EKONOMICKÁ UNIVERZITA V BRATISLAVE

FAKULTA HOSPODÁRSKEJ INFORMATIKY

Evidenčné číslo: 17 300/I/2012/3045545332

TVORBA APLIKÁCIÍ PRE MOBILNÉ ZARIADENIA

Diplomová práca

2012 **Roman Trtek Bc.**

EKONOMICKÁ UNIVERZITA V BRATISLAVE
FAKULTA HOSPODÁRSKEJ INFORMATIKY

TVORBA APLIKÁCIÍ PRE MOBILNÉ ZARIADENIA

Diplomová práca

Študijný program: Manažérske rozhodovanie a informačné technológie

Študijný odbor: 6258 Manažérske rozhodovanie a informačné technológie

Školiace pracovisko: Katedra aplikovanej informatiky

Vedúci záverečnej práce: Marián Šedivý, Ing.

Bratislava 2012 **Roman Trtek Bc.**

Čestné vyhlásenie

Čestne vyhlasujem, že záverečnú prácu som vypracoval samostatne a že som uviedol všetku použitú literatúru.

Dátum: 17.6.2012 **.....**

(podpis študenta)

ABSTRAKT

TRTEK, Roman: *Tvorba aplikácií pre mobilné zariadenia*. – Ekonomická univerzita v Bratislave. Fakulta Hospodárskej Informatiky; Katedra aplikovanej informatiky. – Školiteľ: Ing., Marián Šedivý – Bratislava: FHI, 2012, počet strán 53.

Cieľom záverečnej práce bolo analyzovať podiely mobilných operačných systémov na svetovom trhu mobilných operačných systémov, základné predstavenie a charakteristika jednotlivých operačných systémov a ich ekosystémov. Cieľom bolo tiež vybrať si konkrétnu mobilnú platformu a podrobnejšie ju opísať. Hlavným cieľom však bolo vytvoriť mobilnú aplikáciu a popísať postup jej tvorby. Práca je rozdelená do štyroch kapitol. Obsahuje 2 prílohy.

Prvá kapitola sa zoberá charakteristikou súčasného stavu na trhu mobilných operačných systémov. Stručne popisuje jednotlivé systémy a ich vývojové prostredia.

Druhá kapitola sa zoberá podrobnejším popisom vývojového prostredia na tvorbu aplikácie pre systém Windows Phone a jeho komponentov, požiadavkami na vývoj mobilnej aplikácie v tomto systéme .

V ďalšej časti sa zaoberá analýzou, návrhom vývojom konkrétnej aplikácie, jej testovaním a postupom zverejnenia do ekosystému vybranej platformy.

Výsledkom práce je plne funkčná aplikácia jednoduchého zápisníka úloh, ktorá je zverejnená a voľne prístupná na Microsoft Marketplace a podrobný popis postupu jej tvorby.

Kľúčové slová: tvorba mobilnej aplikácie, mobilný operačný systém, Windows Phone 7, vývojové prostredie Microsoft Visual Studio, dizajn aplikácie, základné požiadavky na vývoj, pravidlá dizajnovania mobilnej aplikácie, zápisník úloh

ABSTRACT

TRTEK, Roman: *Creation of applications for mobile devices* – University of Economics in Bratislava, Faculty of Economics Informatics; Department of Applied Informatics. – Supervisor: Ing., Marián Šedivý – Bratislava: FHI, 2012, number of pages: 53.

The aim of the thesis was to analyze the share of mobile operating systems on the world market for mobile operating systems, and basic performance characteristics of particular operating systems and their ecosystems. The aim was also to choose a specific mobile platform and describe it in more detail. However, the main goal was to create the Mobile Application and description of the procedure. The work is divided into four chapters. Contains 2 appendixes.

The first chapter deals the characteristics of the current state of the mobile operating systems. Briefly describes the systems and development environments.

The second chapter deals the detailed description of the development environment for creating application for Windows Phone and its components. It also concerns the requirements, that need to be satisfied for mobile application to be certified in this system.

The next section deals with the analysis, design, development of specific applicationn, the testing procedure and its publication on the ecosystem of the selected platform.

The result is a fully functional application of task recorder that is published and freely available on the Microsoft Marketplace and a detailed description of its creation.

Keywords: creating mobile applications, mobile operating system, Windows Phone 7 development environment Microsoft Visual Studio, design applications, the basic requirements for development, design rules for creating mobile application, task recorder

Poďakovanie

Touto cestou by som sa chcel poďakovať vedúcemu diplomovej práce Ing. Mariánovi Šedivému za odborné vedenie, ochotu a pomoc pri realizácii tejto práce.

Ekonomická univerzita v Bratislave
Fakulta hospodárskej informatiky

ZADANIE ZÁVEREČNEJ PRÁCE

Meno a priezvisko študenta: Bc. Roman Trtek
Študijný program: Manažérske rozhodovanie a informačné technológie (Jednoodborové štúdium, inžiniersky II. st., denná forma)
Študijný odbor:
Typ záverečnej práce: Inžinierska záverečná práca
Jazyk záverečnej práce: slovenský

Názov: Tvorba aplikácií pre mobilné zariadenia

Anotácia: S príchodom inteligentných mobilných zariadení sa otvárajú nové možnosti ich využitia. Tvorcovia aplikácií získali nový segment trhu, kde môžu realizovať svoje nápady. Práca prezentuje hotovú aplikáciu, ktorá sa dá využiť na mobilnom zariadení, ako aj postup jej tvorby.

Vedúci: Ing. Marián Šedivý
Katedra: KAI FHI - Katedra aplikovanej informatiky FHI
Vedúci katedry: doc. Ing. Gabriela Kristová, CSc.

Dátum zadania: 01.11.2010

Dátum schválenia: 01.10.2011

doc. Ing. Gabriela Kristová, CSc.
vedúci katedry

OBSAH

Úvod

S obrovským rozmachom výpočtovej techniky a v poslednej dobe hlavne presunu používateľov stolových počítačov k mobilným zariadeniam, ktoré dosahujú výkon osobných počítačov z doby pred niekoľkými rokmi, stoja vývojári pred otázkou kam sústrediť svoj intelekt a schopnosti aby dokázali využiť tento posun vo svoj prospech. Preto je žiaduce aby sa vedeli orientovať v problematike platforiem dostupných operačných systémov a vedeli si vybrať správnu platformu pre ktorú je užitočné vyvíjať svoje produkty. Je tiež vhodné vedieť ako sa konkrétnou platformou pracuje respektíve ako sa postupuje pri vývoji konkrétnej aplikácie.

Preto sme sa v našej práci pokúsili potenciálneho vývojára oboznámiť s dostupnými platformami a umožnili sme mu nahliadnuť do procesu tvorby aplikácie pre konkrétnu platformu. Diplomová práca je venovaná tvorbe aplikácií pre mobilné zariadenia konkrétne aplikácie pre Windows phone 7.

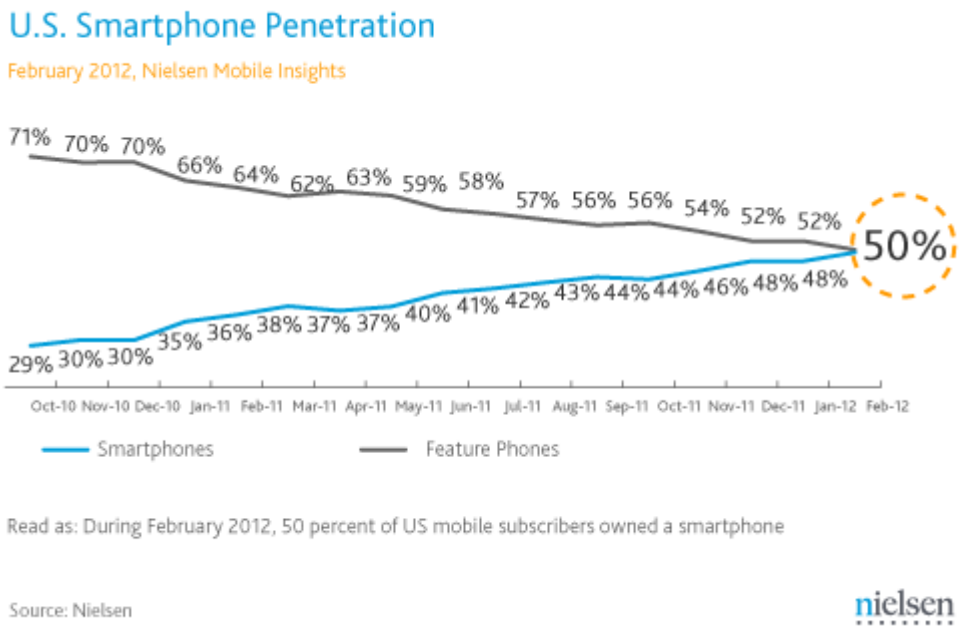
V prvej kapitole diplomová práca pojednáva o súčasnom stave na trhu mobilných operačných systémov, respektíve ich platforiem. Stručne popisuje ich históriu, zákazníkov, počet dostupných aplikácií, vývojové prostredie a programovací jazyk aký sa pri ich vývoji používa..

Druhá kapitola sa zoberá obecným popisom vývojového prostredia na tvorbu aplikácie pre systém Windows Phone a jeho komponentov, požiadavkami na vývoj mobilnej aplikácie v tomto systéme systéme .

V ďalšej časti sa zaoberá analýzou, návrhom vývojom konkrétnej aplikácie, jej testovaním a postupom zverejnenia do ekosystému vybranej platformy.

2. Súčasný stav

V dnešnej dobe sa mobilné telefóny stali nevyhnutnou súčasťou nášho života. Okrem základných funkcií ako je telefonovanie a posielanie SMS správ, majú v sebe čím ďalej tým viac integrovaných množstvo funkcií – od multimediálnych cez pracovné, až po navigačné a čoraz viac sa podobajú na osobné počítače. Majú vlastný operačný systém, podporujú inštaláciu rôznych programov a dokážu sa bezdrôtovo pripojiť na internet. Také telefóny, ktoré túto vymoženosť obsahujú, časom dostali názov múdre telefóny[1]. Trh s týmito telefónmi každoročne rastie a to dokonca v takej miere, že v blízkej budúcnosti objemy predaja smartphonov presiahne objem predaja takzvaných feature telefónov[2]. Napríklad v Spojených štátoch Amerických sa podľa agentúry Nilsen sa tak stane už tento rok[1]. V globálnom miere predaje múdrych telefónov podľa agentúry IDC dokonca predbehnú aj predaj stolných PC[4].



OBRÁZOK 1 VÝVOJ PENTRÁCIE SMARTPHONOV V USA**[1]**

[1] **Z angličtiny smartphones**
[2] **Telefóny s jednoduchým operačným systémom prispôsobeným na konkrétny model**

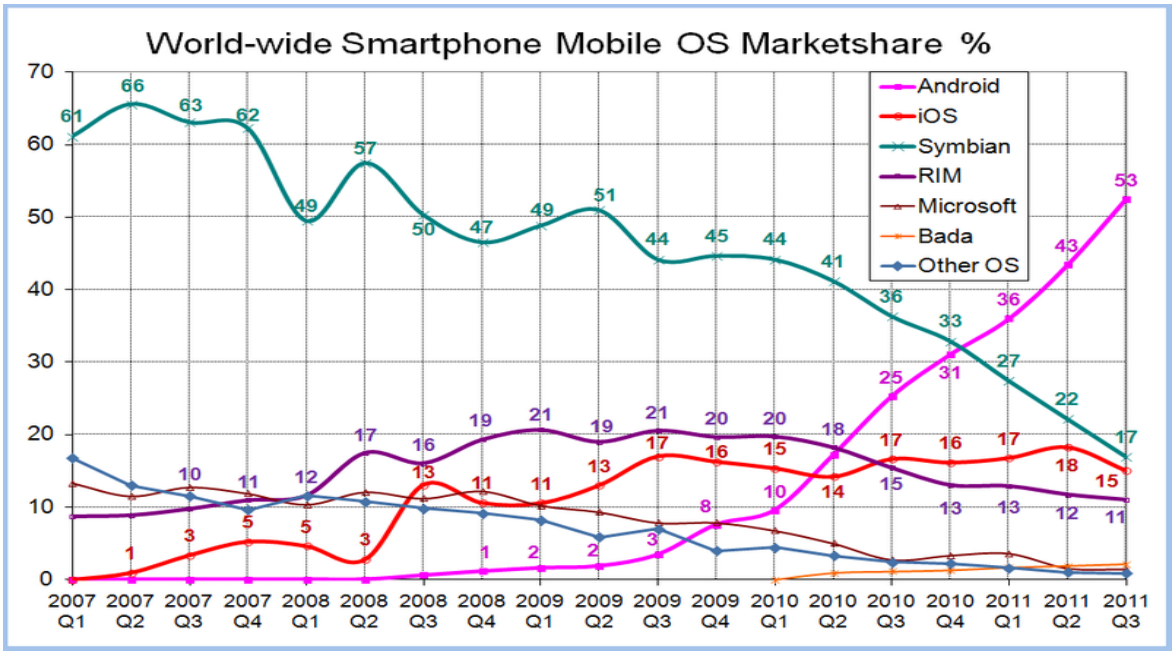
Tento rast dokonca umožnil firme Apple pôvodne sa venujúcej výrobou PC a software stať sa vďaka telefónom najhodnotnejšou firmou sveta[5] .

Ostatní výrobcovia tiež nezaháľajú a predstavili svoje verzie telefónov, či už s vlastným operačným systémom alebo používajú systém dodávaný spoločnosťou Google alebo Microsoft.

Mobilné telefóny boli pôvodne ovládané hardvérovou klávesnicou, postupom času však trend nastoľujúci neustále zväčšovanie displejov spôsobil, že hardvérová klávesnica ustúpila do pozadia a nahradilo ju dotykové ovládanie obrazovky. Táto zmena bola dôležitá nielen z hľadiska potreby zvýšiť počet zobrazených informácií ale aj z dôvodu lepšieho ovládania telefónu prstami.

2.2. Operačné systémy mobilných zariadení

Najbežnejšie softvérové platformy používané na mobilných telefónoch sú Android, iOs, BlackBerry OS, Symbian OS, webOS, Windows Phone, Bada a MeeGo. Zastúpenie jednotlivých operačných systémov na trhu mobilných predstavuje graf na obrázku č.2.



OBRÁZOK 2 PODIEL PREDAJA JEDNOTLIVÝCH OPERAČNÝCH SYSTÉMOV V CELOM SVETE [26]

2.2.1. Android

Android od spoločnosti Google, ktorý bol pôvodne vyvíjaný malou spoločnosťou Android inc. založenou v roku 2003 a neskôr v roku 2005 bol spoločnosťou Google odkúpený. Je to systém založený na Linuxe, je dostupný pod open source licenciou a spadá pod Open Handset Alianciu ktorá združuje vývojárov tejto platformy. Súčasťou tejto aliancie, založenej v roku 2007 sú firmy ako okrem zakladajúcej spoločnosti Google aj spoločnosti HTC, Intel, Samsung a iné. Tým že systém android spadá pod Open Handset Alianciu sa dosiahlo, že ho môžeme nájsť na mobilných zariadeniach ako mobilné telefóny, tablety, hodinky atď. od rôznych výrobcov, ktorí si jeho rôzne verzie môžu upraviť podľa svojich potrieb a možností.

Systém Android začínal verziou 1.0 ktorá bola implementovaná do telefónu T-Mobile G1 ako prvého telefónu s týmto operačným systémom. Postupom času sa objavili ďalšie vývojové vezie tohto systému ktoré pridávali podporu viacdotykového ovládania, podpora VoIP hovorov, podpora Adobe Flash priamo v zabudovanom internetovom prehliadači a ďalšie. Najaktuálnejšia verzia je verzia číslo 4.0.4 s kódovým názvom Ice Cream Sandwich.

Programátori pre systém Android môžu na vytváranie aplikácii použiť vývojársky balík pre Android, ktorý obsahuje sadu knižníc, emulátor zariadení, nástroj na ladenie chýb, sadu knižníc, dokumentáciu, ukážky zdrojových kódov a tiež rôzne návody. Vývojársky balík je voľne stiahnuteľný na internetových stránkach Android Developers. Balík podporuje systémy Linux, Mac OS X 10.4.9 alebo vyšší alebo Windows XP alebo novší. Odporúča sa vývojové prostredie Eclipse 3.5 alebo novšie, pre ktoré existuje zásuvný modul Android Development tools. V tomto modeli sú obsiahnuté všetky potrebné doplnky potrebné na vývoj aplikácií. Aplikácie sú vyvíjané v programovacom jazyku Java ale namiesto štandardného virtuálneho stroja pre beh aplikácií Java – Java Virtual Machine využíva svoj vlastný s názvom Dalvik. Tento bol vytvorený špeciálne pre potreby tohto operačného systému. Je optimalizovaný tak aby mohol fungovať aj na mobilných zariadeniach, to znamená, že počíta s obmedzeniami ktorými sú mobilné zariadenia limitované. Tieto obmedzenia sú napríklad veľkosť dostupnej operačnej pamäte, rýchlosť procesora a sila napájania. Aplikácie môžu vývojári testovať priamo

pomocou pripojeného mobilného zariadenia, prípadne ak ho nemajú k dispozícii, tak pomocou emulátora. Programátor má pri vývoji možnosť pristupovať k viacerým hardvérovým a systémovým funkciám prístroja. Pri vývoji mu platforma umožňuje prístup k posielaniu SMS alebo MMS správam, hovorom, detekcii magnetického poľa, akceleometru, gyroskopu, GPS, kontaktom atď. Tým pádom nie je limitovaný pri vývoji aplikácií, čo podporuje ich rozmanitosť. Pre Android existuje už viac ako 430 000 aplikácií pričom asi jednu tretinu z nich tvoria veľmi jednoduché aplikácie[7][8].

1.2.2 BLACKBERRY OS

Ďalšou platformou v poradí je systém Blackberry OS od kanadskej spoločnosti Research in Motion (RIM). Táto spoločnosť sa však viac orientuje na firemných zákazníkov, keďže má z nich najväčšie príjmy[6]. Firemný zákazníci ju vyhľadávajú najmä kvôli úzkemu prepojeniu prístrojov s internetovými službami, respektíve možnosťou synchronizácie s firemným alebo osobným účtom na ktorom má zákazník kontakty, email, kalendár atď. Tieto účty sú šifrované[9] a dáta na nich sú vždy aktuálne a užívateľom dostupné odkiaľkoľvek kde je dostupné pokrytie umožňujúce dátové prenosy. Aplikácie pre Blackberry OS sa dajú vyvíjať v prostredí Eclipse karé má v sebe zahrnutý zásuvný modul Blackberry Java SDK v6.0. Vývoj prebieha prostredníctvom programovacieho jazyka Java. Vývojári majú k dispozícii emulátory všetkých prístrojov značky. Na nich potom môžu svoje aplikácie v prípade potreby otestovať. Systém Blackberry OS je nainštalovaný len na zariadeniach od spoločnosti RIM, ale keďže jeho základy sú postavené na programovacom jazyku Java v niektorých prístrojoch (tabletoch od tohto výrobcu) je možné prostredníctvom emulátora inštalovať aj aplikácie pôvodne určené pre systém Android. Pre systém Blackberry Os existuje momentálne viac ako 60000[10].

1.2.3. iOS

Mobilný operačný systém ktorý je odvodený od operačného systému používaného na stolných PC a iných zariadeniach od spoločnosti Apple. Je prispôsobená na fungovanie na mobilných zariadeniach a preto neobsahuje všetky súčasti pôvodného systému. Má však niektoré doplnky ako podpora dotykového ovládania, podpora akcelerometra a hlasového ovládania ktoré pôvodný operačný systém neobsahuje.

Prvá verzia systému pôvodne pomenovaného iPhone OS nepodporovala inštaláciu aplikácií tretích strán, vývojári mohli dodávať len webové aplikácie. To sa však zmenilo s príchodom verzie iPhone OS 2.0 kedy bol aj predstavený obchod s aplikáciami AppStore. Aktuálna verzia systému je iOS 5.1 ktorý je už podporovaný viacerými zariadeniami ako tablet iPad a televízia Apple TV umožňuje prepojenie s cloud službami iCloud, pridáva podporu pokročilého hlasového ovládania Siri a iné. Funkcionalita systému respektíve telefónu so systémom iOS sa dá rozšíriť inštaláciou aplikácií, ktoré sú dostupné na stiahnutie prostredníctvom obchodu AppStore. Ten obsahuje veľké množstvo platený ako aj voľne stiahnuteľných aplikácií. V súčasnosti sa počet aplikácií pohybuje okolo hodnoty 500 000 [11].

Aplikácie sa pre Apple programujú v jazyku C alebo Objektovom C, vývojové prostredie však funguje len pod operačným systémom Mac OS X. Vývojové prostredie obsahuje Xcode ktorého súčasťou je nástroj na tvorbu grafického rozhrania vyvíjaného programu, nástroj na hľadanie chýb a nástroj na ladenie výkonu. Súčasťou vývojového prostredia je aj iOS knižnica pre vývojárov, emulátor iPad a iPhone prístroja. Vývojári môžu na úrovni jadra pristupovať k funkciám fotoaparátu, akcelerometra ,GPS, snímaču magnetického poľa, kontaktom atď. Ak chce vývojár testovať aplikáciu priamo na telefóne musí sa zaregistrovať do plateného Apple iOS vývojárskeho programu. Spoločnosti zaplatia za registráciu 299 amerických dolárov, jednotlivci to majú o niečo lacnejšie zaplatia 99 amerických dolárov. Pre študentov a pracovníkov univerzít je určený iOS Developer University Program, kde nemusia platiť registračný poplatok, ale sú obmedzený distribúciou aplikácie – môže ju skúšať len maximálne 200 členný tým a aplikácie takto vytvorené sa nesmú poskytovať komerčne.

Pre vývojárov ktorý sa nechcú viazať na desktopový operačný systém Mac OS X, respektíve nemajú k nemu prístup existuje alternatíva k natívnemu vývojovému prostrediu pre iOS. Poskytuje je spoločnosť Adobe prostredníctvom programu Adobe Flash CS5 ktorá umožňuje kompilovať zdrojové kódy vytvorené pomocou tohto vývojového prostredia do formy ktorú je potom možné spustiť na iOS.

1.2.4. SAMSUNG BADA

Systém ktorý sa používa len na telefónoch značky Samsung. Vznikol doplnením dotykového ovládania, prepracovaním grafického rozhrania, pridaním 3d grafiky a najmä doplnením schopnosti inštalácie aplikácií tretích strán do pôvodného systému, čím sa stal plnohodnotným operačným systémom pre mobilné telefóny.

V súčasnosti sa systém nachádza vo verzii 2.0 ktorá prináša podporu multitaskingu pre všetky aplikácie, blízko poľovej komunikácie[3], podpora WIFI Direct[4], technológie Adobe Flash a iné. Pre systém je v obchode Samsung Apps momentálne dostupných približne 13000 aplikácií[12].

Tie môžu vývojári vyvíjať priamo vo vývojovom prostredí ktoré vydal Samsung. Vývojové prostredie je založené na prostredí Eclipse . Vývojové prostredie v sebe obsahuje aj emulátor, knižnice, konštruktor[3] užívateľského prostredia, príkladné aplikácie, dokumentáciu, príručku a iné. Systém Samsung bada podporuje ovládače pre Adobe Flash, webový prehliadač, pohybový senzor, jemne nastaviteľné vibrácie, a rozpoznávanie tvárí čo otvára nové možnosti pre väčšiu kreativitu a používateľskú interaktivitu. Je určené pre operačný systém Windows od verzie XP. Aplikácie sa vyvíjajú v programovacom jazyku C/C++.

[3] NFC- near field communication

[4] Priame prepojenie prístrojov s WIFI prijímačom bez potreby prístupového bodu

1.2.5. SYMBIAN OS

Bol ešte nedávno (v roku 2010)[13] najrozšírenejší operačný systém medzi mobilnými telefónmi, bol nainštalovaný na mobilných zariadeniach rôznych výrobcov ako Nokia, Samsung a Sony Ericsson a ďalších. Platforma Symbian bola vytvorená spojením a integrovaním aktív spoločnosti Nokia, NTTDoCoMo, Sony Ericsson a Symbian Ltd., zahŕňajúcim aktíva operačného systému Symbian ako jadro, platformu S60 a časti UIQ a MOAP(S) ako používateľské rozhranie. Neskôr ho vzala pod patronát spoločnosť Nokia, ktorá jeho vývoj v roku 2011 outsourcovala spoločnosti Accenture. V súčasnosti však systém rýchlo stráca trhový podiel. Do budúcnosti sa uvažuje s ukončením jeho vývoja, keďže Nokia oznámila, že bude túto platformu podporovať len do roku 2016[14].

Pred rokom 2009 Symbian podporoval viacero užívateľských rozhraní ako napríklad UIQ od UIQ tchnologies, S60 od Nokie a MOAP od NTT DOCOMO. Tie sa v roku 2009 spojili do jedného. Posledné verzie systému boli 9.3 , potom verzia ^2, neskôr verzia ^3 (Symbian OS 9.5) aktuálne je systém označovaný spoločnosťou Nokia ako Nokia Belle (Symbian OS 10.1) ktorá priniesla lepšiu podporu webových technológií, vylepšenú prispôsobiteľnosť užívateľského prostredia používateľom, rýchlejšiu odozvu a vylepšené dotykové ovládanie.

Pre systém Symbian je určené vývojové prostredie Qt, ktoré môže byť použité s prostredím Qt Creator alebo Carbide.c++. Je dostupné pre platformu Windows, Mac OS X ako aj pre operačný systém Linux. Na vývoj aplikácií sa používa programovací jazyk C++ alebo QML[5]. Alternatívne sa aplikácie môžu vytvárať použitím jazyka Python, Adobe Flash Lite, Ruby, .NET, WRT alebo Java ME. Aplikácie sú po vývoji dostupné na stiahnutie v Nokia Obchode, ich počet sa pohybuje okolo čísla 115 000 [15].

1.2.6. WEBOS

Operačný systém od spoločnosti HelwetPackard, založený na Linuxovom Jadre, pôvodne vyvíjaný spoločnosťou Palm ktorá bola spoločnosťou HelwetPackard odkúpená.

[5] **Qt modelovací jazyk**

Systém bol predstavený v roku 2009 ako nástupca Palm OS a odvtedy boli vydané rôzne verzie tohto systému. Posledná verzia systému je 3.0.5, ktorá prináša podporu pre tablety.

Vývojové prostredie webOS SDK vo verzii 3.0.5 umožňuje vývojárom vytvárať aplikácie pre tento systém. Vo vývojovom prostredí sa nachádza aj PDK ktoré poskytuje vývojárom prístup ku kompilátorom, knižniciam, skriptom, utilitám a dokumentom ktoré pomáhajú pri vývoji v jazyku C/C++. Aplikácie môžu byť napísané aj v jazyku HTML5, Java Skripte a CSS použitím Mojo alebo Eyno vývojového prostredia.

Aplikácie sú dostupné cez HP App catalog ktorý slúži ako obchod s aplikáciami. Ďalší zdroj kde sa dajú aplikácie získať je Homebrew, ale ten nie je spoločnosťou HP priamo podporovaný. Podľa vývojového poradcu HP Josha Marinacciho pre systém webOS počet oficiálnych aplikácií v roku 2011 presiahol číslo 1000, dnes ich je podľa stránky webOS Nation[16] viac ako 7000.

1.2.7. MEEGO

Open source operačný systém založený na Debian Linuxe. Primárne určený na mobilné zariadenia, informačné zariadenia pre trhy spotrebnej elektroniky a vstavané zariadenia.. Prvý krát bol oznámený ako nástupca operačného systému Maemo v roku 2010 spoločnosťami Intel a Nokia. Neskôr sa pridali spoločnosti Novell, AMD a Nadácia Linux[6]. Do budúcnosti sa počíta s vývojom pod názvom Tizen ale bez podpory Nokie. Posledná verzia systému bola MeeGo 1.3.[17]

Na vývoj aplikácií pre tento systém sa používa Qt Creator a Qt framework, programuje sa v C++, vývojové prostredie je dostupné pre operačný systém Windows a Linux. Tiež je podporovaná tvorba aplikácií pomocou GTK[7], na kompilovanie aplikácií sa používa openSUSE Build Service. Pre systém je momentálne dostupných približne 480 aplikácií, ale keďže systém je postavený na jadre Debian Linuxu, umožňuje inštaláciu Debian balíčkov[8][18], toto číslo môže byť väčšie.

[6] Linux Foundation
[7] Gimp Tool Kit -multiplatformová sada nástrojov na vytváranie grafického rozhrania
[8] Zdrojové súbory programov zabalené do inštalačného súboru

1.2.8. WINDOWS PHONE 7

Nástupca operačného systému Windows mobile od spoločnosti Microsoft, s ktorým však nemá mnoho spoločného. Je to nová platforma a tým pádom so starým systémom nekompatibilná. Oproti Windows mobile je viac zameraný pre spotrebiteľov ako pre firemnú klientelu. Bol vydaný v roku 2010 a je dostupný pre viacerých výrobcov mobilných zariadení. V roku 2011 oznámil Microsoft partnerstvo so spoločnosťou Nokia a systém Windows Phone 7 sa stal jej hlavným operačným systémom pre múdre telefóny. Vďaka tomu je najviac zariadení na trhu so systémom Windows Phone práve od spoločnosti Nokia[19]. Na tejto skutočnosti sa podieľa aj fakt že telefóny od Nokie obsahujú aplikácie, napríklad Nokia Mapy, ktoré nie sú pre zariadenia od iných výrobcov dostupné. Prvá verzia systému dostupná pre telefóny bola 7.0.7004. Ďalšia významná verzia mala kódové označenie Mango (číslo verzie 7.10.7720) – priniesla veľký počet významných vylepšení ako napríklad podpora behu viacerých aplikácií naraz[9], podpora hlasového ovládania, podpora HTML 5 atď. a zlepšila konkurencieschopnosť platformy. Aktuálne je na trhu verzia s číslom7.10.8107 ktorá priniesla podporu sietí LTE[10].

Aplikácie pre systém Windows Phone 7 môžu byť vytvárané buď technológiou Silverlight prispôsobenou pre mobilné prístroje alebo technológiou XNA. Technológia XNA je vhodná na vývoj 3D a 2D hier a aplikácií ktoré vyžadujú vysoký grafický výkon. Technológia Silverlight je vhodná na vývoj bežných aplikácií pre ktoré je aj svojimi technológiami prispôsobená.

Vývojári môžu programovať svoje aplikácie v jazyku C# prípadne Visual Basic, kdežto grafické rozhranie aplikácie sa vyvíja pomocou značkovacieho jazyka XAML, ktorý je rozšírením značkovacieho jazyka XML. Na vývoj slúži vývojové prostredie Visual Studio 2010 s nástrojmi Windows Phone Developer Tools. Toto prostredie je dostupné pre operačný systém Windows Vista SP2 a novší. Súčasťou vývojového prostredia je aj emulátor, ktorý vie okrem iného emulovať aj rozhranie GPS prijímača, akcelerometra a

[9] Podpora multitaskingu

[10] Long term evolution – moblné dátové siete 4.generácie

magnetometra. Vyvinuté aplikácie môžu vývojári umiestňovať do obchodu Marketplace. V súčasnosti exituje pre túto platformu už viac ako 80 000 aplikácií [20].

1.3 Výber platformy

Platforma Windows Phone 7 je mladý rýchlo sa rozvíjajúci ekosystém, ktorý má do budúcnosti veľký potenciál rastu. Platforma je postavená na moderných technológiách Silverlight a XNA a podporujú programovacie jazyky C#, Visual Basic a XAML[11], ktoré sú programátorsky prívetivé a umožňujú efektívne vytváraní aplikácií. Keďže je táto platforma ešte len vo svojich počiatkoch, neexistuje pre ňu také množstvo aplikácií ako pre iné platformy ako napríklad Android alebo iOs. Napriek tomu si ju zvolil ako svoju hlavnú platformu jeden s najväčších výrobcov mobilných telefónov Nokia.

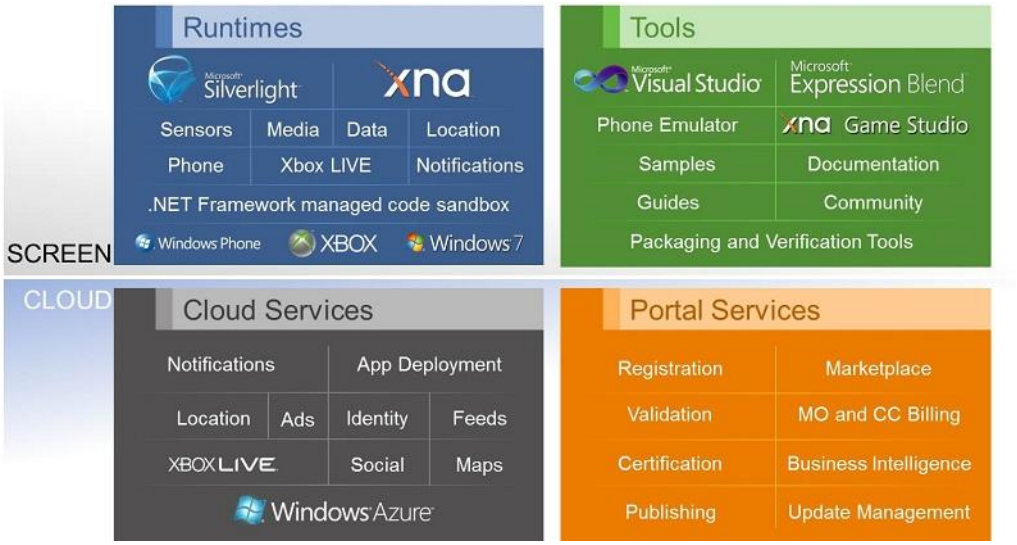
1.4 Architektúra platformy

Architektúra platformy sa skladá zo 4 hlavných komponentov:

- Exekučné prostredia – Silverlight a XNA rámec spolu so špecifickými funkciami
- Nástroje – Visual Studio a Expression Blend a ich pridružené nástroje a dokumentácia – vytvárajú kompletné vývojové prostredie na rýchlu tvorbu, opravu, vypustenie a aktualizovanie aplikácií
- Cloud služby[12]- Windows Azure, Xbox Live služby, Notifikačné služby a Lokalizačné služby umožňujú vývojárom zdieľať dáta prostredníctvom cloudu. Cloud služby sú výhodné pre spotrebiteľov, pretože umožňujú jednoduchý prechod medzi zariadeniami, ktoré používajú. Pripojenie na webové služby tretích strán sú takisto podporované.
- Portálové služby – obchod Windows Phone Marketplace poskytuje vývojárom

[11] Extensible Application Markup Language – rozšíriteľný značkovací aplikačný jazyk – deklaratívny jazyk založený na XML vytvorený spoločnosťou Microsoft
[12] Služby prístupné len prostredníctvom siete internet

možnosť registrovať, certifikovať a predávať svoje aplikácie.



OBRÁZOK 3 ZDROJ: HTTP://MSDN.MICROSOFT.COM/EN-US/LIBRARY/FF402531(V=VS.92).ASPX

1.4.1. EXEKUČNÉ PROSTREDIA

Technológiami Silverlight a XNA je kód písaný v . NET manažovanom kóde[13], ktorý je spúšťaný v chránenom sandboxe [14], čo urýchľuje vývoj bezpečných a dobre zabezpečených aplikácie. Aplikácie, ktoré sú už v Silverlight a XNA napísané, sa po jednoduchých úpravách, ako je zmena rozlíšenia obrazovky a pridaním vlastností špecifických pre telefóny (napr. využitie senzorov) sa dajú používať v systéme Windows Phone.

Platforma Windows Phone 7 ponúka 2 hlavné technológie plus doplnkové komponenty na vývoj aplikácií (ich štruktúra je zobrazená na obr. 4):

1. XNA [15] technológia je zložená zo služieb, softvéru a zdrojov zameraných na vývoj hier pre platformu Microsoftu ktorá zahŕňa Xbox 360, Windows Phone, Zune HD a

[13] Kód ktorý môže byť spustený len pomocou virtuálneho stroja CLR
[14] Bezpečnostný mechanizmus, respektíve druh virtualizácie behu programu, keď program nemá priamy prístup ku zdrojom alebo je jeho prístup značne obmedzený
[15] Xbox New Architecture – nová architektúra hernej konzoly Xbox od Microsoftu

Windows 7. Zahŕňa všetky potrebné API[16] ktoré v 2D umožňujú rotáciu, zmenu rozmerov, naťahovanie a filtrovanie, ako aj API potrebné na generovanie 3D geometrie, textúr, a štandardné osvetlenie a tieňovanie.

2. Silverlight –technológia na vytváranie bohatých používateľských rozhraní s vzhľadom internetovej aplikácie. Silverlight aplikácie sprístupňujú používateľské rozhranie pomocou súboru stránok. Na tvorbu Silverlight aplikácií je určené Visual Studio v ktorom sa dajú tvoriť rozhrania pomocou jazyka XAML. Vo Visual Studio je možné využiť veľký počet Silverlight a Common Base Class[17] knižníc. Platforma Windows Phone umožňuje spájať technológie Silverlight a XNA do jednej aplikácie.

3. Senzory – vývojári môžu použiť pomocou príslušných API výstupné dáta zo senzorov akcelerometra, kompasu, gyroskopu, dotykového senzora, mikrofónu a senzoru polohy.

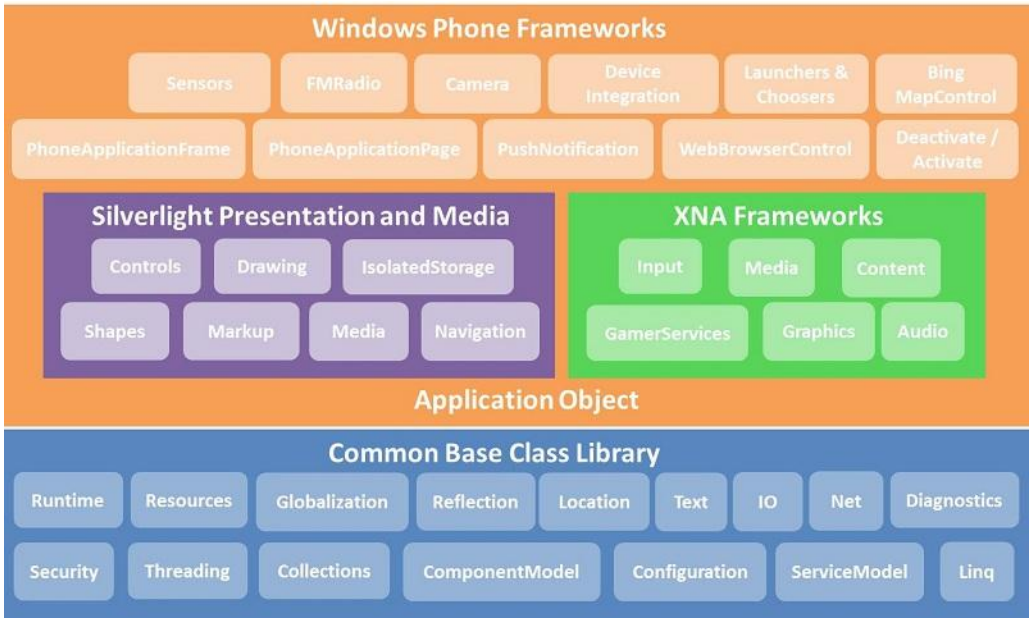
4. Médiá – technológie Silverlight a XNA poskytujú vývojárom model na konštrukciu používateľských rozhraní s bohatou grafikou, animáciami a podporou multimédií.

5. Dáta – izolované úložisko povoľuje aplikáciám vytvárať a uchovávať dáta v izolovanom virtuálnom priečinku. Všetky vstupné a výstupné operácie sú obmedzené len na toto úložisko a aplikácie tretích strán nemajú priamy prístup k súborom operačného systému. Tým sa zabraňujú neautorizovanému prístupu a znehodnoteniu dát ktoré môžu spôsobiť aplikácie tretích strán. Štruktúrované dáta môžu byť pristúpené pomocou LINQ[18] na SQL.

6. Poloha - vývojári môžu pristupovať pomocou služby Microsoft Location Service k dátam o skutočnej fyzickej polohe užívateľa vďaka jedinej API. Môžu tiež pristupovať k dátam o smerovaní, rýchlosti a vypočítať vzdialenosť medzi bodmi,

[16] Application programming interface – aplikačné programové vybavenie
[17] Poskytuje triedy ktoré zapuzdrujú množstvo základných funkcií ako je čítanie a zápis súborv, manipuláciu s XML a ďalšími
[18] Language Integrated Query – komponent NET. Frameworku ktorý pridáva schopnosť natívneho dátového dotazovania

môžu nastaviť potrebnú presnosť výstupných údajov a tiež vytvárať udalosti, ktoré sú vyvolané zmenou polohy.



OBRÁZOK 4., ZDROJ:HTTP://CREATE.MSDN.COM/EN-US/EDUCATION

1.4.2. NÁSTROJE

Vývojár si môže stiahnuť a nainštalovať buď celý balík alebo samostatné aplikácie potrebné na vývoj pre platformu Windows Phone:

- Visual Studio 2010 – vývojové prostredie na tvorbu aplikácií, obsahuje dizajnér, debugger[19] , systém projektov, balíkovač[20] a generátor súpisov
- Expression Blend – nástroj na vytváranie užívateľských rozhraní pomocou technológie Silverlight
- Windows phone emulátor – určený na rýchle a efektívne testovanie vytvorených aplikácií. Obsahuje podporu pre emuláciu GPU[21] a zmenu orientácie
- XNA Game Studio- rozširujúci Visual Studio o XNA technológie a tiež nástroje a knižnice ktoré vkladajú grafický a zvukový obsah potrebný na vývoj hier. Na inštaláciu Visual Studia a jeho súčastí je možné na systémoch Windows XP

[19] **Nástroj na vyhľadávanie chýb v zdrojovom kóde**
[20] **Nástroj na vytváranie inštalačných balíkov**
[21] **Graphical Processing Unit – procesorová jednotka určená na výpočet grafiky**

(x86) SP3 alebo vyšších, s minimálnou hardwarovou konfiguráciou ktorá obsahuje procesor s taktom 1,6GHz alebo rýchlejší, 1GB RAM (2GB pri 64 bitovej architektúre), 3GB dostupného diskového priestoru, 5400 otáčok/min pevný disk alebo rýchlejší, DirectX9 kompatibilnú grafickú kartu schopnou zobraziť aspoň 1024 x 768 alebo lepšiu [21].

- Príklady, dokumentácia, príručky a komunita – na urýchlenie vývoja sú dostupné rôzne vzorové aplikácie, zdrojové kódy, ako na to príručky a hlavne fóra na zdieľanie informácií vo vývojárskej komunite.

1.4.3. CLOUD SLUŽBY

S podporou cloud služieb je umožnené vývojárom vytvárať aplikácie, ktoré sú integrované vo webe. Cloud umožňuje že tieto služby sú stále prístupné, obsahujú viac funkcií, sú škálovateľnejšie a nie sú závislé na napájaní lokálneho zariadenia. Služby postavené na technológii Azure[22] sa môžu použiť na získanie dát do telefónu.

- Notifikácie- Windows Phone platforma poskytuje API na spustenie služieb ktoré oznamujú užívateľovi relevantné udalosti. Prináša podporu notifikácií ktoré obmedzujú potrebu dotazovania sa zariadenia na nové udalosti a tým redukujú spotrebu batérie.

- Lokalizačné cloud služby – spolupracujú s lokalizačnými API čím sa stávajú viditeľnými pre vývojára. Služba využíva dáta z WiFi mobilnej siete a GPS a tiež A-GPS a poskytuje ich ako jediný zdroj o polohe s ktorým môže vývojár pracovať.

- Služby Máp, sociálne služby, zdroje, identity – množstvo webových služieb v cloud umožňuje vzájomnú interakciu, identifikáciu užívateľov na sociálnych sieťach, prijímanie nových dát a použitie máp na navigáciu. Vývojári môžu využiť tieto možnosti na tvorbu nových aplikácií.

[22] Cloud computingová platforma Microsoftu používaná na vytváranie, hostovanie a škálovanie webových aplikácií pomocou datacentier Microsoftu

- Windows Azure – cloud computingová služba hostovaná v Microsoft data centrách, poskytuje škálu funkcií na budovanie aplikácií od bežných používateľov až po firemných zákazníkov. Architektúra podporuje webové štandardy ako REST a SOAP.

- Microsoft vývojové prostredie na tvorbu reklamy pre Windows Phone - umožňuje jednoduché implementovanie banerov[23] a textových reklám do aplikácií a hier pomocou ktorých sa dajú získať financie, maximalizuje odmeny z reklamy vo vnútri aplikácií, poskytuje reporty a cielenú reklamu relevantným zákazníkom

1.4.4. PORTÁLOVÉ SLUŽBY

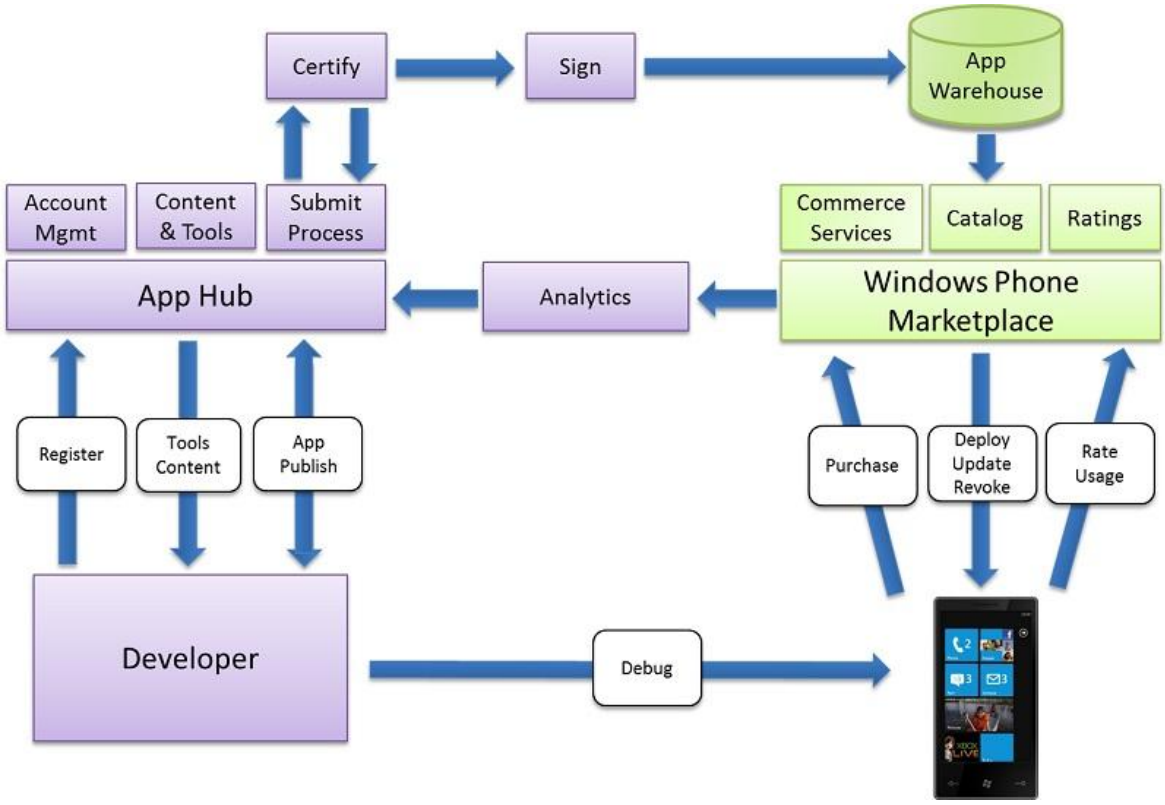
Windows Phone Marketplace je miesto kde vývojári môžu certifikovať svoje aplikácie, môžu určiť na akých trhoch a za akú cenu budú na týchto trhoch vydané. Vývojári majú k dispozícii aj nástroje Business Inteligence, ktoré poskytujú reporty o ich aplikáciách. Zákazníci tu môžu nakupovať a aktualizovať získané aplikácie. Platiť môžu či už prostredníctvom operátora alebo platobnej karty.

1.5. Životný cyklus vývoja aplikácie

App Hub – je štartovacím bodom pre vývojára Windows Phone aplikácií. Po zaregistrovaní sa a vytvorení si účtu na Windows Live ID si môže stiahnuť všetky potrebné nástroje na vývoj v jednom balíku. Môže si tu tiež zaregistrovať svoj telefón a potom na ňom testovať svoje aplikácie. Na App Hub sú tiež dostupné príklady , dokumentácia a vývojárska komunita, ktorá môže uľahčiť vývoj novej aplikácie. Keď má vývojár aplikáciu hotovú, môže ju otestovať a zbaviť chýb buď pomocou emulátora alebo vlastného zariadenia. Po úspešnom otestovaní môže aplikáciu odoslať na certifikáciu do Windows Phone Marketplace ako .xap súbor*(vygenerovaný Visual Studiom), ktorý obsahuje zdrojové kódy, ikonu aplikácie, štartovaciu dlaždicu, metadáta a licenčné podmienky vymedzujúce podmienky použitia programu. Ak sa

[23] **Graficky spracovaný reklamný prúžok**

aplikácia správa tak, že spĺňa kritériá Microsoftu, to znamená, že aplikácia je stabilná a spoľahlivá, efektívne využíva zdroje, aplikácia neinterferuje s funkcionalitou telefónu, neobsahuje zlomyseľný softvér, pracuje správne vo vybraných jazykových mutáciách a nemá negatívny dopad na celkový výkon zariadenia; je schválená a vývojár dostane notifikáciu o jej schválení[27]. Ak k schváleniu aplikácie nedôjde, vývojár dostane záznam chýb a aplikácia nie je publikovaná. Aplikácia musí prejsť certifikáciou aj v tom prípade ak sa jedná len o jej aktualizáciu. Nastavenie cenovej hladiny a politiku predaja si je možné nastaviť cez App Hub. Vývojový cyklus je schematicky zobrazený na obrázku č.5.



OBRÁZOK 5, ZDROJ: HTTP://CREATE.MSDN.COM/EN-US/EDUCATION

1.6. Popis hardvérového zariadenia a základné požiadavky na vzhľad aplikácie

Všetky zariadenia pri vydaní systému Windows Phone 7 Microsoftom označované ako „šasi“ majú definované hardvérové požiadavky ktoré musia spĺňať na to aby na nich mohol systém bežať. Požiadavky sú nasledovné:

- zariadenie musí podporovať multidotykovú obrazovku (schopnú detekovať 4 body dotyku naraz) pod ktorou sa musia nachádzať 3 tlačidlá, zobrazené na obr. 6.

OBRÁZOK 6. ZDROJ: HTTP://CREATE.MSDN.COM/EN-US/EDUCATION

Každé tlačidlo má svoj názov, sprava prvé má názov Späť, potom Štart a nakoniec Vyhľadávanie. Tlačidlo Späť slúži na ukončenie aplikácie(ak sme na úvodnej obrazovke aplikácie) alebo priamo na jej ovládanie (napríklad v prípade webovej aplikácie na návrat späť). Pri dlhšom podržaní tlačidla (viac ako 2s) sa nám zobrazí náhľad spustených aplikácií ktorý nám umožňuje medzi týmito aplikáciami prepínať. Tlačidlo Štart vráti používateľa na štartovaciu obrazovku telefónu, aplikáciám je však neprístupné. Pri dlhšom stlačení tlačidla sa spustí hlasové ovládanie telefónu. Tlačidlo Vyhľadávanie inicializuje vyhľadávaciu funkciu a taktiež nie je prístupné pre vývojárov tretích strán.

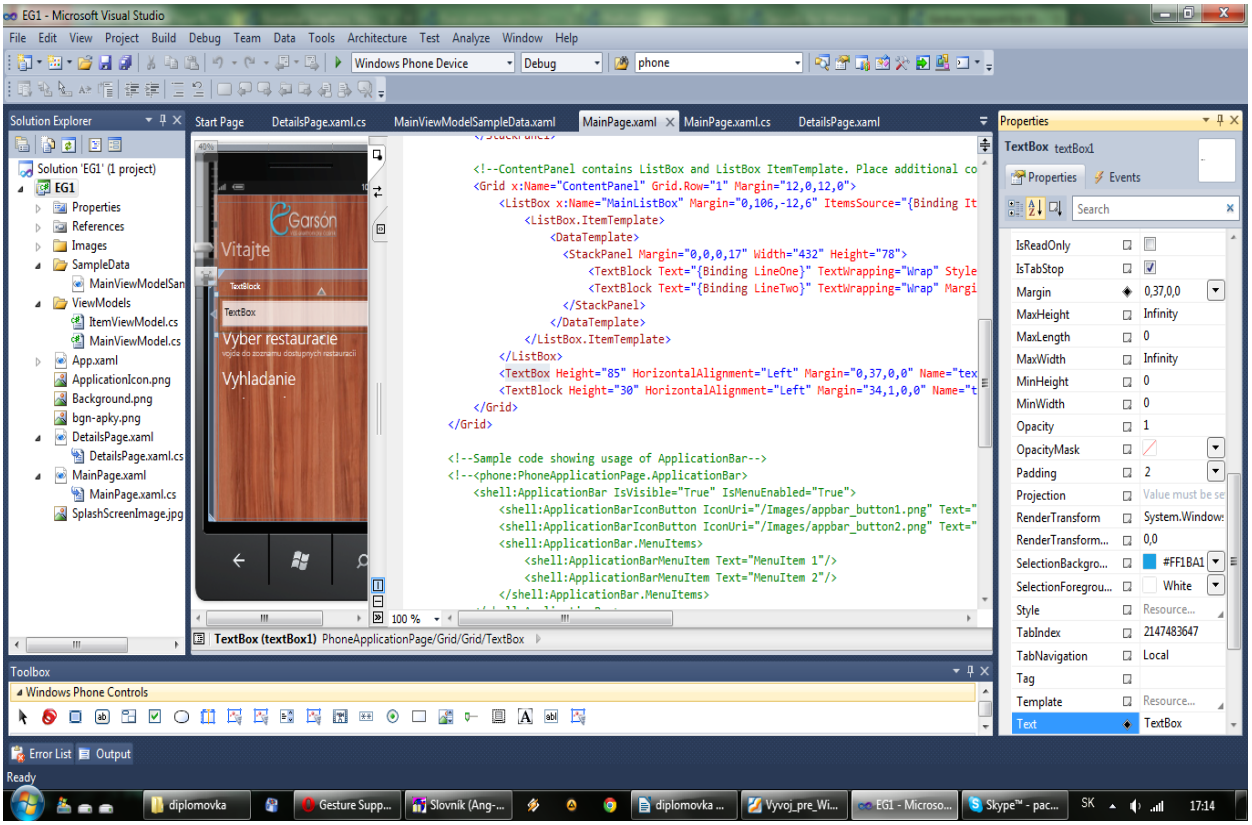
- Okrem predchádzajúcich tlačidiel telefón by mal byť vybavený ešte aspoň tromi vyhradenými tlačidlami – dvojpolohové tlačidlo spúšte fotoaparátu, vypínacie tlačidlo a dvoj tlačidlo hlasitosti.
- DirectX9 kompatibilnou grafickou výpočtovou jednotkou (GPU)
- minimálne 256MB operačnej pamäte a minimálne 4GB Flash pamäte

- Akcelerometer, snímač okolitého osvetlenia, snímač priblíženia, asistované GPS, FM rádio, WiFi a konektivitu do mobilnej dátovej siete

- Všetky zariadenia musia mať rovnaké rozlíšenie obrazovky ktoré je definované na 480 x 800 pixlov (podpora iných rozlíšení sa očakáva až s nasledujúcimi verziami systému). Silverlight aplikácie sú štandardne orientované zvislo a XNA aplikácie vodorovne. Orientáciu aplikácie je možné jednoducho zmeniť zdrojovom kóde aplikácie.

1.7. Popis vývojového prostredia

Vývojové prostredie Visual Studio sa skladá z viacerých okien. Typicky sa pri vývoji využívajú nasledujúce okná (zobrazené na obrázku 7):



1. Okno Solution Explorer zobrazuje adresárovú štruktúru celého projektu. Nachádzajú sa tu priečinky pomocou ktorých sa dá pristupovať k zdrojovým súborom jednotlivých stránok aplikácie a to či už stránok zdrojového kódu grafického rozhrania aplikácie (súbory s príponou .xaml) alebo stránok zdrojového kódu na pozadí (súbory s príponou .cs) . pomocou tohto okna je možné pristupovať aj k obrázkom aplikácie.
2. Okno dizajnu – náhľad na grafické usporiadanie elementov XAML kódu na obrazovke telefónu, okno sa stratí v prípade ak vývojár klikne na záložku zdrojového kódu pozadia
3. Okno zdrojového kódu XAML
4. Okno Properties – zobrazí vlastnosti jednotlivých elementov XAML kódu a udalosti na ne naviazané
5. Okno Toolbox – obsahuje ovládacie prvky respektíve elementy ktoré môžu byť pridané do aplikácie.

Jednotlivé okná si vývojár môže usporiadať podľa svojich potrieb prípadne niektoré z nich odobrať či pridať cez menu View. Vo vrchnom riadku vývojového prostredia si vyberá spôsob otestovania aplikácie. Je tu na výber otestovanie aplikácie pomocou vlastného pripojeného zariadenia alebo pomocou emulátora. Ak si vyberie druhú možnosť otvorí sa mu okno emulátora uvedeného na obr.8:

OBRÁZOK 8

Ako možno vidieť na obrázku vzhľad emulátora zodpovedá približne reálnemu vzhľadu zariadenia. Pomocou ikon napravo od emulátora sa dá upraviť veľkosť zobrazenia, orientácia (vodorovná alebo zvislá) prípadne pristupovať k nastaveniam polohy, akcelerometra a nástroju na vytváranie snímok obrazovky. Hlavný rozdiel respektíve hlavnou výhodou oproti použitiu reálneho zariadenia je možnosť vytváranie snímok obrazovky spustenej aplikácie. Medzi nevýhody emulátora patrí neschopnosť poskytnúť možnosť kvalitne otestovať akcelerometer, multidotykové ovládanie a vstavaný fotoaparát. Takisto neposkytuje plnú funkcionalitu pri spúšťaní externých akcií aplikácií ako napríklad písanie mailu, neposkytuje možnosť otestovať FM rádioprijímač. Pri testovaní na emulátore je otázny aj výkon aplikácie, tá sa môže správať na reálnom zariadení rozdielne ako na emulátore, niekedy na ňom beží rýchlejšie niekedy pomalšie. Preto je na odhad výkonu aplikácie použiť údaje zo zvislého pruhu údajov. Ten je viditeľný v pravom hornom rohu emulátora/telefónu a zobrazuje nám koľko výpočtových prostriedkov zariadenia vytvorená aplikácia spotrebúva, respektíve ako rýchlo funguje. Popis jednotlivých hodnôt môžeme vidieť na obr.9:

OBRÁZOK 9

Na to aby sme spustili aplikáciu na reálnom zariadení musí byť toto zariadenie pripojené k počítaču, zariadenie musí byť odblokované – to znamená že musí byť zaregistrované pomocou konta vývojára, musí byť spustená aplikácia ZUNE ktorá slúži na prepojenie a synchronizáciu dát zo zariadenia do počítača, pripojené zariadenie musí byť aktívne (musí mať rozsvietený displej) a musí mať odomknutú obrazovku (po zapnutí je potrebné zadať prístupový kód - ak ho zariadenie vyžaduje a vysunúť obrazovú roletu nahor)

2. Cieľ práce a metodika práce a metódy skúmania

Cieľom práce bolo preskúmať dostupné alternatívy vhodné na vývoj mobilnej aplikácie, objasniť výber popisovanej platformy na vývoj mobilnej aplikácie, príkladne zhodnotiť aké by mala aplikácia mať atribúty a ako sa má pri vývoji mobilnej aplikácie postupovať od jej návrhu až po umiestnenie do predaja respektíve zverejnenie aplikácie pre konkrétnu platformu. Cieľom tiež bolo opísať vývojové prostredie a postup pri vytváraní konkrétnej aplikácie a to konkrétne návrh, analýzu, implementáciu a testovanie aplikácie.
Pri skúmaní dostupných alternatív sme použili metódu komparácie, indukcie a dedukcie.

3. Výsledky práce

V tejto časti sa budeme zaoberať samotným výsledkom našej práce, čo je vlastne tvorba mobilnej aplikácie. Spomenieme základné dizajnové princípy a požiadavky na mobilnú aplikáciu. Ukážeme si ako sme pri tvorbe postupovali a aké prostriedky sme pritom využili. Samotná tvorba aplikácie pozostáva z niekoľkých častí a to analýza, návrh, testovanie a implementácia. Postupne si tu rozoberieme ako sme pri jednotlivých krokoch postupovali a čo sme vlastne vytvorili.

3.1. Dizajnové princípy a požiadavky na mobilnú aplikáciu

Pri návrhu grafického rozhrania sa treba podľa Raka[22] riadiť nasledujúcimi zásadami:

- Grafické rozhranie by malo byť čo najintuitívnejšie, užívateľ by mal ihneď vedieť ako aplikáciu používať
- Aplikácia by mala byť ľahko použiteľná v akomkoľvek prostredí a situácii
- Aplikácia by mala byť funkčná aj keď zariadenie nie je pripojené do siete internet (to však záleží na aj na type vyvíjanej aplikácie)
- Využitie internetového pripojenia by tiež nemalo kolidovať s fungovaním aplikácie

3.2. Popis aplikácie - analýza

V dnešnej uponáhľanej dobe si potrebujeme mnohokrát poznačiť úlohu ktorú chceme vykonať v blízkej budúcnosti vykonať a momentálne na jej riešenie nemáme čas. Potrebujeme si preto poznačiť čo by sme mali spraviť. Veľakrát však nemáme pri sebe pero a papier kde by sme si mohli niečo rýchlo poznačiť. Ak však vlastníme múdry mobilný telefón, pero a papier nám už nemusí chýbať. S vhodnou aplikáciou si vieme efektívne viesť zoznam úloh ktoré nás ešte čakajú spraviť, ktoré úlohy treba spraviť hneď respektíve ešte v daný deň a tým si lepšie naplánovať rozvrh dňa. Preto sme sa rozhodli spraviť jednoducho vyzerajúcu aplikáciu ktorá spĺňa vyššie uvedené vlastnosti a ešte k tomu pridáva niektoré funkcie navyše.

3.3. Funkcie a vlastnosti aplikácie - návrh

Pri vývoji aplikácii sme sa snažili splniť nasledujúce požiadavky na vlastnosti aplikácie: aplikácia na zaznamenávanie úloh by mala byť jednoduchá, prehľadná, ľahko ovládateľná, rýchla a užívateľsky prívetivá. Bolo by vhodné aby obsahovala nasledujúce funkcie:

- Jednoduché pridanie novej úlohy respektíve poznámky s možnosťou pridania detailného popisu úlohy a farebné označenie úlohy na sprehľadnenie zoznamu pri väčšom množstve úloh
- Zobrazenie zoznamu úloh chronologicky podľa času a dátumu dokedy ich treba spraviť
- Zobrazenie zoznamu úloh ktoré treba ešte vykonať v aktuálny deň
- Zobrazenie zoznamu nestihnutých úloh
- Možnosť jednoduchého označenia úlohy ako splnenej prípadne nesplnenej
- Zobrazenie zoznamu splnených úloh pre lepšiu predstavu užívateľa koľko úloh už má spravených
- Možnosť editácie a mazania úloh
- Možnosť pridávať alebo odstrániť zobrazené filtre na sprehľadnenie aplikáciepr

3.4. Vývoj aplikácie - implementácia

Ako príklad na vývoj aplikácie sme sa rozhodli vyvinúť aplikáciu ktorej úlohou bude ukladať používateľom zadané úlohy do prehľadných zoznamov. Na vývoj sme sa rozhodli použiť štýl aplikácie s takzvaným *pivot menu*, ktoré je určené pre aplikácie zo zoznamom položiek a užívateľ si môže pomocou tohto menu môže dané záznamy filtrovať.

3.4.1. HLAVNÉ ČASTI PROJEKTU WINDOWS PHONE APPLICATION

Aplikácia vytváraná vo Visual Studiu ako nový projekt obsahuje minimálne tieto časti:

- Adresár Properties s Manifestom aplikácie
- Adresár References s odkazmi na použité knižnice
- Adresár SampleData s ukážkou vzorových dát
- Adresár ViewModels s modelmi zobrazenia
- Obrázky

- XAML kód: MainPage.xaml a App.xaml
- C# kód: MainPage.xaml.cs, App.xaml.cs a AssemblyInfo.cs

Adresár Properties obsahuje 3 súbory:

- AppManifest.xml
- AssemblyInfo.cs
- WMAppManifest.xml.

Súbor s názvom WMAppMainfest.xml je manifest aplikácie. Opisuje aplikáciu operačnému systému – jej meno, ako vyzerá, ako sa spúšťa, čo môže a nemôže vykonávať atď. Príklad manifestu je uvedený v nasledujúcom úryvku kódu.

```
<?XML VERSION=”1.0 ” ENCODING =”UTF-8”?>
<DEPLOYMENT XMLNS="HTTP://SCHEMAS.MICROSOFT.COM/WINDOWSPHONE/2009/DEPLOYMENT"
APPPLATFORMVERSION ="7.0 ">
    <APP XMLNS="" PRODUCTID="{2F711986-CFB4-40D3-9B7D-64AA37FAF338} "
TITLE="TALLY"
RUNTIMETYPE ="SILVERLIGHT " VERSION="1.0.0.0" GENRE="APPS.NORMAL "
AUTHOR ="TALLY AUTHOR" DESCRIPTION ="SAMPLE DESCRIPTION " PUBLISHER="TALLY">
      <ICONPATH ISRELATIVE="TRUE" ISRESOURCE="FALSE">APPLICATIONICON.PNG</ICONPATH >
        <CAPABILITIES>
            <CAPABILITY NAME="ID_CAP_IDENTITY_DEVICE"/>
            <CAPABILITY NAME="ID_CAP_IDENTITY_USER"/>
            <CAPABILITY NAME="ID_CAP_LOCATION"/>
            <CAPABILITY NAME="ID_CAP_MICROPHONE"/>
            <CAPABILITY NAME="ID_CAP_NETWORKING"/>
        </CAPABILITIES>
        <TASKS>
            <DEFAULTTASK NAME ="_DEFAULT " NAVIGATIONPAGE="MAINPAGE.XAML "/>
        </TASKS>
        <TOKENS >
            <PRIMARYTOKEN TOKENID="TALLYTOKEN" TASKNAME ="_DEFAULT ">
                <TEMPLATETYPE5 >
                    <BACKGROUNDIMAGEURI ISRELATIVE="TRUE" ISRESOURCE="FALSE">
                        BACKGROUND.PNG
```

```
                </BackgroundImageURI >
                <Count>0</Count>
                <Title>Tally</Title></TemplateType5 ></PrimaryToken></Tokens >
</App ></Deployment>
```

Po nahraní aplikácie do Windows Marketplace sa väčšia časť manifestu prepíše počas certifikačného procesu.

Jednotlivé elementy manifestu majú rôzny význam. *App element* obsahuje *ProductID* Unikátny Globálny Identifikátor (GUID) ktorý slúži na identifikáciu aplikácie a hodnotu *RuntimeType* ktorá indikuje či je aplikácia typu XNA alebo Silverlight – v tomto prípade Silverlight. Hodnota *Title* označuje názov aplikácie, teda ako sa aplikácia bude nazývať v obchode. Ostatné atribúty sa využívajú na zaradenie aplikácie vo Windows Marketplace ale tak isto ako hodnota *Title* sú prepísané hodnotami ktoré sú vývojárom zadané na web stránke Marketplace. Element *IconPath* odkazuje na cestu k obrázku ikony, elemtnt *Tasks* odkazuje na hlavnú Silverlight stránku ktorá sa zobrazí ako prvá pri spustení aplikácie, element *Token* obsahuje informáciu o dlaždici (ktorú užívateľ vidí ak aplikáciu „pripne“ na štartovaciu obrazovku telefónu).

Manifest okrem iného obsahuje element *Capabilities*, v ktorom je uvedený zoznam schopností aplikácie. Sú to vlastne povolenia alebo aj obmedzenia, ktoré používateľ schváli pri inštalácii aplikácie, a tým definuje povolené správanie aplikácie. To znamená či môže aplikácia pristupovať k súkromným dátam, využívať dátové prenosy atď. Manifest vygenerovaný Visual Studiom obsahuje zoznam všetkých schopností, pri certifikačnom procese však automaticky (až na jednu výnimku *ID_CAP_NETWORKING*[24] – ak ju chce vývojár použiť, musí ju explicitne zadať do manifestu) detekuje, ktoré schopnosti aplikácia skutočne vyžaduje a prepíše zoznam schopností tak, aby obsahoval len tie najnutnejšie pre beh aplikácie. Každá zo schopností má potom v Marketplace užívateľsky prívetivé meno a tak *ID_CAP_LOCATION* sa užívateľovi zobrazí ako Lokalizačné služby. V Marketplace sa užívateľ rozhodne už pred stiahnutím aplikácie, či umožní aplikácii využiť všetky schopnosti. Môže buď schváliť všetky schopnosti alebo zamietne stiahnutie a inštaláciu

[24] túto vlastnosť je tiež potrebné zachovať ak chceme aplikáciu testovať priamo na zariadení

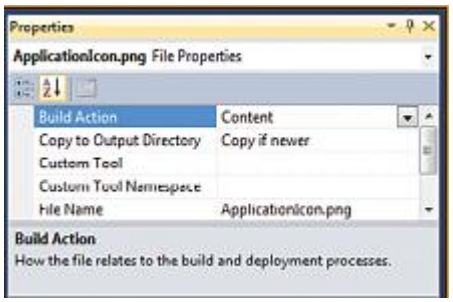
aplikácie. Ak sú už schopnosti schválené, aplikácia si už potom za behu nemusí pýtať od používateľa dodatočné povolenia.

V adresári *Properties* sa nachádza aj súbor *AssemblyInfo.cs* ktorý môžeme využiť na nastavenie čísla verzie aplikácie. Číslo verzie pozostáva zo 4 číslic oddelených bodkou kde prvé číslo zľava vyjadruje hlavné poradie verzie, druhé číslo vyjadruje minoritné poradie, potom číslo zostavenia a nakoniec číslo revízie. Posledné dva čísla môžu byť pomocou hviezdičky nastavené automaticky. Ich inkrementácia je potom vykonaná pri každom zostavení aplikácie.

Novovytvorený projekt obsahuje ešte jeden manifest v adresári *Properties*. Jeho názov je *AppMaingest.xml* a je potrebný len pre infraštruktúru Silverlightu a nie je potrebné ho upravovať.Adresár *References* obsahuje odkazy na knižnice ktoré obsahujú triedy potrebné na vývoj mobilnej aplikácie.
Projekt vygenerovaný Visual Studiom obsahuje adresár s troma obrázkami:

- ApplicationIcon.png – je to hlavná ikona aplikácie keď je už aplikácia nainštalovaná. Jej rozmery by mali byť 62x62 pixlov. Ak bude aplikácia hra (bude umiestnená v Game hub) jej rozmery by mali byť 173x173 pixlov.
- Backgroun.png – dlaždicová ikona s rozmermi 173x173 pixlov, je použitá vtedy ak si používateľ prepne aplikáciu na štartovaciu plochu. Do tejto ikony ja automaticky pridaný text*[text pochádza z manifestu aplikácie z elementu App a vlastnosti Title] s názvom aplikácie do ľavého dolného rohu.
- SplashscreenImage.jpg – obrázok štartovacej obrazovky s rozlíšením 480x800 pixlov ktorý sa zobrazí keď sa aplikácia spúšťa, respektíve načítava do pamäti.

Názov a umiestnenie prvých dvoch obrázkov si môže vývojár prispôsobiť, mali by byť typu JPEG alebo PNG a treba potom patrične upraviť aj manifest. Akýkoľvek z týchto troch obrázkov môže byť vymenený za vlastný, len v tom prípade treba pri každom obrázku cez menu properties zmeniť Build Action z pôvodného Resource na Content ako je ukázané na obr. 10.



Obrázok 10

Tým sa správne umiestni zdrojový súbor do .xap súboru namiesto do DLL[25] knižnice kde by zbytočne zdržoval načítanie aplikácie. Obrázok je vhodné použiť pri náročnejších aplikáciách, kde používateľa uistí že sa aplikácia už načítava. My sme tento obrázok v našej aplikácii nepoužili keďže sa aplikácia spustí takmer okamžite.

Každá aplikácia sa skladá z minimálne jednej alebo viacerých stránok. Pri vytváraní projektu sa vytvorí jeden súbor s názvom *MainPage.xaml a App.xaml*. Súbor MainPage.xaml definuje ako bude vyzerať základné rozloženie hlavnej stránky aplikácie a s úbor App.xaml definuje použité štýly zobrazenia, okrajov a fontov textov a ostatných elementov aplikácie.

Súbor *MainPage.xaml.cs* predstavuje exekučnú časť aplikácie. Sú tu definované triedy, a funkcie adátové položky potrebné na fungovanie aplikácie. V súbore *App.xaml.cs* je definované spôsob práce s aplikáciou, jej spúštanie,vykresľovanie, prerušenie, odchytávanie chýb atď.

3.5. Hlavné časti vyvíjanej aplikácie a postup pri ich vytváraní

Naša aplikácia UrobTO obsahuje všetky hore uvedené časti z ktorých sa minimálne musí skladať, pričom okrem nich sme si vytvorili ďalšie súbory a stránky ktoré boli na beh aplikácie potrebné. Naša aplikácia obsahuje šesť stránok na ktoré môže používateľ pristúpiť. Ukážky týchto stránok sme zahrnuli do prílohy č.1. Stránky obsiahnuté v našom projekte sú nasledovné: Hlavná stránka (MainPage.xaml), stránka pridať/upraviť úlohu (AddEditPage.xaml), stránka detailov (DetailsPage.xaml), stránka inštrukcií

[25] **Dynamicky nalinkovaná knižnica**

(InstuctionsPage.xaml), stránka nastavení (Settings.xaml) a stránka o aplikácii (AboutPage.xaml).

3.5.1. HLAVNÁ STRÁNKA

Stránka MainPage (hlavná stránka) definuje čo užívateľ uvidí po spustení aplikácie. Je implementovaná pomocou dvoch súborov MainPage.xaml ktorá obsahuje používateľské rozhranie a MainPage.xaml.cs ktorá obsahuje aplikačnú logiku(tzv. kód na pozadí*[26]). MainPage.xaml je referencovaná v manifeste WMAppMain.xml takže ak chceme zmeniť názov tejto stránky musíme ho zmeniť aj v manifeste.

Vo Visual Studiu 2010 sme si vytvorili nový projekt, z inštalovaných šablón sme si vybrali z kategórie *Visual C#* podkategóriu *Silverlight for Windows Phone* a šablónu *Windows Phone Pivot Application*. Po zadaní názvu aplikácie a potvrdení sme si zvolili pre akú platformu bude naša aplikácia určená. V našom prípade to je platforma Windows Phone OS 7.1 (staršie verzie už nie sú podporované v Marketplace). Na správne fungovanie všetkých súčastí aplikácie pridáme do referencií odkaz na *Microsoft.Phone.Controls.Toolkit*[27] a *AdamNathanAppsLibrary*[28] ktoré obsahujú triedy ktoré využijeme pri animácii obrazovky, zmeny zobrazenia pri naklonení telefónu, pri využití *Picker boxu*[28] a animovaných voličov dátumu a času.

Po tomto kroku sme pristúpili k návrhu grafického rozhrania. Keďže vo Visual Studiu je aplikácia typu *Pivot* uložená ako šablóna, ušetrilo nám to na začiatku trochu práce.

Výzor aplikácie respektíve jej jednotlivých stránok je možné upravovať pomocou

[26] Backgroud code
[27] Dostupnýna adrese http://silverlight.codeplex.com
[28] Zaškrtávacie políčko použité na výber položiek

grafického dizajnéra alebo pomocou textového editora s podporou Intelisense[29] zabudovaného priamo vo Visual Studiu. My sme použili obidve možnosti. Grafického dizajnéra na hrubý návrh a textový editor na doladenie vzhľadu.

Ako prvý sme upravili jednotlivé položky pivot menu ktoré predstavujú jednotlivé filtre zoznamov. Jednotlivé položky sú ohraničené tagmi[30] *Controls* – to znamená že sme vytvárali prvok pomocou ktorého sa aplikácia ovláda. Na to aby sme mohli umiestniť elementy textu, pozadia a položiek zoznamu bolo potrebné vytvoriť rodičovský element *Grid*[31] . Každú položku pivot menu sme premenovali, zmenili sme obrázok jej pozadia. Jedna položka je definovaná nasledujúcim XAML kódom:

```
<controls:PivotItem Header="všetky">
  <Grid>
```

Tu sa nachádza popis elementu textového bloku ktorý zobrazí text v prípade ak je zoznam prvého filtra pivot menu prázdny. Jeho atribúty definujú jeho názov, viditeľnosť, okraje (ľavý, horný, pravý, spodný) a štýl (v tomto prípade je to odkaz na štýl ktorým sa budeme zaoberať neskôr)

```
        <TextBlock x:Name="NoAllTextBlock" Text="Žiadne úlohy"
Visibility="Collapsed"
          Margin="22,17,0,0" Style="{StaticResource PhoneTextGroupHeaderStyle}"/>
```

Na pridanie obrázku do pozadia zoznamu bolo potrebné vytvoriť elemet grafického objektu, v našom prípade štvorca. Jeho vlastnosti ako nepriehľadnosť, okraje, vertikálne a horizontálne umiestnenie, veľkosť a farbu vyplnenia sme nastavili pomocou príslušných atribútov *Opacity, Margin, VerticalAligment, HorizontalAligment, Width, Height a Fill*. Bolo tiež potrebné si vytvoriť v okne Sollution Explorer nový priečinok s názvom *Images* kde sme potom pridali obrázky ktoré boli v aplikácii použité. Obrázky sme pridali pomocou ľavého kliku myši na novovytvorený priečinok a cez menu add existing item. Cestu k obrázku sme potom zadali do vlastnosti *ImageSource* elementu *Image Brush*.

---

[29] Funkcia Visual Studia ktorá pri písaní zdrojového kódu ponúka alternatívy na dokončenie písaného výrazu
[30] Značkami
[31] Súradnicová sieť

```
  <Rectangle Opacity=".2" Margin="0,0,0,12" VerticalAlignment="Center"
HorizontalAlignment="Center" Width="240" Height="240"
Fill="{StaticResource PhoneForegroundBrush}">
<Rectangle.OpacityMask>
<ImageBrush ImageSource="Images/home.png"/>
</Rectangle.OpacityMask>
                </Rectangle>
```

V rámci elementu *grid* sa nachádza aj element *Listbox* ktorý nám definuje vlastný zoznam úloh, ktorý si bude používateľ sám vytvárať. Má priradený názov *AllListBox* a jeho vlastnosti, respektíve vzhľad je vyjadrený pomocou vlastnosti *ItemTemplate* (ktorá konkrétne odkazuje na *DataTemplate*). Jednotlivé položky zoznamu sú vyjadrené pomocou vlastnosti *ItemSource*. Táto vlastnosť je typu *IEnumerable*[32] a umožňuje priradiť celú kolekciu dát **v** jednom kroku namiesto toho aby sme kolekciu dát priraďovali jednu po druhej. To nám ušetrilo riadky kódu a hlavne nám umožnilo nastaviť hodnotu *ItemSource* pomocou dátovej väzby. Keď použijeme väzobnú syntax[33] bez názvu vlastností znamená že sa má použiť celý objekt nie len jedna hodnota z jeho vlastností. To nám umožní výkonovú optimalizáciu v prípade ak bude množstvo dát (riadkov) v zozname veľké, čo sa pri našej aplikácii predpokladá. Pri pohybe medzi vytvorenými položkami zoznamu nie je žiaduce aby sa nám zmenilo označenie položky pri každom ťuknutí do obrazovky, preto vlastnosť *SelectionChanged* s priradenou hodnotou *ListBox_SelectionChanged* vyčistí práve označenú položku, takže viacero následných dotykov na jednu položku funguje tak ako má. To znamená, že pri dotyku a potiahnutí nám položky *listboxu*[34] sa posúvajú v smere pohybu posúvania, alebo pri dotyku na jednu položku sa táto aktivuje a užívateľ je premiestnený na stránku editácie.

```
<ListBox x:Name="AllListBox" ItemsSource="{Binding}"
                ItemTemplate="{StaticResource DataTemplate}"
                SelectionChanged="ListBox_SelectionChanged"/>
```

[32] **Rozhranie umožňujúce jednoduchú iteráciu ponad celú ne-generickú kolekciu dát**
[33] **Binding syntax**
[34] **Grafický element, ktorý môže obsahovať viac položiek medzi ktorými môže používateľ listovať**

Na prácu s jednotlivými položkami zoznamu sme sa rozhodli použiť kontextové menu. To znamená, že používateľ môže ťuknúť na položku menu a držať na nej prst pokým sa mu nezobrazí menu. Menu sme pridali pomocou elementu *toolkit*[35] . Prvý detský element *toolkit:ContextMenu* vyjadruje, či je menu otvorené alebo zatvorené. Ďalšie elementy predstavujú jednotlivé položky, ktoré si môže používateľ vybrať a tak pracovať s vybraným záznamom. Na výber má možnosť označiť úlohu ako spravenú, upraviť úlohu alebo ju vymazať

```
<toolkit:ContextMenuService.ContextMenu>
<toolkit:ContextMenu Opened="ContextMenu_Opened"
                                    Closed="ContextMenu_Closed">
<toolkit:MenuItem Header="označ ako spravené"
                                   Click="MarkMenuItem_Click"/>
<toolkit:MenuItem Header="upraviť" Click="EditMenuItem_Click"/>
<toolkit:MenuItem Header="vymazať"
                                   Click="DeleteMenuItem_Click"/>
                </toolkit:ContextMenu>
              </toolkit:ContextMenuService.ContextMenu>
```

Toto kontextové menu sme použili pre všetky listboxy okrem listboxu *spravené*. Ten sa odlišuje tým že je v ňom zoznam už spravených úloh, takže logicky jedna položka kontextového menu môže spravené úlohu v prípade požiadavky používateľa presunúť do zoznamu nespravených úloh. Pre tento listbox sme ale museli vytvoriť nový detský element ktorý sa už nenachádzal v rodičovskom elemente *phone:PhoneApplicationPage.Resources* ale zaradili sme ho ako detský element do elementu menu listboxu *spravené*. Nasledujúci úryvok kódu nám ukazuje kontextové menu vytvorené pre prvé štyri položky listbox menu.

```
<toolkit:ContextMenuService.ContextMenu>
 <toolkit:ContextMenu Opened="ContextMenu_Opened"
                                    Closed="ContextMenu_Closed">
```

[35] **Obsiahnutýpomocou referencie vo Windows. Phone. controls**

```
<toolkit:MenuItem Header="označ ako nespravené"
                                  Click="UnmarkMenuItem_Click"/>
<toolkit:MenuItem Header="upraviť" Click="EditMenuItem_Click"/>
<toolkit:MenuItem Header="vymazať" Click="DeleteMenuItem_Click"/>
</toolkit:ContextMenu></toolkit:ContextMenuService.ContextMenu>
```

Tento zoznam je špecifický aj v tom že každá položka zoznamu je prečiarknutá, čo má zvýrazniť, že ju už používateľ označil ako spravenú. Na to nám poslúžil element *line* ktorý predstavuje čiaru prečiarknutia, kde sme si zadefinovali jej pozíciu, dĺžku, okraje a tiež jej hrúbku. Element *line* tiež čerpá svoj štýl (farbu) zo statického zdroja *PhoneForegroundBrush*.

```
<Line X1="-2" X2="800" Y1="32" Y2="32" StrokeThickness="2"
                    Stroke="{StaticResource PhoneForegroundBrush}"/>
```

Na to aby aplikácia spadala do celkovej koncepcie telefónu, respektíve jeho vzhľadu sme prispôsobili titulku aplikácie pomocou elementu *DataTemplate*. Zadefinovali sme jeho okraje a takisto aj štýl.

```
    <DataTemplate>
      <TextBlock Text="{Binding}" Margin="-1,-1,0,-3"
                 Style="{StaticResource PhoneTextTitle0Style}"/>
    </DataTemplate>
  </controls:Pivot.TitleTemplate>
```

Každý listbox našej aplikácie má vo svojej spodnej časti na lište tri centrálne tlačidlá a jedno okrajové, pomocou ktorých užívateľ pristupuje k rozšíreným voľbám aplikácie. Sú to tlačidlá ktorá sú zastúpené piktogramami plus, otáznik, ozubené koliesko a trojbodka. Pomocou tlačidla „plus“ sa prepne na obrazovku kde môže pridať novú úlohu, pomocou tlačidla so symbolom otáznika sa prepne na obrazovku inštrukcií, pomocou tlačidla so symbolom ozubeného kolieska sa prepne na obrazovku nastavení aplikácii a pomocou symbolu trojbodky sa spodná lišta posunie vyššie a odhalí tak popis tlačidiel (v našom prípade sa objavia nápisy „*nová úloha*“, „*inštrukcie*“ a „*nastavenia*“ a tiež sa zobrazí skrytá položka „*o aplikácii*“ pomocou ktorej sa dostaneme na stránku o informácií o aplikácii. Prístup k ďalším položkám menu pomocou trojbodky slúži najmä

na sprehľadňuje aplikácie, keďže zobrazí potrebné položky prístupu a názvy ikon len keď to je potrebné.

Aplikačnú lištu je možné pridať do ktorejkoľvek stránky aplikácie nastavením vlastnosti *ApplicationBar* inštancii *ApplicationBar* objektu. Lišta môže obsahovať maximálne štyri detské elementy tlačídiel (my sme využili tri) separátnu kolekciu položiek menu ktorých počet nie je obmedzený (my sme využili jednu položku menu). Pri vytváraní aplikačnej lišty si môžeme zvoliť aj jej nepriehľadnosť pomocou elementu *Shell:PhoneApplicationBar* a jeho atribútu *opacity*. My sme tomuto atribútu priradili hodnotu 0.9 aby sme dosiahli minimálnu priehľadnosť aplikačnej lišty.

```
<phone:PhoneApplicationPage.ApplicationBar>
  <shell:ApplicationBar Opacity="0.9">
```

Ku každej ikone sme pridali popis pmocou elementu *shell:ApplicationBarIconButton* a jeho atribútu *Text* a takisto sme priradili každému tlačidlu obrázok pomocou atribútu *IconUri.* Voľbu akcie akú má tlačidlo po stlačení vykonať sme nastavili pomocou atribútu Click.

```
<shell:ApplicationBarIconButton Text="NOVÁ ÚLOHA"
  IconUri="/Shared/Images/appbar.add.png" Click="AddButton_Click"/>
<shell:ApplicationBarIconButton Text="INŠTRUKCIE"
  IconUri="/Shared/Images/appbar.instructions.png"
  Click="InstructionsButton_Click"/>
<shell:ApplicationBarIconButton Text="NASTAVENIA"
  IconUri="/Shared/Images/appbar.settings.png"
  Click="SettingsButton_Click"/>
```

Podobne sme pridali aj položku menu „o aplikácii“ len s tým rozdielom že išlo o element *shell:ApplicationBar.MenuItems* a ten neobsahuje cestu k obrázku, keďže to je čistý text a neobsahuje obrázok.

```
<shell:ApplicationBar.MenuItems>
```

```
      <shell:ApplicationBarMenuItem Text="O aplikácii"
Click="AboutMenuItem_Click"/>
      </shell:ApplicationBar.MenuItems>
    </shell:ApplicationBar>
  </phone:PhoneApplicationPage.ApplicationBar>
```

Pri vytváraní aplikácie sme dbali na to aby všetky obrazovky aplikácie mali jednotný grafický dizajn, respektíve fonty, veľkosti písma okraje a iné. To sme dosiahli použitím návrhových vzorov štýlov ktoré sme potom priradili jednotlivým rodičovským elementom pomocou atribútu *Style* tam kde to bolo potrebné. Na vytvorenie takéhoto vzoru, respektíve konkrétneho štýlu stačí vytvoriť rodičovský element *Style* ktorému môžeme priradiť názov pomocou atribútu *x:Key* a typ aký bude zastupovať pomocou atribútu *Targettype*. Jeho detské elementy sa budú označené ako *Setter* – pomocou nich vlastne definujeme ako má daný štýl vyzerať. To sa dá docieliť pomocou definovania atribútov *Property* a *Value*. *Property* predstavuje akú vlastnosť štýlu chceme nastaviť napr. *Font* a *Value* predstavuje konkrétnu hodnotu ktorú jej priradíme prípadne aj zdroj kde je daná hodnota definovaná. Môžeme si to uviesť na príklade kde sme definovali ako bude vyzerať názov aplikácie ktorý je vopred definovaný štýlom *PhoneTextTitle0Style*. Definíciu tohto štýlu môžeme vidieť v nasledujúcom úryvku kódu:

```
<Style x:Key =”PhoneTextTitle0Style” TargetType=”TextBlock”>
<Setter Property =”FontFamily”
Value=”{ StaticResource PhoneFontFamilySemiBold }”/>
<Setter Property =”Foreground”
Value=”{ StaticResource PhoneForegroundBrush}”/>
<Setter Property =”FontSize” Value=”{ StaticResource PhoneFontSizeMedium}”/>
<Setter Property =”Margin” Value=”-1,0,0,0”/>
</Style>
```

Každá stránka použitá v aplikácii spravidla má svoj *.cs súbor so zdrojovým kódom ktorý určuje ako sa budú jednotlivé elementy xaml súboru správať a aký majú medzi sebou vzťah. Cs súbor má taký istý názov ako xaml súbor plus navyše obsahuje koncovku cs. Objekty a ich činnosti vytvorené na hlavnej stránke našej aplikácie *MainPage.xaml* sú definované funkciami a triedami pomocou takzvaného kódu na pozadí v súbore

MainPage.xaml.cs. Najprv sme si tu definovali menné priestory[36] ktoré boli v danom súbore použité a na ktoré sa odkazujeme v referenciách.

```
USING SYSTEM;
USING SYSTEM.WINDOWS;
USING SYSTEM.WINDOWS.CONTROLS;
USING SYSTEM.WINDOWS.NAVIGATION;
USING MICROSOFT.PHONE.CONTROLS;
```

Hlavnou dôvod existencie cs súboru však je že pomocou neho vytvoríme vlastný menný priestor na entity ktoré sú vytvorené v xaml súbore. Menný priestor sme si pomenovali *WindowsPhoneApp*. Ďalej sme si vytvorili verejne prístupnú triedu *MainPage* ktorá dedí z triedy *PhoneApplicationPage*. V tejto triede sme si vytvorili dve premenné typu *bool*. Pomocou prvej premennej *isNavigatingAway* budeme sledovať či používateľ vychádza z aplikácie a pomocou druhej is*ContextMenuOpen* budeme sledovať či je otvorené kontextové menu.

Na incializovanie komponentov a načítanie hlavnej stránky sme použili konštruktor Main page.

```
PUBLIC MAINPAGE()
    {  INITIALIZECOMPONENT();
      THIS.LOADED += MAINPAGE_LOADED;    }
```

Pre spúštanie hlavnej obrazovky pri návrate z menu nastavení sme si vytvorili funkciu OnNavigateTo ktorá nám priradí premennej isNavigatingAway hodnotu nepravda to znamená že z aplikácie neodchádzame, premení nám aktuálny dátum a čas na typ string a priradí objektu text, skontroluje či sú v menu nastavení povolené všetky karty pivot menu a spočíta tie ktoré sú nastavené ako viditeľné. Ak sa počet viditeľných položiek zmení tak bude ich zoznam obnovený. Keď bude používateľ opúšťať aplikáciu funkcia uloží premennej *isNavigatingAway* hodnotu pravdy a uloží nastavenie zobrazenia pivot menu, respektíve tú obrazovku, kde bol používateľ pred opustením aplikácie naposledy.

```
PROTECTED OVERRIDE VOID ONNAVIGATEDFROM(NAVIGATIONEVENTARGS E)
```

[36] **Región v ktorom sú umiestnené deklarácie entít, ktoré v našom programe používame**

```
{ this.isNavigatingAway = true;
  base.OnNavigatedFrom(e);
        Settings.SelectedPivotItemName.Value =
    (this.Pivot.SelectedItem as PivotItem).Name;
  this.Pivot.SelectedIndex = 0; }
```

Na základe nastavení potom funkcia ShowOrHidePivotItem zobrazí alebo skryje položku pivot menu.

```
void ShowOrHidePivotItem(PivotItem item, bool show, ref int insertLocation)
    {           if (show && item.Parent == null)
      this.Pivot.Items.Insert(insertLocation, item);
    else if (!show && item.Parent != null)
      this.Pivot.Items.Remove(item);
    if (show)
      insertLocation++;    }
```

Funkcia *MainPage_Loaded* má na starosti ošetriť chybu systému pri odstraňovaní položiek pivot menu, keďže systém Windows Phone sa ešte nevie veľmi dobre vyrovnať s odstránením jednej alebo viacerých položiek a potom rýchlym návratom na hlavnú obrazovku aplikácie.

Na usporiadanie jednotlivých úloh do zoznamov podľa toho či už nie sú po dátume dokedy mali byť splnené sme si vytvorili funkciu *RefreshLists*. Tá si načíta aktuálny dátum, potom vyčistí zoznamy „dnes*“, „po termíne“* a *„kategorizované*“ a potom ich znovu naplní úlohami podľa toho ako vyšli výsledky porovnávania sa aktuálnym dátumom.

```
void RefreshLists()
{DateTime today = DateTime.Now.Date;
  this.TodayListBox.Items.Clear();
  this.PastDueListBox.Items.Clear();
  this.StarredListBox.Items.Clear();
  foreach (Task item in Settings.TaskList.Value)
  {// dnes
    if (item.DueDate.Date == today)
```

```
            this.TodayListBox.Items.Add(item);
        // po termíne
          if (item.DueDate < DateTime.Now)
            this.PastDueListBox.Items.Add(item);
         // Kategorizované
          if (item.Star != null && item.Star != "ziadne")
            this.StarredListBox.Items.Add(item);      }

        // Zobraz/skryje nápisy žiadne úlohy
        this.NoAllTextBlock.Visibility = Settings.TaskList.Value.Count == 0 ?
          Visibility.Visible : Visibility.Collapsed;
        this.NoTodayTextBlock.Visibility = this.TodayListBox.Items.Count == 0 ?
          Visibility.Visible : Visibility.Collapsed;
        this.NoPastDueTextBlock.Visibility = this.PastDueListBox.Items.Count == 0 ?
          Visibility.Visible : Visibility.Collapsed;
        this.NoStarredTextBlock.Visibility = this.StarredListBox.Items.Count == 0 ?
          Visibility.Visible : Visibility.Collapsed;
        this.NoDoneTextBlock.Visibility = Settings.DoneList.Value.Count == 0 ?
          Visibility.Visible : Visibility.Collapsed;    }
```

Ďalej sme si vytvorili funkciu ktorá používateľa po ťuknutí na úlohu presunie na stránku podrobností o úlohe, respektíve zruší kontextové menu, ak používateľ klikne na ďalšiu položku. Pre zistenie, či je používateľ otvoril kontextové menu, sme si vytvorili pomocné funkcie *ContextMenu_Opened* a *ContextMenu_Closed*. Keďže používateľ potrebuje jednotlivé úlohy presúvať do zoznamu splnených úloh a naopak, vytvorili sme funkcie *MarkMenuItem_Click* a *UnmarkMenuItem_Click*. Po každom presunutí úlohy sa spustí funkcia *RefreshLists* ktorá zabezpečí znovuusporiadanie nových zoznamov.

```
void MarkMenuItem_Click(object sender, RoutedEventArgs e)
    { Task task = (sender as MenuItem).DataContext as Task;
      Settings.TaskList.Value.Remove(task);
      Settings.DoneList.Value.Add(task);
 RefreshLists();    }
```

Pre ďalšie položky kontextového menu, ktorými používateľ upravuje jednotlivé úlohy

sme vytvorili funkciu na editovanie *EditMenuItem_Click*. Táto funkcia presunie používateľa na obrazovku editácie

```
void EditMenuItem_Click(object sender, RoutedEventArgs e)
    { Settings.CurrentTask.Value = (sender as MenuItem).DataContext as Task;
      this.NavigationService.Navigate(new Uri("/AddEditPage.xaml",
        UriKind.Relative));
```

Poslednú položku kontextového menu „*vymazať*“ obsluhuje funkcia *DeleteMenuItem_Click*, ktorá vymaže požadovanú položku z menu úloh a pre istotu aj z menu spravených úloh. Pre zachovanie bezpečnosti sa aplikácia používateľa opýta či si je istý že chce vymazať vybranú úlohu.

```
void DeleteMenuItem_Click(object sender, RoutedEventArgs e)
    {if (MessageBox.Show(
        "Ste si istý že chcete natrvalo odstrániť túto úlohu?", "Vymazať úlohu",
        MessageBoxButton.OKCancel) == MessageBoxResult.OK)
      {Settings.TaskList.Value.Remove((sender as MenuItem).DataContext as Task);
        Settings.DoneList.Value.Remove((sender as MenuItem).DataContext as Task);
        RefreshLists();}}
```

Vyššie uvedené funkcie sú spúšťané až po nastaní určitej udalosti , v našom prípade je to ťuknutie prstom na konkrétnu položku kontextového menu. Posledná skupina funkcií ktoré sme vytvorili na obsluhu Hlavnej stránky sú funkcie, ktoré obsluhujú kliknutie na tlačidlá spodnej lišty. Ich funkcia je jednoduchá a to preniesť používateľa na konkrétnu obrazovku, ktorú daná ikona reprezentuje. Slúžia vlastne ako odkazy na nasledujúce obrazovky respektíve podstránky aplikácie: „*pridať/upraviť*“, „*inštrukcie*“, „*nastavenia*“, a „*o aplikácii*“. Keďže zdrojový kód všetkých odkazov je veľmi podobný ukážeme si len jeden z nich:

```
void AddButton_Click(object sender, EventArgs e)
{ Settings.CurrentTask.Value = null;
      this.NavigationService.Navigate(new Uri("/AddEditPage.xaml",
UriKind.Relative));    }
```

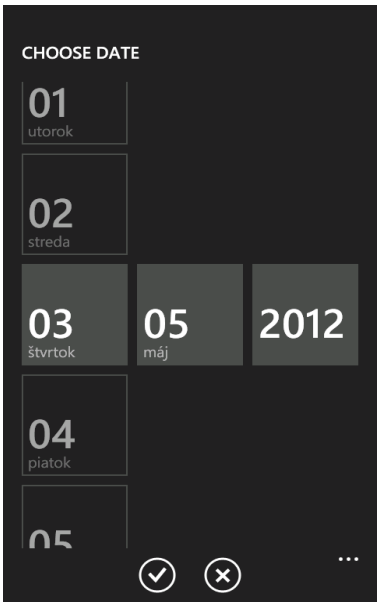
3.5.2. STRÁNKA PRIDAŤ/UPRAVIŤ ÚLOHU

Obrazovka pridať/upraviť úlohu sa používateľovi zobrazí buď po ťuknutí na tlačidlo *plus* v spodnej lište alebo po aktivovaní položky kontextového menu editovať. Stránku sme si pridali kliknutím pravého tlačidla myši v Solution Exploreri na položku projektu *UrobTO* a vybrali sme voľbu *add*, potom *add new item,* kde sme z okna vybrali voľbu *Windows Phone Portrait Page*. Do poľa meno sme dali názov *AddEditPage* a stlačili tlačidlo *Add*. Túto obrazovky sme navrhli prevažne pomocou grafického rozhrania, kde sme postupne z lišty nástrojov pridávali jednotlivé elementy. Najprv sme umiestnili textové polia do ktorých používateľ zadáva údaje, potom sme ich pomocou karty *Properties* pomenovali. Po vytvorení prvého *textboxu* [37] nám Visual Studio uľahčovalo prácu pri pridávaní ďalších elementov, zobrazením pomocných čiar, respektíve hraníc, ktoré vymedzovali priestor na ich umiestnenie. Potom sme im popridávali nadpisy aby mal používateľ prehľad o tom čo ktoré pole znamená. Umiestnili sme ich tak aby boli prehľadné, zrozumiteľné a aby bola práca s nimi čo najjednoduchšia. Prvý textbox má titulku „*Názov úlohy*“ a pole do ktorého sa môže vložiť názov konkrétnej úlohy. Pod neho sme vložili ďalší textbox s titulkou „*Popis*“ a väčším textovým poľom aby mal používateľ možnosť podrobnejšie popísať úlohu. Pod elementom „*Popis*“ sme vložili roletové menu s titulkom „*Vybrať farebné označenie*“. Tu sme už pomocou módu textového editora vložili pomocou elementu *ImageBrush* a jeho vlastnosti *ImageSource* cestu k obrázku, ktorý sa bude zobrazovať v prípade farebného označenia úlohy. Takisto sme pridali element *toolkit:ListPicker.ItemTemplate*. Tu sme nastavili zoznam farieb ktoré si používateľ môže pre sprehľadnenie úloh vybrať.

Na výber času a dátumu sme použili z toolkitu nástroj *DatePicker aTimePicker*. Tieto nástroje neumožňujú zadať hodnoty pomocou klávesnice ale pomocou rýchleho posunu hodnôt kváziroletovým respktíve valcovým[38] menu. Ukážku tohto menu môžeme vidieť na obrázku č.

[37] Grafický element určený na zobrazenie a vkladanie textu

[38] Hodnoty na výber sa javia ako keby boli umiestnené na valcoch, používateľ si vyberie želanú hodnotu otáčaním valca.



OBRÁZOK 11

Na spodnú časť stránky sme pridali lištu s tlačidlom „*uložiť*“.

V kóde na pozadí stránky sme si vytvorili triedu *AddItemPage* ktorá takisto ako hlavná trieda *MainPage* dedí z triedy *PhoneApplicationPage*. Zadefinovali sme si premenné *saveButtonna* na manipuláciu s tlačidlom uloženia a premenné *pendingChosenDate a pendingChosenTime* na manipuláciu s vybratým časom a dátumom. Potom sme si vytvorili konštruktor ktorý nám inicializuje tlačidlo uložiť.

```
PUBLIC ADDITEMPAGE()
    {INITIALIZECOMPONENT();
      THIS.SAVEBUTTON = THIS.APPLICATIONBAR.BUTTONS[0]
        AS IAPPLICATIONBARICONBUTTON; }
```

Napokon sme si vytvorili funkciu *OnNavigatedFrom* ktorá nám zabezpečí aby sa nám zachovali údaje vložené používateľom pri opustení stránky, respektíve aplikácie.

```
PROTECTED OVERRIDE VOID ONNAVIGATEDFROM(NAVIGATIONEVENTARGS E)
    {BASE.ONNAVIGATEDFROM(E);
       THIS.STATE["TITLE"] = THIS.TITLETEXTBOX.TEXT;
      THIS.STATE["DESCRIPTION"] = THIS.DESCRIPTIONTEXTBOX.TEXT;
      THIS.STATE["STAR"] = THIS.STARLISTPICKER.SELECTEDITEM;
```

```
        this.State["DueDate"] = this.DueDatePicker.Value;
        this.State["DueTime"] = this.DueTimePicker.Value; }
```

Pri návrate na stránku alebo pri jej prvom spustení sa aktivuje funkcia *OnNavigatedTo*. Táto funkcia obsahuje rozhodovacie algoritmy ktoré zabezpečujú uchovanie času a dátumu po návrate z objektu ich voľby.

```
if (this.pendingChosenDate.HasValue)
 this.State["DueDate"] = this.pendingChosenDate;
if (this.pendingChosenTime.HasValue)
 this.State["DueTime"] = this.pendingChosenTime;
```

Funkcia sa tiež rozhodne či ide užívateľ nový údaj vytvoriť alebo len upraviť pôvodné a podľa toho buď zobrazí prázdnu stránku s titulkom „*nová úloha*“ alebo načíta už uložené údaje a sprístupní ich na úpravu.

```
if (Settings.CurrentTask.Value == null)
        {this.PageTitle.Text = "nová úloha"; }
        else
        {this.PageTitle.Text = "upraviť";
          this.TitleTextBox.Text = Settings.CurrentTask.Value.Title;
          this.DescriptionTextBox.Text = Settings.CurrentTask.Value.Description;
          this.StarListPicker.SelectedItem = Settings.CurrentTask.Value.Star;
          this.DueDatePicker.Value =
            Settings.CurrentTask.Value.DueDate.LocalDateTime;
          this.DueTimePicker.Value =
            Settings.CurrentTask.Value.DueDate.LocalDateTime; }
```

Funkcia tiež obsahuje pomocné rozhodovacie príkazy *if*, ktoré kontrolujú, či sú jednotlivé polia vyplnené a ak sú, tak sú jednotlivé hodnoty uložené prislúchajúcim premenným.

```
if (this.State.ContainsKey("Title"))
          this.TitleTextBox.Text = (string)this.State["Title"];
        if (this.State.ContainsKey("Description"))
          this.DescriptionTextBox.Text = (string)this.State["Description"];
        if (this.State.ContainsKey("Star"))
```

```
    THIS.STARLISTPICKER.SELECTEDITEM = (STRING)THIS.STATE["STAR"];
  IF (THIS.STATE.CONTAINSKEY("DUEDATE"))
    THIS.DUEDATEPICKER.VALUE = (DATETIME?)THIS.STATE["DUEDATE"];
  IF (THIS.STATE.CONTAINSKEY("DUETIME"))
    THIS.DUETIMEPICKER.VALUE = (DATETIME?)THIS.STATE["DUETIME"];
```

Funkcia *OnNavigatedTo* využíva verejne prístupnú triedou *Task* (definovanou v samostatnom súbore *Task.cs*) ktorá definuje polia do ktorých sa načítajú zadané údaje a funkcie (vlastnosti) ktoré zabezpečia načítanie týchto údajov a aj ich prípadné vyvolanie.

Funkcia *TitleTextBox_TextChanged* obsahuje algoritmus na kontrolu vyplnenia názvu úlohy, ak názov úlohy nie je zadaný tak sa nesprístupní tlačidlo *uložiť*.

Na vykonanie akcie tlačidla *uložiť* sme vytvorili funkciu *SaveButton_Click*. Táto nám najprv združí dátum a čas do jednej hodnoty *DateTime* a potom vytvorí nový objekt triedy *Task* a naplní jej atribúty z polí zadaných užívateľom. Funkcia má v sebe kontrolný algoritmus, ktorý zisťuje či sa jedná o upravenie úlohy alebo vytvorenie novej, priradí jej dátum vytvorenia, respektíve dátum editovania (ten sa prepíše tým najaktuálnejším). Na základe tohto dátumu je neskôr táto úloha zaradená do príslušného zoznamu.

```
IF (SETTINGS.CURRENTTASK.VALUE != NULL)
      {IF (SETTINGS.TASKLIST.VALUE.REMOVE(SETTINGS.CURRENTTASK.VALUE))
        {ITEM.CREATEDDATE = SETTINGS.CURRENTTASK.VALUE.CREATEDDATE;
           SETTINGS.TASKLIST.VALUE.ADD(ITEM);
          SETTINGS.CURRENTTASK.VALUE = ITEM;}
        ELSE
        {SETTINGS.CURRENTTASK.VALUE.TITLE = ITEM.TITLE;
          SETTINGS.CURRENTTASK.VALUE.DESCRIPTION = ITEM.DESCRIPTION;
          SETTINGS.CURRENTTASK.VALUE.STAR = ITEM.STAR;
          SETTINGS.CURRENTTASK.VALUE.MODIFIEDDATE = ITEM.MODIFIEDDATE;
          SETTINGS.CURRENTTASK.VALUE.DUEDATE = ITEM.DUEDATE;} }
      ELSE
      { ITEM.CREATEDDATE = ITEM.MODIFIEDDATE;
        SETTINGS.TASKLIST.VALUE.ADD(ITEM); }
      IF (THIS.NAVIGATIONSERVICE.CANGOBACK)
```

```
        this.NavigationService.GoBack();  }
```

Na záver sa otestuje či je splnená podmienka návratu späť na hlavnú stránku, ak je splnená tak sa spustí funkcia **GoBack** ktorá to umožní.

3.5.3. Stránka detailov

Stránka detailov *DetailsPage.xaml* po prekliknutí z pivot menu zobrazí podrobnosti o vytvorenej úlohe. Tu sme znovu použili už šablónu z ponuky *pridať novú položku*. Doplnili sme tu *textbloky* ktoré zobrazujú názov úlohy, jej popis. Pod ne sme vložili objekt *rectangle* (obdĺžnik) ktorého vnútro je farebne zvýraznené a obsahuje tri riadky s dátumami a časmi.

```
<Rectangle x:Name="AccentRectangle" Grid.Row="1" Grid.ColumnSpan="2"
                    Grid.RowSpan="4" Fill="{StaticResource PhoneAccentBrush}"/>
```

Sú to dátumy dokedy má byť úloha splnená, kedy bola úloha vytvorená a kedy bola editovaná. Na správne zobrazenie dátumov sme si vytvorili v pomocnom súbore *DateConverter.cs* triedu *Dateconverter* ktorá nám konvertuje dátum načítaný z hodnoty zadanej používateľom na formát *dátum@čas*.

```
public class DateConverter : IValueConverter
  {public object Convert(object value, Type targetType, object parameter,
      CultureInfo culture)
    {DateTimeOffset date = (DateTimeOffset)value;
           return date.LocalDateTime.ToShortDateString() + " @ "
          + date.LocalDateTime.ToShortTimeString();}
public object ConvertBack(object value, Type targetType, object parameter,
      CultureInfo culture)
    {return DependencyProperty.UnsetValue; }}
```

Do stránky detailov sme vložili aj obrázok hviezdy, ktorý sa zobrazí len vtedy ak si používateľ pri vytváraní úlohy túto úlohu označil farebne výberom príslušnej položky z menu. Obrázok nadobudne takú istú farbu akú si používateľ zvolil.

```
<Rectangle Grid.Row="2" Width="240" Height="240" Fill="{Binding Star}"
      VerticalAlignment="Center" HorizontalAlignment="Stretch" Margin="0,0,0,12">
```

```
      <Rectangle.OpacityMask>
        <ImageBrush ImageSource="Images/bigStar.png"/>
      </Rectangle.OpacityMask>
    </Rectangle>
```

Aby si mohol používateľ úlohu upraviť aj z tejto obrazovky, prípadne ju označiť ako *spravenú* pridali sme do spodnej lišty tlačidlá *„spravená“, „upraviť“ a „vymazať“*.

Na vykonanie akcií týchto tlačidiel slúži súbor *DetailsPage.xaml.cs* kde je definovaná trieda *DetailsPage*. Tá obsahuje objekt triedy *IAplicationBarIconButton markUnmarkButton*, konštruktor ktorý tento objekt inicializuje. Na to aby sa nám správne zobrazilo tlačidlo ktorým sa označuje úloha ako spravená, respektíve nespravená, sme si vytvorili funkciu *MarkUnmarkButton_Click*. Táto metóda kontroluje či je úloha spravená alebo nie a podľa toho priradí tlačidlu *markUnmarkButton* správny popis a obrázok. V prípade ak je spravená, zobrazí sa obrázok fajky so symbolom mínus a *titulkom nespravená* alebo ak je spravená tak sa zobrazí ten istý obrázok, len so symbolom plus a titulkom *spravená*. Toto tlačidlo má opačný popis preto aby používateľ vedel, čo s týmto tlačidlom môže vykonať. Ak používateľ ťukne na tlačidlo vykonajú sa funkcie, ktoré úlohu z pôvodného zoznamu vymažú a pridajú do nového zoznamu. Používateľovi sa tak javí ako keby sa úloha premiestnila zo zoznamu nedokončených úloh do zoznamu spravených úloh ak ju označil ako spravenú.

```
void MarkUnmarkButton_Click(object sender, EventArgs e)
    {if (this.markUnmarkButton.Text == "spravená")
      {this.markUnmarkButton.IconUri = new Uri("/Images/appbar.unmarkDone.png",
          UriKind.Relative);
        this.markUnmarkButton.Text = "nespravená";
                Settings.TaskList.Value.Remove(Settings.CurrentTask.Value);
        Settings.DoneList.Value.Add(Settings.CurrentTask.Value);
      else
      {this.markUnmarkButton.IconUri = new Uri("/Images/appbar.markDone.png",
          UriKind.Relative);
        this.markUnmarkButton.Text = "spravená";
                Settings.DoneList.Value.Remove(Settings.CurrentTask.Value);
```

```
        Settings.TaskList.Value.Add(Settings.CurrentTask.Value); }}
```

V triede *DetailsPage* sa nachádza aj funkcia *EditButton_Click* obsluhujúca tlačidlo *upraviť*, táto v zásade len presunie používateľa na obrazovku *AddEditPage* kde môže byť táto úloha upravená.

```
void EditButton_Click(object sender, EventArgs e)
    {this.NavigationService.Navigate(new Uri("/AddEditPage.xaml",
        UriKind.Relative)); }
```

Tlačidlo so symbolom *smetného koša* je naviazané na funkciu *DeleteButton_Click*, ktorá po zobrazení výstražného hlásenia a jeho potvrdení vymaže aktuálnu úlohu.

```
void DeleteButton_Click(object sender, EventArgs e)
    {if (MessageBox.Show(
        "Ste si istý že chcete natrvalo odstrániť túto úlohu?", "Vymazať úlohu",
        MessageBoxButton.OKCancel) == MessageBoxResult.OK)
      {Settings.TaskList.Value.Remove(Settings.CurrentTask.Value);
        Settings.DoneList.Value.Remove(Settings.CurrentTask.Value);
if (this.NavigationService.CanGoBack)
          this.NavigationService.GoBack();}}
```

V triede *DetailsPage* sa nachádza aj funkcia *AboutMenuItem_Click*, ktorej úlohou je zobraziť obrazovku o aplikácii. Na správne zobrazenie dátumu pri prechode na stránku podrobností zo stránky upraviť sme vytvorili funkciu *OnNavigatedto*. Táto nám skontroluje či sa dátum zmenil alebo nie a podľa toho ho potom zobrazí v textbloku. Funkcia nám tiež zabezpečí správne zobrazenie ikony *spravená/nepravená* pri návrate na túto obrazovku.

```
protected override void OnNavigatedTo(NavigationEventArgs e)
    {base.OnNavigatedTo(e);
      this.DataContext = Settings.CurrentTask.Value;
if (Settings.CurrentTask.Value != null)
      {if (Settings.CurrentTask.Value.CreatedDate ==
            Settings.CurrentTask.Value.ModifiedDate)
        {this.ModifiedLabelTextBlock.Visibility = Visibility.Collapsed;
```

```
            this.ModifiedTextBlock.Visibility = Visibility.Collapsed; }
        else
        {this.ModifiedLabelTextBlock.Visibility = Visibility.Visible;
            this.ModifiedTextBlock.Visibility = Visibility.Visible; }
if (Settings.DoneList.Value.Contains(Settings.CurrentTask.Value))
        {this.markUnmarkButton.IconUri =
            new Uri("/Images/appbar.unmarkDone.png", UriKind.Relative);
            this.markUnmarkButton.Text = "nespravená";}}
```

3.5.4. *Stránka nastavení*

Pomocou stránky nastavení sme používateľovi umožnili nastaviť si aké položky pivot menu bude chcieť mať zobrazené. Stránku sme vytvorili tak ako predchádzajúce cez menu *add*, pridali sme hlavičku pomocou elementu *StackPanel*. Na výber položiek pivot menu sme sa rozhodli použiť zaškrtávacie políčka. Pre každú položku pivot menu sme vytvorili jedno políčko, pričom políčko „*všetky*“ sme nechali predvolene zaškrtnuté, zatmavené a neprístupné. Tým sme zabránili aby používateľ nechtiac neodznačil všetky položky a zabezpečili aby aspoň jedna položka pivot menu ostala vždy viditeľná.

Jednotlivé zaškrtávacie políčka sme pridali pomocou elementu *CheckBox* ktorý sme spolu umiestnili do rodičovského elemntu *ScrollViewer*. Jednotlivé popisy checkboxov sme pridali pomocou atribútov checkboxov *Content*. Na obrazovke je pridaný aj obrázok a to pomocou elementu *Rectangle*. Na to aby zaškrtávacie políčka fungovali, sme si vytvorili triedu *SettingsPage* v súbore *SettingsPage.xaml.cs*. Tá obsahuje metódu *OnNavigatedFrom* ktorá zabezpečí uloženie nastavení pri opustení obrazovky nastavení. Využíva pritom verejnú statickú triedu *Settings* ktorú sme si definovali v súbore *Settings.cs*

```
protected override void OnNavigatedFrom(NavigationEventArgs e)
    { base.OnNavigatedFrom(e);
        Settings.IsTodayVisible.Value = this.TodayCheckBox.IsChecked.Value;
        Settings.IsPastDueVisible.Value = this.PastDueCheckBox.IsChecked.Value;
        Settings.IsStarredVisible.Value = this.StarredCheckBox.IsChecked.Value;
        Settings.IsDoneVisible.Value = this.DoneCheckBox.IsChecked.Value; }
```

Na to aby nám trieda *SettingsPage* akceptovala nastavenie pri prechode z inej stránky na stránku nastavení sme vytvorili funkciu *OnNavigatedTo*.

```
PROTECTED OVERRIDE VOID ONNAVIGATEDTO(NAVIGATIONEVENTARGS E)
    {BASE.ONNAVIGATEDTO(E);
      THIS.TODAYCHECKBOX.ISCHECKED = SETTINGS.ISTODAYVISIBLE.VALUE;
      THIS.PASTDUECHECKBOX.ISCHECKED = SETTINGS.ISPASTDUEVISIBLE.VALUE;
      THIS.STARREDCHECKBOX.ISCHECKED = SETTINGS.ISSTARREDVISIBLE.VALUE;
      THIS.DONECHECKBOX.ISCHECKED = SETTINGS.ISDONEVISIBLE.VALUE;
```

Pri tejto funkcii tiež použijeme triedu *Settings* definovanú v súbore *settings.cs* a to preto, lebo pri opustení stránky nastavení sa nám vymažú zaškrtnuté voľby. Trieda settings zabezpečí, že voľby sú okamžite uložené a nestratia sa. Trieda *settings* je verejne prístupná statická trieda čo umožňuje že môže byť prístupná z ostatných tried. Prvých päť nastavení triedy *settings* udržuje stav pivot menu na hlavnej stránke a ďalšie nastavenie (*CurrentTask*) sprístupňuje detaily stránke pridať/upraviť o práve vybratej položke na hlavnej stránke.

```
PUBLIC STATIC CLASS SETTINGS
  {PUBLIC STATIC READONLY SETTING<STRING> SELECTEDPIVOTITEMNAME =
      NEW SETTING<STRING>("SELECTEDPIVOTITEMNAME", "ALL");
PUBLIC STATIC READONLY SETTING<BOOL> ISTODAYVISIBLE =
      NEW SETTING<BOOL>("ISTODAYVISIBLE", TRUE);
    PUBLIC STATIC READONLY SETTING<BOOL> ISPASTDUEVISIBLE =
      NEW SETTING<BOOL>("ISPASTDUEVISIBLE", TRUE);
    PUBLIC STATIC READONLY SETTING<BOOL> ISSTARREDVISIBLE =
      NEW SETTING<BOOL>("ISSTARREDVISIBLE", TRUE);
    PUBLIC STATIC READONLY SETTING<BOOL> ISDONEVISIBLE =
      NEW SETTING<BOOL>("ISDONEVISIBLE", TRUE);
 PUBLIC STATIC READONLY SETTING<TASK> CURRENTTASK =
      NEW SETTING<TASK>("CURRENTTASK", NULL);
```

Posledné dva *settingy* triedy *Setting* sú dva rozdielne typy kolekcií. Prvý typ je základná kolekcia úloh, kde nič nie je zoraďované. Naproti tomu druhý typ je kolekcia, ktorá

automaticky usporiadava položky zoznamu nespravených úloh podľa dátumu do kedy ich treba urobiť.

```
PUBLIC STATIC READONLY SETTING<OBSERVABLECOLLECTION<TASK>> DONELIST = NEW
SETTING<OBSERVABLECOLLECTION<TASK>>("DONELIST",
        NEW OBSERVABLECOLLECTION<TASK>());
PUBLIC STATIC READONLY SETTING<SORTEDTASKCOLLECTION> TASKLIST =
      NEW SETTING<SORTEDTASKCOLLECTION>("TASKLIST",
        NEW SORTEDTASKCOLLECTION());}
```

Obidve kolekcie sú typu observable[39], pretože hlavná stránka spolieha na notifikáciách založených na zmenách kolekcie, aby bola vždy aktuálna pri pridávaní alebo odstránení úloh.

Na zoradenie úloh podľa dátumu sme si vytvorili verejne prístupnú triedu *SortTaskCollection* v súbore *SortTaskCollection.cs* ktorá implementuje funkciu *InsertItem*. Táto funkcia, keď je zavolaná prejde celým poľom úloh a zoradí ich chronologicky.

```
PUBLIC CLASS SORTEDTASKCOLLECTION : OBSERVABLECOLLECTION<TASK>
  {PROTECTED OVERRIDE VOID INSERTITEM(INT INDEX, TASK ITEM)
    {INT I = 0;
      FOR (I = 0; I < THIS.COUNT; I++)
      {DATETIMEOFFSET D = THIS[I].DUEDATE;
        IF (D > ITEM.DUEDATE)
          BREAK; }
      BASE.INSERTITEM(I, ITEM); }}
```

3.5.5. STRÁNKA O APLIKÁCII

Táto stránka je dôležitá najmä pre vývojára, keďže pomocou nej ho môže používateľ aplikácie kontaktovať prípadne sa oboznámiť s ďalšími aplikáciami ktoré vývojár vytvoril. Spravidla obsahuje ikonu aplikácie, číslo verzie aplikácie, odkaz na prístup do Marketplace, ktorý ho presmeruje na ďalšie vytvorené aplikácie od toho istého

[39] **Trieda ktorá má vstavanú implementáciu rozhrania INotifyCollectionChanged ktoré poskytuje notifikácie pri obnovení kolekcie**

autora a poprípade kontakt na vývojárov aplikácie. My sme všetky tieto údaje do stránky zahrnuli, pričom ťuknutím na mailovú adresu sa používateľovi otvorí mailový klient s vyplnenou adresou prijímateľa ktorou je emailová adresa autora aplikácie. Naša stránka okrem toho pri každom zostavení aplikácie má automaticky prepísané posledné päťčíslie verzie aplikácie. Stránku sme navrhli pomocou elementu *StackPanel* do ktorého sme umiestnili elementy *textblock* ktoré svojimi vlastnosťami definujú názov stránky, aplikácie a ich štýl. Pod nadpis sme umiestnili element *ScrolViewer* do ktorého sme umiestnili element *Image* ktorý predstavuje ikonu aplikácie. Pod ním sú elementy *textblock* s menom autora, mailom a odkazom na ďalšie aplikácie. Na to aby tieto linky fungovali vytvorili sme si triedu *AboutPage* v súbore *AboutPage.xaml.cs*. Tu sme si vytvorili konštruktor, ktorý sa stará o zapísanie inkrementácie verzie aplikácie. Pomocou metódy *OnNavigateTo* sme zabezpečili nastavenie názvu aplikácie do hlavičky stránky.

```
PROTECTED OVERRIDE VOID ONNAVIGATEDTO(NAVIGATIONEVENTARGS E)
    {      BASE.ONNAVIGATEDTO(E);
      IF (THIS.NAVIGATIONCONTEXT.QUERYSTRING.CONTAINSKEY("APPNAME"))
      {THIS.APPLICATIONNAME.TEXT =
          THIS.NAVIGATIONCONTEXT.QUERYSTRING["APPNAME"].TOUPPERINVARIANT();}}
```

Pre zobrazenie ďalších aplikácií v Marketplace sme vytvorili funkciu *ViewMoreTextBlock*. Táto inicializuje spustenie Marketplace a vyhľadá aplikácie od zadaného autora. Na odoslanie spätnej odozvy autorovi sme vytvorili funkciu ktorá zavolá emailovú aplikáciu a pomocou dotazu v nej vyplní predmet a cieľovú adresu.

3.5.6. STRÁNKA INŠTRUKCIÍ

Obrazovku inštrukcií sme vytvorili kompletne pomocou grafického editora pridaním elementov *StackPanel* a *ScrollViewer* a doplnením potrebného textu. Keďže stránka neobsahuje ovládacie prvky nebolo potrebné vytvárať kód na pozadí. Stránka obsahuje jednoduchý návod ako aplikáciu ovládať.

3.6. Testovanie

Aplikáciu sme otestovali pomocou vstaveného emulátora Visual Studia a aj pomocou samotného zariadenia. Potvrdia sa nám teória, že testovanie na reálnom zariadení je vierohodnejšie keďže aplikácia nám tu fungovala svižnejšie. Na základné otestovanie však emulátor postačuje. Podarilo sa nám odhaliť mnoho chýb, môžeme napríklad spomenúť odhalenie nekorektného správania tlačidla na stránke podrobností úlohy, kde sa tlačidlo označenia úlohy správalo presne naopak ako sme pôvodne zamýšľali. Pri označení úlohy ako hotovej v hlavnom menu sa táto skutočnosť nepremietla do zobrazenia tohto tlačidla na stránke podrobností. Na chybu sme ale rýchlo prišli a opravili sme ju v kóde na pozadí v súbore *Details.xaml.cs*. Oprava spočívala v správnom zadaní názvu tlačidla na ktoré sa odkazovalo v xaml zdrojovom kóde. Spustenie testovania sa vykoná výberom zariadenia na hornej lište Visual Studia (okno Select target for Windows Phone projects) a výberom akcie *Debug* (okno solution configurations) a stlačením klávesy F5 pričom sa spustí buď emulátor alebo priamo pripojené mobilné zariadenie.

Po úspešnom otestovaní sme nechali aplikáciu zostaviť výberom akcie *Release.* Po tomto kroku sa nám vygeneroval XAP súbor ktorý sme pomocou nášho konta nahrali do Marketplace. Tu sme pridali aj popis aplikácie, snímky obrazovky, ikony aplikácie a potvrdili odoslanie aplikácie na schválenie. Po troch dňoch nám prišiel informačný mail o tom, že aplikácia bola schválená a že nasledujúci deň bude zverejnená v Marketplace.

4. Diskusia

Tvorba aplikácií pre mobilné zariadenia sa stáva čoraz častejším pojmom vo svete vývojárov. Keďže predaje inteligentných mobilných zariadení postupne dobiehajú a niekde predbiehajú predaje stolných počítačov a notebookov je veľmi výhodné pre vývojárov sa preorientovať na vývoj aplikácií práve pre takéto zariadenia. Vývojári majú na výber pomerne široké množstvo platforiem, ktoré sa medzi sebou líšia nielen použitým programovacím jazykom, ale aj rozdielnymi vývojovými prostrediami, ekosystémom aplikácií, poskytovanou podporou a v konečnom dôsledku aj množstvom potenciálnych zákazníkov. Táto posledná skutočnosť má pritom často najväčšiu váhu pri rozhodovaní sa

potenciálnych začínajúcich vývojárov, ktorú platformu si vyberú. Je to najmä preto lebo výber vhodnej platformy s veľkým množstvom zákazníkov im môže priniesť dodatočné finančné prostriedky. Odradiť ich však môže z tohto dôvodu zvýšená konkurencia. Vývojári však môžu aj preferovať platformy na základe iných kritérií ako je napríklad vyspelosť vývojového prostredia, dostupnosť náučných materiálov respektíve dokumentácie prípadne potenciál rozvoja platformy. Týmito posledne menovanými kritériami sme sa snažili riadiť v našom prípade.

V našej práci sme si vybrali jednu z štyroch najpopulárnejších platforiem, ktoré sú v súčasnosti na trhu (nie však tú najväčšiu) Windows Phone, pretože má perspektívu rastu zákazníkov a medzi vývojármi pri nej nie je až taká silná konkurencia. Naša voľba padla na túto platformu aj preto, lebo na jej vývoj nebolo potrebné platiť vstupné poplatky a preto, lebo poskytuje príjemné a vysoko funkčné vývojové prostredie Visual Studio.

V zahraničnej literatúre sa môžeme stretnúť s niekoľkými autormi, ktorí sa danou témou zaoberali spomenúť môžeme napríklad Adam Nathan alebo Charles Petzold, na slovensku však dosiaľ nevyšla žiadna kniha s touto tematikou aj keď niektorí autori ako Ľuboslav Lacko sa jej stručne venovali. To môže byť spôsobené aj tým že táto platforma je pomerne nová a len v poslednom období vstupuje na náš trh.

Cieľom našej práce bolo aj predstaviť platformu Windows Phone, jej súčasti, zloženie, vývojové prostredie. Snažili sme sa aj ukázať ako vývoj aplikácie na platforme Windows Phone prebieha a akými prostriedkami sa to dá dosiahnuť. Vývoj aplikácie nám bol z časti uľahčený vývojovým prostredím, napriek tomu začiatky pri vývoji aplikácií nebývajú najľahšie. Aj preto sme vytvorili jednoduchú aplikáciu na poznamenávanie úloh a popísali sme postup pri vývoji jej jednotlivých súčastí. Tak sme názorne ukázali ako sa dá takáto jednoduchá aplikácia vytvoriť. Uvedomujeme si ,že používateľovi aplikácie sa môže zdať až príliš jednoduchá, a možno mu v nej bude chýbať nejaká funkcionalita, ale aspoň ho nebude zbytočne zaťažovať a zdržiavať svojou zložitosťou. Výsledok a vydarenosť našej práce môžu posúdiť a ohodnotiť mimo odbornej verejnosti aj používatelia našej aplikácie, ktorú si mi môžu voľne stiahnuť na slovenskom Marketplace.

Záver

V práci sme sa pokúsili opísať súčasná stav na trhu platforiem mobilných operačných systémov, ich podiely na trhu, a smerovanie. Stručne sme predstavili jednotlivé mobilné operačné systémy , ich aplikačnú základňu a vývojové prostredia. Ďalej sme si vybrali jednu platformu konkrétne Windows Phone a tú sme podrobnejšie opísali. Zamerali sme sa na jej vývojové prostredie, na jej komponenty. Vo výsledkoch práce sme popísali analýzu, návrh vývoj a testovanie konkrétnej aplikácie. Objasnili sme tvorbu jej jednotlivých obrazoviek a aj zdrojového kódu ktorý zaručuje jej správne fungovanie.

Zoznam použitej literatúry:

1. Brownlow,M. Smartphone statistics and market share [on-line][cit.2012.04.20] Dostupné na internete: < http://www.email-marketing-reports.com/wireless-mobile/smartphone-statistics.htm>
2. Global mobile statistics 2012: all quality mobile marketing research, mobile Web stats, subscribers, ad revenue, usage, trends… [on-line] [cit.2012.04.8] Dostupné na internete: <http://mobithinking.com/mobile-marketing-tools/latest-mobile-stats>
3. Chloe Albanesius, Smartphone Shipments Surpass PCs for First Time. What's Next? [on-line] [cit.2012.02.8] Dostupné na internete: <http://www.pcmag.com/article2/0,2817,2379665,00.asp>
4. The Most Valuable Company in the World [on-line] 16.2.2012 [cit.2012.02.16] Dostupné na internete: < http://www.forbes.com/sites/marketshare/2012/02/16/the-most-valuable-company-in-the-world/>
5. Telegraph staff and agencies [on-line] [cit. 30.3. 2012] Dostupné na internete: <http://www.telegraph.co.uk/finance/newsbysector/mediatechnologyandtelecoms/electronics/9175006/BlackBerry-maker-Research-in-Motion-to-focus-on-business-customers-after-revenue-falls-25pc.html>
6. Android Market Tops 400000[on-line] [cit. 30.3. 2012] Dostupné na internete: <http://www.pcworld.com/article/247247/android_market_tops_400000_apps.html>
7. Ian Paul, PCWorld [on-line] [cit. 4.1. 2012] Dostupné na internete: <http://www.pcworld.com/article/247247/android_market_tops_400000_ap

ps.html>

8. Algorithms the BE Solution uses to encrypt data [on-line] [cit. 4.1. 2012] Dostupné na internete: <http://docs.blackberry.com/en/admin/deliverables/7325/Algorithms_the_BESolution_uses_to_encrypt_data_806442_11.jsp>
9. Zeis, A., Blackberry app world stats devcon europe [on-line] [cit. 7.2. 2012] Dostupné na internete: < http://crackberry.com/blackberry-app-world-stats-devcon-europe>
10. App-store [on-line] [cit. 7.2. 2012] Dostupné na internete: <http://www.apple.com/iphone/built-in-apps/app-store.html>
11. Satapathy, A., [on-line] The Mobile Indian, New Delhi, March 28, 2011 [cit. 2012.03.10.] Dostupné na internete: <http://www.themobileindian.com/news/883_Samsung-bada-app-store-crosses-100-million-downloads>
12. Egham, UK, [on-line] February 15, 2012 [cit.2012.02.18] Dostupné na internete: <http://www.gartner.com/it/page.jsp?id=1924314>
13. Flann [on-line] on May 26, 2011 [cit.2012.02.8] Dostupné na internete: <http://conversations.nokia.com/2011/05/26/stephen-elop-in-china-time-for-a-challenger-mindset-video/>
14. Nokia Ovi Store [on-line]Dec 27, 2011 [cit.2012.02.8] Dostupné na internete: <http://www.distimo.com/appstores/app-store/21-Nokia_Ovi_Store>
15. Apps [on-line] [cit.2012.03.8] Dostupné na internete: <http://www.webosnation.com/apps>
16. N900 [on-line] [cit.2012.03.8] Dostupné na internete: <http://wiki.meego.com/N900>
17. How to install MeeGo Hartman [on-line]August 9, 2011[cit.2012.02.8]

Dostupné na internete: < http://qt-project.org/wiki/How_to_Install_deb_To_MeeGo_Harmattan#a0d4b7e5582a446ad31fbe1a2305db20>

18. Neil Mawston [on-line] 23. 2. 2012 [cit.2012.02.8] Dostupné na internete: <http://www.strategyanalytics.com/default.aspx?mod=ReportAbstractViewer&a0=7122>
19. Blandford, R. Windows Phone Marketplace Pass [on-line] 3.4. 2012 [cit.2012.04.8] Dostupné na internete: <http://allaboutwindowsphone.com/news/item/14554_Windows_Phone_Marketplace_pass.php>
20. System requirements [on-line] [cit.2012.02.8] Dostupné na internete: <http://www.microsoft.com/visualstudio/en-us/products/2010-editions/ultimate/system-requirements>
21. Rak,D., Ako nevyvinúť zle mobilnú aplikáciu [on-line] Infoware,3. 2012. [cit.2012.03.8] Dostupné na internete: <http://www.qbsw.sk/tlacove-spravy/news/ako-nevivinut-zle-mobilnu-aplikaciu/25986b9aa9>

23. LaMarche ,J.,Mark,D., 2010 ,*Průvodce vývojem aplikací pro iPhone a iPod touch* · Praha, Computer Press, 2010,
24. Mark L. Murphy 2011, *Android 2 Průvodce programováním mobilních aplikací 2011*
25. Lacko,L. Ako stvoriť Android[on-line]15.3.2011 [cit.2012.03.8] Dostupné na internete: <http://www.itnews.sk/tituly/infoware/2011-03-15/c138709-ako-stvorit-android-1.-cast-zaciname Ako stvoriť Android / 1. časť: Začíname >
26. Egham, Gartner Says Worldwide Smartphone Sales Soared in Fourth Quarter of 2011 With 47 Percent Growth [on-line] 15.2 [cit.2012.03.8] Dostupné na internete: <http://www.gartner.com/it/page.jsp?id=1924314>
27. Application Certification Requirements for Windows Phone [on-line] March 22, 2012 [cit.2012.03.8] Dostupné na internete: <http://go.microsoft.com/fwlink/?LinkID=183220>

28. Natham, A., 2011, *101 Windows ® Phone 7 Apps*, 1.vydanie, Indiana 46240 USA, , Sams, 2011, 1136 s. ISBN-10: 0-672-33552-2

Prílohy:

1.

```
<phone:PhoneApplicationPage x:Class="WindowsPhoneApp.MainPage"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:phone="clr-namespace:Microsoft.Phone.Controls;assembly=Microsoft.Phone"
    xmlns:shell="clr-namespace:Microsoft.Phone.Shell;assembly=Microsoft.Phone"
    xmlns:controls="clr-
namespace:Microsoft.Phone.Controls;assembly=Microsoft.Phone.Controls"
    xmlns:toolkit="clr-
namespace:Microsoft.Phone.Controls;assembly=Microsoft.Phone.Controls.Toolkit"
    xmlns:local="clr-namespace:WindowsPhoneApp"
    FontFamily="{StaticResource PhoneFontFamilyNormal}"
    FontSize="{StaticResource PhoneFontSizeNormal}"
    Foreground="{StaticResource PhoneForegroundBrush}"
    SupportedOrientations="PortraitOrLandscape" shell:SystemTray.IsVisible="True"
mc:Ignorable="d" xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
d:DesignHeight="696" d:DesignWidth="480">

  <!-- Lišta aplikácie,s 3 tlačítkami a jedným tlačidlom menu -->
  <phone:PhoneApplicationPage.ApplicationBar>
    <shell:ApplicationBar>
      <shell:ApplicationBarIconButton Text="nová úloha"
        IconUri="/Shared/Images/appbar.add.png" Click="AddButton_Click"/>
      <shell:ApplicationBarIconButton Text="inštrukcie"
        IconUri="/Shared/Images/appbar.instructions.png"
        Click="InstructionsButton_Click"/>
      <shell:ApplicationBarIconButton Text="nastavenia"
        IconUri="/Shared/Images/appbar.settings.png"
        Click="SettingsButton_Click"/>
      <shell:ApplicationBar.MenuItems>
        <shell:ApplicationBarMenuItem Text="o aplikácii"
Click="AboutMenuItem_Click"/>
      </shell:ApplicationBar.MenuItems>
    </shell:ApplicationBar>
  </phone:PhoneApplicationPage.ApplicationBar>

  <phone:PhoneApplicationPage.Resources>
    <!--Dátová šablóna zdieľaná prvými 4 listboxami -->
    <DataTemplate x:Key="DataTemplate">
      <StackPanel Orientation="Horizontal" local:Tilt.IsEnabled="True">

        <!-- Pridanie kontextového menu položke -->
        <toolkit:ContextMenuService.ContextMenu>
          <toolkit:ContextMenu Opened="ContextMenu_Opened"
                               Closed="ContextMenu_Closed">
            <toolkit:MenuItem Header="úloha spravená" Click="MarkMenuItem_Click"/>
            <toolkit:MenuItem Header="upraviť" Click="EditMenuItem_Click"/>
            <toolkit:MenuItem Header="vymazať" Click="DeleteMenuItem_Click"/>
          </toolkit:ContextMenu>
        </toolkit:ContextMenuService.ContextMenu>
```

```
        <!-- Hviezda, s farbou na základe druhu položiek -->
        <Rectangle Fill="{Binding Star}" Width="26" Height="25" Margin="0,0,0,10">
          <Rectangle.OpacityMask>
            <ImageBrush ImageSource="Images/star.png"/>
          </Rectangle.OpacityMask>
        </Rectangle>
        <!-- Titulka -->
        <TextBlock Text="{Binding Title}" Margin="8,0,0,16"
                   Style="{StaticResource PhoneTextExtraLargeStyle}"/>
      </StackPanel>
    </DataTemplate>
  </phone:PhoneApplicationPage.Resources>

  <controls:Pivot x:Name="Pivot" Title="UROB TO">

    <!--Vylepšenie titulku UROB TO  aby zodpovedal vstavaným aplikáciám-->
    <controls:Pivot.TitleTemplate>
      <DataTemplate>
        <TextBlock Text="{Binding}" Margin="-1,-1,0,-3"
                   Style="{StaticResource PhoneTextTitle0Style}"/>
      </DataTemplate>
    </controls:Pivot.TitleTemplate>

    <!-- Pivot položka #1 -->
    <controls:PivotItem Header="všetky">
      <Grid>
        <TextBlock x:Name="NoAllTextBlock" Text="Žiadne úlohy"
Visibility="Collapsed"
          Margin="22,17,0,0" Style="{StaticResource PhoneTextGroupHeaderStyle}"/>
                <Rectangle Opacity=".2" Margin="0,0,0,12"
VerticalAlignment="Center"
                   HorizontalAlignment="Center" Width="240" Height="240"
                   Fill="{StaticResource PhoneForegroundBrush}">
                    <Rectangle.OpacityMask>
                        <ImageBrush ImageSource="Images/home.png"/>
                    </Rectangle.OpacityMask>
                </Rectangle>
                <ListBox x:Name="AllListBox" ItemsSource="{Binding}"
                 ItemTemplate="{StaticResource DataTemplate}"
                 SelectionChanged="ListBox_SelectionChanged"/>
      </Grid>
    </controls:PivotItem>

    <!-- Pivot položka #2 -->
    <controls:PivotItem x:Name="TodayPivotItem" Header="dnes">
      <Grid>
        <TextBlock x:Name="NoTodayTextBlock" Text="Dnes netreba nič spraviť"
          Visibility="Collapsed" Margin="22,17,0,0"
          Style="{StaticResource PhoneTextGroupHeaderStyle}"/>
        <!-- Ukáže dnešný dátum pod list boxom -->
        <TextBlock x:Name="TodayTextBlock" Opacity=".2" Margin="0,0,0,4"
          HorizontalAlignment="Center" VerticalAlignment="Bottom"
FontWeight="Light"
          FontSize="{StaticResource PhoneFontSizeExtraExtraLarge}"/>
```

```
    <ListBox x:Name="TodayListBox"
             ItemTemplate="{StaticResource DataTemplate}"
             SelectionChanged="ListBox_SelectionChanged"/>
  </Grid>
</controls:PivotItem>

<!-- Pivot položka #3 -->
<controls:PivotItem x:Name="PastDuePivotItem" Header="nestihnuté">
  <Grid>
    <TextBlock x:Name="NoPastDueTextBlock" Visibility="Collapsed"
      Text="Žiadne nestihnuté úlohy!" Margin="22,17,0,0"
      Style="{StaticResource PhoneTextGroupHeaderStyle}"/>
    <!-- Zobraz ikonu hodín pod list boxom -->
    <Rectangle Opacity=".2" Margin="0,0,0,12" VerticalAlignment="Center"
               HorizontalAlignment="Center" Width="240" Height="240"
               Fill="{StaticResource PhoneForegroundBrush}">
      <Rectangle.OpacityMask>
        <ImageBrush ImageSource="Images/clock.png"/>
      </Rectangle.OpacityMask>
    </Rectangle>
    <ListBox x:Name="PastDueListBox"
             ItemTemplate="{StaticResource DataTemplate}"
             SelectionChanged="ListBox_SelectionChanged"/>
  </Grid>
</controls:PivotItem>

<!-- Pivot položka #4 -->
<controls:PivotItem x:Name="StarredPivotItem" Header="kategórie">
  <Grid>
    <TextBlock x:Name="NoStarredTextBlock" Text="Žiadne zvýraznené úlohy"
      Visibility="Collapsed" Margin="22,17,0,0"
      Style="{StaticResource PhoneTextGroupHeaderStyle}"/>
    <!-- Zobrazí hviezdu pod list boxom -->
    <Rectangle Opacity=".2" Margin="0,0,0,12" VerticalAlignment="Center"
               HorizontalAlignment="Center" Width="240" Height="240"
               Fill="{StaticResource PhoneForegroundBrush}">
      <Rectangle.OpacityMask>
        <ImageBrush ImageSource="Images/bigStar.png"/>
      </Rectangle.OpacityMask>
    </Rectangle>
    <ListBox x:Name="StarredListBox"
             ItemTemplate="{StaticResource DataTemplate}"
             SelectionChanged="ListBox_SelectionChanged"/>
  </Grid>
</controls:PivotItem>

<!-- Pivot položka #5 -->
<controls:PivotItem x:Name="DonePivotItem" Header="spravené">
  <Grid>
    <TextBlock x:Name="NoDoneTextBlock" Text="Nič nie je hotové. Urob TO!"
      Visibility="Collapsed" Margin="22,17,0,0"
      Style="{StaticResource PhoneTextGroupHeaderStyle}"/>
    <!-- zobrazí znak spravenia pod list boxom -->
    <Rectangle Opacity=".2" Margin="0,0,0,12" VerticalAlignment="Center"
               HorizontalAlignment="Center" Width="277" Height="240"
```

```
            Fill="{StaticResource PhoneForegroundBrush}">
    <Rectangle.OpacityMask>
      <ImageBrush ImageSource="Images/done.png"/>
    </Rectangle.OpacityMask>
  </Rectangle>
  <ListBox x:Name="DoneListBox" ItemsSource="{Binding}"
           SelectionChanged="ListBox_SelectionChanged">
    <ListBox.ItemTemplate>
                <!-- špeciálny vzor pre list box "spravené"  -->
      <DataTemplate>
        <StackPanel Orientation="Horizontal" Background="Transparent"
                    local:Tilt.IsEnabled="True">

          <!-- Pridá kontextové menu položke -->
          <toolkit:ContextMenuService.ContextMenu>
            <toolkit:ContextMenu Opened="ContextMenu_Opened"
                                 Closed="ContextMenu_Closed">
              <toolkit:MenuItem Header="označ ako nespravené"
                                Click="UnmarkMenuItem_Click"/>
              <toolkit:MenuItem Header="upraviť" Click="EditMenuItem_Click"/>
              <toolkit:MenuItem Header="vymazať"
                                Click="DeleteMenuItem_Click"/>
            </toolkit:ContextMenu>
          </toolkit:ContextMenuService.ContextMenu>

          <!-- Obrázok symbolu spravenej úlohy -->
          <Rectangle Width="48" Height="48"
                     Fill="{StaticResource PhoneForegroundBrush}">
            <Rectangle.OpacityMask>
              <ImageBrush ImageSource="Shared/Images/normal.done.png"/>
            </Rectangle.OpacityMask>
          </Rectangle>

          <Grid>
            <StackPanel Orientation="Horizontal" Margin="8,0,0,0">
              <!-- Označenie farbou podľa typu úlohy -->
              <Rectangle Fill="{Binding Star}" Width="26" Height="25">
                <Rectangle.OpacityMask>
                  <ImageBrush ImageSource="Images/star.png"/>
                </Rectangle.OpacityMask>
              </Rectangle>
              <!-- Titulka -->
              <TextBlock Text="{Binding Title}" Margin="8,0,0,6"
                         Style="{StaticResource PhoneTextExtraLargeStyle}"
                         HorizontalAlignment="Left" />
            </StackPanel>
            <!--vetikálna čiara prečiarknutia titulky spravenej úlohy -->
            <Line X1="-2" X2="800" Y1="32" Y2="32" StrokeThickness="2"
                  Stroke="{StaticResource PhoneForegroundBrush}"/>
          </Grid>
        </StackPanel>
      </DataTemplate>
    </ListBox.ItemTemplate>
  </ListBox>
</Grid>
```

```
    </controls:PivotItem>
  </controls:Pivot>
</phone:PhoneApplicationPage>
```

2.

úloha spravená

upraviť

vymazať