

Multiplatformní aplikace pro mobilní zařízení

Cross-platform Applications for Mobile Devices

Bc. Václav Nepožitek

Diplomová práce
2013



Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

*** nascannované zadání str. 1 ***

*** nascannované zadání str. 2 ***

ABSTRAKT

V současné době existuje několik mobilních operačních systémů. Roste množství aplikací, které je potřeba vytvořit na každý z nich. Existuje myšlenka tvorby multiplatformní aplikace. V této práci je provedena analýza možností tvorby multiplatformních aplikací na současná zařízení a jsou zde popsány výhody a nevýhody jednotlivých přístupů. Práce popisuje také problémy, se kterými se musí vývojář vypořádat při vývoji jednoduché nativní a multiplatformní aplikace. Na obou aplikacích jsou provedeny výkonnostní testy. Práce pomůže při rozhodování, zda se vydat cestou vývoje nativní nebo multiplatformní aplikace.

Klíčová slova: multiplatformní aplikace, nativní aplikace, uživatelské rozhraní, PhoneGap, jQuery Mobile, HTML 5, JavaScript, Sencha, Xamarin, Appcelerator, Titanium, Android, Windows Phone

ABSTRACT

There are several mobile operating systems these days and the number of applications that need to be created for each of them is growing. There is an idea of creating cross-platform application. This thesis analyzes possibilities of creating cross-platform applications for mobile devices and describes advantages and disadvantages of each approach. The thesis describes problems which the developer must deal with during the development of simple cross-platform and native application. Performance tests are executed for both applications. The thesis will help to make decision whether to go through native or cross-platform development.

Keywords: cross platform application, native application, user interface, PhoneGap, jQuery Mobile, HTML 5, JavaScript, Sencha, Xamarin, Appcelerator, Titanium, Android, Windows Phone

Děkuji panu Ing. Radku Valovi za cenné rady a připomínky k vývoji aplikací a čas, který mi věnoval při realizaci této diplomové práce.

Prohlašuji, že

- beru na vědomí, že odevzdáním diplomové/bakalářské práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb. o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších právních předpisů, bez ohledu na výsledek obhajoby;
- beru na vědomí, že diplomová/bakalářská práce bude uložena v elektronické podobě v univerzitním informačním systému dostupná k prezenčnímu nahlédnutí, že jeden výtisk diplomové/bakalářské práce bude uložen v příruční knihovně Fakulty aplikované informatiky Univerzity Tomáše Bati ve Zlíně a jeden výtisk bude uložen u vedoucího práce;
- byl/a jsem seznámen/a s tím, že na moji diplomovou/bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- beru na vědomí, že podle § 60 odst. 1 autorského zákona má UTB ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- beru na vědomí, že podle § 60 odst. 2 a 3 autorského zákona mohu užít své dílo – diplomovou/bakalářskou práci nebo poskytnout licenci k jejímu využití jen s předchozím písemným souhlasem Univerzity Tomáše Bati ve Zlíně, která je oprávněna v takovém případě ode mne požadovat přiměřený příspěvek na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše);
- beru na vědomí, že pokud bylo k vypracování diplomové/bakalářské práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tedy pouze k nekomerčnímu využití), nelze výsledky diplomové/bakalářské práce využít ke komerčním účelům;
- beru na vědomí, že pokud je výstupem diplomové/bakalářské práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá. Neodevzdání této součásti může být důvodem k neobhájení práce.

Prohlašuji,

- že jsem na diplomové práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků budu uveden jako spoluautor.
- že odevzdaná verze diplomové práce a verze elektronická nahraná do IS/STAG jsou totožné.

Ve Zlíně

.....
podpis diplomanta

OBSAH

ÚVOD.....	9
I TEORETICKÁ ČÁST	10
1 MOBILNÍ OPERAČNÍ SYSTÉMY	11
1.1 ANDROID.....	11
1.1.1 Vývojová prostředí.....	12
1.1.1.1 Android SDK	12
1.1.1.2 Android Development Tools	12
1.1.2 Uživatelské rozhraní a struktura aplikace	12
1.1.2.1 Activity - Aktivity.....	12
1.1.2.2 Services - Služby.....	13
1.1.2.3 Content Providers - Poskytovatelé obsahu	13
1.1.2.4 Broadcast receivers	13
1.1.3 Specifika systému Android	13
1.2 APPLE IOS.....	14
1.2.1 Vývojová prostředí.....	15
1.2.2 Uživatelské rozhraní.....	15
1.2.3 Specifika systému iOS	17
1.3 WINDOWS PHONE.....	17
1.3.1 Vývojové prostředí.....	18
1.3.2 Uživatelské rozhraní.....	18
1.3.3 Specifika systému.....	18
1.3.4 Design	19
2 NATIVNÍ APLIKACE.....	20
3 MULTIPLATFORMNÍ APLIKACE	21
3.1 MULTIPLATFORMNÍ KOMPILÁTORY	21
3.1.1 Appcelerator – Titanium	21
3.1.2 Xamarin.....	22
3.2 WEBOVÉ APLIKACE	24
3.2.1 Sencha Touch	24
3.3 HYBRIDNÍ APLIKACE.....	26
3.3.1 PhoneGap	26
3.4 UŽIVATELSKÉ ROZHRAŇÍ A RESPONZIVNÍ WEB DESIGN	28
4 PŘÍPADOVÉ STUDIE	29
4.1.1 Příklad č. 1	29
4.1.2 Příklad 2	30
4.1.3 Příklad 3	30
II PRAKTICKÁ ČÁST.....	32
5 VÝVOJ APLIKACE	33
5.1 NÁVRH APLIKACE.....	33
5.1.1 Databáze	33

5.1.2	Webová služba	35
5.2	NATIVNÍ APLIKACE	35
5.2.1	Popis uživatelského rozhraní.....	36
5.2.2	Logika Aplikace	37
5.3	PHONEGAP APLIKACE PRO WINDOWS PHONE A ANDROID.....	38
5.3.1	Struktura projektu.....	38
5.3.2	Přihlášení.....	39
5.3.3	PhoneGap – Ukládání dat.....	41
5.3.4	Přidání nového záznamu	41
5.3.5	Přenos aplikace na Android.....	41
5.4	VÝKON NATIVNÍ A MULTIPLATFORMNÍ APLIKACE	44
5.4.1	Metodika měření	44
5.4.2	Výsledky měření.....	45
5.4.2.1	Nativní aplikace	45
5.4.2.2	Multiplatformní aplikace	46
ZÁVĚR		49
ZÁVĚR V ANGLIČTINĚ.....		50
6 SEZNAM POUŽITÉ LITERATURY		51
SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK		53
SEZNAM OBRÁZKŮ		54
SEZNAM TABULEK.....		55
SEZNAM PŘÍLOH.....		56

ÚVOD

S rozvojem mobilních operačních systémů, telekomunikačních služeb a mobilních telefonů dochází zároveň k rozvoji tvorby aplikací a her na jednotlivé mobilní telefony. Existuje celá řada výrobců, kteří se snaží na trhu prorazit právě se svým operačním systémem a ekosystémem služeb. Výrobce vydává nativní vývojové nástroje, které mohou skrze operační systém využívat hardware telefonu. Uživatelé telefonů instalují aplikace, které jim usnadňují život, které je baví a hrají hry. Zpravodajské portály, sociální sítě, internetové obchody prodávají své zboží či služby a cílí reklamu na uživatele mobilních telefonů. Poptávka po mobilní aplikaci roste. Snahou vývojáře je vytvořit aplikaci tak, aby byla rychle, levně, přitom kvalitně a konkurence-schopně napsaná. S rozrůstajícím se počtem mobilních operačních systémů roste i počet aplikací, které je potřeba vytvořit. Objevuje se myšlenka tvorby multiplatformní aplikace. Aplikace, která bude vytvořena pomocí jednoho kódu a zároveň bude spustitelná na několika různých platformách. Práce se zabývá současnými možnostmi tvorby multiplatformní aplikace, jakým způsobem se multiplatformní vývojové nástroje snaží vypořádat s odlišnostmi jednotlivých operačních systémů a jaká jsou úskalí jednotlivých přístupů a problémy související s vývojem.

I. TEORETICKÁ ČÁST

1 MOBILNÍ OPERAČNÍ SYSTÉMY

V současné době existuje celá řada operačních systémů na architektuře ARM:

- Apple iOS
- Android od Google
- BlackBerry
- Windows Phone
- Symbian
- Web OS
- Bada
- Tizen
- Firefox OS

Z hlediska prodejnosti současnému trhu dominuje Android se 74.4% podílem trhu. Následován dlouhodobě iOS a na třetím a čtvrtém místě jsou to telefony s operačním systémem BlackBerry a Windows Phone.

Tabulka 1: Prodeje mobilních telefonů podle operačního systému. [1]

Operační systém	1. Čtvrtletí 2013 tis. kusů	1. čtvrtletí 2013 Podíl na trhu (%)	1. čtvrtletí 2013 tis. kusů	1. čtvrtletí 2012 Podíl na trhu (%)
Android	156,186.0	74.4	83,684.4	56.9
iOS	38,331.8	18.2	33,120.5	22.5
BlackBerry	6,218.6	3.0	9,939.3	6.8
Microsoft	5,989.2	2.9	2,722.5	1.9
Bada	1,370.8	0.7	3,843.7	2.6
Symbian	1,349.4	0.6	12,466.9	8.5
Others	600.3	0.3	1,242.9	0.8
Total	210,046.1	100.0	147,020.2	100.0

1.1 Android

Společnost byla založena v roce 2003 v kalifornii firmou Android Inc. V roce 2005 se firma stala dceřinou společností Google. Pod vedením Andyho Rubina byl vyvinut v roce 2007 Android OS, operační systém založený na Linuxovém jádře. 5. listopadu 2007 bylo založeno uskupení Open Handset Alliance. Vzniklo konsorcium tvořené dosavadními výrobci mobilních telefonů, navigací a Googlu, které si dalo za cíl vytvořit otevřený standard pro mobilní zařízení. Byl vydán Android SDK pro vývoj a tvorbu aplikací pod

licencí open source. První telefon s operačním systémem Google Android byl uveden v říjnu 2008 a byl to HTC G1. [2]

1.1.1 Vývojová prostředí

1.1.1.1 Android SDK

Android SDK (Software development kit) obsahuje nezbytné nástroje pro vytvoření a kompilaci aplikace. Většina z těchto nástrojů se ovládá z příkazového řádku.

Android SDK zároveň obsahuje emulátor mobilního telefonu nebo tabletu, na kterém je možné si aplikaci vyzkoušet. Pomocí nástroje android AVD Manager (Android Virtual Device Manager) lze definovat parametry vytvářeného zařízení.

Dále android SDK obsahuje tzv ADB (Android Debug Bridge), který k virtuálnímu zařízení připojí vytvořenou aplikaci. [3]

1.1.1.2 Android Development Tools

Pro vývoj aplikací poskytuje Google zdarma Android Developer Tools (ADT). Ty obsahují vývojové prostředí Eclipse. ADT je tedy set komponent, nebo doplňků, které rozšiřují standardní vývojové prostředí Eclipse o možnost tvorby aplikací pro Android.

Oficiálním jazykem pro tvorbu aplikace je Java. Zdrojové soubory jazyka Java jsou konvertovány do tříd standardním Java kompilátorem. Všechny tyto třídy jsou zabaleny pomocí nástroje zvaného dx do jednoho komprimovaného souboru (**.dex**). Tento soubor je spolu se zdroji, jako například obrázky, jazykovými soubory a jinými soubory, použit ke tvorbě souboru **.apk**, který se používá pro instalaci aplikace do zařízení.

1.1.2 Uživatelské rozhraní a struktura aplikace

Aplikace na androidu se skládá z komponent. Existují 4 různé typy komponent

1.1.2.1 Activity - Aktivita

Aktivita reprezentuje jednu obrazovku s uživatelským rozhraním pro ovládání části aplikace. Například emailový klient obsahuje aktivitu pro seznam všech emailů. Druhá aktivita zobrazuje email zobrazený pro čtení, třetí aktivita slouží k psaní emailu. Jednotlivé

aktivity dovedou existovat samostatně. Aktivita spolu nemusí spolupracovat pouze v rámci jedné aplikace, ale mohou komunikovat i mezi aplikacemi navzájem. Je tedy možné spustit aktivitu na psaní emailu z aplikace fotoaparátu, pokud mu to aktivita pro psaní emailu dovolí.

1.1.2.2 Services - Služby

Služba je komponenta, která zajišťuje běh aplikace na pozadí. Nejedná se o uživatelské rozhraní. Služba může být například hudba hrající na pozadí při používání jiné aplikace, například internetového prohlížeče. Neblokuje uživatele při interakci s aktivitami. Aktivita spouští Službu.

1.1.2.3 Content Providers - Poskytovatelé obsahu

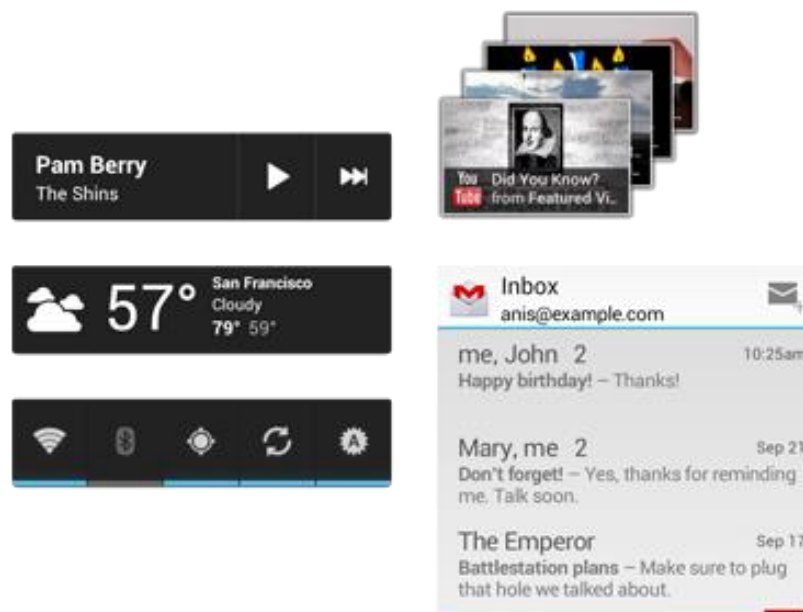
Komponenta zajišťuje správu a sdílení dat v rámci aplikace. Data je možné ukládat pomocí SQLite databáze na web nebo do lokálního úložiště. Pokud to Content Provider povolí, k datům mohou přistupovat i jiné aplikace. Příkladem je Provider, který poskytuje data o uživateli v telefonním seznamu. Mohou k němu přistupovat i jiné aplikace a využít tato data k vizualizaci.

1.1.2.4 Broadcast receivers

Komponenta reagující na systémové požadavky typu *broadcast*. Takovým požadavkem může být například zhasnutí displeje. *Broadcast* požadavek může spustit samotná aplikace jako upozornění v notifikační liště o skončené činnosti a podobně. [3] [5]

1.1.3 Specifika systému Android

Mezi specifika je potřeba řadit prvky operačního systému, které nelze najít u konkurenčních platforem. V případě Androidu jsou to především Widgety. Jedná se o vizuální prvky, které lze umístit na hlavní obrazovku telefonu. Tyto prvky mají schopnost nejen zobrazovat, ale i ovládat aplikaci, ke které jsou přidružené. [4]



Obrázek 1: Android Widget. [10]

Mohou mít různou velikost a tvar. Mohou běžet samostatně, nemusí patřit k aplikaci. Některé lze umístit na zamykací obrazovku. Mezi další specifika patří notificační centrum. Notificační centrum zobrazuje aktualizace a informace o změnách v systému a v aplikacích. [4]

1.2 Apple iOS

V roce 2007 byl firmou Apple představen světu telefon iPhone s operačním systémem iPhone OS. Z počátku na něj nebylo možné psát nativní aplikace a tak první aplikace byly webové. Měly velmi omezený přístup k funkcím a možnostem systému. Až s příchodem tzv. AppStore (obchod s aplikacemi) přibýlo vývojové prostředí Xcode, ve kterém pomocí jazyka Objective C, bylo možné tvořit plnohodnotné mobilní aplikace. Od čtvrté verze systému došlo ke vzniku softwarové platformy iOS, která byla instalovaná nejen na telefony iPhone, ale také na iPad a iPod Touch. IOS je odlehčenou verzí systému Mac OS X. Tento systém nepodporuje všechny funkce plnohodnotného systému iOS. Přidává podporu pro dotykové ovládání.

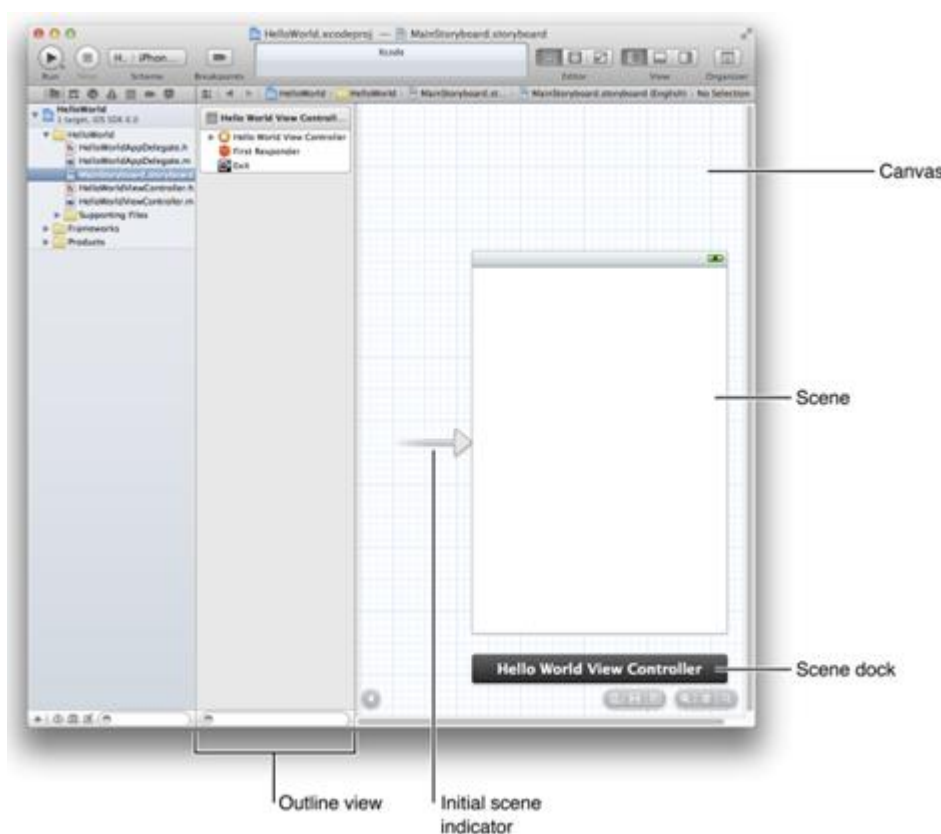
1.2.1 Vývojová prostředí

Podmínkou pro vývoj aplikace pro mobilní platformu iOS je vlastnit počítač od firmy Apple. Doporučené vývojové nástroje jsou Xcode a oficiální jazyk pro tvorbu nativních aplikací je Objective C.

1.2.2 Uživatelské rozhraní

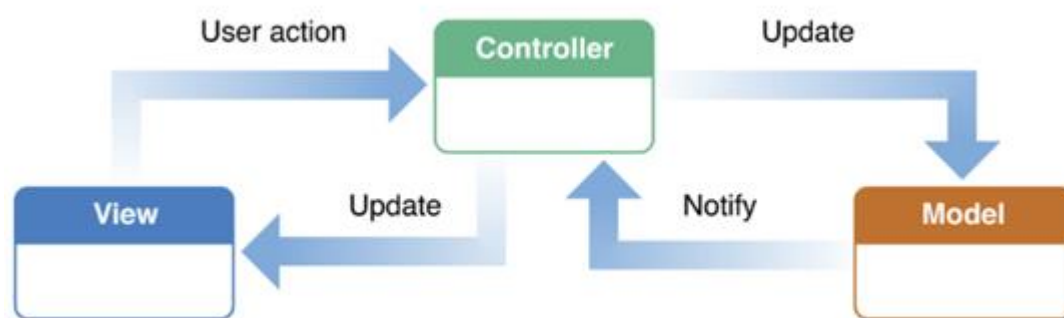
Grafické rozhraní aplikace tvoří View-Controller, který je složen z tzv. **Scény (Scene)** a **Přechodu (Seque)**. Scéna zobrazuje a vizualizuje obsah aplikace, přechod je animace prováděná při změně scén.

Oblast, která se nachází pod scénou, se nazývá **dok scény** (orig. Scene Dock) a má pevnou polohu. Může obsahovat text, tlačítka, ikony atd.



Obrázek 2: Vývojové prostředí Xcode a popis základních prvků aplikace. [6]

Návrhový vzor doporučený Apple pro vývoj aplikací je Model–View–Controller (MVC). Tento návrhový vzor definuje, jakou roli budou mít objekty v aplikaci. Jednotlivé objekty jsou od sebe odděleny abstraktní hranicí, přes kterou spolu zároveň i komunikují.



Obrázek 3: Návrhový vzor MVC. [6]

Existuje oficiální průvodce programováním iOS, který detailně popisuje jak se vypořádat s tvorbou grafického rozhraní aplikace. Je to seznam nejrozličnějších rad a technik jak vytvářet aplikace pro zařízení běžící na tomto systému. [6] [7]

Kvalitní design aplikace podle Applu by měl splňovat tyto základní funkce:

Estetická integrita – aplikace by se měla chovat tak, jak se od ní očekává.

Konzistence – uživatel je zvyklý na chování systému. Aplikace by měla toto chování respektovat a následovat. Pokud je uživatel zvyklý na to, že tlačítko zpět je v systému zobrazeno vlevo nahoře, aplikace by ho měla mít na stejném místě.

Přímá manipulace – uživatel by měl dobře znát ovládací prvek, který právě využívá a ten by měl dělat právě to, co uživatel očekává.

Zpětná vazba – ovládací prvky by měly poskytovat zpětnou vazbu uživateli. Ten by měl vědět, že právě stiskl dané tlačítko. Zpětnou vazbu lze zajistit například různými animacemi.

Metafora – aplikace by se měla snažit napodobovat svůj reálný protějšek ve skutečném světě. Příkladem může být aplikace Notes – poznámkový blok nebo kalendář, který se snaží napodobovat kalendář z reálného světa. Dalším příkladem je snaha napodobit otočení listu v aplikaci pro čtení elektronických knih. [7]

1.2.3 Specifika systému iOS

Většina těchto funkcí patří mezi základní zásady vývoje kvalitního uživatelského rozhraní, avšak z hlediska multiplatformního vývoje se dostáváme do rozporu například v bodech **konzistence** a **metafory**. Zatímco Apple doporučuje napodobovat v aplikacích vzhled reálných předmětů, Microsoft u Windows Phone upřednostňuje jednoduché minimalistické grafické rozhraní zapadající do vzhledu operačního systému.



Obrázek 4: Aplikace FM rádio ve stylu iOS a aplikace rádio ve stylu WP7. [8][9]

V případě konzistence je důležité u iOS zařízení umístit tlačítko „zpět“ do levého rohu, u Androidu nebo Windows Phone je tlačítko zpět hardwarovým prvkem. Z hlediska vzhledu je při vývoji multiplatformní aplikace důležité chytře použít ovládací prvky a přizpůsobit grafické prostředí tak, aby co nejvíce splňovalo doporučení výrobců.

1.3 Windows Phone

Windows Phone je označení pro operační systémy firmy Microsoft. Systém Windows Phone 7 byl na trh uveden 21. 10. 2010 a stejně jako Windows Mobile vychází z jádra předchozího Windows CE. Windows Phone 8 byl představen jako nástupce Windows Phone 7 a je vychází z upraveného jádra Windows NT. Uživatelské rozhraní je

přizpůsobeno ovládání prsty a úhlopříčce displeje mobilního telefonu. Pro větší úhlopříčky displeje je tu Windows RT.

1.3.1 Vývojové prostředí

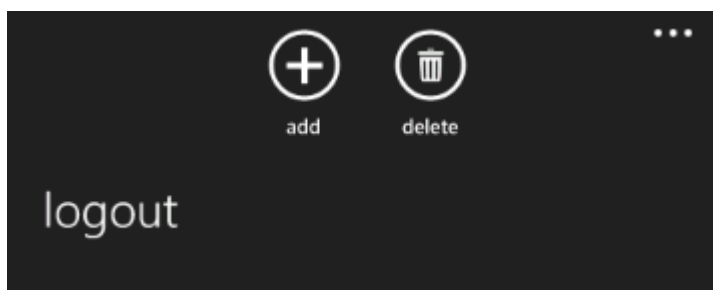
Microsoftem doporučený nástroj pro vývoj software na platformu Windows Phone je Microsoft Visual Studio. Software je zdarma v omezené verzi zvané Visual Studio Express v balíčku vývojových nástrojů Windows Phone Developer Tools. Vytvořit aplikace lze pouze na operačním systému Windows. Dále je potřeba vytvořit vývojářský účet, který je placený. Aby bylo možné vyvíjet a testovat na reálném zařízení, je potřeba toto zařízení odemknout pro vývoj. [11]

1.3.2 Uživatelské rozhraní

Uživatelské rozhraní Windows Phone aplikací je popsáno na webu Design Library for Windows Phone. Většina aplikací dodržuje návrhový vzor MVVM (Model View View-Model). [8]

1.3.3 Specifika systému

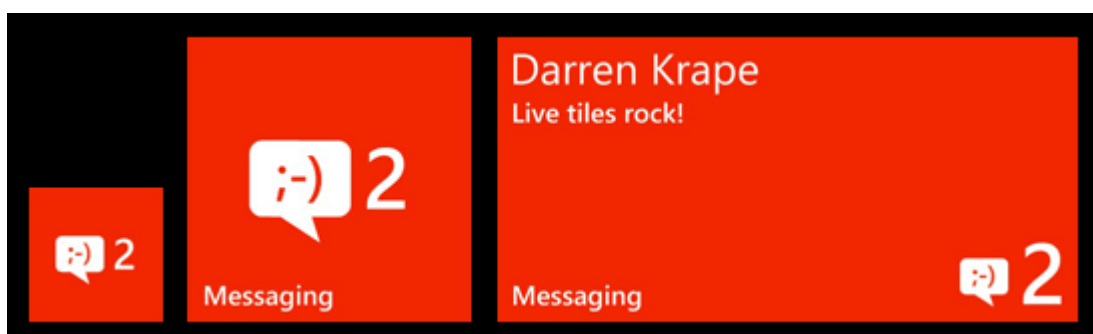
Aplikace z hlediska struktury vypadá velice podobně jako webová stránka. ApplicationBar je specifický zobrazovací prvek, který je tvořen lištou s tlačítky a položkami menu. ApplicationBar může obsahovat maximálně 4 tlačítka a doporučuje se použít maximálně pět položek menu. [12]



Obrázek 5: ApplicationBar se dvěma tlačítky a jednou položkou menu (zdroj vlastní).

Další specifickou vlastností Windows Phone je tzv. živá dlaždice. Jedná se o způsob, jakým může aplikace sdělit, že uvnitř došlo ke změně obsahu. Zobrazuje náhledy obsahu, který je většinou uvnitř aplikace popsán detailněji. Živá dlaždice je schopna zobrazit pouze text, číslo a obrázek.

Živá dlaždice nemusí být pouze spouštěčem aplikace, ale může sloužit jako odkaz na libovolnou stránku aplikace. Pro aplikaci zobrazující informace o počasí může dlaždice znamenat zkratku pro počasí v daném městě a podobně. [14]



Obrázek 6: Příklad živé dlaždice – Live tile v různých velikostech.[13]

1.3.4 Design

Na webu Microsoftu existuje několik článků zabývajících se návrhem aplikací, přičemž, podobně jako u Apple, se snaží doporučovat vývojáři, jak by měl při návrhu aplikace uvažovat. Vždy je kladen důraz na to, aby při tvorbě uživatelského rozhraní šel do popředí obsah nad grafickou stránkou aplikace, aby ovládací prvky aplikace byly vždy na dosah prstu a aby aplikace dobře zapadla do prostředí, do kterého je cílena (velká tlačítka pro aplikace typu navigace do auta). Většina těchto doporučení patří mezi obecné zásady vývoje kvalitního uživatelského rozhraní aplikace. Pokud se snažíme dodržovat všechny rady a doporučení, výsledkem je aplikace, u které je na první pohled zřejmé, že se jedná o aplikaci pro Windows Phone. [8]

2 NATIVNÍ APLIKACE

Nativní aplikace využívají ke své tvorbě vývojové prostředí a jazyk určený výrobcem. Nástroje pro vývoj nativní aplikace pro jednotlivé operační systémy byly představeny v předchozí kapitole.

Výhody vývoje nativní aplikace oproti multiplatformnímu vývoji jsou především:

- Výkon – webová nebo hybridní aplikace nedokáže běžet tak rychle, jako aplikace nativní
- Dostupná dokumentace a přístup k hardwarovým funkcím telefonu
- Nativní vzhled a chování aplikace
- Při vývoji nativní aplikace lze snadno splnit všechny podmínky pro přijetí aplikace do obchodu s aplikacemi.

Nevýhody

- Multiplatformní aplikace vytvořená pomocí nativního kódu vyžaduje oficiální vývojové prostředí pro každou platformu.
- Vývojová prostředí, metody vývoje, návrhové vzory a zásady správného vývoje jsou často natolik odlišné, že je občas potřeba zahrnout několik různých vývojových týmů.
- Je potřeba psát, testovat a udržovat více kódu, což může mít za následek zanesení více chyb.

3 MULTIPLATFORMNÍ APLIKACE

Mobilní platforma označuje kombinaci hardwaru (chytrého mobilního telefonu) a mobilního operačního systému.

Multiplatformní programování je programování s využitím SW prostředků pro rychlý a efektivní vývoj aplikací, které je možné překládat nebo přímo spouštět na různých platformách. Výsledkem takového multiplatformního programování bývá software, který je schopen běžet na více počítačových platformách. Při vývoji pro mobilní platformy se však častěji setkáváme s pojmem aplikace.

Hardwarová platforma dnešních mobilních zařízení je prakticky sjednocena na architekturu typu ARM. Operačních systémů schopných běžet na této platformě je však několik druhů.

Primárním cílem výrobců komerčních aplikací je prodat svoji aplikaci co největšímu množství lidí. Z toho plyne snaha o podporu co největšího počtu mobilních operačních systémů. Se vzrůstajícím počtem těchto softwarových platforem se začala objevovat snaha výrobců komerčních aplikací sjednotit vývoj. [14]

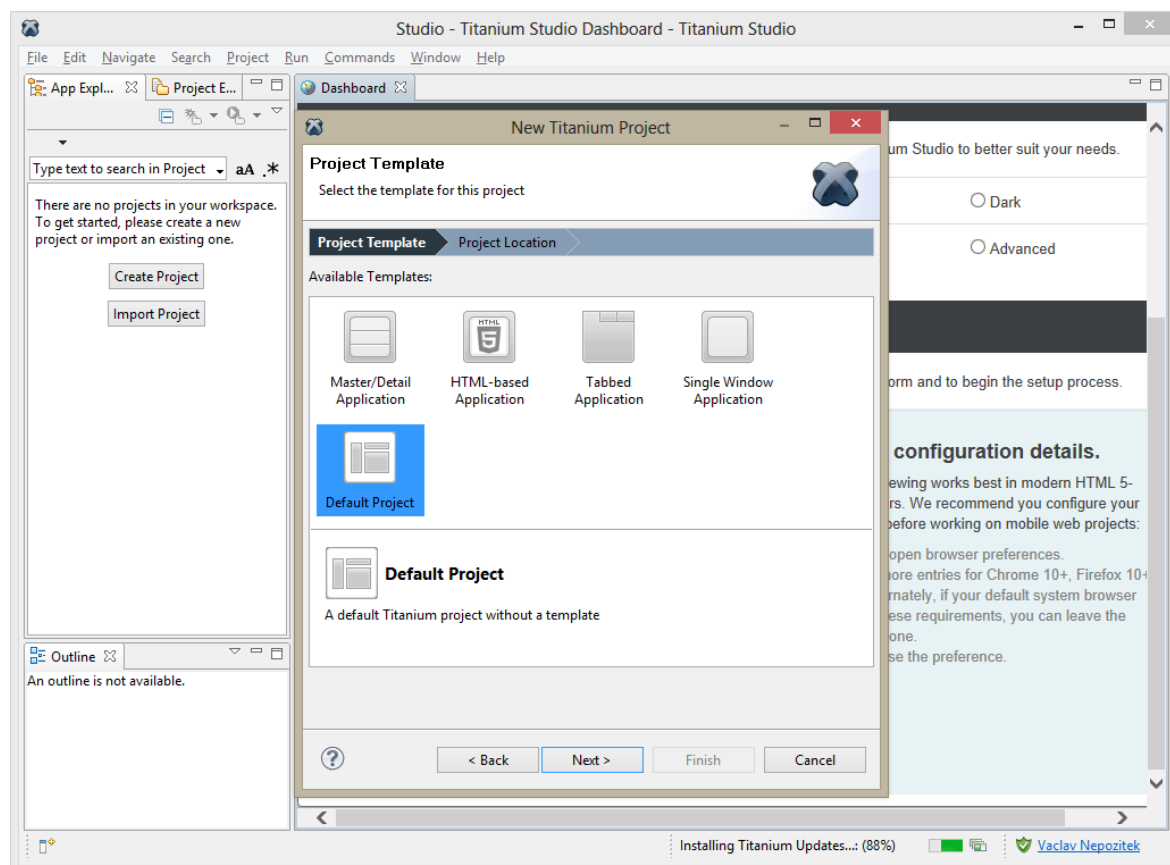
V současné době můžeme rozdělit vývoj multiplatformních aplikací na tři hlavní podkategorie lišící se především přístupem k vývoji.

3.1 Multiplatformní kompilátory

Nejvyšší výkon v dnešní době poskytují multiplatformní aplikace vyvinuté s využitím multiplatformních kompilátorů. Lze použít kód psaný na nejnižší úrovni pomocí jazyka C, který je převeden do nativního kódu. Výhoda spočívá v psaní jednoho kódu, který je překompilován tak, aby podporoval API jednotlivých platforem.

3.1.1 Appcelerator – Titanium

Firma založená v roce 2006 v Atlantě v USA propaguje strategii tzv. "Continuous Mobile Innovation" což je nový přístup pro vytváření, dodávání a analyzování vytvořených aplikací. Pomocí vývojových nástrojů **Titanium SDK** a vývojového studia **Titanium Studio** lze vytvářet aplikace tzv. "Design driven development". Cílem tohoto vývojového procesu je vytvořit aplikaci v krátkých iteracích. Technologie používané při vývoji jsou JavaScript, CSS, XML.



Obrázek 7: Vývojové prostředí Appcelerator (zdroj vlastní).

Kód napsaný v tomto jazyce je překompilován:

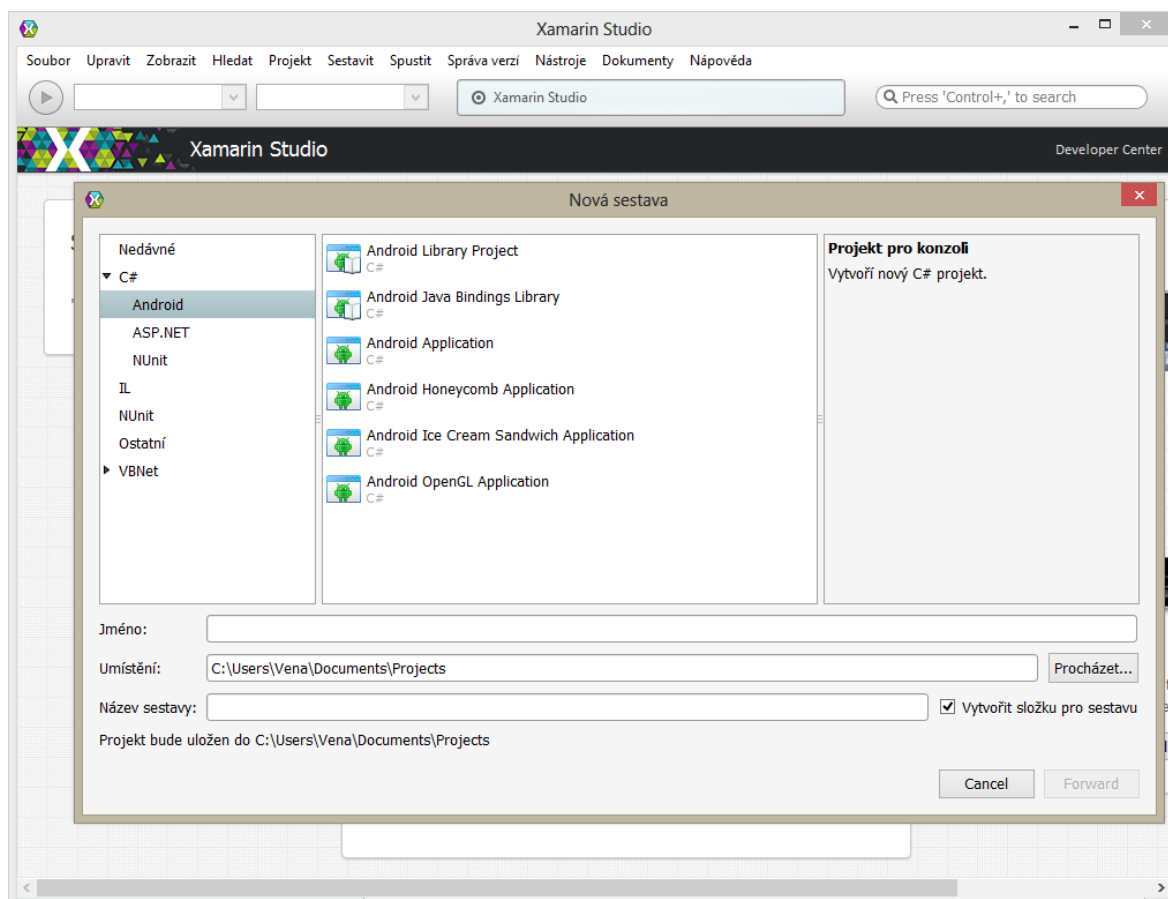
- do nativního kódu – nutností je tedy instalace nativního SDK pro danou mobilní platformu.
- do webové aplikace

Podporované mobilní operační systémy jsou iOS, Android, BlackBerry a Tizen. [17]

3.1.2 Xamarin

Xamarin studio je vývojové prostředí pro vývoj multiplatformních aplikací pomocí C#. Studio je propojeno s Android a iOS SDK, je tedy možné testovat aplikace jak v emulátoru, tak na reálném zařízení.

Jedná se o moderní vývojové prostředí podporující týmovou spolupráci, nápovědu při psaní kódu a ladící nástroje.



Obrázek 8: Xamarin studio – Vytváření nového projektu Android aplikace (zdroj vlastní).

Jedna z výhod, které Xamarin poskytuje je obchod s hotovými komponentami tzv. Xamarin Component Store. Ten obsahuje například nástroje pro připojení ke službám Windows Azure, hotové komponenty pro tvorbu grafů a podobně.

Výhodou může být i možnost využít Microsoft Visual Studio při vývoji aplikace. Toho lze využít pouze v placené verzi. Xamarin je zdarma jen ve verzi Starter Edition, která nabízí pouze základní Xamarin Studio. [16]

Výhody

- Možnost vyvíjet aplikace v jazyce, který je vývojáři blízký
- Aplikaci lze nahrát do obchodu s aplikacemi
- Možnost využít hardware telefonu

- Výkon aplikace a její vzhled jsou velice podobné nativním aplikacím

Nevýhody

- Nejsou podporovány všechny platformy
- Některé vývojové nástroje nejsou zdarma
- Potřeba instalace nativního SDK

3.2 Webové aplikace

Všechny moderní operační systémy dnes nabízejí velice pokročilé internetové prohlížeče, které jsou schopny pracovat efektivně s HTML 5. Pomocí JavaScriptu lze přistupovat k hardwarovým funkcím telefonu jako například k informaci o poloze, fotoaparátu, gyroskopu, obrázkům a hudbě, úložišti telefonu a tedy pracovat bez připojení k síti.

Stinnou stránkou takového vývoje je nemožnost publikovat webové aplikace v obchodech s aplikacemi. Aplikace však mohou vydělávat jiným způsobem, například reklamou umístěnou v aplikaci, nebo na službě, kterou skrz webovou aplikaci ovládáme.

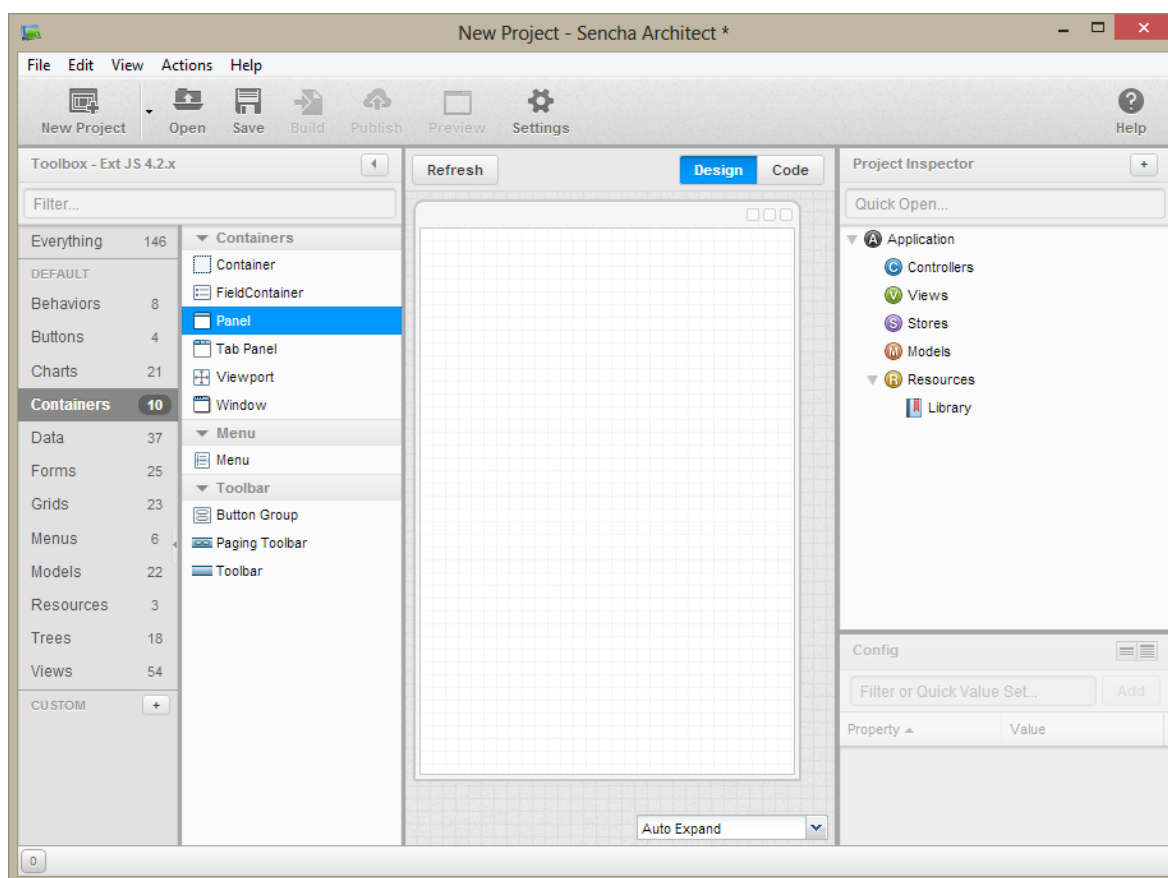
3.2.1 Sencha Touch

K vývoji pomocí softwaru Sencha Touch je potřeba mít elementární znalosti HTML, CSS, JavaScriptu a znát obecná pravidla pro vývoj webu. Pomocí tohoto nástroje lze vytvářet aplikace pro Android, iOS, BlackBerry. Jediné co stačí, je nainstalovat Sencha Touch na počítač s Windows, OS X, nebo Linux a vytvořit lokálně běžící web server. Sencha poskytuje kvalitní dokumentaci a spoustu příkladů jak začít s vývojem. Návrhový vzor používaný pro vývoj webu je klasický MVC. [19]

Pro vývoj aplikace pomocí nástroje Sencha Touch 2 je potřeba:

- Sencha Touch 2 SDK
- SDK Tools
- Webserver, běžící na lokálním počítači

- Internetový prohlížeč



Obrázek 9: Vývojové nástroje Sencha Touch (zdroj vlastní).

Výhody řešení webových aplikací

- Vývojové nástroje jsou dostupné zdarma, je jich dostatek a ladit aplikaci lze téměř v každém prohlížeči.
- Dokumentace k použité technologii HTML 5 a JavaScript je dostatek.
- HTML 5 se stává standardem, bez kterého moderní prohlížeče nemohou existovat. Má schopnost běžet na jakémkoliv zařízení s webovým prohlížečem podporujícím standardy HTML5.
- Webové aplikace lze snadněji převést na aplikace hybridní.

Nevýhody

- Schopnost využít hardware telefonu je značně omezená. Lze využít informace o geografické poloze a natočení displeje, problémem je kalendář, telefonní seznam, fotoaparát a podobně.
- Aplikace nelze distribuovat přes obchody s aplikacemi.
- Problém dosáhnout nativního vzhledu a vysokého výkonu při náročnějších operacích. Rychlost webové aplikace je závislá na schopnosti prohlížeče pracovat s HTML5.

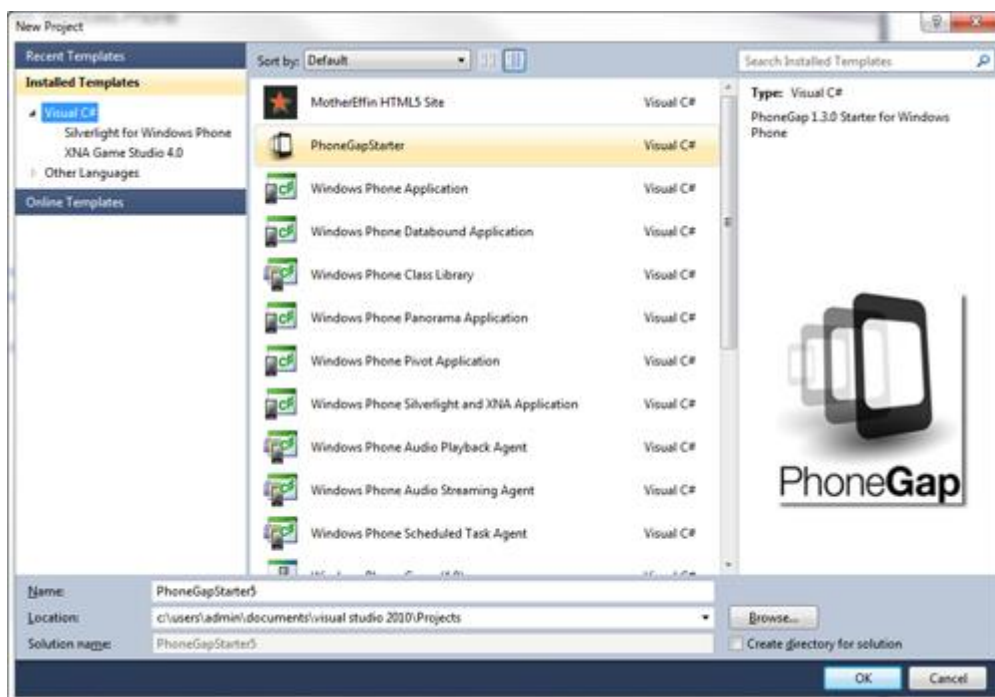
3.3 Hybridní aplikace

Hybridní aplikace je většinou psána pomocí HTML 5 a JavaScriptu. Při kompilaci je zabalena do nativního kódu a výsledkem je balíček nerozeznatelný od nativní aplikace. Výhodou takového řešení je využití jak možnosti psát aplikace pomocí HTML 5, tak využití hardwarových funkcí telefonu. K vývoji aplikace je však potřeba vývojových nástrojů pro danou platformu.

3.3.1 PhoneGap

PhoneGap je framework, který slouží jako prostředník mezi nativním a webovým vývojovým prostředím aplikace. Samotná aplikace je vyvíjena pomocí HTML5, CSS a JavaScriptu a potom použitím nástroje PhoneGap je konvertována do nativního kódu pro zvolenou mobilní platformu.

V současné době lze PhoneGap využít pro vývoj aplikace pro Android, iOS, BlackBerry OS, WebOS, Symbian a Windows Phone. [15] [18]



Obrázek 10: PhoneGap projekt integrovaný do Visual Studia (zdroj vlastní).

Multiplatformní aplikace je tak schopna se dostat do obchodu s aplikacemi. Během návrhu uživatelského prostředí je nutné dodržovat doporučení výrobců a zároveň respektovat odlišnosti v ovládání jednotlivých mobilních operačních systémů.

Při rozhodování, kterou strategii vývoje zvolit je proto nutné se zamyslet nad cílem aplikace a nad požadavky na aplikaci.

Výhody hybridních aplikací

- Vývojové nástroje jsou dostupné zdarma, je jich dostatek a ladit aplikaci lze téměř v každém internetovém prohlížeči.
- Dokumentace k použité technologii HTML 5 a JavaScript je dostatek
- Možnost přistupovat k hardware telefonu

Nevýhody

- Nutnost instalovat nativní SDK
- Přetrvávají některá omezení týkající se specifik operačního systému (AppBar na WP7)

- Nejsou podporovány všechny platformy

3.4 Uživatelské rozhraní a responzivní web design

Jak bylo zmíněno v předchozích kapitolách, Apple, Android a Windows Phone kladou důraz na kvalitu uživatelského rozhraní v aplikaci. Displeje současných zařízení se mohou lišit hustotou jednotlivých pixelů, která je závislá na velikosti úhlopříčky displeje, rozlišením obrazovky, poměrem stran, maticí subpixelů (RGB, RBGB), barevným podáním a další. Framework pro vývoj nativní aplikace je většinou na tento problém připraven. Multiplatformní, zejména webové a hybridní aplikace psané pomocí HTML5, JavaScriptu a kaskádových stylů, mohou mít s touto rozmanitostí problém.

Snahu vyřešit tento problém má tzv. responzivní web design. Jedná se o způsob stylování HTML dokumentu, které zaručí, že zobrazení stránky bude optimalizováno pro všechny druhy nejrůznějších zařízení (mobilní telefony, notebooky, tablety atd.).

Responzivní web design má tři základní úrovně:

- Flexibilní struktura – rozměry jednotlivých prvků nejsou zadávány v pixelech ale v procentech
- Flexibilní obrázky
- Media Queries – Především díky vlastnosti Media Queries, která je zahrnuta ve specifikaci CSS3, lze rozpoznat vlastnosti zařízení, na kterém je stránka prohlížena a přizpůsobit tak samotnou stránku a její obsah.

Příklad:

Media Query na detekci šířky vepsaná přímo do odkazu na kaskádový styl

```
<link rel="stylesheet" href="mobile.css" media="screen and (max-width: 480px)">
```

Media Query vepsaná do stylu

```
@media only screen and (max-device-width : 480px) { ... }
```

4 PŘÍPADOVÉ STUDIE

Při rozhodování jakou technologii pro vývoj zvolit je nutné znát

- Výhody a nevýhody jednotlivých přístupů.
- Přesná specifikace vyvíjené aplikace.
 - K čemu bude aplikace určena?
 - Bude potřeba využít hardware telefonu?
 - Jak budou probíhat aktualizace aplikace?
 - Bude potřeba nativní vzhled a chování?
 - Prodáváme aplikaci nebo službu?
- Cílové platformy.

4.1.1 Příklad č. 1

Zpravodajský server Ihned.cz žije především z návštěvnosti svých stránek, na jejichž základě prodává reklamu. Lidé sice mohou prohlížet web aplikace přes mobilní telefon, ale je to pro ně nepohodlné, protože grafické rozhraní webu není přizpůsobené na menší úhlopříčky displejů. Reklama není vidět, protože většina mobilních prohlížečů nepodporuje technologii Flash a objem dat při načítání webové stránky je velký, což způsobuje pro uživatele rychlejší dosažení limitu pro přenos dat. Potřebuje proto aplikaci pro mobilní telefony.

Cíle:

Aplikace musí podporovat co největší počet zařízení. Je potřeba se zaměřit nejen na rozdíly v jednotlivých platformách, ale také na rozdílné úhlopříčky a rozlišení displejů.

Aktualizace aplikace bude potřeba provádět okamžitě. Pro zpravodajský server je důležitá rychlá reakce na aktuální dění, přidávání nových specializovaných rubrik pro volby, speciální sportovní události a podobně.

Řešení:

Typu zadání nejvíce odpovídá tvorba webové aplikace. Aktuální verze aplikace bude dostupná okamžitě na webové adrese serveru. V jeden okamžik bude aplikace vypadat

stejně na každém zařízení s prohlížečem podporujícím HTML 5. Z hardwaru zařízení je potřeba využít pouze modul pro získání informací o poloze.

4.1.2 Příklad 2

Firma Adidas plánuje vytvořit aplikaci na mobilní platformy, která bude měřit sportovní výkony na základě GPS a zároveň bude mít reprezentační a reklamní funkci na firmu Adidas.

Cíle:

Aplikace bude zaznamenávat uběhnutou trasu do mapy a zároveň bude měřit rychlost, nadmořskou výšku a uběhnutou vzdálenost. Aplikace bude zaznamenané informace odesílat na předem známou službu. Záznamy je možné sdílet s ostatními uživateli, ke každému záznamu je možno přidat fotografii z trasy.

Řešení:

Splnění zadání nejvíce vyhovuje nativní aplikace, nebo hybridní aplikace programovaná v jednom jazyce a překompilovaná na více platform. U aplikace je kladen důraz na výkon. Bude obsahovat efektní animace a na pozadí poběží několik výpočtů zároveň. Aplikace bude využívat všech možností operačního systému. Pro platformu Android bude vytvořen Widget na hlavní obrazovku. V případě Windows Phone aplikace je vytvořena živá dlaždice zobrazující souhrnnou statistiku za poslední měsíc. Na iPhone naopak existuje několik externích doplňků, které lze k telefonu připojit. Vzhledem k variabilitě této aplikace a snaze využít z každé platformy maximum, bude nejvhodnější zvolit cestu vývoje nativní aplikace. Priorita vývoje bude záležet na rozšířenosti jednotlivých platform

4.1.3 Příklad 3

Firma zabývající se vývojem webových stránek, plánuje svému zákazníkovi nabídnout mobilní aplikaci pro jeho internetový obchod.

Cíle:

Aplikace bude schopna pomocí fotoaparátu číst čárové kódy a bude nabízet stejný sortiment zboží jako internetový obchod.

Řešení:

Dle zadání je možné začít psát nativní aplikaci pro každou platformu. Vhodnější však bude použití multiplatformní aplikace, která bude psaná pomocí HTML 5 a JavaScriptu, protože firma disponuje programátory se zkušenostmi v této oblasti. Bude snadné využít funkci fotoaparátu a geolokačních služeb telefonu. Získáme rychlost na úrovni webového prohlížeče, která je pro aplikaci tohoto typu dostačující. Vzhledem k existenci webových stránek je možné využít stávajícího kódu.

II. PRAKTICKÁ ČÁST

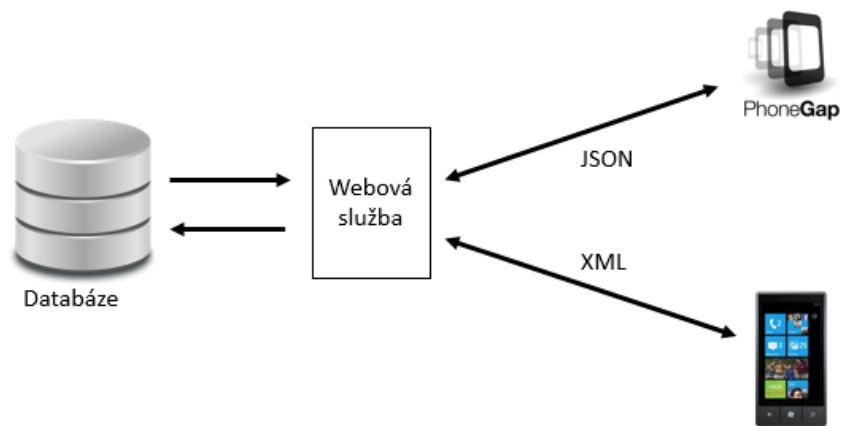
5 VÝVOJ APLIKACE

Aplikace Fuel Manager slouží k ukládání záznamů o tankování paliva a vytváří statistiku tankování.

5.1 Návrh aplikace

Data jsou uložena v databázi na vzdáleném úložišti. Ta komunikuje skrze WCF službu.

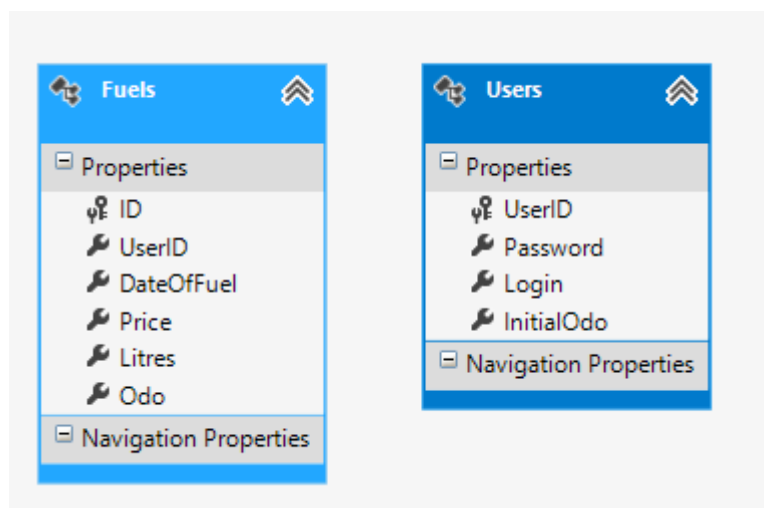
Tato webová služba poskytuje data jak ve formátu XML tak ve formátu JSON.



Obrázek 11: Komunikace (zdroj vlastní).

5.1.1 Databáze

Databáze je uložena na vzdáleném serveru hostovaném na Microsoft Azure. Skládá se ze dvou jednoduchých tabulek. Tabulka uživatelů *Users* a tabulka záznamů o tankování paliva *Fuels*.



Obrázek 12: Schéma databáze (zdroj vlastní).

Databáze obsahuje uložené procedury pro získávání záznamů v tabulce pro daného uživatele v daném časovém rozmezí.

Tělo uložené procedury pro získání záznamů o tankování:

```
Select Top 10
    Id,
    Userid,
    DateOfFuel,
    Price,
    Litres,
    Odo,
    ROUND((Price * Litres),1) as TotalPrice --výpočet celkové ceny
From Fuels
Where Fuels.UserId = @userid
and Fuels.DateOfFuel >= @dateFrom
and Fuels.DateOfFuel <= @dateUntil
order by Id desc
```

Tělo uložené procedury pro získání statistiky o tankování:

```
select
    UserId,
    round(avg(Price),2) as AveragePrice,
    --celková útrata včetně posledního tankování
    sum(Price * Litres) as TotalPrice,
    --cena spočítána: Průměrná cena * počet litrů - poslední celková platba za tankování
    ((avg(Price) * sum(Litres))-
    ((select top 1 Price
    from Fuels
    where Fuels.UserId = @userid order by Fuels.DateOfFuel desc ) *
    (select top 1 Litres
    from Fuels
    where Fuels.UserId = @userid order by Fuels.DateOfFuel desc )
    )
    ) as TotalPriceWithoutLast,
    --počet spotřebovaných litrů je celkový počet litrů bez posledního záznamu
    (sum(Litres) - (select top 1 Litres
    from Fuels where Fuels.UserId = @userid order by Fuels.DateOfFuel desc )
    ) as Litres,
    --počet kilometrů bez prvního záznamu
```

```

(select Top 1
  (Odo - (select top 1 Odo from Fuels where Fuels.UserId = @userid))
  from Fuels
  where Fuels.UserId = @userid
  order by Fuels.DateOfFuel desc
) as km
from Fuels
where Fuels.UserId = @userid
group by UserId

```

5.1.2 Webová služba

Webová služba odkazuje na knihovnu *Dip.Dal*, která je určena k manipulaci s daty v databázi. Databáze je připojena pomocí REST rozhraní. Do projektu je přidán ADO.NET Entity Data Model, který vytvoří na základě existující databáze její objektový model. Knihovna dále definuje svoje vlastní objekty typu *cUser*, *FuelRecord*, *Stats*, které jsou posílány přes webovou službu do klientských aplikací. Konfigurační soubor obsahuje nastavení pro posílání dat jak ve formátu XML, tak ve formátu JSON. [21] [22]

Příklad rozhraní webové služby u metody pro získání historie tankování:

Verze XML:

```

[OperationContract]
List<FuelRecord> GetUserFuels(int userID);

```

Verze JSON:

```

[OperationContract]
[WebInvoke(Method = "GET",
  ResponseFormat = WebMessageFormat.Json,
  RequestFormat = WebMessageFormat.Json,
  BodyStyle = WebMessageBodyStyle.WrappedRequest)]
List<FuelRecord> GetUserFuelsJson(string userID);

```

Pro posílání dat ve formátu JSON je potřeba přidat do konfiguračního souboru element:

```

<endpointBehaviors>
  <behavior name="jsonBehavior">
    <enableWebScript />
  </behavior>
</endpointBehaviors>

```

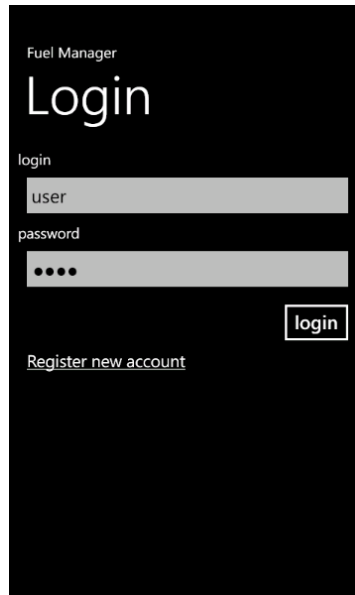
Poté je služba schopna zpracovávat data pomocí AJAX.

5.2 Nativní Aplikace

Nativní aplikace je vytvořena pro platformu Windows Phone 7.x. Je využíván návrhový vzor MVVM.

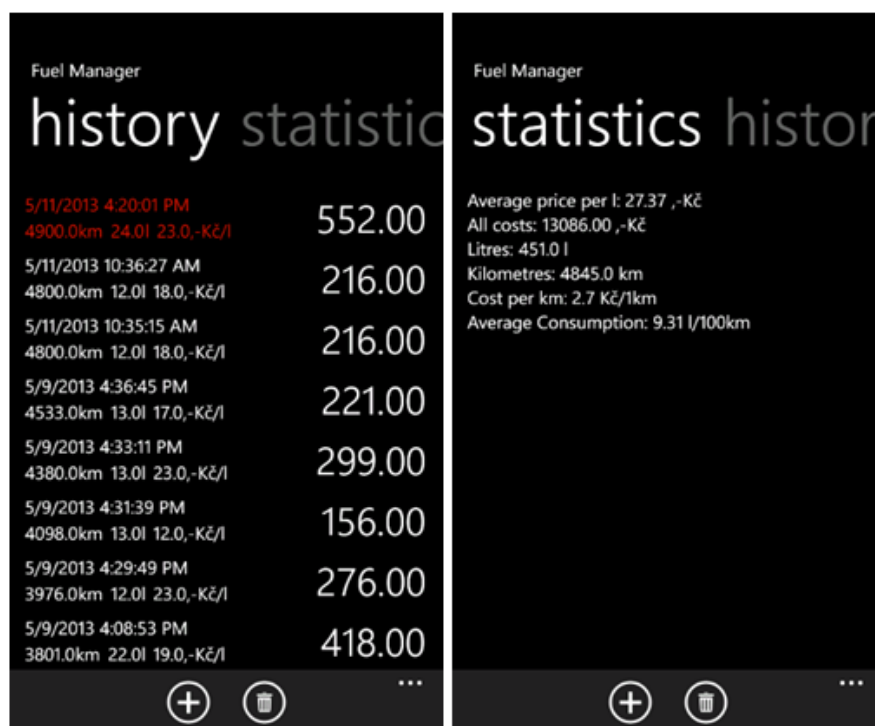
5.2.1 Popis uživatelského rozhraní

- **LoginPage** – Stránka umožňuje zadání uživatelského jména a hesla.



Obrázek 13: Přihlašovací obrazovka (zdroj vlastní).

- **CreateUserPage** – Stránka vytvořená pro registraci nového uživatele.
- **MainPage** – Tato stránka obsahuje dvě záložky. Na první záložce je zobrazena historie tankování. Na druhé záložce je zobrazena průběžná statistika. Při zobrazení stránky **MainPage** je v aplikaci k dispozici ApplicationBar, který obsahuje tlačítko pro smazání již existujícího záznamu a tlačítko pro přidání nového záznamu. Toto tlačítko přesměruje na stránku **AddFuelRecordPage**.



Obrázek 14: Obrazovka zobrazující historii tankování a statistiku (Zdroj vlastní).

- **AddFuelRecordPage** – je určena k přidání nového záznamu o tankování.

5.2.2 Logika Aplikace

Aplikace odkazuje na webovou službu vytvořením instance klienta webové služby (WCF service). Má k dispozici objekty *User*, *FuelRecord*, *Stats*. Aplikace využívá techniky *DataBinding*. Každá třída, která definuje objekt, který je potřeba zobrazovat, implementuje rozhraní *INotifyPropertyChanged* a pomocí události *OnPropertyChanged* je uživatelské rozhraní informováno o změně hodnot. [20]

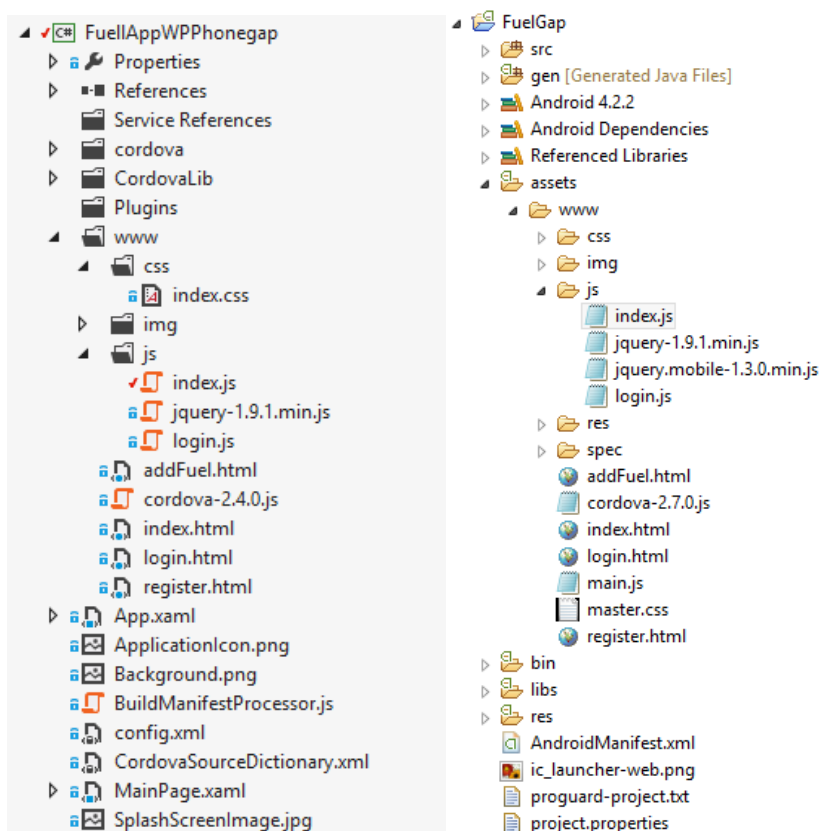
Nativní aplikace je vytvořena v Jazyce C#, uživatelské rozhraní pomocí značkovacího jazyka XAML. Aplikace je kompatibilní s Platformou Windows Phone 7, 7.1, 7.8 a lze ji překompilovat i na systém Windows Phone 8. WP7.x a WP8 jsou v jádru odlišné, protože WP7.x je založeno na jádru Windows CE zatímco WP8 na Windows NT. Dalo by se hovořit o multiplatformní aplikaci v rámci jednoho výrobce.

5.3 PhoneGap aplikace pro Windows Phone a Android

Pro vývoj pomocí nástroje PhoneGap je nutné instalovat nativní vývojové nástroje, protože je potřeba zkompileovat výsledný projekt tak, aby byl vytvořen balíček, který je možné nahrát do obchodu s aplikacemi. Dále je potřeba stáhnout ze stránek výrobce balíček PhoneGap API, který obsahuje jak knihovny pro vývoj na jednotlivé platformy.

5.3.1 Struktura projektu

Pro Windows Phone i pro Android existují již vytvořené struktury projektu a šablony, které usnadní organizaci projektu pro vývoj. Při zakládání nového projektu mobilní aplikace je přímo v nabídce Visual Studia nabídka PhoneGap. Tyto šablony obsahuje balíček ke stažení na stránkách PhoneGap.



Obrázek 15: Struktura projektu mobilní aplikace PhoneGap pro Windows Phone a Android (zdroj vlastní).

Veškerý vývoj probíhá v adresáři `www`. Ten obsahuje:

- vizuální styly – v adresáři `css`

- jednotlivé HTML stránky
- JavaScript kód – v adresáři *js*

5.3.2 Přihlášení

LOGIN	BACK NEW USER
LOGIN <input type="text"/>	LOGIN <input type="text"/>
PASSWORD <input type="text"/>	PASSWORD <input type="text"/>
<input type="button" value="LOGIN"/>	REPEATE PASSWORD <input type="text"/>
<input type="button" value="REGISTER"/>	<input type="button" value="CREATE"/>

Obrázek 16: Obrazovka pro přihlášení uživatele do aplikace a obrazovka pro registraci nového uživatele (zdroj vlastní).

Aplikace získává data prostřednictvím webové služby. K přenosu je použito JSON – (JavaScript Object Notation), kde objekty jsou převedeny na řetězec JSON. Jedná se o prostý text, v němž jsou jednotlivé objekty odděleny speciálními znaky.

Příklad volání webové služby prostřednictvím asynchronního JavaScriptu:

```
function GetStatistics(id) {  
    $.support.cors = true;  
    $.ajax({  
        type: 'GET',  
        data: '{}',  
        cache: false,  
        url: serviceAddress + 'GetFuelStatsJson?userId=' + id,  
        contentType: 'application/json',  
        dataType: 'json',  
        success: UserGetFuelSucceeded,  
        error: UserGetFuelCreateFailed  
    });  
}
```

```
});
}
```

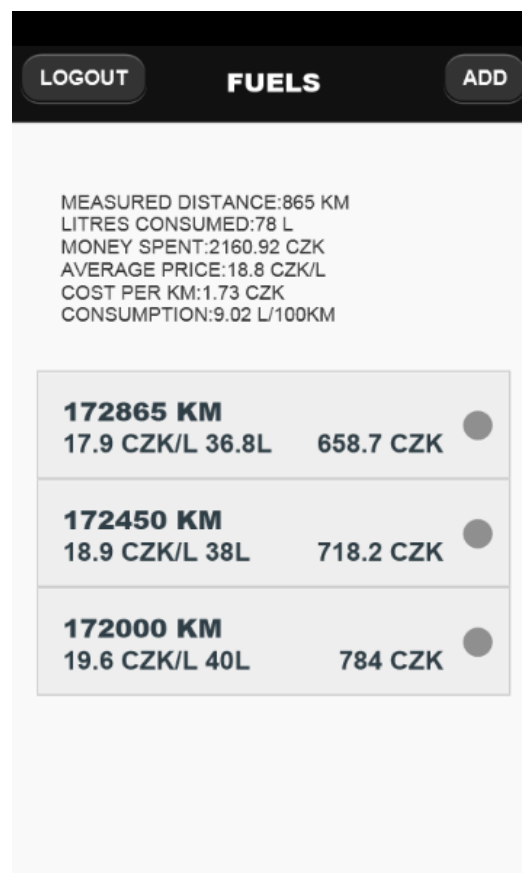
Příklad JSON řetězce pro předchozí volání:

```
{"d":{"__type":"Stats:#Dip.Dal.Entities","AveragePrice":27.370370,"Km":4845.0,"TotalLitres":451.0,"TotalPrice":13086.00,"UserId":3}}
```

Pokud je volání JSON úspěšné, vykoná se obsluha metody UserGetFuelSucceeded

```
function UserGetFuelSucceeded(data) {
    //do proměné content vytvoříme html kód do které jsou zapsána získaná data
    var content = '<p>Total kilometres:</p>' + data.d.Km + '<br />';
    content += 'Litres fueled:</p>' + data.d.TotalLitres + '<br />';
    content += 'Money spent:</p>' + data.d.TotalPrice + '<br />';
    content += 'Average price:</p>' + data.d.AveragePrice + '</p>';
    //pomocí selectoru přiřadíme content existujícímu elementu statsDiv
    $('#statsDiv').html(content);
    //zavolání metody pro získání záznamů o tankování
    GetFuelRecords(data.d.UserId);
}
```

Po získání celkové statistiky a historie tankování jsou data zobrazena v aplikaci.



Obrázek 17: Zobrazení statistiky a historie tankování (zdroj vlastní).

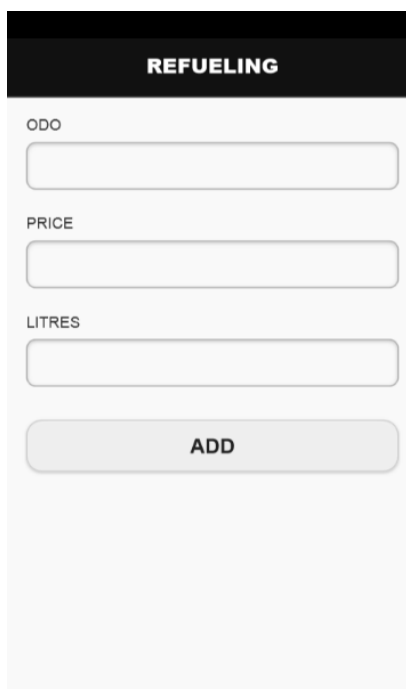
5.3.3 PhoneGap – Ukládání dat

PhoneGap API pro ukládání dat je založeno na W3C Web SQL Databázi a W3C Web Storage API.

Protože dosavadní verze knihovny nedokáže využít SQL databázi pro vývoj na Windows Phone, aplikace využívá lokální úložiště založené na Web Storage API. V aplikaci jsou ukládány informace o přihlášeném uživateli. Při příštím startu aplikace není potřeba se znovu přihlašovat. Tyto informace jsou v paměti telefonu uchovány, dokud se uživatel sám neodhlásí. [15] [18]

5.3.4 Přidání nového záznamu

Stránka *AddFuel.html* slouží k přidání nového záznamu. Na stisk tlačítka *Add* je volána metoda *AddNewFuelRecord()*, která přidá nový záznam do databáze. Pokud tato metoda skončí úspěšně, je znova zobrazena hlavní stránka *index.html*. Při zobrazení této stránky jsou opět pomocí metod *GetStatistics(id)* a *GetFuelRecords(id)* aktualizována data.



Obrázek 18: Karta pro přidání nového záznamu o tankování (zdroj vlastní).

5.3.5 Přenos aplikace na Android

Multiplatformní aplikace byla zpočátku psaná a testovaná pouze na platformě Windows Phone. Pro přenos aplikace na Android bylo potřeba nejprve nastavit vývojové nástroje.

Studio Eclipse je součástí Android SDK. Pro vytvoření PhoneGap projektu je potřeba přidat odkaz na Java knihovnu PhoneGap (cordova.x.jar). Dále přidat složku **www**, kde je obsažena JavaScript knihovna cordova.x.js a upravit metodu třídy *MainActivity.java* tak, aby byla jako úvodní stránka aplikace spuštěna html stránka *index.html*.

```
import org.apache.cordova.DroidGap;

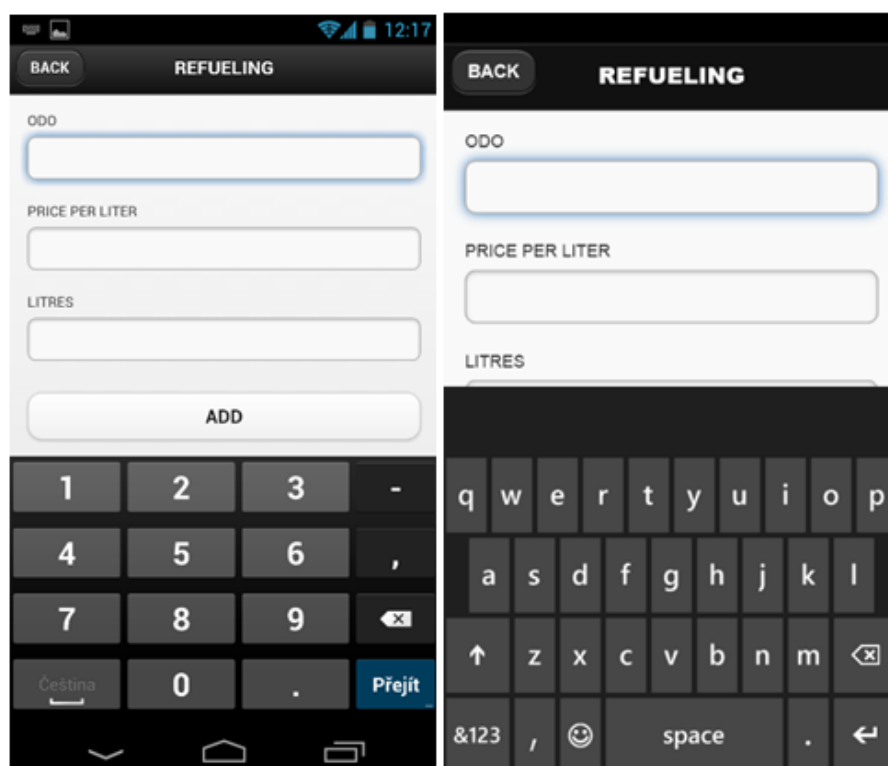
public class MainActivity extends DroidGap {

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        super.loadUrl("file:///android_asset/www/index.html");
    }
}
```

Pro přenesení aplikace je potřeba překopírovat obsah složky **www** z Windows Phone projektu do Android projektu.

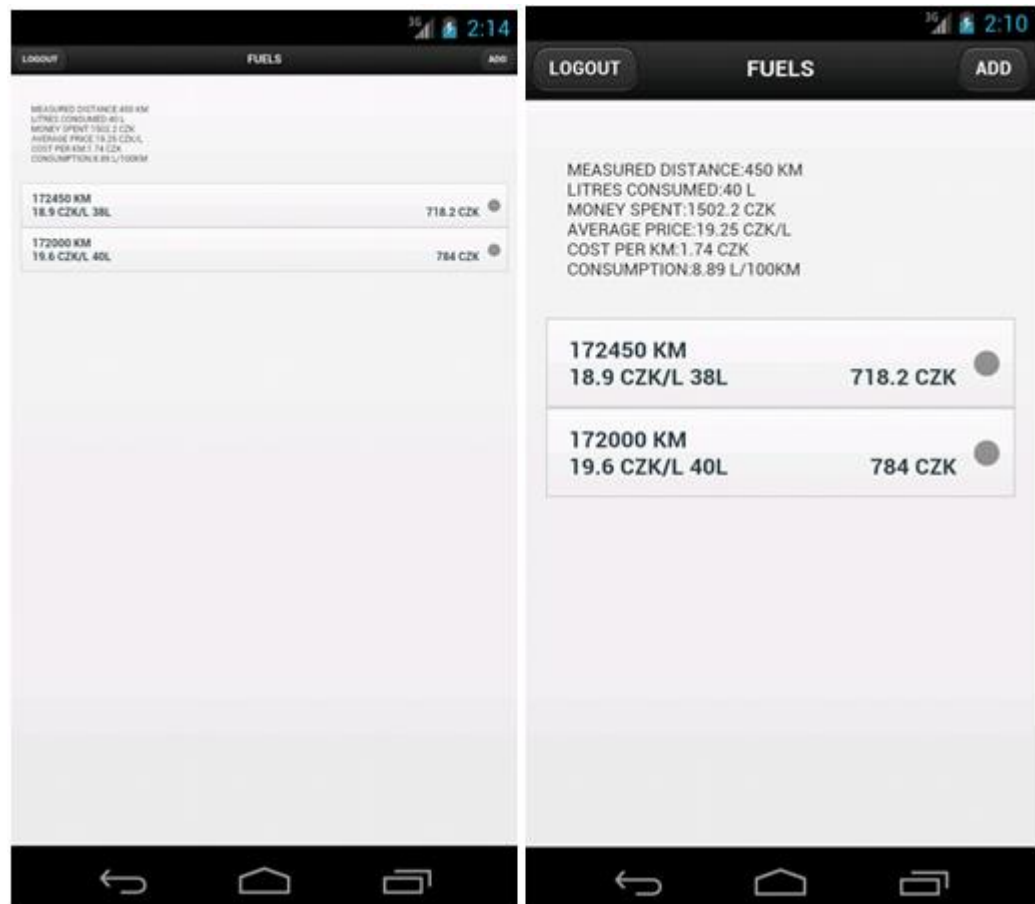
Po přenosu aplikace bylo zjištěno několik odlišností:

- Testovaný Android verze 4.x.x podporuje na formuláři hodnotu atributu *type='number'* elementu *input*. Ten slouží k tomu, že při vkládání je rovnou zobrazena numerická klávesnice. Telefon s Windows Phone zobrazí pouze standartní klávesnici. Oproti nativní aplikaci je tedy potřeba řešit navíc validace vstupů.



Obrázek 19: Zobrazení klávesnice pro psaní (zdroj vlastní).

- Zařízení s Androidem mají mnoho velikostí a rozlišení displejů. Je potřeba upravit styl aplikace tak, aby byl text dobře čitelný a tlačítka dostatečně velká.

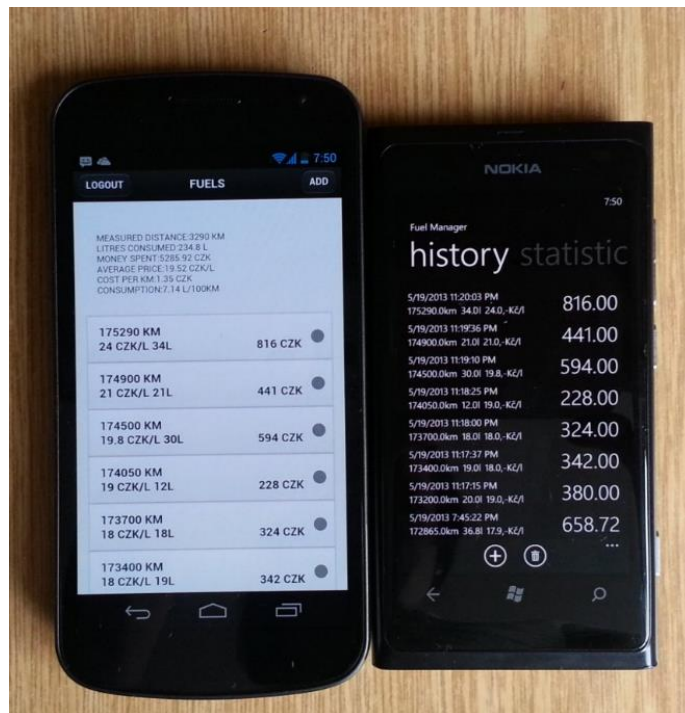


Obrázek 20: Aplikaci je potřeba upravit pro vysoká rozlišení (zdroj vlastní).

- Při návratu na předchozí stránku tedy bylo nutné zjistit, je-li možné využít třídy *navigator.app* a podle toho zavolat správný kód.

```
function goBack() {
  if (typeof (navigator.app) !== "undefined") {
    navigator.app.backHistory();
  } else {
    window.history.back();
  }
}
```

5.4 Výkon nativní a multiplatformní aplikace



Obrázek 21: Aplikace Fuel Manager na telefonech Samsung Galaxy Nexus (multiplatformní) a Nokia Lumia 800 (nativní verze). (zdroj vlastní)

5.4.1 Metodika měření

Výkon aplikace můžeme zjistit tak, že změříme dobu, která je potřeba pro návrat ze stránky pro přidání nového záznamu na hlavní stránku aplikace. Čas je zaznamenán při stisku tlačítka pro návrat nebo tlačítka pro zrušení obrazovky pro zadávání (je volána stejná metoda).

Nativní aplikace loguje čas pomocí funkce *LogTime(string s)*.

```
public long LogTime(string text)
{
    //počítám čas v milisekundách od 1.1.1970
    var epochStart = new DateTime(1970, 1, 1, 0, 0, 0, DateTimeKind.Utc);
    //počet milisekund
    long returnvalue = (DateTime.UtcNow - epochStart).Ticks / 10000;
    //zápis do logu
    Debug.WriteLine("TIME " + text + " is " + (returnvalue));
    return returnvalue;
}

protected override void OnBackPressed(System.ComponentModel.CancelEventArgs e)
{
    base.OnBackPressed(e);
    App.ViewModel.LogTime("Back navigation");
}
```

Měřený čas je zapsán do logu. Druhý čas je zaznamenán v době, kdy je načtena hlavní stránka aplikace. V případě Windows Phone se jedná o obsluhu události Loaded.

```
private void MainPage_Loaded(object sender, RoutedEventArgs e)
{
    App.ViewModel.LogTime("Main page load");
    ...
}
```

V případě PhoneGap jsou měřeny časy při načtení stránky:

```
$(document).delegate("#home", "pageinit", function () {
    initPageTime = new Date().getTime();
});

$(document).delegate("#home", "pageshow", function () {
    showPageTime = new Date().getTime();
});
```

A následně logovány pomocí:

```
console.log('TIME: INIT ' + initPageTime);
```

V případě PhoneGap aplikace byl zaznamenán i čas události *deviceready*. Jedná se událost, která signalizuje kdy je PhoneGap načten a lze využívat jeho API. [21]

Měření je opakováno 10x a z výsledků vypočten aritmetický průměr.

5.4.2 Výsledky měření

5.4.2.1 Nativní aplikace

Tabulka 2: Výsledky měření výkonu nativní aplikace pro Nokia Lumia 800 –
Windows Phone 7.8

pokus	Doba od stisku tlačítka zpět po načtení stránky (ms)
1	150
2	155
3	123
4	150
5	131
6	123
7	121
8	152
9	154
10	125
Průměrná hodnota	138,4

Dle subjektivního pozorování nativní aplikace nejeví žádné známky zpomalení. Reakce na stisk tlačítka je okamžitá. Jediné zpomalení znamená získávání dat prostřednictvím webových služeb, které je ale na rychlé síti také velmi rychlé.

5.4.2.2 *Multiplatformní aplikace*

Tabulka 3: Výsledky měření pro Nokia Lumia 800 - Windows Phone 7.8

zpět – pageinit (ms)	zpět – pageshow (ms)	pageinit – deviceready (ms)
1001	1100	334
987	1125	365
999	1140	369
996	1129	370
1010	1146	398
1000	1136	381
995	1180	428
1005	1141	403
991	1122	383
995	1125	400
997,9	1134,4	383,1

Průměrný čas, který uplyne mezi voláním funkce *goBack()* a zobrazením stránky je 1134,4ms. V porovnání s rychlostí nativní aplikace je tedy cca 8,2 krát pomalejší.

Tabulka 4: Testované zařízení Samsung Galaxy S3 - Android 4.1.2

zpět – pageinit (ms)	Zpět – pageshow (ms)	pageinit – deviceready (ms)
865	922	170
790	832	153
722	762	389
1102	1155	188
640	680	330
656	694	399
841	893	307
631	669	289
872	922	319
711	749	300
830	867	137
787,2727273	831,3636364	271

Průměrný čas, který uplyne mezi voláním funkce *goBack()* a zobrazením stránky je 832,36ms. V porovnání s rychlostí nativní aplikace je 6 krát pomalejší.

Tabulka 5: Testované zařízení Samsung Galaxy Nexus – Android 4.2.2

zpět – pageinit (ms)	zpět – pageshow (ms)	pageinit – deviceready (ms)
806	845	117
714	746	317
597	628	278
801	833	293
723	754	324
617	649	291
767	799	271
665	697	282
616	646	304
785	816	268
709,1	741,3	274,5

Průměrný čas, který uplyne mezi voláním funkce *goBack()* a zobrazením stránky je 741,3 ms. V porovnání s rychlostí nativní aplikace je to 5,36 krát pomalejší. Multiplatformní aplikace na zařízení s Android běží rychleji než na Windows Phone 7.8, rychlosti nativní aplikace však nedosahuje. Rychlost je závislá především na integrovaném prohlížeči a na výkonu použitého zařízení.

Dle subjektivního pozorování chování multiplatformní aplikace na telefonu s Windows Phone bylo zjištěno mírné zpomalení při interakci s uživatelským rozhraním. Zpomalení bylo zjištěno při přechodu mezi jednotlivými stránkami. Jistá prodleva byla i při zobrazování klávesnice. Windows Phone telefon pomaleji reagoval při zobrazení nativního dialogu pro smazání záznamu o tankování. Občas se stalo, že se dialogy objevily se zpožděním dva přes sebe, což způsobilo nemožnost ovládat aplikaci.

Pro porovnání byla měřena i rychlost komunikace s webovou službou. První čas je zaznamenán při volání metody pro získání záznamů o tankování a druhý čas je získán při odpovědi webové služby.

Rychlost komunikace s webovou službou je znázorněna v tabulce.

Tabulka 6: Srovnání rychlosti stahování 10 záznamů o tankování.

Volání – odpověď (XML) Doba trvání v ms	Volání – odpověď (JSON) Doba trvání v ms
245	132
140	117
134	116
150	112
222	124
245	111
139	120
129	113
175,5	118,125

ZÁVĚR

V současné době existuje několik možností tvorby multiplatformních aplikací. Ve většině případů je využito společné schopnosti všech platforem pracovat s HTML5 a JavaScriptem. Nemalou výhodou tohoto přístupu je existence velkého množství webových vývojářů a přenositelnost mezi platformami. Nevýhoda multiplatformních aplikací jako především rychlost, plynulost ovládání a schopnost přizpůsobit se na různá rozlišení mobilních displejů je v současné době víceméně řešitelná, přestože oproti nativnímu vývoji může v některých případech přinést práci navíc. Navigace mezi stránkami byla v multiplatformní aplikaci 8 krát pomalejší než v nativní, ale aplikace byla stále velmi dobře použitelná. Na telefonech s Androidem byla multiplatformní aplikace rychlejší než na telefonech s Windows Phone. To ukazuje potenciál pro další zlepšení. Větší překážkou jsou odlišnosti jednotlivých platforem z hlediska filozofie ovládání a návrhu uživatelského rozhraní. Dosáhnout na dané platformě nativního vzhledu je možné, ale v rozporu se základní myšlenkou multiplatformního vývoje – psát pouze jeden kód. Pro vývoj multiplatformní aplikace je proto nutné znát nejen všechny možnosti vývoje, výhody a nevýhody jednotlivých přístupů, ale také jaké může špatně zvolený způsob vývoje přinést následky. Nejdůležitější je především provést důkladnou analýzu zadání aplikace, kterou se tímto způsobem rozhodneme vyvíjet.

ZÁVĚR V ANGLIČTINĚ

There are several options for creating multiplatform applications these days. In most cases the ability of working with HTML5 and JavaScript that all platforms have in common is used. The great advantage of this attitude is that there are a lot of web developers and the code is usable on different platforms. Disadvantages like speed, operating fluency and adaptability to different mobile screen resolution is nowadays solvable but can mean additional work contrary to native development. Navigation between pages in cross-platform application is 8 times slower than in native on the same device, but still very responsive. Application was faster on the mobile phones with Android than on mobile phones with Windows Phone. It shows the potential for further improvements. The bigger barrier are differences of particular platforms from the user view – controllability and graphical user interface. It is possible to reach native look on particular platform but in contradiction to the basic idea of multiplatform development – to write only single code. To be able to write multiplatform application, it is necessary to know all possibilities of development, advantages and disadvantages of particular approaches and also what impact can badly chosen method have. The most important is to perform detailed analysis of assignment of application that we are about to develop with this method.

6 SEZNAM POUŽITÉ LITERATURY

- [1] GARTNER. *Pacific Led Worldwide Mobile Phone Sales to Growth in First Quarter of 2013* [online]. 2013 [cit. 2013-05-20]. Dostupné z: <http://www.gartner.com/newsroom/id/2482816>
- [2] ELGIN, Ben. Google Buys Android for Its Mobile Arsenal. *Businessweek* [online]. 2005 [cit. 2013-03-14]. Dostupné z: <http://www.businessweek.com/stories/2005-08-16/google-buys-android-for-its-mobile-arsenal>
- [3] GOOGLE INC. *Developer Tools / Android Developers* [online]. 2012 [cit. 2013-05-20]. Dostupné z: <http://developer.android.com/tools/index.html>
- [4] GOOGLE INC. *App Widgets / Android Developers* [online]. 2011 [cit. 2013-05-3]. Dostupné z: <http://developer.android.com/guide/topics/appwidgets/index.html>
- [5] GOOGLE INC. *App Components / Android Developers* [online]. 2013 [cit. 2013-03-15]. Dostupné z: <http://developer.android.com/guide/components/index.html>
- [6] APPLE INC. *Developer Tools Overview - Apple Developer* [online]. 2010 [cit. 2013-04-10]. Dostupné z: <https://developer.apple.com/technologies/tools/>
- [7] APPLE INC. *IOS Dev Center - Apple Developer* [online]. 2009 [cit. 2013-04-02]. Dostupné z: <https://developer.apple.com/devcenter/ios/index.action>
- [8] MICROSOFT CORP. *Implementing Windows Phone app design* [online]. 2012 [cit. 2013-04-12]. Dostupné z: [http://msdn.microsoft.com/en-us/library/windowsphone/design/hh202895\(v=vs.105\).aspx](http://msdn.microsoft.com/en-us/library/windowsphone/design/hh202895(v=vs.105).aspx)
- [9] iPhone radio remote app. In: *Flickr* [online]. 2009 [cit. 2013-05-12]. Dostupné z: <http://www.flickr.com/photos/ntwojp/3790072265/>
- [10] GOOGLE INC. *App Widget Design Guidelines / Android Developers* [online]. 2010 [cit. 2013-05-20]. Dostupné z: http://developer.android.com/guide/practices/ui_guidelines/widget_design.html
- [11] MICROSOFT CORP. *Windows Phone Dev Center* [online]. 2011 [cit. 2013-05-12]. Dostupné z: <https://dev.windowsphone.com/en-us>

- [12] MICROSOFT CORP. *In-app navigation for Windows Phone* [online]. 2013 [cit. 2013-04-21]. Dostupné z: [http://msdn.microsoft.com/en-us/library/windowsphone/develop/ff402536\(v=vs.105\).aspx](http://msdn.microsoft.com/en-us/library/windowsphone/develop/ff402536(v=vs.105).aspx)
- [13] MEADOWS, Larry. A Closer Look at Windows Phone 8X and 8S. In: *HTC Blog* [online]. 2012 [cit. 2013-05-20]. Dostupné z: <http://blog.htc.com/2012/09/windows-phone-8-qa/>
- [14] MICROSOFT CORP. *Tiles for Windows Phone* [online]. 2013 [cit. 2013-05-14]. Dostupné z: [http://msdn.microsoft.com/en-US/library/windowsphone/develop/hh202948\(v=vs.105\).aspx](http://msdn.microsoft.com/en-US/library/windowsphone/develop/hh202948(v=vs.105).aspx)
- [15] WARGO, John M. *PhoneGap essentials: building cross-platform mobile apps*. Upper Saddle River, NJ: Addison-Wesley, c2012, xxiv, 359 p. ISBN 03-218-1429-0.
- [16] XAMARIN INC. *Introduction to Mobile Development / xamarin* [online]. 2012 [cit. 2013-05-20]. Dostupné z: http://docs.xamarin.com/guides/cross-platform/getting_started/introduction_to_mobile_development
- [17] APPCELERATOR. *Titanium 3.X - Appcelerator Docs* [online]. 2013 [cit. 2013-05-16]. Dostupné z: <http://docs.appcelerator.com/titanium/latest/>
- [18] PHONEGAP. *PhoneGap API Documentation* [online]. 2013 [cit. 2013-05-16]. Dostupné z: <http://docs.phonegap.com/en/2.6.0/index.html>
- [19] SENCHA INC. *Architect 2 - Sencha Docs* [online]. 2013 [cit. 2013-05-15]. Dostupné z: <http://docs.sencha.com/architect/2/#!/guide/intro>
- [20] CAMERON, Rob. *Pro Windows Phone 7 development*. New York: Distributed to the book trade worldwide by Springer Science Business Media, c2011, xvi, 464 p. ISBN 14-302-3219-6.
- [21] MEYER, Jeanine. *HTML5 and JavaScript projects*. New York: Distributed to the book trade by Springer, c2011, xv, 432 p. Expert's voice in Web development. ISBN 978-1-4302-4033-4.
- [22] BROWN, By Tracy Steven. *Dynamic Apache with Ajax and JSON*. Sebastopol, California: O'Reilly, 2006. ISBN 978-059-6528-393.

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

REST	Representational State Transfer
HTTP	Hypertext Transfer Protocol
JS	JavaScript
JSON	JavaScript Object Notation
XML	Extensible Markup Language
AJAX	Asynchronnous JavaScript
SDK	Software Development Kit
WCF	Windows Communication Foundation
RWD	Responsive Web Design
MVC	Model-View-Controller
MVVM	Model-View-ViewModel
DX	Dalvik executable
API	Application Programming Interface

SEZNAM OBRÁZKŮ

Obrázek 1: Android Widget. [10]	14
Obrázek 2: Vývojové prostředí Xcode a popis základních prvků aplikace. [6]	15
Obrázek 3: Návrhový vzor MVC. [6]	16
Obrázek 4: Aplikace FM rádio ve stylu iOS a aplikace rádio ve stylu WP7. [8][9].....	17
Obrázek 5: ApplicationBar se dvěma tlačítky a jednou položkou menu (zdroj vlastní).....	18
Obrázek 6: Příklad živé dlaždice – Live tile v různých velikostech.[13]	19
Obrázek 7: Vyvojové prostředí Appcelerator (zdroj vlastní).....	22
Obrázek 8: Xamarin studio – Vytváření nového projektu Android aplikace (zdroj vlastní).....	23
Obrázek 9: Vývojové nástroje Sencha Touch (zdroj vlastní).	25
Obrázek 10: PhoneGap projekt integrovaný do Visual Studia (zdroj vlastní).....	27
Obrázek 11: Komunikace (zdroj vlastní).	33
Obrázek 12: Schéma databáze (zdroj vlastní).....	34
Obrázek 13: Přihlašovací obrazovka (zdroj vlastní).	36
Obrázek 14: Obrazovka zobrazující historii tankování a statistiku (Zdroj vlastní).	37
Obrázek 15: Struktura projektu mobilní aplikace PhoneGap pro Windows Phone a Android (zdroj vlastní).	38
Obrázek 16: Obrazovka pro přihlášení uživatele do aplikace a obrazovka pro registraci nového uživatele (zdroj vlastní).	39
Obrázek 17: Zobrazení statistiky a historie tankování (zdroj vlastní).	40
Obrázek 18: Karta pro přidání nového záznamu o tankování (zdroj vlastní).	41
Obrázek 19: Zobrazení klávesnice pro psaní (zdroj vlastní).	43
Obrázek 20: Aplikaci je potřeba upravit pro vysoká rozlišení (zdroj vlastní).	43
Obrázek 21: Aplikace Fuel Manager na telefonech Samsung Galaxy Nexus (multiplatformní) a Nokia Lumia 800 (nativní verze). (zdroj vlastní)	44

SEZNAM TABULEK

Tabulka 1: Prodeje mobilních telefonů podle operačního systému. [1]	11
Tabulka 2: Výsledky měření výkonu nativní aplikace pro Nokia Lumia 800 – Windows Phone 7.8.....	45
Tabulka 3: Výsledky měření pro Nokia Lumia 800 - Windows Phone 7.8.....	46
Tabulka 4: Testované zařízení Samsung Galaxy S3 - Android 4.1.2	46
Tabulka 5: Testované zařízení Samsung Galaxy Nexus – Android 4.2.2.....	47
Tabulka 6: Srovnání rychlosti stahování 10 záznamů o tankování.....	48

SEZNAM PŘÍLOH

P1: CD s Aplikacemi a uživatelským manuálem

PŘÍLOHA P I: CD S APLIKACEMI A UŽIVATELSKÝM MANUÁLEM

- Aplikace – Adresář obsahuje instalační balíčky hotových aplikací.
- Src – Obsahuje zdrojové kódy aplikací, webové služby, databáze.
- Doc – Obsahuje text diplomové práce, manuály k ovládání aplikace.