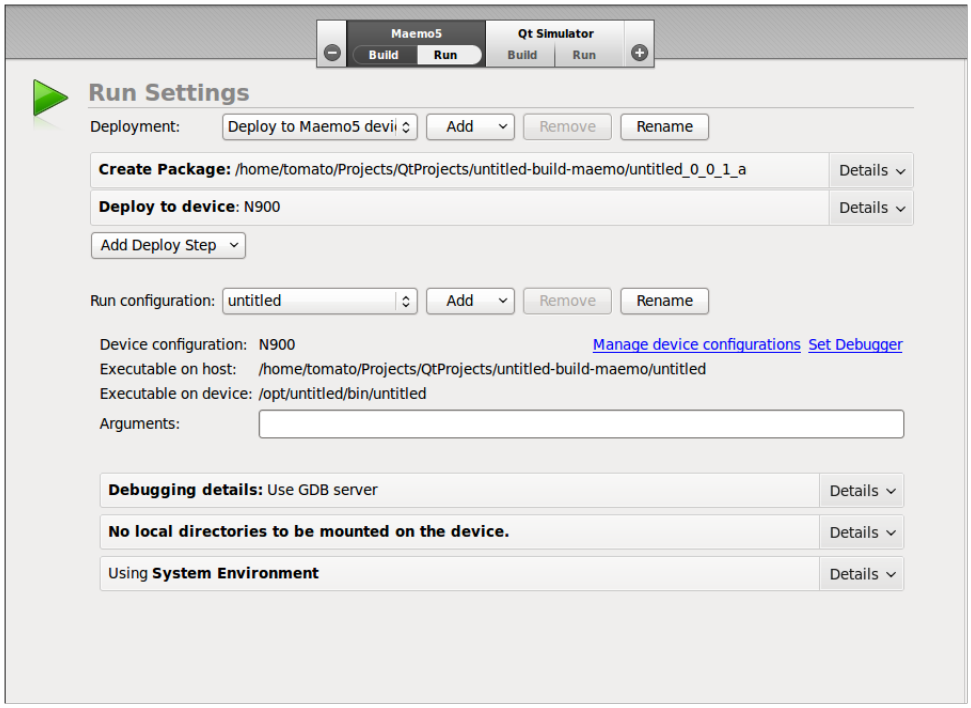
Dokončíme sprievodcu a zariadenie sa otestuje, či obsahuje všetky potrebné knižnice pre vývoj a chod Qt aplikácií. Po úspešnom otestovaní zariadenia je párovanie ukončené. Na obrazovke sa zobrazia zosumarizované nastavenia zariadenia 2.5.



Obrázek 2.5: Obrazovka so zosumarizovanými informáciami o pripojenom zariadení

Po vrátení sa do projektu v paneli s výberom módu vyberieme záložku *Projects*, v ktorej sa zobrazia vybrané cieľové zariadenia. Pri vývoji na mobilný telefón obrazovka bude obsahovať dve cieľové zariadenia a to Maemo (alebo MeeGo / Harmattan) a QtSimulator 2.6. Po vybratí nastavení behu aplikácie pre Maemo 5 vyberieme zariadenie, ktoré sme úspešne spárovali s QtCreator-om. V tomto bode pri výbere cieľa v ľavom paneli môžeme vybrať zariadenie Maemo 5 a aplikáciu preložiť a spustiť.

V prípade neúspešného pripojenia na zariadenie je nutné v MAD Developer-i spustiť



Obrázek 2.6: Nastavenia pre beh a preklad aplikácie na mobilnom zariadení

USB Networking, vybrať USB modul (Unix alebo Windows) a po zobrazení IP adresy je zariadenie plne pripravené. Okrem možnosti prepojiť počítač s mobilným zariadením cez USB, sú k dispozícií pripojenia cez bezdrôtovú sieť, alebo pomocou GPRS služby.

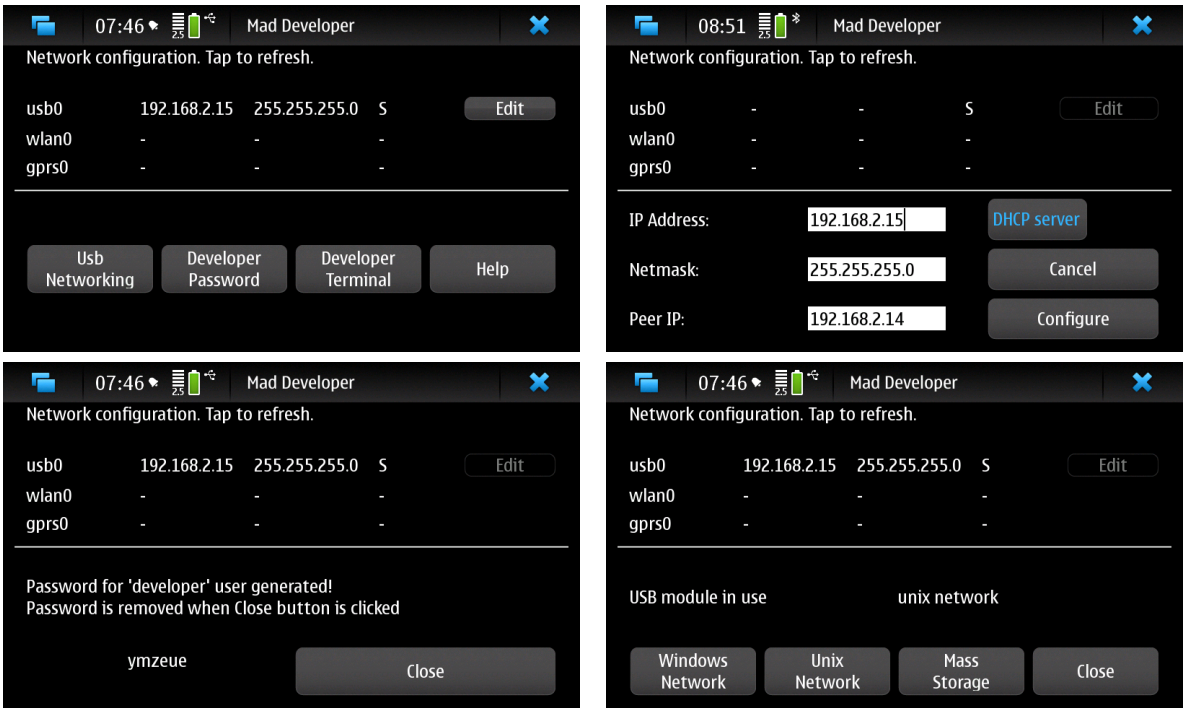
Kompiláciu programu zabezpečuje program *qmake* [2]. Program qmake vytvára projekty nezávislé na platforme a to generovaním Makefile súboru z projektového súboru, ktorý popisuje vytváranú aplikáciu vo forme nezávislej na operačnom systéme na ktorom je aplikácia kompilovaná. Vygenerovaný Makefile súbor obsahuje všetky potrebné informácie pre kompiláciu C++ aplikácie na špecifikovanej platforme. Program qmake rozoznáva niekoľko typov premenných, ktoré môžu byť obsiahnuté v projektovom súbore. Premenná CONFIG obsahuje základné nastavenia projektu, DESTDIR premenná obsahuje cestu k adresáru, do ktorého sa uloží preložený program. FORMS obsahuje zoznam súborov s vytvorenými formulármi pre užívateľské rozhrania, vytvorenými pomocou nástroja UIC (User Interface Creator). Premenná HEADERS obsahuje zoznam hlavičkových súborov s koncovkou .h , ktoré sú použité pri zostavovaní projektu. SOURCES premenná obsahuje zoznam zdrojových súborov. Premenná RESOURCES obsahuje .rc súbory v ktorých je uložený zoznam súborov, ktoré sú zahrnuté do výsledného projektu. Typicky sú v .rc súbore zapísané použité obrázky, animácie, videá a pod. V TEMPLATE premennej je uložená hodnota šablóny, ktorá bola použitá pri vytváraní projektu. Podľa tejto hodnoty sa vytvára aplikácia, knižnica alebo doplnok (plugin). Pokiaľ premenná CONFIG obsahuje hodnotu *mobility* musíme do .pro súboru pridať premennú MOBILITY a v nej sa uchovávajú hodnoty jednotlivých častí Qt Mobility frameworku, ktoré sú použité v aplikácii. Ak uložíme hodnotu *qt* do premennej CONFIG, qmake zapne podporu pre Qt aplikácie. Následne môžeme ukladať do QT premennej možnosti, ktoré symbolizujú použitie jednotlivých modulov QT frameworku.

Možnosť *core* obsahuje modul QtCore, ktorý sprístupňuje celú funkcionalitu, ktorá nie je spojená s grafickým užívateľským rozhraním. Všetky ostatné moduly sú závislé na module QtCore, pretože sprístupňuje základné triedy pre prácu so znakmi, reťazcami, abstraktnými datovými typmi, ukazovateľmi a podobne. Modul QtGui, sprístupnený hodnotou *gui*, obsahuje všetky triedy potrebné pre vytvorenie a zobrazenie grafického užívateľského rozhrania, triedy pre vytváranie a spracovávanie rôznych udalostí a triedy pre zobrazovanie grafických položiek. Sieťový modul QtNetwork sa sprístupňuje pomocou hodnoty *network* a obsahuje triedy pre sieťovú komunikáciu. QtNetwork umožňuje vytvárať sieťové programy jednoducho a bez rozdielu medzi unixovými a windowsovými socketmi. Hodnota *opengl* sprístupňuje modul QtOpenGL. QtOpenGL je API pre renderovanie 3D grafiky , ktorý ale neposkytuje žiadnu podporu pre programovanie užívateľského rozhrania. Ak využívame databázu na permanentné ukladanie dát je potrebné do premennej QT uložiť hodnotu *sql*, ktorá sprístupní triedy v module QtSql. Modul slúži na vytvorenie a prácu s SQL databázou. Podporu pre .svg súbory zabezpečíme pomocou hodnoty *svg* a sprístupnením modulu QtSvg. Hodnota *xml* sprístupňuje QtXml modul pre prácu s xml súbormi. QtXmlPatterns modul slúži pre prácu pre Xpath, Xquery, XSLT a XML Schéma validáciu a sprístupňuje sa hodnotou *xmlpatterns*. V prípade portovania aplikácie z Qt3 do Qt4 je možné použiť hodnotu *qt3support*. Takto sprístupnený modul Qt3Support je podpora pri migrácií aplikácií a nemal by byť používaný pri produkčnom kóde. Projektový súbor nám umožňuje ukladať hodnoty pre rôzne typy operačných systémov, pomocou hodnoty *win32:* pre Windows a pomocou *unix:* pre linuxové distribúcie. Po spustení qmake na počítači s operačným systémom Windows sa vygenerujú 3 súbory: Makefile, Makefile.Debug a Makefile.Release. V tomto prípade je Makefile metasúbor, ktorý odkazuje na zvyšné dva súbory, ktoré popisujú ako make program zloží projekt dohromady. Ak sa spustí qmake na Unixovej platforme vygeneruje sa len jeden súbor, Makefile, ktorý vytvorí spustiteľnú verziu programu, ktorý zahrňuje aj debuggovací výstup po spustení programu make. Ak chceme vytvoriť aj verziu podporujúci debuggovanie, potrebujeme mať nainštalované debuggovacie knižnice. To sa týka Unixových systémov (knižnice s _debug.so v názve), ako aj systému Windows (odpovedajúce DLL knižnice). Po nainštalovaní potrebných knižníc je potrebné nakonfigurovať projekt pridaním riadku CONFIG += debug_and_release. Takto nakonfigurovaný projekt vytvorí dve verzie kompilovaného programu. Jedna verzia podporujúca debuggovanie a druhá verzia bez extra podpory pre hľadanie chýb, vhodná pre uvoľnenie aplikácie pre koncového užívateľa. Tieto verzie zostaveného programu sa dajú vytvárať pomocou príkazov *make debug* alebo *make release*. Qt Creator používajúci multiplatformný Qt framework poskytuje podporu pre zostavovanie a beh aplikácii na desktopových prostrediach a na mobilných zariadeniach. Nastavenia zostavovania aplikácie povoľuje prepínať medzi zostavovacími cieľmi. Ak zostavujem aplikáciu pre mobilné zariadenie, ktoré je pripojené k počítaču na ktorom sa aplikácia vyvíja, Qt Creator automaticky vygeneruje inštalačný balíček, ktorý nainštaluje na zariadenie a spustí nainštalovanú aplikáciu. Qt Creator používa C++ kompilér z kolekcie GNU Compiler na Linuxe a FreeBSD a na systéme Windows využíva MinGW alebo MSVC kompilér.

2.6 Qt Simulator

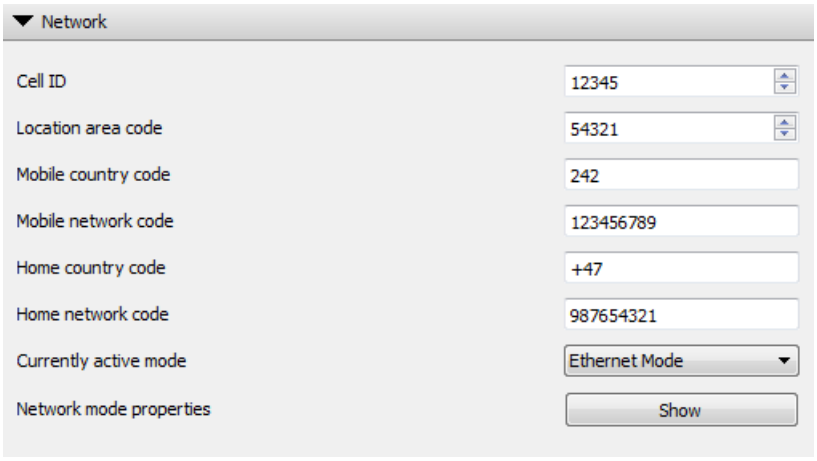
Integrované vývojové prostredie Qt creator ponúka množstvo užitočných nástrojov. Okrem VCS je mocnou zbraňou Qt Simulator, ktorý umožňuje testovanie Qt aplikácií pre mobilné telefóny v prostredí podobnému prostrediu, ktoré je na cieľovom zariadení.

Qt Simulator [8] nepodporuje API špecifických zariadení. Ak aplikácia beží v simulá-



Obrázek 2.7: MAD Developer, nastavenia siete, developer password a vyberanie USB sieťovania

tore, bude bežať aj na mobilnom zariadení, ktoré obsahuje a podporuje Qt a Qt Mobility knižnice. Qt Simulator je mocný nástroj, ktorý umožňuje sledovať ako aplikácia vyzerá a ako sa správa na rôznych typoch mobilných zariadení bez nutnosti ich fyzicky vlastniť. Qt Simulator podporuje komponenty Qt Mobility knižnice (napríklad fotoaparát, kontakty, pripojenia, senzory atď.). Qt Simulator umožňuje debuggovanie Qt mobilných aplikácií, ale nepodporuje zachytávanie únikov pamäte, ktorá nie je uvoľnená. Na zachytávanie týchto únikov sa dá použiť napríklad program Valgrind. Pri vytváraní aplikácie na mobilné zariadenie musíme mať na pamäti, že vytvárame aplikáciu pre zariadenie, ktoré sa môže používať v rozdielnych podmienkach a prostrediach. Qt Simulator dokáže simulovať správanie aplikácie na rozdielnych typoch zariadení a v rozličných podmienkach. Mobilné zariadenie nie je konštantne pripojené na napájací zdroj a preto by mala byť výrazná snaha o vývoj aplikácií, ktoré budú spotrebovávať čo najmenej zdrojov. Energetickú náročnosť aplikácie a jej správanie pri nízkom stave batérie v mobilnom zariadení môžeme otestovať v Qt Simulator nastavením hodnôt Battery level a Power state 2.9. Qt Simulator poskytuje možnosť testovať správanie aplikácie pre tichý a offline profil. Programovo môžeme nastaviť platformu, jazyk, vybavenie, rôznu verziu softwaru či doplnky zariadenia. Pomocou zmeny hodnôt Color depth a Brightness môžeme sledovať vzhľad aplikácie pri rôznych farebných nastaveniach. Aplikácie vyžadujúce pripojenie na sieť sa dajú testovať vďaka simulácii sieťového pripojenia 2.8. Qt Simulator pracuje s rôznymi sieťovými službami, ako Bluetooth, WLAN či mobilná sieť. V sieťových nastaveniach Qt Simulatoru je možné nastaviť rôzne hodnoty pre sieťový status jednotky, silu signálu a pod.



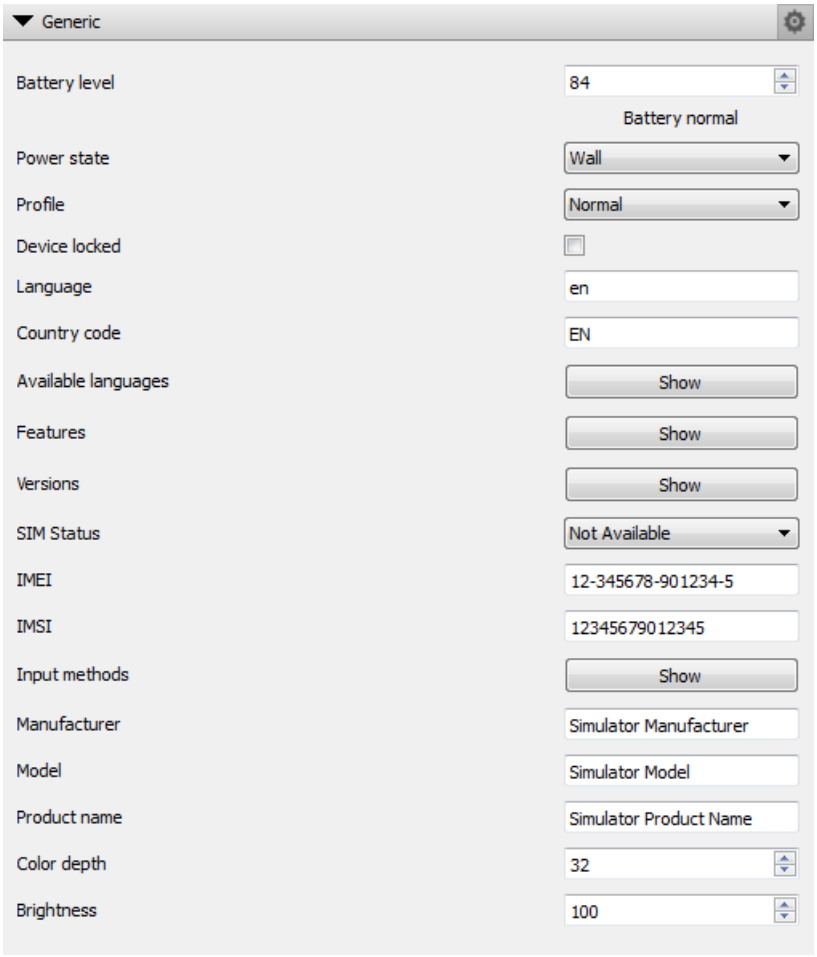
Obrázek 2.8: Príklad nastavenia sieťových nastavení Qt Simulator

2.7 Qt Mobility

Qt Mobility [11] pokrýva škálu rôznych funkcií a technológií. Nie je to len jedno API (application programming interface, teda rozhranie pre programovanie aplikácií), ale je to kolekcia viacerých API a frameworkov. Do Qt Mobility patria tieto komponenty: konektivita, kontakty, galéria dokumentov, spätná väzba, lokalizačné API, posielanie a príjem správ, multimédia, organizér, API pre publikovanie a odoberanie, framework pre Qt služby, QtMobility QML elementy, QtMobility QML pluginy, senzory, systémové informácie, Versit. Každá z častí Qt Mobility bude v ďalej v texte rozpísaná. Qt Mobility je používané vývojármi aplikácií bez ohľadu na to, či cieľová platforma je mobilná, alebo nie. Pomocou Qt Mobility je možné dosiahnuť jednoduchý prenos užívateľskej skúsenosti (user experience,UX) z telefónu do počítača a naopak. API pre konektivitu poskytuje súpravu rozhraní pre komunikáciu s lokálnymi zariadeniami. API obsahuje podporu pre Bluetooth a pre NFC (Near Field Communication). Bluetooth je bezdrôtová technológia s krátkym dosahom (menej ako 100 metrov) s možnosťou dátového prenosu do 2.1 Mbit/s. Bluetooth API obsiahnutá v API pre konektivitu poskytuje triedy pre ovládanie lokálneho Bluetooth zariadenia, objavovanie zariadení a služieb poskytovaných týmito zariadeniami. Ďalej poskytuje podporu pre zverejňovanie služieb zariadenia a podporu pre L2CAP[1] a RFCOMM[2]protokoly. NFC je bezdrôtová technológia, pracujúca v extrémne krátkom dosahu (približne 1 centimeter) s maximálnym dátovým prenosom 424 kbit/s. NFC je ideálne pre prenos malého množstva dát medzi dvoma, dotýkajúcimi sa zariadeniami. API pre kontakty umožňuje klientovi žiadať kontaktné dáta z lokálnych alebo vzdialených backendov. Poskytuje sprístupnenie kontaktných údajov pomocou metód nezávislých na platforme a na dátovom úložisku. Pod pojmom kontaktný údaj sa rozumie kontakt, detail a vzťah medzi kontaktmi. Kontakt je digitálna reprezentácia osoby, skupiny alebo entity, uložená špecifickým spôsobom pre každú platformu. Detail je jedinečná jednotka informácií uložená v kontakte. Kontakty sa

[1]L2CAP [9] (Logical link control and adaptation protocol) protokol prepúšťa pakety do HCI (Host Controller Interface) alebo priamo do Link Managera/ACL spoja, ak zariadenie neobsahuje HCI. L2CAP poskytuje QoS (Quality of Service) pre protokoly na vyšších vrstvách, multiplexuje dáta medzi rôznymi protokolmi na vyšších vrstvách, segmentuje a zostavuje pakety a poskytuje jednosmerné vysielanie multicastových dát pre ďalšie Bluetooth zariadenia.

[2]RFCOMM [10] (Radio frequency communication) je jednoduchá súprava transportných protokolov. RFCOMM je súčasť L2CAP protokolu, poskytujúci emulované RS-232 sériové porty.



Obrázek 2.9: Príklad nastavenia všeobecných nastavení a nastavení batérie v Qt Simulator

medzi sebou môžu podieľať na vzťahoch medzi sebou. Je niekoľko možných vzťahov medzi kontaktmi, napríklad či je kontakt členom skupiny. Galéria dokumentov poskytuje API pre navigáciu a dopyty medzi dokumentmi pomocou metadát. Bežne sa toto API používa pre priradenie správnych súborov pre hudobný prehrávač, alebo prehliadač obrázkov. V galérii dokumentov by mali byť prístupné všetky súbory, ktoré sú obsiahnuté v užívateľských priečinkoch. Základné typy súborov pre galériu dokumentov sú audio, video, obrázok a textový dokument, teda kategorizovanie súborov podľa programu, ktorý je schopný daný súbor spracovať. Iný pohľad na kategorizáciu súborov v galérii dokumentov kategorizuje súbory ako súbor, priečinok, text a zoznam skladieb. Podľa metadát súborov môžeme vytvoriť metatypy ako umelec, album, hudobný žáner a fotoalbum. API pre spätnú väzbu umožňuje ovládať a poskytovať dotykovú a audio spätnú väzbu ako odpoveď na užívateľové akcie. Spätná väzba zahŕňa ovládanie vibračnej jednotky zariadenia, kde vibrácia môže byť použitá ako spätná väzba po stlačení dotykovej obrazovky. Lokalizačné API poskytuje knižnice pre lokalizáciu, správu orientačných bodov, mapy a navigáciu. Lokalizačné dáta presne špecifikujú pozíciu na zemskom povrchu pomocou súradníc pre zemepisnú šírku a dĺžku. Na získanie lokalizačných dát môžeme použiť rôzne metódy, ako napríklad GPS (Global positioning system) alebo Cell ID, ktoré používa jedinečný identifikátor bunky siete tvorenej anténami a inými elektronickými komunikačnými zariadeniami na výpočet približnej

lokácie. Časť lokalizačného API pre orientačné body zahŕňa vytvorenie, získavanie, aktualizáciu a mazanie orientačných bodov z ľubovoľných zdrojov. API pre mapy a navigáciu je založená na rôznych doplnkoch pre zariadenie. API pre posielanie a príjem správ zabezpečuje prístup k službám pre posielanie správ, príjem správ, vyhľadávanie a radenie správ a spúšťanie preddefinovaných klientov pre zobrazenie existujúcej správy, napísanie novej správy alebo rýchlu odpoveď na existujúcu správu. API pre posielanie a príjem správ poskytuje jednotné rozhranie pre manipuláciu a ukladanie SMS, MMS správ, MIME a TNEF emailov. API pre multimédia, QtMultimediaKit, poskytuje súpravu rozhraní umožňujúcich nahrávanie, prehrávanie a správu mutlimediálneho obsahu. API poskytuje jednoduché rozhranie pre zobrazenie obrázku, prehranie videa, nahranie videa alebo zvuku. Rozhranie pracuje s veľkou flexibilitou multimediálnych zdrojov, ktoré môžu byť nahrané v pamäti zariadenia, alebo môžu byť streamované zo vzdialenej lokácie a identifikované pomocou URL. Pri nahrávaní zvuku pomocou tohto rozhrania je možné nastaviť kódovanie súboru (napr. MP3 alebo Ogg), nastaviť bitrate, počet kanálov, kvalitu a vstupný zdroj. Pri prehrávaní zvukového súboru nám rozhranie poskytuje možnosť vytvoriť objekt prehrávača, nastaviť hlasitosť, vybrať prehrávaný súbor a iné parametre. Pri prehrávaní videa máme možnosť nastaviť jas, kontrast, odtieň a sýtosť farieb a mód prehrávania. API podporuje vytváranie, správu a prehrávanie zoznamov skladieb, či už audio alebo video skladieb. API pre multimédia poskytuje rozhranie pre kameru, teda pre vytváranie videa, alebo fotografií. API pre organizér poskytuje rozhranie pre kalendár, časový rozvrh a osobné dáta pomocou metód nezávislých na platforme a na dátovom úložisku. API pre publikovanie a odoberanie poskytuje nástroje, ktoré umožňujú aplikáciám reagovať a čítať rôzne upozornenia pomocou hodnôt v Qt Value Space. Qt Value Space zjednocuje rôzne zdroje dát do konzistentného modelu. Qt Value Space má stromovú hierarchiu, kde každý uzol, alebo list môže obsahovať nejakú hodnotu zo zadanej cesty v zariadení (napr. cesta k tlačítku). Následne môžeme pracovať s hodnotami v uzloch/listoch, ktoré sa menia na základe zmeny odoberanej hodnoty jednotky definovanej cestou. Framework pre Qt služby definuje jednotný spôsob implementácie, hľadania a prístupu k službám medzi rôznymi platformami. Služba je chápaná ako nezávislá komponenta, ktorá slúži na vykonanie predefinovanej operácie. Vyhľadávanie služieb je založené na ich mene, verzii a rozhraní, ktoré je implementované pomocou služby. Po identifikovaní služby je táto služba spustená a je vrátený pointer na spustenú službu. Rozhranie k frameworku pre Qt služby je tvorené QserviceManager-om. Poznatky získané frameworkom pre Qt služby slúžia na unifikovanie prístupu k špecifikovanej implementácii služby v multi platformnom prostredí. API pre senzory pracuje s low-level senzormi, pracujúcimi v reálnom čase, ako je napríklad akcelerometer, ale aj s higher-level senzormi. Zariadenie môže obsahovať niekoľko typov senzorov a nie všetky senzory musia byť týmto API podporované. Senzory podporované API pre senzory pracujú v kartézkom súradnicovom systéme pre 3 smery. Ak senzory merajú všetkých 6 smerov, sú použité aj záporné hodnoty. API pre systémové informácie poskytuje súpravu API pre získanie systémových informácií a kapacít. API pracuje s niekoľkými informačnými kategóriami ako sú informácie o verzii, hardwarové vybavenie, informácie o sieťových pripojeniach, informácie o displeji, informácie o úložných zariadeniach, informácie o šetriči obrazovky a informácie o zariadení. API Versit poskytuje knižnicu na konverziu vCard súborov na QContacs triedu a naopak a na konverziu iCalendar súborov na QOrganizerItems triedu a naopak.

Kapitola 3

Návrh a implementácia aplikácie

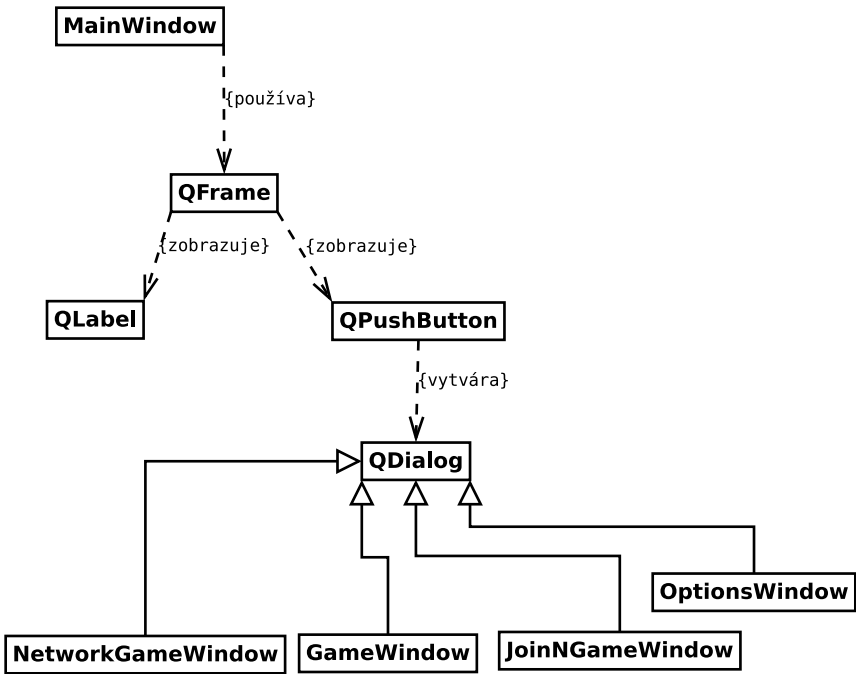
Navrhovaná aplikácia sa dá rozdeliť na niekoľko podproblémov, ktoré bolo nutné navrhnúť a naimplementovať. Pred samotným spustením hry sa užívateľ dostane do kontaktu s grafickým užívateľským rozhraním, ktoré mu ponúka nastavenia aplikácie, výber herného módu a sprevádza užívateľa až do okamihu vytvorenia a spustenia hry. Po spustení hry sa vytvorí hracia plocha, ktorá zobrazuje a združuje vytvorené prvky hry. Na hracej ploche je ústredný bod záujmu užívateľa, jeho hracia postavička, ktorá sa dá ovládať dvoma možnými spôsobmi, ktoré si môže užívateľ vybrať v ponuke užívateľského rozhrania. Posledná časť aplikácie je sieťová komunikácia medzi dvoma zariadeniami, ktoré sú na seba pripojené a na oboch zariadeniach prebieha hra v rovnakom časovom okamihu.

3.1 Princíp hry

Cieľom tejto práce je navrhnúť a naimplementovať hru, podobnú známej sérii hier Bomberman. Hra obsahuje dve možné módy, singleplayer, teda úlohou hráča je zneškodniť počítačom riadených protivníkov v čo najkratšom možnom čase a multiplayer, kde proti sebe stoja dvaja hráči a úlohou je zneškodniť svojho protihráča. Systém hry je rovnaký ako v predlohe. Hráč začína na štartovacej pozícii na hracej ploche, s možnosťou položenia jednej bomby s minimálnym dosahom explózie. Hracia plocha obsahuje okrem hracej postavičky aj prekážky a nepriateľov. Prekážky na hracej ploche sú tvorené dvoma typmi blokov, blokmi, ktoré explózia bomby zničí a blokmi, ktoré sú nezničiteľné. Nezničiteľné bloky tvoria spolu s ohraničením plochy základné prekážky, ktoré vymedzujú voľné plochy po ktorých sa hráč môže pohybovať. Pohyb hracej postavičky je možný len po voľných plochách. Ničením blokov sa vytvára ďalšia plocha pre pohyb a zároveň po zničení bloku, je šanca, že sa objaví na hracej ploche vylepšenie hracej postavičky, ktoré sa aplikuje na postavu po prechode postavičky po tomto vylepšení. Vylepšení je niekoľko typov a medzi základné vylepšenia patrí zvýšenie počtu naraz uložených bômb, zvýšenie dosahu explózie, zvýšenie rýchlosti hracej postavičky a zvýšenie počtu životov, teda možnosť prežiť kontakt s explóziou bomby. Tieto vylepšenia sa kumulujú do určitej hodnoty. Špeciálne vylepšenia sa nekumulujú a ich efekt sa prejaví len raz, aj keď hráč pozbiera viac rovnakých vylepšení. Medzi tieto vylepšenia patrí možnosť prechádzať cez bomby a možnosť odkopávať bomby. V danom časovom okamihu je aktívne len jedno špeciálne vylepšenie. Pokiaľ sa hráč rozhodne nevyužiť vylepšenie, ktoré je zobrazené na hracej ploche, je možné ho zničiť explóziou bomby.

3.2 Návrh GUI

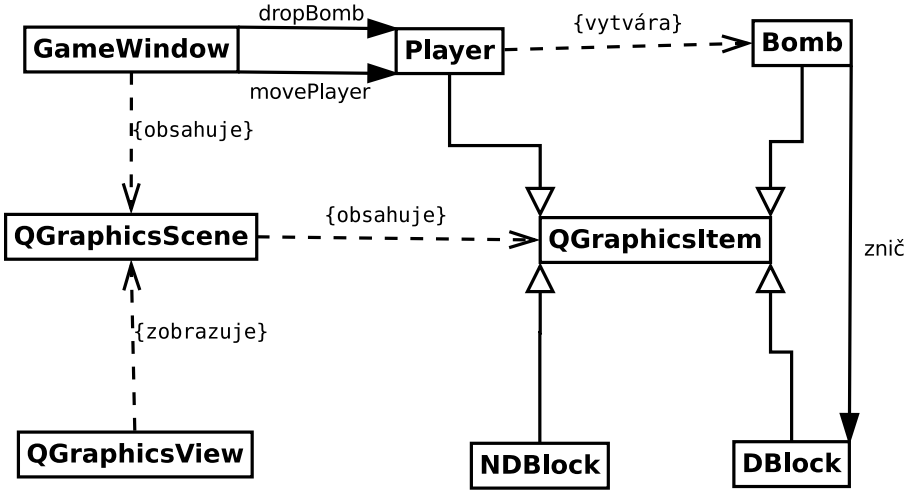
Grafické užívateľské rozhranie má plniť funkciu prvotnej komunikácie s užívateľom. Pri návrhu užívateľského rozhrania je kladený dôraz na jednoduchosť a intuitívne ovládanie aplikácie. Návrh ráta s využívaním tlačítok, ktorých označenie jednoznačne definuje akciu, ktorú tieto tlačidlá vykonávajú. Užívateľ by mal byť užívateľským rozhraním jasne a jednoducho navigovaný k spustení samotnej hry, bez využitia pomocníka. Užívateľské prostredie bude tvorené už spomínanými, vhodne označenými tlačítkami a oknami, ktoré budú podrobnejšie rozširovať akciu tlačidiel. Okná budú obsahovať aj textové návestia, ktoré budú užívateľovi pomáhať presne definovať správny sled akcií v okne. Grafické užívateľské rozhranie bude vytvorené pomocou tried z modulu QtGui. Úvodné okno bude tvorené triedou QMainWindow, ktorá bude obsahovať tlačítka, vytvorené pomocou triedy QPushButton. Ďalšie okná tvoriace užívateľské rozhranie sa budú vytvárať pomocou triedy QDialog, ktorá ponúka vytvorenie dialógového okna. V prípade nutnosti výberu aspoň jednej možnosti sa budú využívať zaškrtávacie tlačidlá z triedy QCheckBox a plný textový vstup užívateľa bude poskytovať trieda QLineEdit. Informačné textové návestia bude zastupovať trieda QLabel. Úvodné okno sa bude vytvárať pomocou nástroja UICreator, ktorý ponúka možnosť vytvárať užívateľské rozhrania jednoduchým pridávaním prvkov do formulára. Do prázdneho formulára sa teda budú postupne vkladať všetky elementy, ktoré budú tvoriť užívateľské rozhranie. Rozloženie jednotlivých elementov budú zabezpečovať layouty, ktoré odbremeňujú programátora od nutnosti implementovať vlastné výpočty pre vhodné uloženie prvkov a prvky usporadúvajú a menia ich veľkosti podľa voľného miesta v danom layoute.



Obrázek 3.1: Diagram tried návrhu hlavného okna

3.3 Návrh hracej plochy

Úlohou hracej plochy v aplikácii bude zobrazovať grafiku jednotlivých častí hry (postavičiek, blokov, bômb...). Ako grafické zobrazenie som zvolil animované .gif obrázky pri pohyboch, či vyjadrení činnosti a statické obrázky pre statické predmety na hracej ploche, ako sú bloky. Navrhnutú hraciu plochu zastupuje dvojica tried QGraphicsScene(scéna) a QGraphicsView(pohľad). Scéna slúži na vykresľovanie jednotlivých grafických predmetov a poskytuje rýchly prístup k zobrazeným predmetom. Na scénu môžeme jednoducho pridávať predmety, odstraňovať ich a meniť ich pozíciu. Pohľad slúži na zobrazenie určitej časti scény. Scéna môže byť potencionálne nekonečná a práve pohľad nám umožňuje nahliadať na tú časť scény, na ktorej sú vykreslené predmety. Scénu si môžeme predstaviť ako celý svet a pohľad ako okno, cez ktoré na tento svet pozeráme. Vzhľadom na možnosť sieťovej hry potrebujeme zaručiť, aby obaja pripojení hráči mali možnosť vidieť aj pohyb a akcie protihráča. A túto záruku som vyriešil tým, že scéna, teda herný svet a pohľad, teda okno do tohto sveta, budú rovnako veľké. Ak by scéna bola väčšia ako pohľad, musel by mať každý hráč prispôsobený pohľad na svoju hraciu postavičku a teda na tú časť scény v ktorej sa v danom okamihu nachádza. Ak by sa aplikácia navrhla a implementovala týmto spôsobom, stávalo by sa často, že hráči by nemali prehľad o pohybe svojho súpera a ten by mohol, z herného hľadiska, získať nie úplne fair-play výhodu, ak by sa zrazu zjavil v hráčovom okne. Ďalší dôvod zavedenia rovnakej veľkosti scény a okna je aj inšpirácia v pôvodnej hre a ostatných Bomberman-like derivátov. Pôvodná hra bola vyvíjaná na osobný počítač, ktorého monitor, teda zobrazovacia plocha, je niekoľko násobne väčšia ako zobrazovacia plocha na mobilnom telefóne, a preto si vývojári mohli dovoliť zvoliť rovnakú veľkosť scény a pohľadu a vizuálna stránka hry neutrpela žiadnu ujmu. Napriek tomu, že moja aplikácia je vyvíjaná na mobilný telefón, ktorého zobrazovacia plocha je relatívna malá, a teda aj grafické predmety budú veľkosťou prispôsobené veľkosti plochy. Rozlíšenie predmetov bude teda nižšie a tým pádom sa znižuje náročnosť na detaily predmetov a ich grafickú zložitosť.



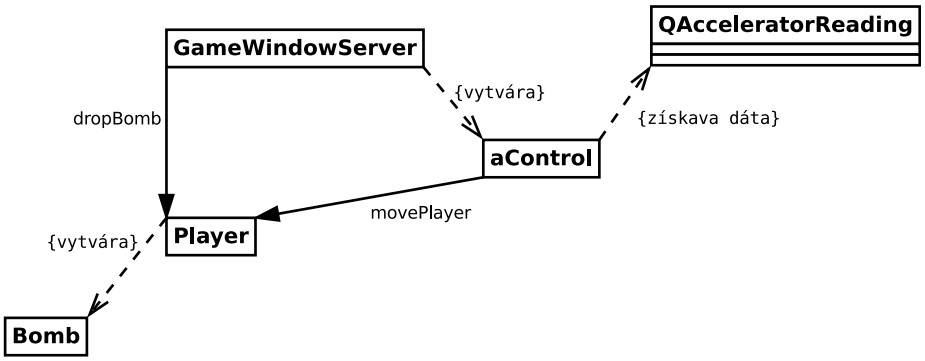
Obrázek 3.2: Diagram tried návrhu hracej plochy

3.4 Návrh hracej postavičky, blokov a bomby

Hracia postavička, bloky a bomby, teda grafické predmety, ktoré budú pridané do scény a zobrazované pohľadom sú navrhnuté ako triedy, ktoré dedia z triedy QGraphicsItem, teda, zo základnej triedy, s ktorou pracuje scéna. Grafické predmety sú uložené v súradnicovom systéme scény. Predmety, ktoré majú možnosť pohybu po scéne, musia byť schopné pri vykonávaní pohybu určiť, či daný pohyb môžu vykonať, či im v ceste nestojí prekážka, ktorú nemôžu prekonať. Tieto predmety pred vykonaním pohybu zistia, či majú voľnú cestu dotazom na scénu, či pred nimi neleží iný predmet. Ak je cesta voľná a predmetom nič nebráni v pohybe je pohyb vykonaný.

3.5 Návrh ovládania

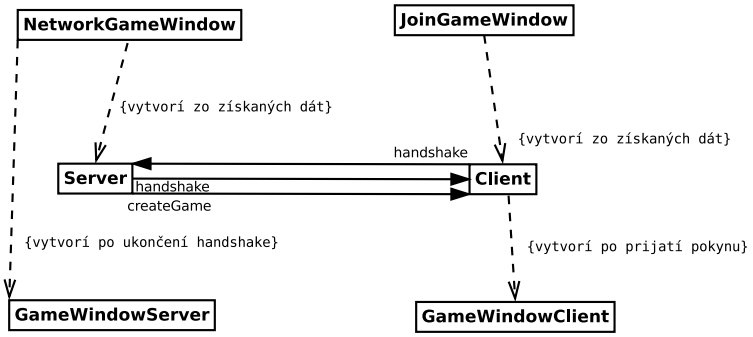
Ovládanie pohybu hracej postavičky po ploche je navrhnuté s cieľom na jednoduchosť a pohodlie užívateľa. Navrhol som dve možnosti ovládania, ovládanie pomocou dotykových plôch a ovládanie pomocou akcelerometra. Pri ovládaní pomocou dotykových plôch je displej telefónu rozdelený na niekoľko zón. Zóny sú umiestnené intuitívne na dotykovom displeji telefónu. Po kliknutí na ľavú časť displeja hracia postavička vykoná pohyb doľava, po kliknutí na pravú časť, doprava. To isté s hornou a spodnou časťou displeja. Pre uloženie bomby je nutné kliknúť do stredovej časti displeja. Jednotlivé dotykové plochy zaberajú tretinu displeja, takže by mali byť dostatočne veľké a užívateľ by nemal mať problém s preklikmi do iných častí. Ovládanie pomocou akcelerometra využíva hodnoty z akcelerometra zariadenia. Toto ovládanie som sa rozhodol pridať pre zvýšenie zaujímavosti aplikácie. Signál o pohybe sa vyšle vtedy, ak je hodnota na danej osi za polovicou maximálnej hodnoty v oboch smeroch(pracuje sa s hodnotami aj na kladnej, aj na zápornej časti osi). Ak hodnota ostane za prahovou hodnotou vyvolania pohybu aj po ukončení pohybu hracej postavičky je znova vyslaný signál o pohybe v danom smere. Hodnoty z akcelerometra je nutné cyklicky opakovať v krátkych časových intervaloch, aby bola zaistená okamžitá reakcia na zmenu hodnôt. Hodnoty z akcelerometra získavam pomocou API pre senzory z QtMobilty, aby bolo možné aplikáciu používať aj na iných zariadeniach, ktoré podporujú QtMobility. Bez využitia API Sensors by bolo možné získavať dáta z akcelerometra priamo zo zariadenia pristupovaním do súboru /sys/class/i2c-adapter/i2c-3/3-001d/coord na zariadení N900, ale pri tomto spôsobe by vzniklo reálne riziko, že by ovládanie pomocou akcelerometra nefungovalo na iných typoch zariadení, ktoré by mali hodnoty uložené v inom súbore.



Obrázek 3.3: Diagram tried návrhu ovládania pomocou akcelerometra

3.6 Návrh sieťovej komunikácie

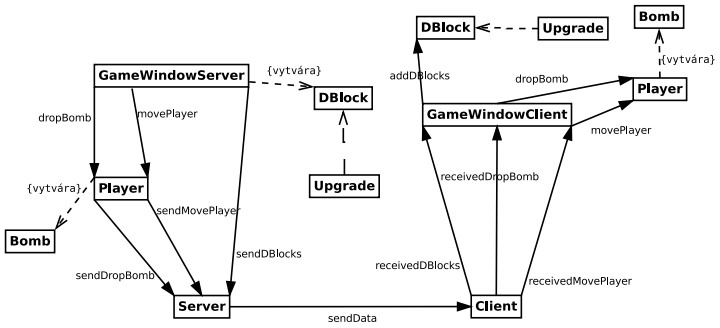
Pri návrhu sieťovej komunikácie bolo potrebné riešiť niekoľko problémov. Prvý problém bol v rozhodnutí, ktorý protokol použiť. Ak by sa sieťová komunikácia implementovala pomocou TCP protokolu bolo by zaistené jednoduché naviazanie spojenia, eliminovali by sa prípadné straty paketov či ich chybné poradie, keďže TCP protokol kontroluje poradie paketov a po prijatí sa posiela späť na odosielateľa potvrdenie o prijatí. Použitím TCP protokolu by sa vytratili možné rozdiely v stave hracej plochy medzi spojenými zariadeniami. Rozhodol som sa miesto TCP protokolu použiť protokol UDP. Hoci implementácia TCP protokolu by umožnila pohodlné prepojenie zariadení a spoľahlivé spojenie by zaručovalo správne poradie paketov, mohol by nastať problém, ak by sa museli pakety preposielať a pri pomalšom pripojení, by mohol narastať rozdiel medzi stavom hry na zariadeniach a vznikali by lagy, teda hra by kvôli spojeniu prestala byť plynulá a nastalo by skokové správanie. UDP ako prenosový protokol bol zvolený pretože je vo všeobecnosti používaný pri vytváraní real-time sieťových aplikácií a absenciou potvrdzovania prijatých paketov sa oproti TCP protokolu odľahčí sieťová záťaž. Pri použití UDP protokolu sa však musí vhodne ustanoviť spojenie medzi oboma zariadeniami a zvoliť vhodnú architektúru. Zvolená nebola klient-server architektúra, ale peer-to-peer. Klient-server architektúra by nebola vhodná vzhľadom na výkon telefónov. Ak by som použil klient-server architektúru, server by bol zaťažený viac oproti klientovi, lebo by musel zvládať výpočet celej aplikácie. Pri peer-to-peer architektúre je možné výpočtovú záťaž rovnomerne rozdeliť medzi obe zariadenia. Napriek zvolenej architektúre je jedno zariadenie pomenované ako server(host) a druhé ako klient. Toto rozdielne pomenovanie je zvolené kvôli ustanoveniu správneho spojenia 3.4. Obe zariadenia



Obrázek 3.4: Diagram tried návrhu ustanovenia spojenia

majú vytvorené dva UDP sockety, kde jeden je naviazaný na port, na ktorom prijíma prichádzajúce datagramy a druhý slúži na odosielanie datagramov zo zariadenia. Tento spôsob umožňuje pružnejšie reagovať v komunikácií medzi zariadeniami, keďže sa nemusí čakať pri čítaní prichádzajúceho datagramu na ukončenie akcie čítania pri posielaní datagramu, ale jednoducho sa použije druhý soket na poslanie dát. Ustanovenie spojenia spočíva v poslaní IP adresy a portu, na ktorom čaká soket na datagramy. Po rezervovaní portu na serveri , server čaká kým mu príde *handshake* paket od klienta s IP adresou a portom, ktorý si rezervoval klientský soket. Keď príde *handshake* paket od klienta, server pošle svoj *handshake* paket a na oboch zariadeniach sa uložia hodnoty IP adresy a portu svojho náprotivku. V užívateľskom rozhraní užívateľ s klientským zariadením zadá IP adresu hostiteľa a každých niekoľko sekúnd sa pošle *handshake* paket na IP adresu hostiteľa. Prvá správa sa pošle na port 12345 a pokiaľ nepríde odpoveď, posiela sa *handshake* paket na nasledujúci port. Po prijatí *handshake* paketu od servera sa ukončí pravidelné posielanie *handshake* paketov.

Server skúša rezervovať porty od hodnoty 12345 a klient od hodnoty 15432. Začiatočné hodnoty voľných portov by mali byť ihneď k dispozícii. Zvyšovanie hodnoty portu sa udeje v prípade, že zvolený port je už používaný inou aplikáciou.



Obrázek 3.5: Diagram tried návrhu sieťovej komunikácie

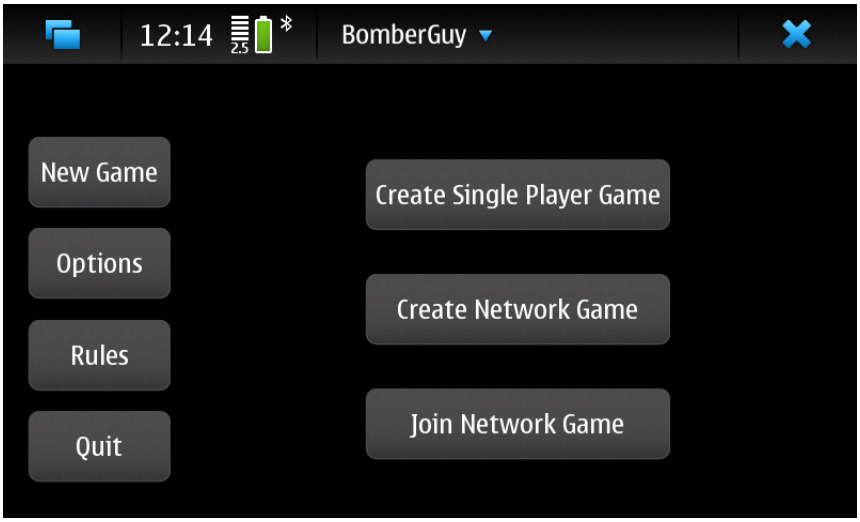
3.7 Implementácia

Obsahom tejto kapitoly je podrobný popis implementácie navrhnutej aplikácie. Kapitola teda popisuje postupnú implementáciu podľa návrhu od grafického užívateľského rozhrania, cez hraciu plochu a predmety, ktoré grafická scéna a pohľad na ňu na hracej ploche zobrazuje, implementáciu pohybu až po implementáciu sieťovej komunikácie. Implementácia prebiehala priamo na mobilnom zariadení Nokia N900 cez QtCreator. Sieťová komunikácia bola testovaná s využitím QtSimulator.

3.8 Implementácia GUI

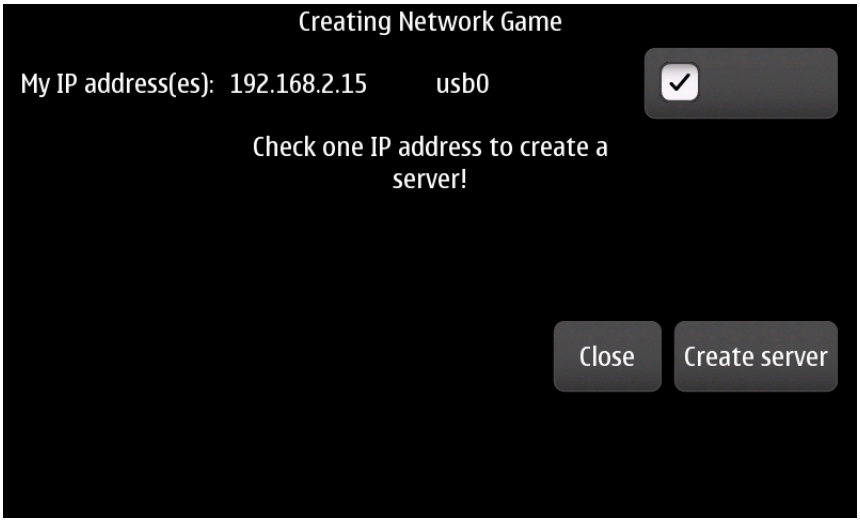
Grafické užívateľské rozhranie je prvá časť aplikácie s ktorou príde užívateľ do styku. Slúži na orientáciu v aplikácii a ovládanie prvotných akcií užívateľa, ktoré vedú k vytvoreniu hry. V tejto časti budú popísané jednotlivé prvky, ktoré tvoria užívateľské rozhranie. Prvé dialógové okno, MainWindow, je vytvorené pomocou triedy QMainWindow. Obsah okna MainWindow je tvorený užívateľským rozhraním ui, ktoré je vytvorené pomocou integrovaného nástroja UI Designer, ktorý vytvorí .ui súbor v XML formáte. Užívateľské rozhranie MainWindow 3.6 je tvorené troma rámcami (Qframe). Prvý rámec obsahuje štyri tlačítka vytvorené pomocou triedy QPushButton, ktoré tvoria hlavné menu aplikácie. Tlačítka *New Game* a *Rules* slúžia na zobrazenie príslušných rámcov. Rámec s menu je viditeľný vždy, pokiaľ nie je MainWindow prekryté iným dialógovým oknom. Rámec *Rules* obsahuje základné informácie o aplikácii.

Tlačítko *Options* vytvorí a zobrazí dialógové okno v ktorom aplikácia ponúka možnosť výberu ovládania hry cez dotykové plochy, alebo pomocou akcelerátora. Tieto nastavenia sa ukladajú permanentne do textového konfiguračného súboru, ktorý sa vytvorí v domovskom priečinku užívateľa skrytý priečinok .bomberGuy a v tomto priečinku je uložený konfiguračný súbor. Pri zmene možnosti ovládania sa táto zmena uloží do konfiguračného súboru a táto zmena pretrvá aj po ukončení aplikácie. Vždy musí byť zvolená práve jedna možnosť ovládania a preto sa QCheckBox prvky pri zaškrtnutí vzájomne vylučujú a v danom časovom okamihu je zaškrtnutý práve jeden QCheckBox. Ukladanie konfiguračných informácií



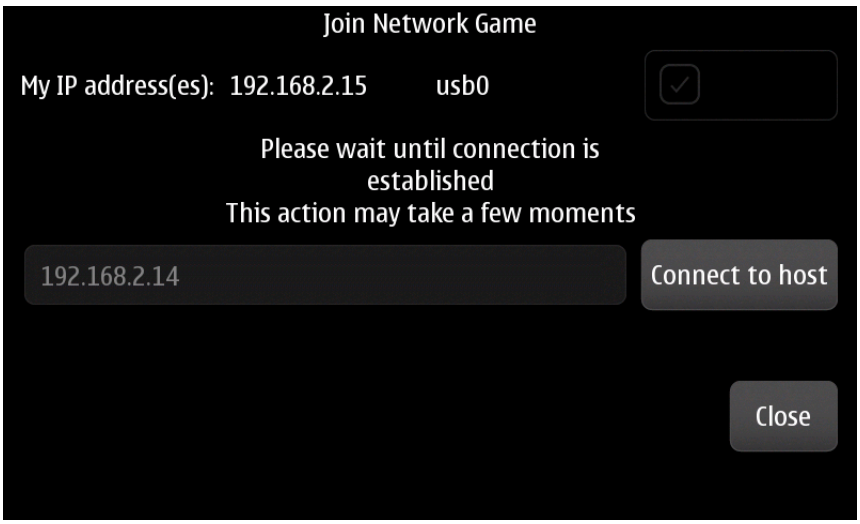
Obrázek 3.6: Hlavné menu v grafickom užívateľskom rozhraní

do textového súboru je zvolené kvôli jednoduchosti a malému množstvu ukladaných informácií. Rámec *New Game* obsahuje ďalšie tri tlačítka. Tlačítko *Create Single Player Game* vytvorí dialógové okno s hracou plochou pre hru proti počítačom riadeným protivníkom. Tlačítka *Create Network Game* a *Join Network Game* slúžia na zobrazenie dialógových okien, ktoré slúžia na vytvorenie hostiteľa pre sieťovú hru, respektíve na možnosť pripojiť sa na hostiteľa ako klient. Pri stlačení tlačítka *Create Network Game* sa vytvorí a zobrazí okno súžiace na vytvorenie servera, na ktorý sa bude pripájať klient. Po zobrazení dialógového okna 3.7 aplikácia navedie užívateľa aby vyznačil práve jedno aktívne sieťové rozhranie na ktorom chce vytvoriť server. Zobrazené sú všetky aktívne sieťové rozhrania a ich IP adresy. Užívateľ si môže vybrať na ktorom rozhraní bude aplikácia komunikovať so svojím náprotivkom. Pri každom rozhraní je uvedená jeho IP adresa pre pohodlie užívateľa. Po vybratí sieťového rozhrania sa odblokuje tlačítko *Create server*.



Obrázek 3.7: Okno Create Network Game so zvoleným sieťovým rozhraním pripravené na spustenie servera

Ak v rámci *New Game* stlačí tlačítko *Join Network Game* zobrazí sa dialógové okno 3.8 s prvkami, potrebnými pre pripojenie zariadenia na druhé zariadenie. Podobne ako v okne so spustením serveru, aj v tomto okne užívateľ nájde všetky aktívne sieťové rozhrania a ich IP adresy. Po vybratí rozhrania sa odblokuje prvok *MyLineEdit*, ktorý rozširuje triedu *QLineEdit*. Pokiaľ užívateľ klikne na prvok *MyLineEdit* má možnosť zadať IP adresu druhého zariadenia na ktoré sa chce pripojiť. Po kliknutí sa aktivuje event *focusInEvent* a úvodný text sa odstráni. Zároveň je na tomto prvku aktivovaný validátor IP adresy, ktorý nedovolí užívateľovi zadať iný vstup ako IP adresu. Po validácii užívateľského vstupu sa odblokuje tlačítko *Connect to host*, ktoré umožní pripojiť sa na druhé zariadenie. Viac o sieťovej komunikácii a jej prvkoch sa čitateľ dozvie v samostatnej kapitole.



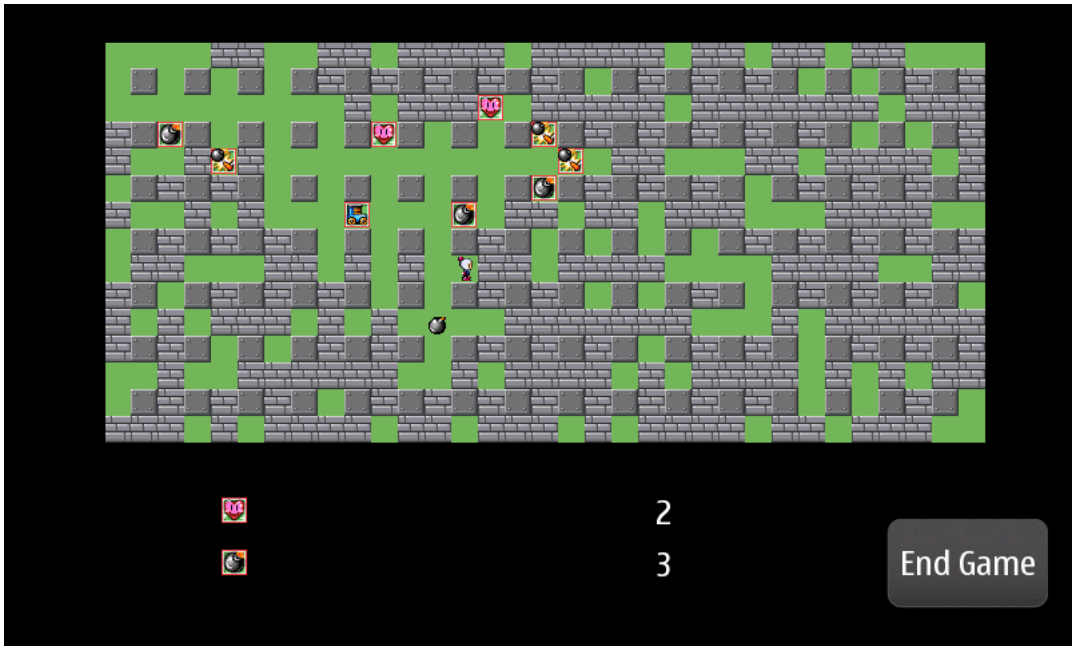
Obrázek 3.8: Okno Join Network Game so zvoleným sieťovým rozhraním a zadanou IP adresou servera, čaká na ustanovanie spojenia

Po stlačení tlačítka *Create Single Player Game* sa vytvorí hracia plocha, umožňujúca hrať hru bez využitia siete proti jednoduchým protivníkom ovládaných počítačom.

3.9 Implementácia hracej plochy

Hracia plocha je implementovaná ako dialógové okno, ktoré obsahuje grafickú scénu *QGraphicsSCene* a pohľad na scénu *QGraphicsView*. Pri vytváraní hracej plochy sa z konfiguračného súboru načíta zvolený spôsob ovládania a vypočítajú sa dotykové plochy. Pohľad obsahuje celú scénu, bez možnosti posúvania sa. Do scény sa pridajú základné prekážky, ktoré vytvárajú hranice scény a bloky, ktoré sa nedajú zničiť na vymedzenie vnútorného priestoru. Nasleduje vytvorenie zničiteľných blokov. Využívaním generovania náhodných čísel a obsadením len dvoch tretín voľného miesta na ploche sa zabezpečí, že každá hra má rozdielne usporiadanie zničiteľných blokov. Navyše v každom bloku je len tretinová pravdepodobnosť, že bude po zničení obsahovať vylepšenie pre hráča. Je zabezpečené, že obe štartovacie pozície sú rovnaké a neobsadené, aby bolo možné začať hru. Ak nie je zvolený sieťový mód aplikácie do scény sa pridávajú aj počítačom ovládaní nepriatelia.

Nepriatelia A.3 sú naimplementovaný bez umelej inteligencie. Fungujú ako príšery, ktoré sa pohybuju po hracej ploche náhodným pohybom a majú možnosť prechádzať cez bloky, ktoré sa dajú zničiť výbuchom. Nepriatelia sú rôzne rýchli a sú schopni odolať niekoľkým ex-



Obrázek 3.9: Ukážka zobrazujúca hracie okno s hracou plochou, obsahujúcou hraciu postavičku, bombu a vylepšenia

plóziám. Na vytvorenie náhodného pohybu bolo využité náhodné generovanie čísel pomocou funkcie *qrand()* a so stanovením seedu pre generovanie *qsrand((int)time.msec())*. Nepriatelia sa pohybujú štyrmi čiastkovými pohybmi, aby bol vykonaný plný posun o jednu pozíciu. Náhodná zmena pohybu sa deje vždy o n plných posunov. Číslo n je náhodne generované v rozmedzí 1 až 10 plných pohybov. Nepriateľ je na grafickej scéne reprezentovaný grafickou animáciou a jeho pohyb je počítaný vo vlastnom vlákne *QThread*.

3.10 Implementácia hracej postavičky, blokov a bomby

Hracia postavička je tvorená triedou Player. Pohyb hracej postavičky po scéne je riadený prijímaním signálu s hodnotou smeru pohybu(1 v ľavo, 2 v pravo, 3 hore a 4 nadol) do slotu, ktorý podľa hodnoty smeru spojí vytvorený časovač so slotmi, ktoré vykonávajú pohyb. Animácia pohybu je riešená čiastkovým pohybom po obdržaní signálu *timeout()* od spusteného časovača. Čím rýchlejšie sa hracia postavička pohybuje, tým je kratší interval časovača. Pre dokončenie pohybu sa vykonajú štyri čiastkové pohyby, časovač sa zastaví a odpojí sa od spojeného slotu. Pred vykonaním samotného pohybu sa skontroluje, či je možné vykonať pohyb do zadaného smeru. Zistia sa aktuálne súradnice hráča a zo scény sa zistí, či niečo neleží pred hráčom vo zvolenom smere. Pokiaľ je možné na danú pozíciu vykonať pohyb, pohyb sa vykoná spustením časovača. Ak trieda Player obdrží signál, ktorý dáva pokyn na uloženie bomby, tak sa volá funkcia, ktorá najprv skontroluje, či už nie je na daných súradniciach jedna bomba položená. Bomby sa na seba skladať nedajú, takže v danom časovom okamžiku môže byť na zadaných súradniciach práve jedna bomba. Ak sú súradnice vhodné na uloženie bomby, tak sa nová bomba pridá do scény. Po uložení novej bomby do scény sa spustí časovač, ktorý po uplynutí zadaného času spustí funkciu, ktorá vykreslí explóziu bomby. Explózia bomby sa šíri v smere horizontálnej a vertikálnej osi

bomby, pokiaľ v osiach neležia prekážky. Prekážky v smeroch explózie sa zisťujú rovnakým princípom ako pri pohybe hráča. Ak explózia zasiahne hráča, alebo zničiteľný blok či vylepšenie, pošle sa signál zasiahnutému prvku scény na zničenie. Ak explózia zasiahne inú, ešte nevybuchnutú bombu pošle sa signál zasiahnutej bombe o okamžitej explózii. Grafika hracej postavičky a bomby je animovaná pomocou .gif animácií. Grafická scéna QGraphicsScene nepodporuje zobrazenie animovaných obrázkov, či videí a preto bolo potrebné použiť triedu QGraphicsProxyWidget. Vďaka tejto triede je možné v grafickej scéne zobrazovať okrem tried, ktoré dedia z QGraphicsItem aj triedy, ktoré dedia zo základnej triedy QWidget. Teda je možné pomocou QGraphicsProxyWidget zobraziť na scéne tlačítka, riadky na editáciu textu a podobne. Na zobrazenie animovaného .gif súboru v grafickej scéne potrebujem použiť triedu QMovie, ktorá bude animáciu prehrávať a triedu QLabel, ktorá bude slúžiť ako kontajner pre animáciu. Nastavením priehľadného pozadia nebude QLabel rušiť ostatné predmety na scéne. Trieda Player, odvodená od triedy QGraphicsProxyWidget, obalí vytvorený QLabel s animáciou a vytvorí rozhranie medzi obsiahnutými QWidgetmi a grafickou scénou. Pri pohybe sa zvolí .gif animácia v smere pohybu. Animácie bomby je riešená rovnakým sposôbom. Animovaná bomba mení svoju veľkosť kým nevybuchne. Pri explózii sa zmení grafika a zobrazí sa animácia explózie, ktorá je tvorená statickými obrázkami, ktoré sa v rýchlom časovom slede pridávaju na grafickú scénu a zmiznú. Grafické obrázky a animácie boli vytvorené pomocou programu GIMP. Podklady tvoria voľne stiahnuteľné textúry.

3.11 Implementácia ovládania

Pokiaľ je zvolené ovládanie pomocou dotykových plôch, ktoré sú vypočítané pri tvorbe hracieho okna, na zasielanie signálov hracej postavičky s hodnotou smeru pohybu sa využíva *eventFilter* na hracej ploche. Pri vygenerovaní udalosti *MouseButtonPress* sa zistia súradnice udalosti a pokiaľ súradnice ležia vo vnútri jednej z definovaných dotykových plôch, vyšle sa signál do triedy Player s hodnotou pohybu, či signál na uloženie bomby. Ak je zvolené ovládanie pomocou akcelerometra, vytvorí sa nová trieda *aControl*, ktorá rozširuje triedu *Qthread*. Funkcia *run*, teda beh vlákna sa spúšťa v intervale 20 milisekúnd pomocou časovača, ktorého signál *timeout()* je pripojený na slot triedy *aControl*, ktorá pomocou triedy *QAccelerometerReading* získa hodnoty akcelerometra z osí x a y. Os z nie je pre pohyb hráča podstatná. Pokiaľ získané hodnoty prekročia prahovú hodnotu, ktorá je určená ako polovica maximálnej hodnoty z akcelerometra vyšle sa signál s hodnotu smeru pohybu do triedy *Player*. Ak sú hodnoty z oboch relevantných ôs za prahovou hodnotou vykoná sa pohyb v tom smere, v ktorom je viac prekročená prahová hodnota. Zároveň sa nastaví hodnota premennej *enabled* na *false*, čím sa zabráni v ďalšom intervale vyslať signál na pohyb hráča, kým nie je predchádzajúci pohyb dokončený.

3.12 Implementácia sieťovej komunikácie

Sieťová komunikácia medzi dvoma zariadeniami prebieha pomocou dvoch soketov na oboch stranách. Jeden soket, označený ako *receiver* prijíma datagramy a druhý soket, označený ako *sender* datagramy posiela. V časti, v ktorej sa písalo o návrhu aplikácie, je vysvetlené pomenovanie oboch účastníkov komunikácie. Aj keď sa jedná o spojenie peer-to-peer, kde sú si obe zariadenia rovnocenné, na jednom zariadení je vytvorená trieda *Server* a na druhom zariadení je vytvorená trieda *Client*. Toto označenie som vybral preto, aby bolo jasne dané,

ktorá strana zahajuje *podanie rúk*. Pri vytvorení servera a úspešnej rezervácii portu, server čaká kým mu na rezervovaný port príde výzva na spárovanie sa. Okrem textového reťazca, ktorý jasne definuje o akú sieťovú správu ide, obsahuje výzva na spárovanie sa aj IP adresu odosielateľa, teda klienta, a číslo portu, ktorý si rezervoval soket *receiver* na strane klienta. Následne server odošle na IP adresu a port definovaný v správe od klienta správu so svojou IP adresou a svojím rezervovaným portom. Po vytvorení klienta a rezervovaní portu pre *receiver* soket začne klienta posielať *handshake* správy na IP adresu servera, ktorú užívateľ zadal v dialógovom okne *Join Network Game*. Keďže existuje možnosť, že hneď prvý port, ktorý sa server pokúsil zaregistrovať pre svoj *receiver* soket je už obsadený a správa na server nepríde, tak sa v pravidelnom intervale posiela zahajujúca správa pre spárovanie sa na port vždy o 1 väčší ako bol predchádzajúci port. Ak server klientovi odpovie, časovač zodpovedný za interval v ktorom sa posiela správa sa stopne. V tejto chvíli sú obe zariadenia spárované a vedia, na ktorú IP adresu a na ktorý port majú posielať dáta. Správy, ktoré sa posielajú sú rozdelené na niekoľko častí, kde každá časť je oddelená znakom —. V prvej časti sa nachádza znakový reťazec, ktorý jasne definuje o aký typ správy sa jedná a v ďalšej časti správy sú od seba oddelené upresňujúce informácie. Zariadenia si vymieňajú niekoľko typov správ. Po úspešnom spárovaní zariadení sa v dialógovom okne na strane servera odblokuje tlačítko *Start*, ktoré po stlačení vyvolá akciu, ktorá pošle na klienta správu s hodnotou *createGame* a vytvorí hraciu plochu na serveri. Keď klient obrdží túto správu vyvolá sa akcia, ktorá vytvorí hraciu plochu na klientovi A.4

Aplikácia je vytvorená v architektúre peer-to-peer v snahe o rozloženie výpočtovej záťaže na obe spárované zariadenia. Implementácia tejto architektúre v tejto aplikácií znamená, že obe zariadenia sú zodpovedné za svoj obsah v grafickej scéne a zároveň, pri každej zmene, vyvolanej užívateľom sa táto zmena pošle na svoj náprotivok. Po pridaní hráča do hracej plochy sa okamžite pošle správa *addPlayer* s hodnotami x a y súradnice na ploche a s číslom hráča, kde 1 je číslo hráča na serveri a 2 je číslo hráča na klientovi. Zariadenie, ktoré túto správu obdržalo vytvorí na svojej grafickej scéne hráča. Rovnako je to aj s pohybom hráča, či uložením bomby.Po poslaní signálu do triedy *Player* sa pošle správa druhému zariadeniu, aby vykonalo pohyb aj vo svojej scéne. Pri poslaní správy s pohybom hráča, sieťové datagramy obsahujú znakový reťazec *movePlayer* nasledovaný číselnou hodnotou smeru pohybu typu *integer* a pre kontrolu správa obsahuje aj číselne hodnoty súradníc osy x a osy y, na ktorých sa nachádza herná postavička pred vykonaním pohybu vo zvolenom smere. Tieto hodnoty sú posielané pre prípad stratených datagramov a týmto systémom sa redukuje prípadný rozdiel v stave hry medzi zariadeniami pri slabom dátovom prenose. Pokiaľ sa poslané hodnoty líšia od aktuálnej pozície hracej postavičky, vykoná sa korekcia pozície a následna po tejto korekcii sa vykoná zvolený pohyb. Správa, ktorá obsahuje informácie o uložení bomby, obsahuje textový reťazec *dropBomb* nasledovaný číselnými hodnotami súradníc, na ktorých sa má bomba v grafickej scéne zobraziť. Aby herná plocha obsahovala na oboch zariadeniach rovnaké predmety, či už ide o rozmiestnenie blokov, alebo o vylepšenia, ktoré tieto bloky obsahujú. Server pri vytváraní svojej hernej plochy vytvára tieto bloky a ukladá do zoznamov ich x-ovú, y-ovú súradnicu a číslo vylepšenia, ktoré blok po zničení zanechá na ploche. Hodnota 0 znamená, že blok neobsahuje žiadne vylepšenie. Prehľad vylepšení obsahuje tabuľka 3.10.

Všetky vytvorené bloky sa posielajú v jednej správe, ktorá obsahuje znakový reťazec *addDBlock*, hodnoty súradníc, na ktorých vytvorený blok leží a číselnú hodnotu vylepšenia. V správe sú od seba jednotlivé bloky oddelené znakom !. V prípade, že sa počas prebiehajúcej hry jeden z hráčov rozhodne nepokračovať a hru predčasne ukončí na svojom zariadení, pošle sa spárovanému zariadeniu správa s textovým reťazcom *endGame*. Server na túto správu

0	Žiadne vylepšenie	
1	Pridaj život	
2	Pridaj bombu	
3	Zvýš explóziu	
4	Možnosť odkopávať bomby	
5	Zvýšenie rýchlosti	
6	Možnosť prechádzať cez bomby	

Obrázek 3.10: Tabuľka s prehľadom vystykujúcich sa vylepšení v hre

zareaguje zavretím hracieho okna a vytvorením okna, ktoré nesie informáciu o predčasnom ukončení hry druhou stranou. Prehľad všetkých posielaných správ medzi zariadeniami pri sieťovej komunikácií obsahuje tabuľka 3.11

handshakeServer	IPserver	portServer		
handshakeClient	IPklient	portClient		
addDBlock	actual xPos	actual yPos	upgrade	!
addPlayer	actual xPos	actual yPos		
movePlayer	move	actual xPos	actual yPos	
dropBomb	actual xPos	actual yPos		
createGame				
endGame				

Obrázek 3.11: Tabuľka s prehľadom posielaných správ a ich atribútov medzi zariadeniami pri sieťovej komunikácií

Tento spôsob implementácie odľahčuje množstvo výpočtov, ktoré by musel robiť server, ako je to pri aplikáciách, ktoré sú postavené na architektúre klient-server. Pri rôznych internetových hrách s touto architektúrou je vždy server zastúpený samostatným strojom s obrovskou výpočtovou silou, ktorý je prispôsobený na zvládanie veľkého počtu pripojených klientov a veľké množstvo vykonaných výpočtov.

Kapitola 4

Záver

Cieľom práce bolo navrhnúť a vytvoriť sieťovú hru na mobilný telefón N900 pomocou knižnice Qt vo vývojovom prostredí QtSDK. Hra umožňuje hrať naraz dvom hráčom proti sebe pomocou sieťovej komunikácie a hrať tzv. single-player mód, teda bez ľudského oponenta na druhom zariadení. Pred návrhom a samotnou implementáciou bolo potrebné naštudovať informácie o hardware a software zariadenia. Po získaní týchto vedomostí som nadobudol presvedčenie, že firma Nokia odviedla výbornú prácu na tomto mobilnom telefóne a dovolím si tvrdiť, že predbehla svoju dobu. Po získaní informácií o telefóne nasledovalo štúdium knižnice Qt a vývojového štúdia QtSDK. Informácie mi poskytovala najmä výborne napísana dokumentácia knižnice Qt, ktorá obsahuje nemalý počet príkladov slúžiacich k jednoduchšiemu pochopeniu jednotlivých tried a modulov knižnice. Samotná implementácia a testovanie aplikácie na telefóne by nebola možná bez naštudovania a zdokumentovania postupu ako správane prepojiť QtCreator s mobilným telefónom. Pri návrhu som zúročil informácie získane z dokumentácie a rôznych príkladov a čiastkových aplikácií, ktoré mi pomohli rýchlo pochopiť riešený problem a vybrať jeho najvhodnejšie riešenie. Po návrhu nasledovala implementácia a každou naimplementovanou časťou sa aplikácia začala viac podobať navrhovanému riešeniu. Azda najkrajšia časť implementácie bolo pridávanie grafickej časti. Aplikácia sa po pridaní grafiky začala viac podobať na svoju predlohu. Pridaná sieťová časť podľa môjho názoru zvyšuje atraktivitu aplikácie, kedže vzniká možnosť hrať hru s inými ľudmi po celom svete. Grafické užívateľské rozhranie je navrhnuté a naimplementované s dôrazom na intuitívne ovládanie, ktoré je umožnené už nazbieranými skúsenosťami z používania aplikácií na mobilných telefónoch. Ovládanie hracej postavičky pomocou akcelerometra by malo byť príjemným oživením, no zároveň sa môže zdať náročné a bez tréningu môže byť ťažkopádne. Preto je možnosť ovládať hraciu postavičku aj klikaním na vymedzené dotykové plochy mobilného telefónu. Bolo by odvážne tvrdiť, že aplikácia je dokonalá a že neostal žiaden priestor na zlepšovanie. V dnešnej dobe je užívateľ zvyknutý hodnotiť aplikácie hlavne podľa vzhľadu. Preto si myslím, že zlepšenie by si zaslúžila grafická stránka hry. Moje grafické schopnosti nie sú na dostatočnej úrovni, aby som vytvoril úplne novú a nápaditú grafiku, ale napriek tomu si myslím, že použitú grafiku ocenia pamätníci a fanúšikovia prvých sérií Bombermana. Porovnávanie vytvorenej aplikácie s podobnými aplikáciami dostupnými v aplikačných marketoch pre rôzne mobilné platformy (Bomberman Zombies a Bomberman Dojo pre Android, Bomberman Touch iOS pre iOS) odhalí nedostatky v už spomenutej grafickej stránke hry. Spoločnosti, vytvárajúce tieto aplikácie totiž spolupracujú s grafikmi na vysokej úrovni. Princíp hry je na všetkých spomenutých aplikáciach rovnaký. Teda likvidácia nepriateľov a protihráčov pomocou explózií bômb. Ovládanie hracích postavičiek sa uskutočňuje pomocou dotykových gest na

dotykovom displeji, preto ovládanie pomocou akcelerometra je relatívne exotické a zaujímave. Ak konkurenčné aplikácie ponúkajú možnosť sieťovej hry (Bomberman Touch), tak je postavená na komunikácii cez rozhranie Bluetooth a teda hráči sú limitovaný vzdialenosťou dosahu Bluetooth zariadenia. Na druhej strane využitie Bluetooth komunikácie odstraňuje problém nekvalitného, či chýbajúceho sieťového pripojenia. V budúcnosti by som rád pokračoval vo vývoji tejto aplikácie a hlavnou motiváciou je vytvoriť hru, ktorá sa bude dať hrať aj na ostatných mobilných platformách ako je Android, iOS či Windows Phone. Pri dostatku financií by v budúcnosti mohol vzniknúť aj server, ktorý bude hostiť vytvárané hry a zmení architektúru sieťovej komunikácie na klient-server.

Literatura

[1] Jon Masters, Richard Blum: *Linux profesionálne - programovaní aplikací*. Zoner Press, 2008, ISBN 978-80-86815-71-8.

[2] Molketin, D.: *The Book of Qt 4: The Art of Building Qt Applications*. Open Source Press GmbH, 2006, ISBN 978-3-937514-12-3.

[3] PDF dokument: Nokia N900: Návod na použitie. `http://nds1.nokia.com/phones/files/guides/Nokia_N900_UG_sk.pdf`, [cit. 2011-20-12].

[4] WWW stránky: Oficiálne stránky mobilného telefónu Nokia N900. `http://www.nokia.sk/produkty/telefony/nokia-n900/specifikacia`, [cit. 2011-20-12].

[5] WWW stránky: Oficiálne stránky operačného systému Maemo. `http://maemo.org/intro/platform/`, [cit. 2011-20-12].

[6] WWW stránky: Oficiálne stránky Texas Industries, špecifikácia OMAP3430. `http://www.ti.com/general/docs/wtbu/wtbuproductcontent.tsp?contentId=14649&navigationId=12643&templateId=6123`, [cit. 2011-20-12].

[7] WWW stránky: Oficiálne stránky Qt frameworku, Qt Creator. `http://qt.nokia.com/products/developer-tools`, [cit. 2011-21-12].

[8] WWW stránky: Dokumentácia nástroja Qt Simulator. `http://doc.qt.nokia.com/qtsimulator/index.html`, [cit. 2012-01-06].

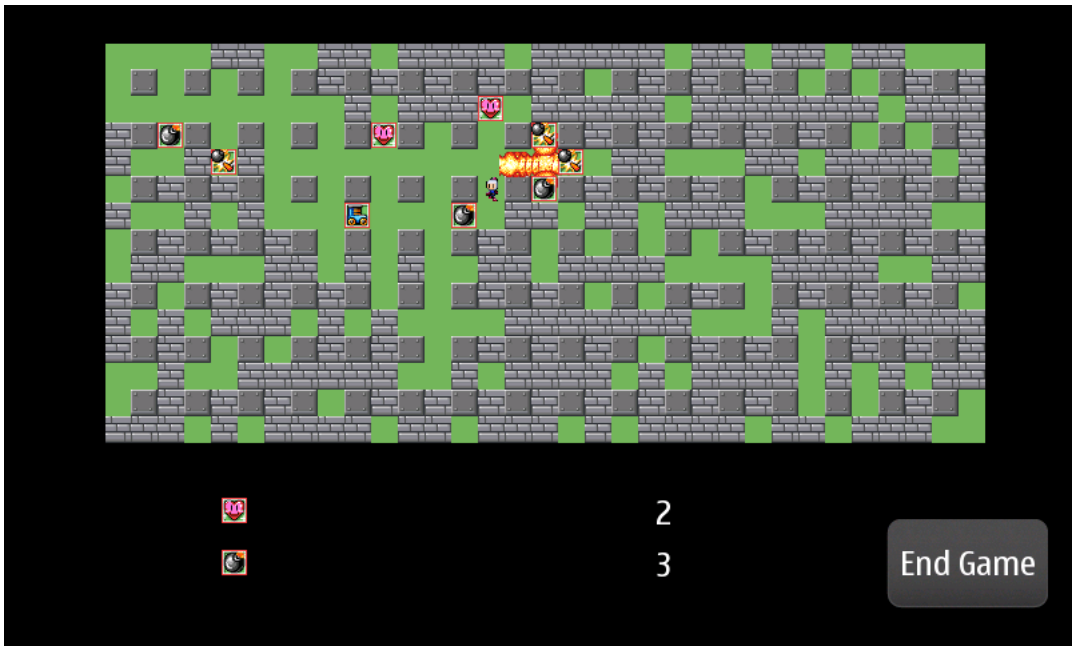
[9] WWW stránky: Stránky pre vývojárov aplikácií využívajúcich Bluetooth. `http://developer.bluetooth.org/KnowledgeCenter/TechnologyOverview/Pages/L2CAP.aspx`, [cit. 2012-01-06].

[10] WWW stránky: Stránky pre vývojárov aplikácií využívajúcich Bluetooth. `http://developer.bluetooth.org/KnowledgeCenter/TechnologyOverview/Pages/RFCOMM.aspx`, [cit. 2012-01-06].

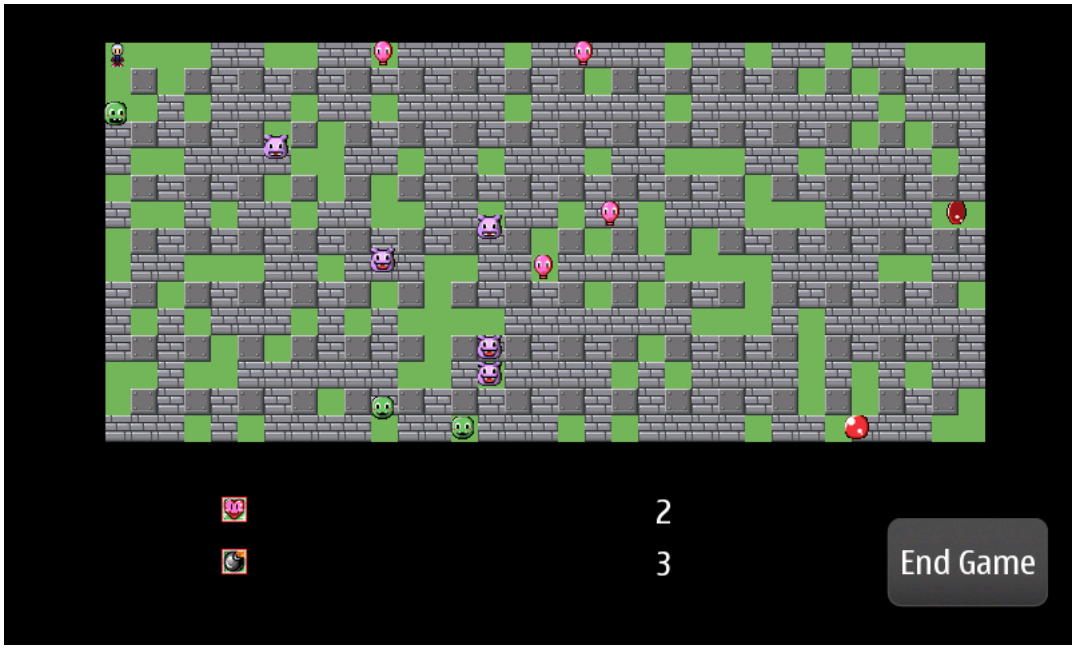
[11] WWW stránky: Dokumentácia API Qt Mobility. `http://doc.qt.nokia.com/qtmobility-1.2/index.html`, [cit. 2012-01-07].

Příloha A

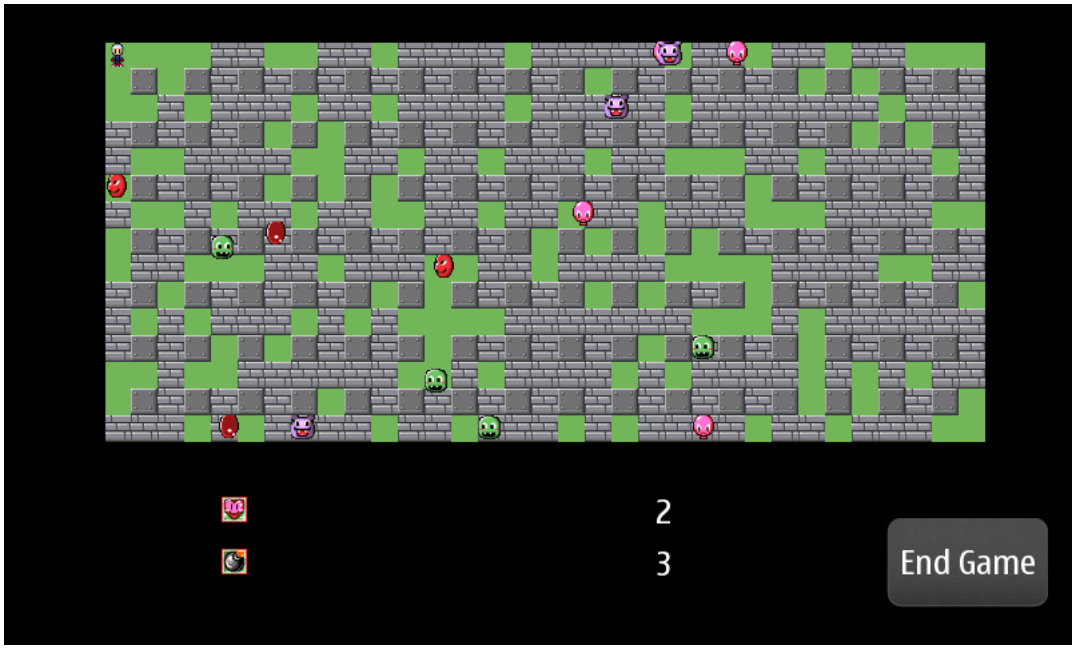
Obrázky z aplikácie



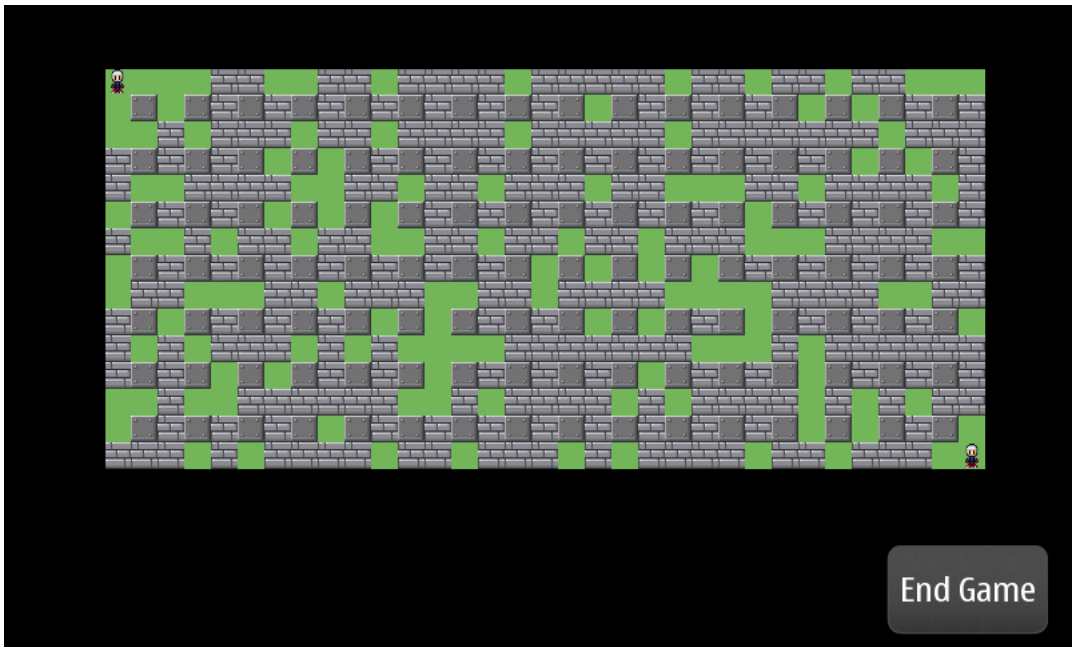
Obrázek A.1: Screenshot z aplikácie s prebiehajúcou explóziou bomby. Pri explózii sú zničené prekážky v horizontálnom a vertikálnom smere.



Obrázek A.2: Screenshot zo single player módu s pridanými monštrami



Obrázek A.3: Screenshot zo single player módu s pridanými monštrami



Obrázek A.4: Stav na hracej ploche po vytvorení sieťovej hry. Štartovacia pozícia protihráčov je v protiľahlých rohoch hraciej plochy.

Obrázek A.5: Screenshot zo sieťovej hry, výbuch zasahuje hráča. Dochádza k jeho zničeniu, prípadne zníženiu počtu životov. Po zničení hráča oponent vyhráva a hra je ukončená.

Příloha B

Obsah CD

1. Text bakalárskej práce vo formáte PDF
2. Text bakalárskej práce vo formáte TEX
3. Zdrojové kódy aplikácie
4. Inštalačný balíček aplikácie
5. Obrázky použité pri tvorbe textu práce
6. Grafika použitá v aplikácii
7. Krátke demonštračné video