



Ján Hanák

Vývoj moderných WinRT-programov
pre systém Windows 8



Windows 8

Ján Hanák

Vývoj moderných WinRT-programov pre systém Windows 8

Microsoft
2012

Obsah

Úvod.....	3
Pre koho je táto kniha určená.....	5
Obsahová štruktúra knihy.....	5
Typografické konvencie.....	7
Podakovanie.....	9
1 Prvý pohľad na operačný systém Windows 8.....	11
1.1 Nové rozhranie systému Windows.....	11
1.2 Inovované rozhranie Aero a štandardná pracovná plocha.....	15
1.3 Elektronický obchod Windows.....	18
1.3.1 Praktická ukážka prevzatia programu z obchodu Windows.....	20
1.3.2 Použitie programu získaného z obchodu Windows.....	27
2 Paradigmy a technológie na vývoj programov v štýle WinRT.....	32
3 Životný cyklus programu v štýle WinRT.....	39
3.1 Životný cyklus vývoja programu v štýle WinRT.....	39
3.1.1 Analýza požiadaviek na program v štýle WinRT.....	39
3.1.2 Návrh softvérovej architektúry programu v štýle WinRT.....	42
3.1.3 Implementácia navrhutej softvérovej architektúry programu v štýle WinRT.....	43
3.1.4 Testovanie programu v štýle WinRT.....	45
3.1.5 Nasadenie a predaj programu v štýle WinRT.....	46
3.2 Životný cyklus spracovania programu v štýle WinRT.....	46
4 Vývoj prvého programu v štýle WinRT v jazykoch C# a Visual Basic.....	52
4.1 Charakteristika prvého programu v štýle WinRT.....	52
4.2 Náčrt vizuálneho rozhrania prvého programu v štýle WinRT.....	53

4.3 Založenie projektu nového programu v štýle WinRT v jazykoch C# a Visual Basic ...	55
4.4 Analýza projektových súčastí programu v štýle WinRT	57
4.5 Vývoj grafického rozhrania prvého programu v štýle WinRT	60
4.5.1 Úprava vzhľadu hlavnej stránky prvého programu v štýle WinRT.....	63
4.6 Implementácia logiky prvého programu v štýle WinRT	71
4.7 Konfigurácia grafických zdrojov prvého programu v štýle WinRT.....	73
4.8 Zostavenie a spustenie programu štýle WinRT v systéme Windows 8	75
5 Vývoj prvého programu v štýle WinRT v jazyku C++	81
5.1 Charakteristika prvého programu v štýle WinRT	81
5.2 Náčrt vizuálneho rozhrania prvého programu v štýle WinRT.....	81
5.3 Založenie projektu nového programu v štýle WinRT v jazyku C++	83
5.4 Analýza projektových súčastí programu v štýle WinRT	85
5.5 Vývoj grafického rozhrania prvého programu v štýle WinRT	88
5.5.1 Úprava vzhľadu hlavnej stránky programu v prostredí produktu Blend for Visual Studio 2012	90
5.6 Implementácia logiky prvého programu v štýle WinRT	96
5.7 Konfigurácia grafických zdrojov prvého programu v štýle WinRT.....	98
5.8 Zostavenie a spustenie programu v štýle WinRT v systéme Windows 8.....	100
Záver.....	103
O autorovi.....	104

Úvod

Rok 2012 sa do histórie počítačového priemyslu nezmazateľne zapíše ako rok uvedenia zbrusu nového operačného systému Windows 8 od spoločnosti Microsoft. Hoci softvéroví inžinieri firmy Microsoft vyvíjajú operačný systém Windows už takmer tri dekády, každá nová verzia tohto softvérového produktu prichádza s niečím novým a dosiaľ nevídaným. V prípade systému Windows 8 je zásadných novinek hneď niekoľko, avšak snád' najväčšou z nich je neobvyčajné vizuálne rozhranie.

Nové rozhranie systému Windows 8 je primárne orientované na tablety a počítačové systémy s dotykovo citlivými obrazovkami. Pre počítače tohto typu je prirodzené, že ich používatelia ovládajú dotykovými gestami, čím sa komunikačný dialóg medzi človekom a počítačom dostáva na sofistikovanejšiu úroveň. Nové prostredie nie je len paralelný ekvivalent dobre známeho rozhrania Aero: v skutočnosti ide o nové rozhranie, za ktorým stojí dômyselná stratégia s koncentráciou na vytváranie a prehliadanie obsahu, a to všetko rýchlo, plynulo a efektívne.

Pre nové Windows-rozhranie je typická inovovaná štartovacia ponuka, ktorej dominantou je pás s dlaždicami nainštalovaných programov. Tieto programy v štýle WinRT, ako ich zvykneme nazývať, predstavujú nový druh aplikačného typového softvéru. WinRT-programy sú v mnohých ohľadoch špecifické. Napríklad, používatelia tieto programy spúšťajú, no už si nemusia robiť starosti s ich ukončovaním (túto akciu za nich vykoná operačný systém Windows 8 sám). Každý program v štýle WinRT disponuje svojou dlaždicou, ktorá môže byť naprogramovaná ako dynamická. Vtedy dokáže program vysielat' používateľovi užitočné informácie aj bez toho, aby používateľ tento program vôbec spustil. Pre WinRT-programy je tiež príznačné, že bežia v celoobrazovkovom režime a svojich používateľov zdravia štýlovými úvítacími obrazovkami. V neposlednom rade možno postrehnúť dôrazné nasadenie paradigmy asynchrónneho programovania. Vďaka správnej implementácii asynchronizmu dokážu programy v štýle WinRT spracúvať svoje úlohy svižne, bez vzniku kolíznych stavov, ktoré boli špecifické pre synchrónne bežiacie programy predchádzajúcej generácie.

Technicky sú programy v štýle WinRT softvérovými aplikáciami, ktoré bežia buď v riadenom, alebo v natívnom prostredí. Pre vývojárov počítačového softvéru to znamená, že WinRT-programy môžu vytvárať s podporou jednak riadených a rovnako aj natívnych technológií. Prvý variant využíva programovacie jazyky Visual Basic 2012 a C# 5.0, zatiaľ čo v druhom prípade je k dispozícii jazyk C++ (s komponentovými rozšíreniami CX) a webové technológie spočívajúce na HTML5, CSS3 a jazyku JavaScript. Pre nové prostredie systému Windows 8 je dokonca možné programovať počítačové hry s podporou rozhrania DirectX (za týchto okolností je hlavným programovacím jazykom C++).

V tejto knihe sa zameriavame na splnenie nasledujúcich vzdelávacích cieľov:

1. Poskytujeme základný kurz fungovania nového operačného systému Windows 8 s inkorporovaným novým rozhraním a programami, ktoré bežia v tomto prostredí.
2. Rozoberáme paradigmy a technológie na vývoj programov v štýle WinRT.
3. Skúmame životné cykly vývoja a spracovania programov v štýle WinRT.
4. Opisujeme praktické nasadenie riadených a natívnych jazykov, nástrojov, platforiem a technológií pri vývoji nových WinRT-programov v programovacích jazykoch Visual Basic, C# a C++.

V publikácii vás najskôr oboznámime s programami v štýle WinRT z pohľadu finálneho používateľa. Len čo pochopíme pracovné princípy WinRT-programov, zmeníme pohľad a pozrieme sa na z pohľadu softvérového vývojára. Pritom však používame pozvoľný štýl výkladu preberanej problematiky so zameraním na začínajúcich vývojárov, ktorí by sa radi naučili, ako zhotovovať programy kompatibilné s novým rozhraním systému Windows 8.

Ján Hanák

Bratislava august 2012

Pre koho je táto kniha určená

Táto kniha je určená pre začínajúcich programátorov v novom prostredí systému Windows 8. Ako „začínajúceho programátora“ profilujeme v tomto kontexte vývojára, ktorý vyhovujeme týmto požiadavkám:

- Ovláda princípy objektového programovania v jednom z programovacích jazykov Visual Basic, C# a C++.
- Pozná paradigmy vizuálneho a udalostného programovania.
- Má skúsenosti s vývojovou platformou Microsoft .NET Framework.

Pri písaní tejto knihy sme kládli dôraz na zaradenie maximálneho možného počtu praktických postup, algoritmov, odporúčaní a rád, ktoré by mal absorbovať každý vývojár, ktorý sa podujme naprogramovať svoj prvý program v štýle WinRT. Hoci vravíme, že táto publikácia je určená pre začínajúceho programátora v novom rozhraní systému Windows 8, neznamená to, že je vhodná pre úplného nováčika v oblasti tvorby počítačového softvéru. Úplným nováčikom odporúčame najskôr preštudovať iné knižné zdroje, ktoré sa venujú základnom algoritmickej a objektového programovania. Tematika vykladaná v tejto knihe je prístupná až po nadobudnutí uvedených znalostí.

Obsahová štruktúra knihy

Knihu tvorí dovedna 5 kapitol s nasledujúcim tematickým zameraním:

1. kapitola: **Prvý pohľad na operačný systém Windows 8.** Táto kapitola opisuje nové rozhranie systému Windows 8 a pracovné princípy programov, ktoré v tomto rozhraní bežia. Pozornosť je venovaná aj inováciám v rozhraní Aero či elektronickému obchodu Windows.

2. kapitola: **Paradigmy a technológie na vývoj programov v štýle WinRT.** Táto kapitola rozoberá paradigmy vývoja riadených a natívnych programov v štýle WinRT v programovacích jazykoch Visual Basic, C#, C++ a JavaScript.
3. kapitola: **Životný cyklus programu v štýle WinRT.** Obsahová náplň tejto kapitoly je rozdelená na charakteristiku životného cyklu vývoja nového programu v štýle WinRT a na opis životného cyklu spracovania programu v štýle WinRT v operačnom systéme Windows 8.
4. kapitola: **Vývoj prvého programu v štýle WinRT v jazykoch C# a Visual Basic.** Kapitola podáva rozsiahly praktický návod, ako zhotoviť medicínsky WinRT-program v jazykoch C# a Visual Basic, ktorý bude schopný diagnostikovať ejekčnú frakciu srdca pacienta. Čitatelia sa zoznámia s celým vývojovým cyklom, charakteristikou programu počnúc a jeho nasadením končiac.
5. kapitola: **Vývoj prvého programu v štýle WinRT v jazyku C++.** Kapitola poskytuje rozsiahly praktický návod, ako vyvinúť WinRT-program, ktorý bude pôsobiť ako rýchly prehliadač webových stránok. Primárnym programovacím jazykom je jazyk C++ s komponentovými rozšíreniami (CX), známy tiež ako C++/CX. Vývojovou platformou je Windows Runtime (respektíve WinRT).


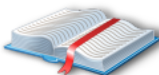

Typografické konvencie

Aby sme vám čítanie tejto knihy spríjemnili v čo možno najväčšej miere, prijali sme kódex typografických konvencií, pomocou ktorých došlo k štandardizácii a unifikácii použitých textových štýlov a grafických symbolov. Veríme, že prijaté konvencie napomôžu zvýšenie prehľadnosti a používateľskej prívetivosti výkladu. Prehľad použitých typografických konvencií uvádzame v tab. A.

Tab. A: Prehľad použitých typografických konvencií	
Typografická konvencia	Ukážka použitia typografickej konvencie
Štandardný text výkladu, ktorý neoznačuje zdrojový kód, identifikátory, modifikátory a kľúčové slová jazykov Visual Basic, C# a C++, ani názvy iných syntaktických elementov a entít, je formátovaný týmto typom písma.	Vývojová platforma Microsoft .NET Framework 4.5 vytvára spoločne s jazykmi Visual Basic, C# a C++ jednotnú technologickú bázu na vytváranie moderných riadených a natívnych programov v štýle WinRT.
Názvy ponúk, položiek ponúk, ovládacích prvkov, komponentov, dialógových okien, podporných softvérových nástrojov, typov projektov ako aj názvy ďalších súčastí grafického používateľského rozhrania sú formátované tučným písmom.	Projekt nového programu v štýle WinRT založíme v prostredí jazyka Visual Basic 2012 takto: <ol style="list-style-type: none">1. Na úvodnej stránke Start Page klepneme na položku New Project.2. V dialógovom okne New Project rozviníme v stromovej štruktúre Templates položku Visual Basic a klikneme na položku Windows Store.3. Zo súpravy projektových šablón vyberieme šablónu Grid App (XAML).4. Do textového poľa Name zapíšeme názov nového projektu a stlačíme tlačidlo OK.

Symboly klávesov, klávesových skratiek a ich kombinácií sú uvádzané VELKÝMI PÍSMENAMI.	Ak chceme otvoriť už existujúci projekt jazyka Visual Basic 2012, použijeme klávesovú skratku CTRL+O.
--	---

Okrem typografických konvencií predstavených v tab. A sa môžeme na stránkach knihy stretnúť aj s informačnými ostrovčekmi, ktoré ponúkajú hodnotné informácie súvisiace s práve preberanou problematikou. Zoznam použitých informačných ostrovčiek a s nimi asociovaných grafických symbolov je zobrazený v tab. B.

Tab. B: Prehľad použitých informačných ostrovčiek		
Grafický symbol informačného ostrovčeka	Názov informačného ostrovčeka	Charakteristika
	Dôležité	Upozorňuje čitateľov na dôležité skutočnosti, ktoré by mali mať v každom prípade na pamäti, pretože od nich môže závisieť pochopenie ďalších súvislostí alebo úspešné uskutočnenie postupu či pracovného algoritmu.
	Poznámka	Oboznamuje čitateľov s podrobnejšími informáciami, ktoré sa spájajú s vysvetľovanou problematikou. Hoci je miera dôležitosti tohto informačného ostrovčeka menšia ako pri jeho predchodcovi, vo všeobecnosti sa odporúča, aby čitatelia venovali doplňujúcim informačným vyhláseniam svoju pozornosť. Môžu sa tak dozvedieť nové fakty, alebo nájsť skryté súvislosti medzi už známymi poznatkami.
	Tip	Poukazuje na lepšie, rýchlejšie a efektívnejšie splnenie úlohy alebo postupu. Keď čitatelia uvidia v texte knihy tento informačný ostrovček, môžu si byť istí, že nájdu spôsob, ako produktívne dosiahnuť požadovaný cieľ.

Pod'akovanie

Na tomto mieste by autor knihy rád vyjadril svoje poďakovanie pánovi Mgr. Miroslavovi Kubovčíkovi, ktorý je kmeňovým členom vývojárskeho tímu slovenskej pobočky spoločnosti Microsoft, za výbornú niekoľkoročnú spoluprácu, ktorá priniesla vývojárskej komunite veľa hodnotných produktov.



Kapitola 1

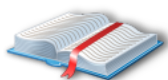
Prvý pohľad na operačný systém Windows 8

1 Prvý pohľad na operačný systém Windows 8

Nový operačný systém Windows 8 od spoločnosti Microsoft prichádza so zásadnými inováciami, a to nielen pre finálnych používateľov, ale tiež pre vývojárov aplikačného softvéru. V tejto kapitole preskúmame hlavné novinky, s ktorými sa vývojári musia oboznámiť ešte predtým, ako začnú budovať svoje nové programy pre systém Windows 8.

1.1 Nové rozhranie systému Windows

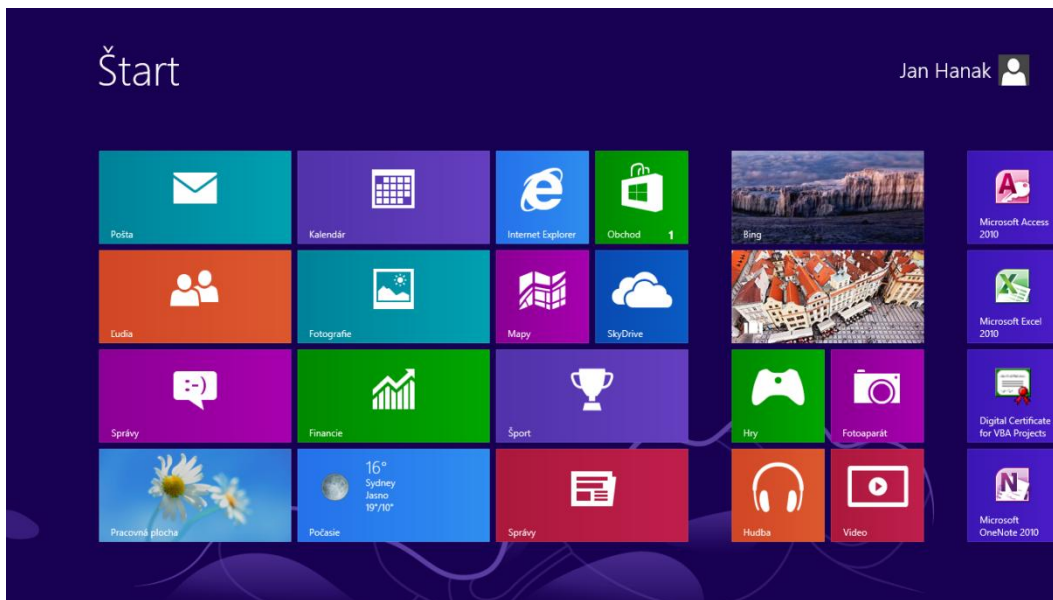
Snáď najväčšou novinkou systému Windows 8 je jeho nová prezentačná vrstva, ktorú tvorí nové vizuálne rozhranie. Nové Windows-rozhranie je určené predovšetkým pre tie typy počítačových systémov, ktoré môžu byť ovládané dotykmi a gestami finálnych používateľov. V súčasnosti ide najmä o tablety, počítače typu All-In-One (teda „všetko v jednom“) a inteligentné mobilné telefóny (v nich je nové Windows-rozhranie prístupné prostredníctvom systému Windows Phone 7.5). Pre stroje citlivé na dotyk sa nové Windows-rozhranie stáva základným komunikačným prvkom v procese interakcie človeka a počítača. Na druhej strane však existujú milióny štandardných počítačov, ktoré nedisponujú schopnosťou analyzovať a spracúvať dotykové gestá používateľov. Samozrejme, aj na týchto počítačoch bude možné nové operačný systém nainštalovať. Ak nie je určené inak, Windows 8 vždy integruje dve základné prezentačné vrstvy: jednak ide o nové rozhranie a jednak o štandardné „desktopové“ rozhranie Aero, na ktoré sme (či už ako používatelia, alebo ako vývojári) zvyknutí zo systému Windows 7. Navyše, medzi obidvomi rozhraniami sa dá plynulo prepínať, a tak využívať výhody tu jednej a tam zase druhej formy vizuálnej komunikácie s počítačom.



Poznámka: V čase písania tejto knihy (august 2012) bola pre záujemcov k dispozícii ostrá verzia produktu Microsoft Windows 8 s príznakom RTM (angl. *Release To Manufacturing*). Radi by sme uviedli, že všetky informácie, obrázky,

ilustrácie a pracovné postupy, s ktorými sa stretnete na nasledujúcich stranách tejto príručky, vychádzajú práve z uvedenej verzie systému Windows 8.

Po spustení a prihlásení sa do systému Windows 8 sme okamžite konfrontovaní s úvodnou obrazovkou nového prezentačného rozhrania (obr. 1.1).



Obr. 1.1: Úvodná obrazovka nového rozhrania po spustení systému Windows 8

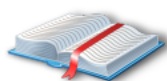
Nové Windows-rozhranie má zaujímavé vlastnosti, ktoré budeme v stručnosti charakterizovať:

1. **Všetky WinRT-programy sídlia na páse dlaždíc.** Pás dlaždíc je variabilne mohutný, no vždy platí, že sa po ňom môžeme posúvať dotykcom, respektíve myšou pomocou posúvacieho ovládacieho prvku. Pás dlaždíc rastie podľa toho, koľko programov si používateľ nainštaluje. Povedané inak, po prvotnom spustení operačného systému Windows 8 je pás dlaždíc kratší, no s dodaním každej novej aplikácie sa rozširuje. Používatelia (a rovnako aj vývojári) môžu pás dlaždíc sledovať pri variabilnom rozlíšení. Vždy, keď sa rozlíšenie zväčší, zväčší sa aj viditeľný výsek pásu dlaždíc, ktorý efektívne vyplní dostupnú zobrazovaciu plochu. Vďaka schopnosti dynamického prepínania miery vizuálnej granularity pásu dlaždíc sa vieme orientovať aj v systéme obsahujúcom veľké množstvo

nainštalovaných programov. Práve opísaná vlastnosť je v novom Windows-rozhraní známa ako sémantický objektív (angl. *semantic zoom*). Sémantický objektív funguje ako objektív s premenlivou ohniskovou vzdialenosťou, a tak nám umožňuje zobrazovať väčšie alebo menšie porcie žiadaného informačného obsahu.

2. Každému WinRT-programu patrí jedna dlaždica, ktorá môže byť dynamická.

To znamená, že program dokáže prostredníctvom svojej dynamickej dlaždice poskytovať finálnemu používateľovi rozmanitú spätnú väzbu. Napríklad, e-mailový klient využije plochu dlaždice na zobrazenie doručených alebo zatiaľ neprečítaných správ elektronickej pošty. Bežecký manažér je schopný používateľa informovať o metrikách posledného tréningu (môže ísť napríklad o dĺžku zabehnutej trasy, celkový čas tréningu či priemerné bežecké tempo). Skutočnosť, že dlaždice programov sa vedia správať dynamicky, je významná, pretože používateľ sa dozvie základné informácie (ktoré sú mu automaticky poskytnuté) aj bez toho, aby program v skutočnosti spustil.



Poznámka: Životný cyklus programu, ktorý beží v novom rozhraní systému Windows 8, je odlišný od spôsobu života štandardnej aplikácie. Poznamenajme, že program v štýle WinRT sa vtedy, keď

s ním používateľ nepracuje, prepína do špeciálneho hibernačného režimu. V tomto režime program síce stále žije (disponuje fyzickým procesom, programovými vláknami a pamäťovými vrstvami), no nevyvíja žiadnu aktivitu (v jeho vláknach nie sú realizované žiadne činnosti). Viac o problematike riadenia životných cyklov programov pôsobiacich v novom Windows-rozhraní povieme v kapitole 3 *Životný cyklus programu v štýle WinRT*.

S dlaždicou patriacou WinRT-programu možno vykonávať kontextovo viazané akcie. Napríklad, po vykonaní gesta, ktoré vyvolá zobrazenie možností kontextovej ponuky, sa dá program odinštalovať. Alebo po označení dlaždice programu môžeme plynulým posunom zmeniť aktuálnu pozíciu tejto dlaždice na páse dlaždíc. Takisto je možné zväčšiť či naopak zmenšiť plochu dynamickej dlaždice. Týmto spôsobom účinne vplývame aj na rozsah vizuálne prezentovanej spätnej väzby, ktorú nám program môže poskytnúť.

3. **Nové rozhranie systému Windows 8 je rýchle a plynulé a presne také sú aj WinRT-programy, ktoré v ňom bežia.** Keďže nové Windows-rozhranie je v prvom rade orientované na systémy s dotykovými obrazovkami, je pochopiteľné, že jeho používanie je úplne podriadené jednotlivým a združeným dotykom a tiež dotykovým gestám. Hoci nové Windows-rozhranie vieme ovládať aj štandardnými perifériami, dotyk je pre toto rozhranie najprirodzenejšou alternatívou.

Práca s programami v štýle WinRT je vskutku rýchla, pričom systém je vždy maximálne citlivý na vstupy používateľa. Dojem plynulosti je umocnený interaktívnymi animáciami, ktoré sa vyznačujú čistými dizajnovými linkami a hladkými prechodmi, čím pozitívne vplývajú na maximalizáciu subjektívnej rýchlosti programu v očiach používateľa.

4. **Pre program v štýle WinRT sú typické tieto atribúty:**

- Ak nie je určené inak, program v štýle WinRT beží štandardne v celoobrazovkovom móde. Hoci operačný systém Windows 8 implementuje moderné techniky preemptívneho viacúlohového spracovania svojich programov, pri práci v novom rozhraní navrhujeme programy tak, aby používateľ priamo komunikoval s práve jedným (zvoleným) programom. Zmyslové zacielenie na jeden program je prínosné, pretože napomáha maximalizácii zobrazeného informačného obsahu.



Tip: Pre úplnosť dodajme, že nové Windows-rozhranie dovoľuje vzájomnú koexistenciu dvoch, prípadne aj viacerých programov vedľa seba. V tomto prípade sa obrazovka rozdelí pomocou vodiacich liniek na viacero regiónov a do každého regiónu sa umiestni prezentačná vrstva práve jedného programu v štýle WinRT.

- Komunikácia s programom je možná dotykmi, gestami alebo štandardnými vstupnými zariadeniami počítača.

- Program má vždy deterministicky definované vizuálne rozvrhnutie svojej prezentačnej vrstvy. Túto vrstvu formuje grafické rozhranie programu, ktorého funkcionality je založená na princípe dynamickej mriežky. Mriežka vytvára základné hierarchické usporiadanie jednak kolekcii objektov a tiež aj samostatných objektov týchto kolekcii. Výhodou mriežkového usporiadania je okrem logickej kategorizácie objektových kolekcii aj schopnosť automatického prispôsobovania sa vzhľadu programu v priamej korelácii na dostupnú zobrazovaciu plochu a aktuálne rozlíšenie zariadenia, na ktorom je program práve spracúvaný.
- Program sa koncentruje na zobrazovanie a správu informačného obsahu. Exkluzívna koncentrácia na informačný obsah je snád' druhým najväčším špecifikom každého správne navrhnutého programu v štýle WinRT. Používatelia sa posúvajú po racionálne členenej obsahovej štruktúre a intuitívne vyberajú „tie pravé“ objektové kolekcie s požadovaným obsahom. Cieľom vývojárov je naprogramovať také aplikácie, ktoré umožnia používateľovi dotknúť sa informačného obsahu a okamžite s ním priamo manipulovať.

1.2 Inovované rozhranie Aero a štandardná pracovná plocha

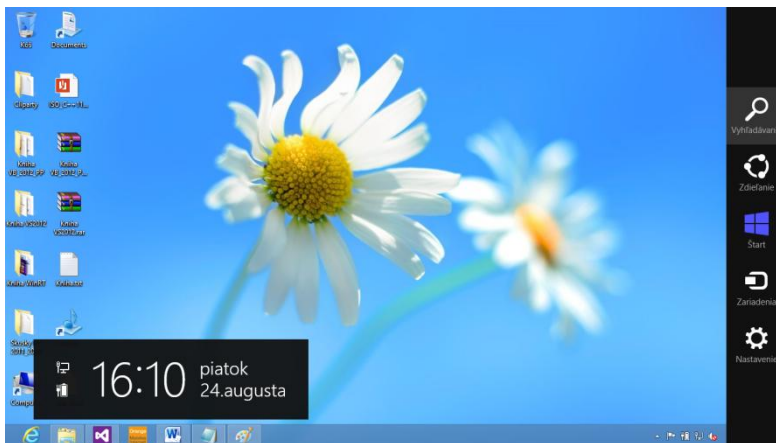
Z nového Windows-rozhrania sa môžeme kedykoľvek prepnúť na pracovnú plochu systému Windows 8 (obr. 1.2). Pracovná plocha je vykreslená v rozhraní Aero, ktoré je veľmi podobné svojej implementácii v systéme Windows 7. Aj keď to nemusí byť na prvý pohľad úplne zrejmé, i na štandardnej pracovnej ploche došlo k určitým zmenám. S tými najvýznamnejšími sa zoznámime na nasledujúcich riadkoch.



Obr. 1.2: Pracovná plocha rozhrania Aero systému Windows 8

Pracovná plocha obsahuje nasledujúce novinky:

1. Na hlavnom paneli sa nenachádza tlačidlo **Štart**.
2. Absencia tohto tlačidla je vynahradená pracovným modelom tzv. horúcich rohov (angl. *hot corners*). Napríklad, po prejdení kurzorom myši do oblasti pravého horného rohu pracovnej plochy sa objaví postranný panel s kľúčovými tlačidlami (obr. 1.3).



Obr. 1.3: Postranný panel s kľúčovými tlačidlami

Panel s kľúčovými tlačidlami (angl. *charm bar*) obsahuje 5 tlačidiel:

1. **Vyhľadávanie:** tlačidlo slúži na vyhľadávanie programov, súborov alebo informačného obsahu. Vyhľadávanie je veľmi senzitívne a jeho výsledky sú dynamicky predkladané pred používateľa. Napríklad, po zadaní vyhľadávacieho reťazca „Visual“ nám systém Windows 8 v zlomku sekundy oznámi, že toto slovo sa vyskytuje v názvoch 2 aplikácií, 5 nastavení a 819 súborov. Po doplnení textového reťazca na „Visual Studio“ sa štatistika zmení: 2 aplikácie, 0 nastavení a 817 súborov.
2. **Zdieľanie:** tlačidlo je nápomocné pri konfigurácii nastavení zdieľania priečinkov a zariadení na aktuálnom počítačovom systéme.
3. **Štart:** tlačidlo nás preniesie z pracovnej plochy (a rozhrania Aero) do úvodnej obrazovky nového Windows-rozhrania. Ihneď tak získavame prístup k pásu s dlaždicami, ktoré reprezentujú programy v štýle WinRT.
4. **Zariadenia:** tlačidlo umožňuje realizovať nastavenia periférnych zariadení, ktoré sú pripojené k počítaču. Týmto spôsobom môžeme napríklad uskutočniť inicializačné nastavenie pracovnej plochy dátového projektora.
5. **Nastavenie:** tlačidlo sprístupní panel s konfiguračnými nastaveniami. Veľmi rýchlo vieme upravovať tieto nastavenia: pripojenie k sieti, úroveň hlasitosti, jas obrazovky či správu notifikácií.



Tip: Postranný panel s kľúčovými tlačidlami zobrazíme rýchlo pomocou klávesovej skratky Windows+C. Dovedna s panelom je vždy zobrazený aj menší obdĺžnikový región, v ktorom sa objavujú informácie o systémovom čase, dátume, aktívnych sieťových pripojeniach a stave batérie počítača.

Keď sa kurzor myši dostane do ľavého spodného rohu pracovnej plochy, zviditeľní sa tlačidlo na prechod do nového Windows-rozhrania (obr. 1.4).



Obr. 1.4: Tranzit do nového rozhrania systému Windows 8

Stačí jedno kliknutie a razom sa ocitneme v prostredí pásu dlaždíc a WinRT-programov.



Tip: Keď v ľavom spodnom horúcom rohu stisneme pravé tlačidlo myši, uvidíme kontextovú ponuku s užitočnými odkazmi (obr. 1.5).



Obr. 1.5: Kontextová ponuka s užitočnými odkazmi

Vďaka týmto odkazom môžeme rýchlo spravovať sieťové pripojenia, upravovať možnosti napájania, spustiť správcu úloh alebo naštartovať prieskumníka operačného systému.

1.3 Elektronický obchod Windows

S operačným systémom Windows 8 prichádza aj nový elektronický obchod Windows. Ten bude predstavovať počítačový mega-market s tisíckami softvérových produktov, ktoré budú konformné s novým Windows-rozhraním, alebo budú pôsobiť ako bežné programy so štandardným vizuálnym rozhraním. Zatiaľ čo cieľovým prostredím všetkých nových programov bude nové Windows-rozhranie, cieľovým prostredím ostatných programov bude rozhranie Aero.

Vzhľadom na vysoko dominantnú trhovú penetráciu operačného systému Windows 8 bude elektronický obchod Windows schopný maximalizovať zásah požadovaných trhových segmentov produktmi vývojárov. V tejto chvíli spoločnosť Microsoft hlási, že vývojárom bude po spustení ostrej prevádzky obchodu k dispozícii 231 trhových segmentov v 100 rôznych jazykových mutáciách. Štatistiky jasne hovoria o sile nového obchodného modelu: vývojármi vytvorené WinRT-programy si môže prevziať respektíve zakúpiť akýkoľvek záujemca z ktoréhokoľvek kontinentu našej planéty.

Okrem toho, obchod Windows ponúka vývojárom premyslené riadenie celého logistického procesu zhotovených programov. Pre programy v štýle WinRT bude obchod Windows jediným distribučným kanálom. Do predajného strediska obchodu Windows môžu byť nahraté nasledujúce typy programov:

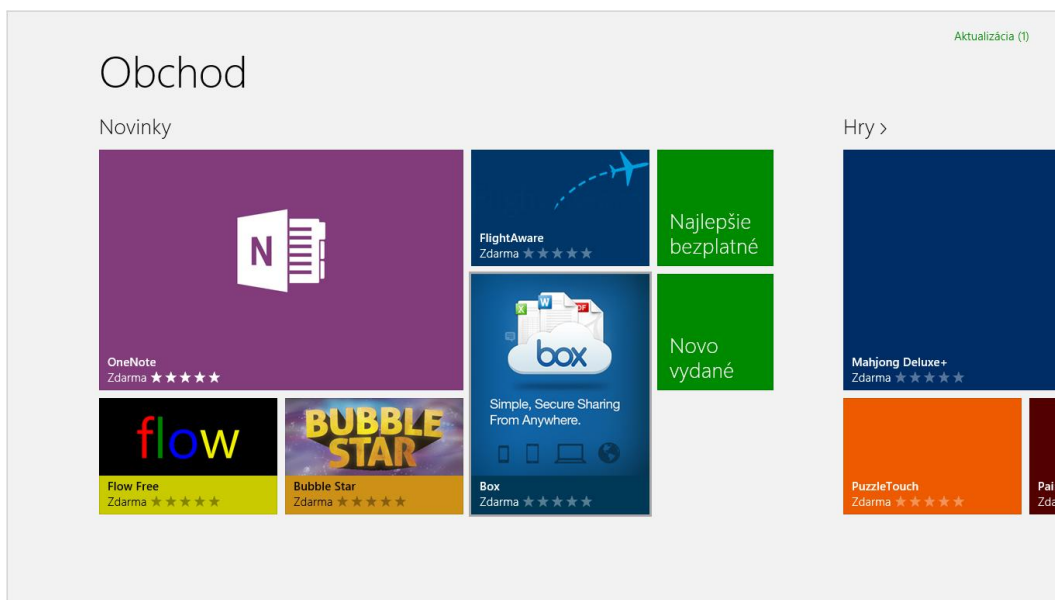
- **Voľne dostupné programy.** Tieto programy sú vývojármi ponúkané zadarmo a ich prevzatie nie je podmienené žiadnymi poplatkami.
- **Voľne dostupné verzie komerčných programov s časovým obmedzením funkčnosti.** Tieto programy fungujú na vopred jasne stanovený čas ako rýdzo komerčné softvérové produkty. Len čo uplynie skúšobná lehota, programy tohto typu prestávajú byť aj naďalej funkčné.
- **Voľne dostupné verzie komerčných programov s funkčným obmedzením.** Tieto programy nie sú obmedzené časovo, ale z pohľadu poskytovaných funkcií. Na rozdiel od svojich plnohodnotných komerčných ekvivalentov sú programy tejto kategórie funkčne obmedzené, takže poskytujú len zlomok portfólia kompletnej funkcionality.
- **Komerčné programy.** Tieto programy sú platené, pričom disponujú úplnou funkcionalitou ako aj všetkými ďalšími atribútmi, ktoré platia pre komerčné softvérové produkty.

1.3.1 Praktická ukážka prevzatia programu z obchodu Windows

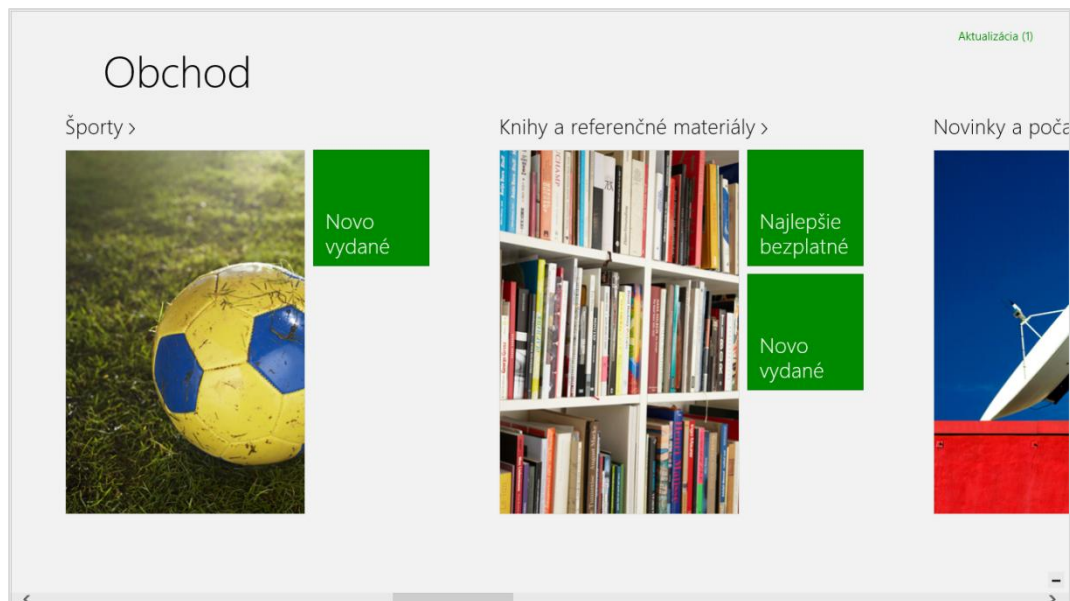
V čase písania tejto publikácie (august 2012) ešte nie je spustená ostrá prevádzka elektronického obchodu Windows. Obchod beží v testovacom režime, pričom ponúka iba voľne dostupné programy v štýle WinRT. Kvôli vyššej efektívite správy programov sú tieto členené do samostatne pôsobiacich homogénnych kategórií. Po vstupe do elektronického obchodu Windows uvidíme pás programových kategórií (obr. 1.6 – obr. 1.9).



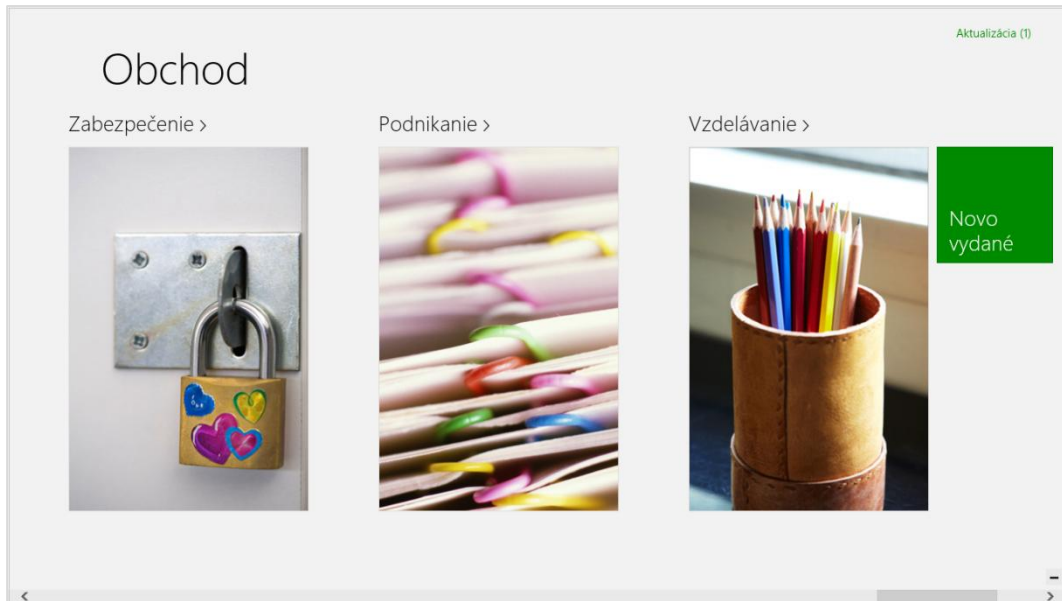
Obr. 1.6: Uvítacia obrazovka obchodu Windows



Obr. 1.7: Pás programových kategórií v prostredí obchodu Windows (1. fáza)



Obr. 1.8: Pás programových kategórií v prostredí obchodu Windows (2. fáza)

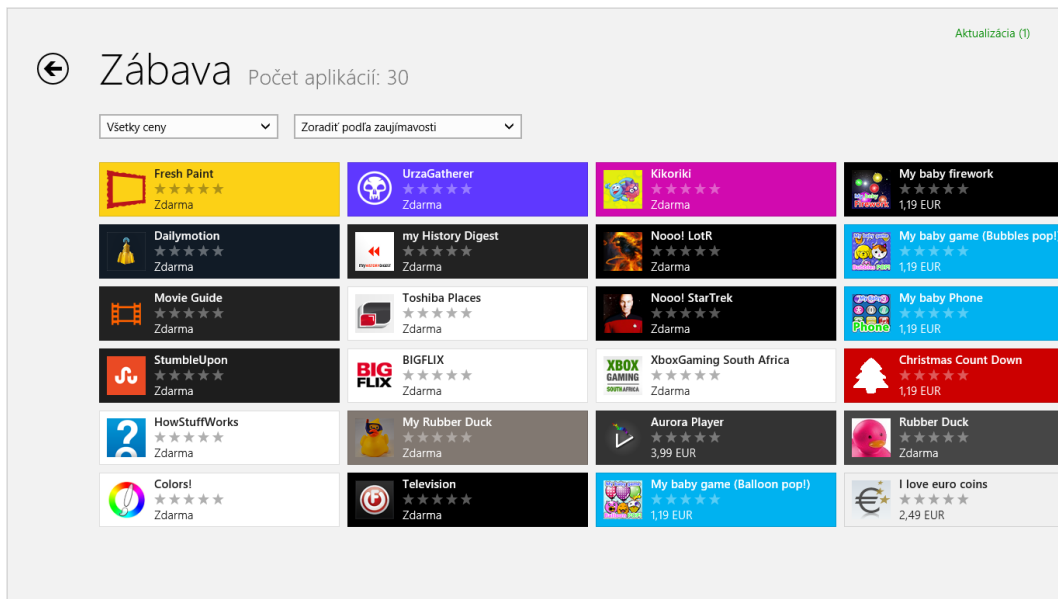


Obr. 1.9: Pás programových kategórií v prostredí obchodu Windows (3. fáza)

Zaujímavé je, že elektronický obchod Windows sa sám podobá na program v štýle WinRT. Obsahuje veľké dlaždice (s ilustračnými obrázkami a jednotným typom písma), ktoré sú kompozične naukladané tak, aby tvorili jeden súvislý pás. Po páse môžeme rolovať buď dotykovým gestom, myšou, alebo klávesnicou.

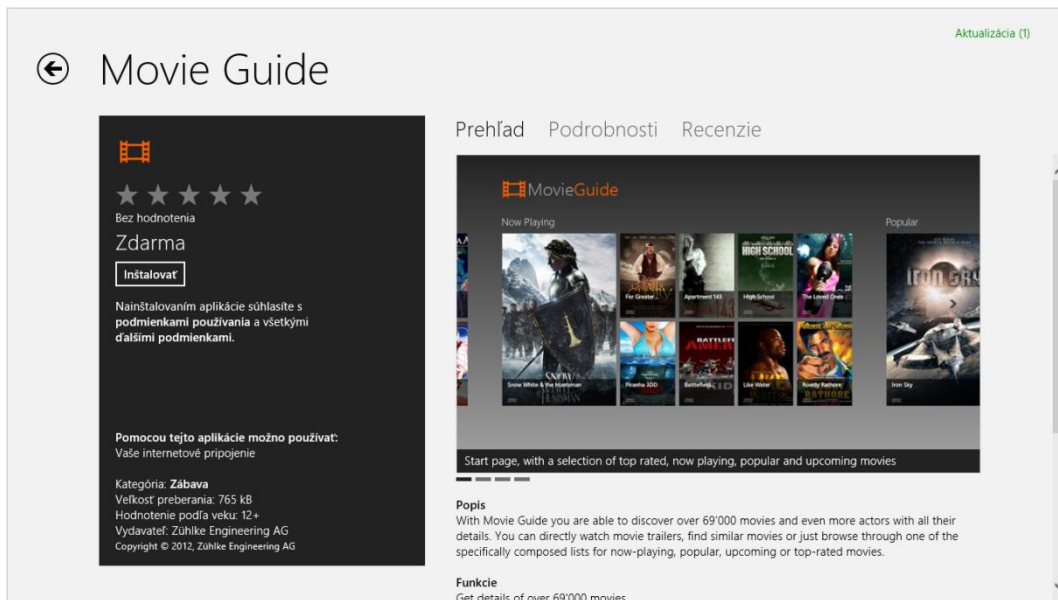
Prvá kategória obchodu Windows Store nesie meno **Novinky** a prezentuje najlepšie najnovšie programy, ktoré by zákazníci určite nemali minúť. Za kategóriou **Novinky** nasledujú ďalšie nemenej interesantné kategórie ako sú **Hry, Zábava, Fotografie, Hudba a video** a iné.

Povedzme, že sa vydáme bližšie preskúmať programovú kategóriu **Zábava** (obr. 1.10).


Obr. 1.10: Inšpekcia programovej kategórie **Zábava**

Každá kategória poskytuje používateľom miniatúrne náhľady všetkých programov, ktoré sa v tej ktorej kategórii vyskytujú. Presnejšie povedané, v kategórii **Zábava** je dostupných dovedna 30 programov v štýle WinRT. Jednotlivým programom prináležia dlaždice podávajúce základné informácie, ako je názov programu, barometer jeho obľúbenosti a dosiahnuteľnosť programu (teda vyjadrenie skutočnosti, či ide o voľný, alebo platený program).

Ďalej opíšeme proces prevzatia, inštalácie a použitia WinRT-programu s názvom **Movie Guide**. Tento program združuje informácie o rozmanitých filmových tituloch. Po výbere dlaždice s názvom programu sa pred nami rozprestrie jeho základná karta (obr. 1.11).


Obr. 1.11: Základná karta programu **Movie Guide**

Z informácií, ktoré nám základná karta programu **Movie Guide** poskytuje, sa dozvedáme, že tento program je voľne dostupný, má právo na využívanie internetového pripojenia počítačového systému a zatiaľ nebol hodnotený používateľmi.

Keď sa prepne na kartu **Podrobnosti**, zistíme, že program **Movie Guide** môže byť spracovaný na 32- a 64-bitových procesorových architektúrach typu x86, x64 a ARM. Spomínaná karta nám rovnako prezradí, že program **Movie Guide** je aktuálne dostupný iba v anglickej jazykovej mutácii.

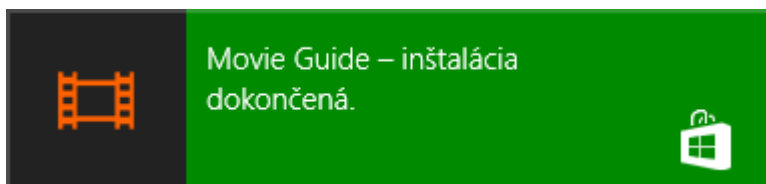
Používatelia, ktorí si WinRT-programy už stihli nainštalovať na svoje počítače, ich môžu hodnotiť. Zbierku reakcií, názorov a kanálov spätnej väzby objavíme pod poslednou kartou **Recenzie**. Poznamenajme, že komentáre používateľov sa dajú filtrovať podľa rôznych preferencií (napríklad podľa najvyššieho skóre obľúbenosti programu).

Inštalačný proces programu sa začne vo chvíli, keď klepneme na tlačidlo **Inštalovať**. Prevzatie programu a jeho nainštalovanie sú akcie, ktoré sú vykonané za niekoľko málo okamihov.



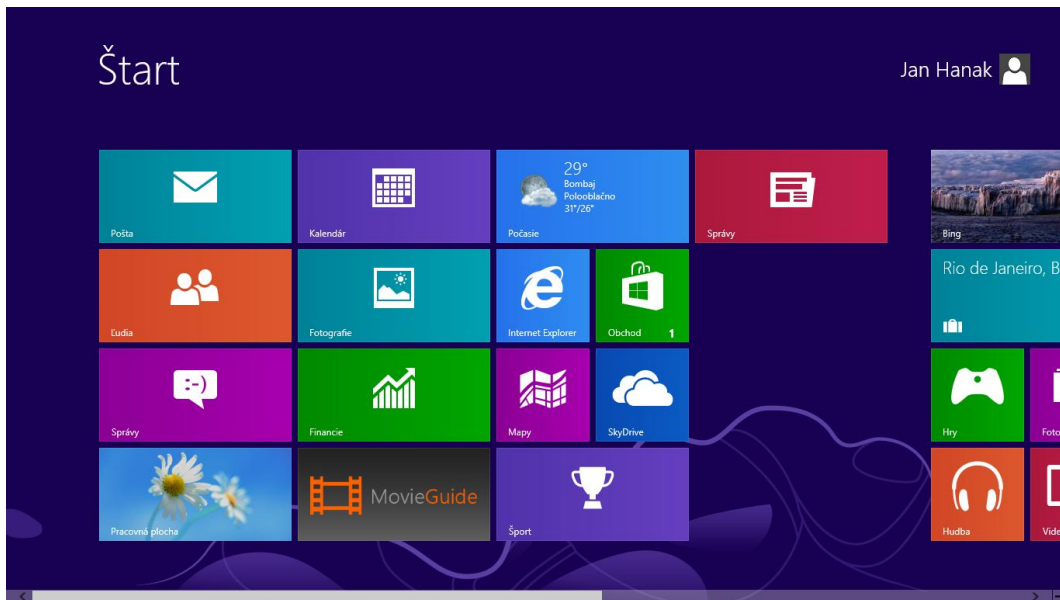
Poznámka: V tomto smere je promptne aplikovaná smernica spoločnosti Microsoft, ktorá konštatuje, že proces inštalácie nového WinRT-programu z elektronického obchodu Windows by mal trvať iba niekoľko sekúnd. Takýto prístup je rozumný, pretože čím skôr je program prevzatý a nainštalovaný, tým skôr s ním používateľ môže začať pracovať. Navyše, rýchla inštalácia zvyšuje mieru subjektívnej spokojnosti používateľa s programom.

Po skončení inštalácie sa v pravom hornom rohu obrazovky objaví správa o úspešne vykonanej inštalácii programu v štýle WinRT (obr. 1.12).



Obr. 1.12: Notifikácia o zdarene uskutočnenej inštalácii WinRT-programu

Program v štýle WinRT sa po svojom nainštalovaní usídli na páse dlaždíc. My sme zmenili polohu dlaždice tohto programu, a to tak, aby bola bližšie k začiatku pásu dlaždíc (obr. 1.13).

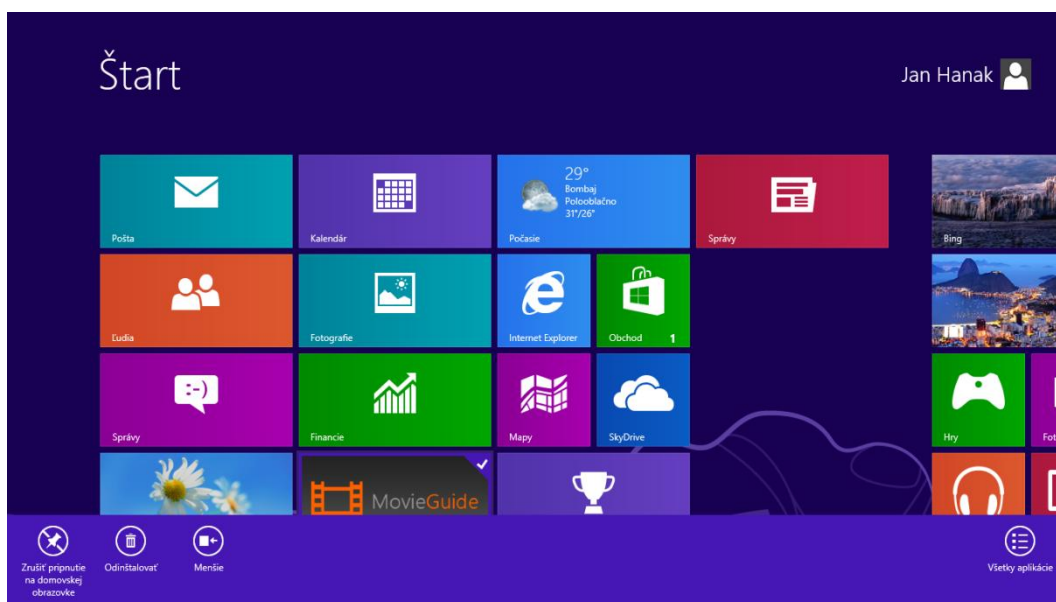


Obr. 1.13: Dlaždica programu **Movie Guide** na páse dlaždíc

Keď na dlaždicu programu **Movie Guide** klikneme pravým tlačidlom myši (alebo vyvoláme ekvivalentné dotykové gesto), v spodnej časti obrazovky sa zviditeľní aplikačný panel (angl. *application bar* alebo len *app-bar*). V súvislosti s programom **Movie Guide** nám nové rozhranie systému Windows 8 dovoľuje vykonať tri kontextovo viazané akcie:

- Zrušiť pripnutie programu na domovskej obrazovke.
- Odinštalovať program.
- Zmenšiť dlaždicu programu.

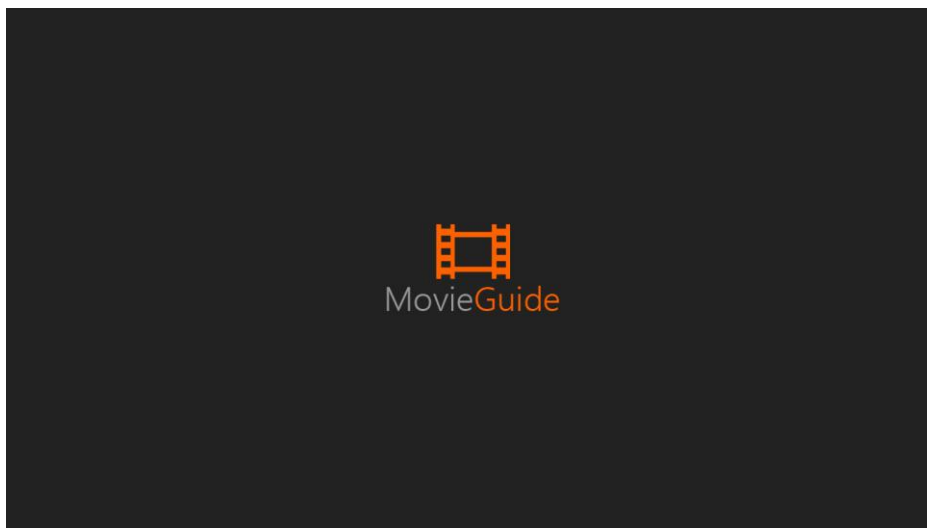
Aplikačný panel je znázornený na obr. 1.14.



Obr. 1.14: Aplikačný panel s možnosťami pre program **Movie Guide**

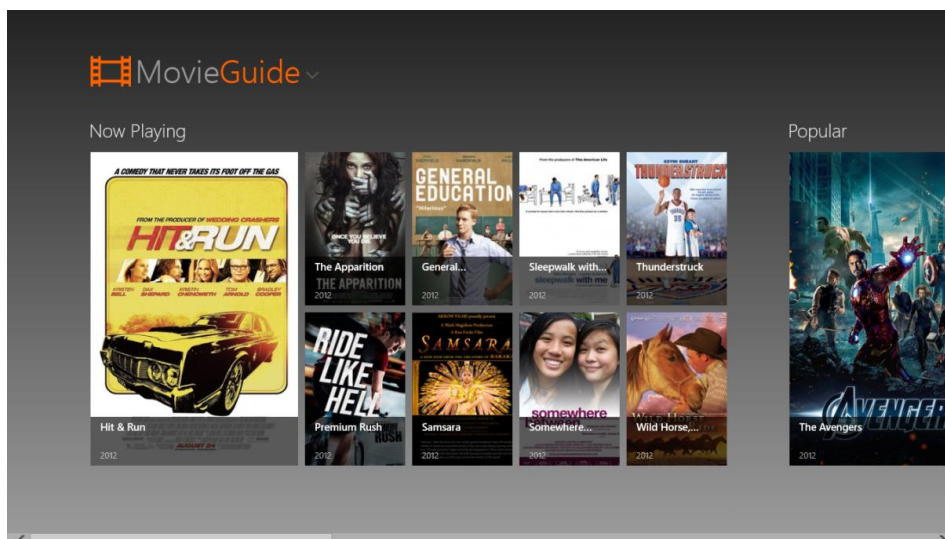
1.3.2 Použitie programu získaného z obchodu Windows

Program **Movie Guide** spustíme klepnutím na jeho dlaždicu. Implicitne program beží v režime celej obrazovky, čoho dôkazom je už úvodná obrazovka programu (angl. *splash-screen*), ako dokumentuje obr. 1.15.



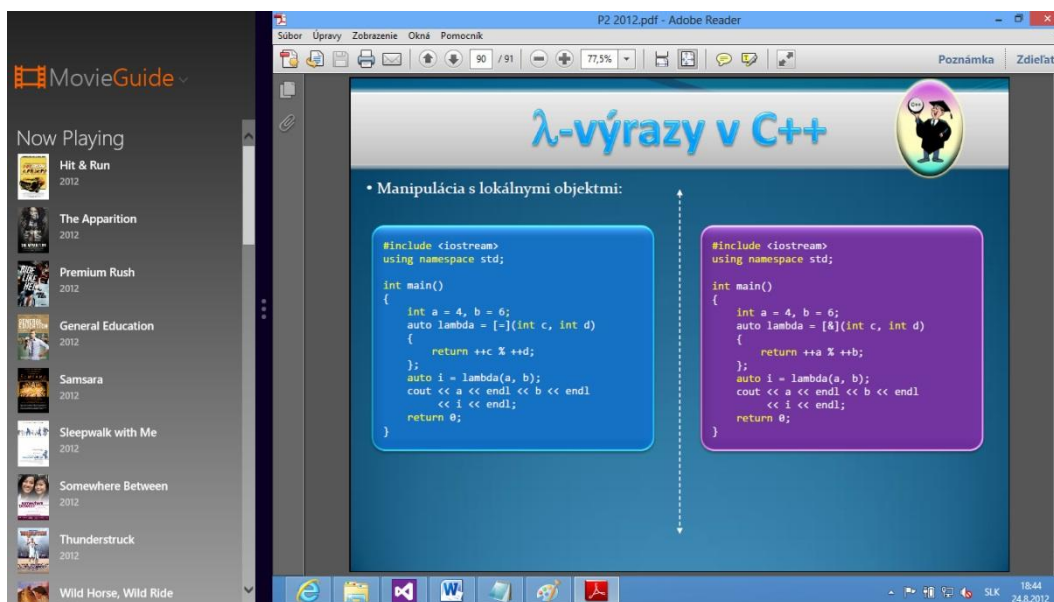
Obr. 1.15: Uvítacia obrazovka WinRT-programu **Movie Guide**

Vizuálny motív programu je generovaný mriežkovým rozvrhnutím v štýle nového Windows-rozhrania (obr. 1.16).



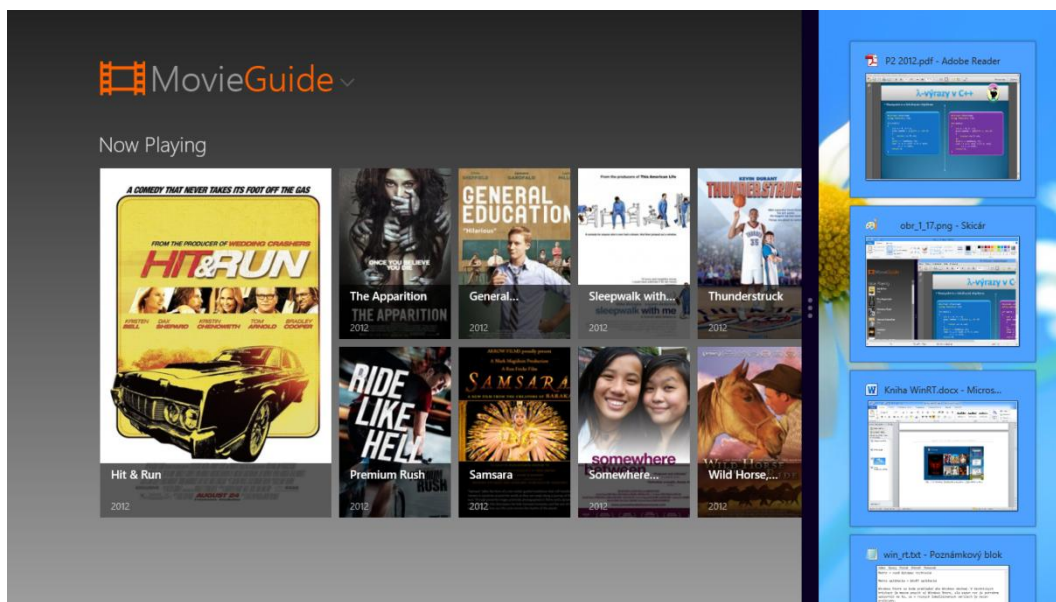
Obr. 1.16: Grafické používateľské rozhranie programu v štýle WinRT

Ako používatelia sa môžeme posúvať po jednotlivých bunkách mriežky, v ktorých sa nachádzajú filmové plagáty s ich upútavkami. Program **Movie Guide** (a v skutočnosti ktorýkoľvek program, ktorý je konformný so zásadami nového Windows-rozhrania) smie byť zobrazený paralelne s iným programom. Na obr. 1.17 pozorujeme koexistenciu filmového programu **Movie Guide** s prezentáciou zobrazenou v aplikácii Adobe Reader.



Obr. 1.17: Paralelné zobrazenie dvoch programov vedľa seba

V tejto situácii používatelia vidia dva programy, a teda aj dve prezentačné vrstvy týchto programov. Zatiaľ čo naľavo je situovaný program v štýle nového Windows-rozhrania, na pravej strane beží štandardný program v štýle rozhrania Aero. Samozrejme, veľkosti zobrazovacích plôch pre zúčastnené programy sú konfigurovateľné. Hoci WinRT-program na obr. 1.17 zaberá približne jednu štvrtinu obrazovky, nie je problém dodať mu väčšiu zobrazovaciu kapacitu. Pochopiteľne však v tomto smere platí princíp nepriamej úmery, teda o čo zväčšíme plochu pre WinRT-program, o to budeme musieť zmenšiť plochu pre program rozhrania Aero (obr. 1.18).



Obr. 1.18: Paralelné zobrazenie programov v štýle WinRT a Aero



Kapitola 2

Paradigmy a technológie na vývoj
programov v štýle WinRT

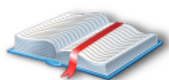
2 Paradigmy a technológie na vývoj programov v štýle WinRT

Spoločnosť Microsoft pripravila pre vývojárov programov v štýle WinRT sofistikované technologické a nástrojové portfólio, ktoré im umožní plánovať, navrhovať, programovať a nasadzovať zbrusu nové aplikácie určené pre operačný systém Windows 8. Podpora tvorcov softvérových produktov je veľmi široká, pretože firma Microsoft myslí na rozmanité pracovné tímy vývojárov, ktoré využívajú rôzne programovacie jazyky, technológie a paradigmy.

Keďže spoločnosť Microsoft uplatňuje stratégiu kompletného pokrytia všetkých dôležitých vývojárskych segmentov, môžeme diferencovať nasledujúce paradigmy a s nimi spriaznené technológie, ktoré nám umožnia vytvárať programy v štýle WinRT:

1. **Vývoj riadených programov v štýle WinRT použitím programovacích jazykov C# a Visual Basic.** Spoločne s operačným systémom Windows 8 je uvedená nová platforma Microsoft .NET Framework (vo verzii 4.5) a, prirodzene, rovnako aj nové prostredie pre vývojárov Microsoft Visual Studio 2012. Programátori, tvoriaci aplikácie pre platformu .NET Framework, pracujú najviac s programovacími jazykmi C# a Visual Basic. Výstupmi prekladačov a spojovacích programov týchto programovacích jazykov sú riadené programy. Adjektívum „riadené“ vyjadruje skutočnosť, že tieto programy sú spracúvané pomocou virtuálneho exekučného systému CLR, ktorý poskytuje riadeným programom všetky požadované nízkoúrovňové služby (akými sú správa aplikačných domén, manažment programových vlákien, riadenie dynamických pamäťových vrstiev atď.). Programovať aplikácie v štýle WinRT dokážu vývojári, ktorí sú znalí princípov objektovo orientovaného programovania v prostredí jazyka C# alebo Visual Basic. V súvislosti s platformou .NET Framework 4.5 využijú vývojári najnovšie verzie uvedených programovacích jazykov: C# 5.0 a Visual Basic 2012.

2. **Vývoj natívnych programov v štýle WinRT použitím programovacieho jazyka C++.** V balíku Microsoft Visual Studio 2012 je prítomný produkt Visual C++ 2012, ktorý implementuje programovací jazyk C++. Implementácia jazyka C++ je konformná s najnovším ISO štandardom pre tento jazyk. Finálny ISO štandard pre jazyk C++ je známy ako C++11 a bol publikovaný v septembri 2011 ako ISO/IEC 14882:2011.



Poznámka: Produkt Visual C++ 2012 obsahuje prekladače pre dovedna 4 programovacie jazyky:

1. Prekladač pre programovací jazyk C (implementácia jazyka je parciálne konformná s ISO štandardom C99).
2. Prekladač pre programovací jazyk C++ (implementácia jazyka je parciálne konformná s ISO štandardom C++11).
3. Prekladač pre programovací jazyk C++, ktorý obsahuje nové natívne rozšírenia Component Extensions, v skratke označované ako C++/CX.
4. Prekladač pre programovací jazyk C++/CLI (implementácia jazyka je úplne konformná so štandardom ECMA-372).

Na vývoj natívnych WinRT-programov je priamo určený programovací jazyk C++, a to buď samostatne, alebo v kooperácii s rozšíreniami C++/CX, ktoré preukazujú svoje silné stránky pri budovaní programov so zacielením na platformu Windows Runtime (alebo skrátené WinRT).

Vývoj natívnych programov v štýle WinRT použitím jazyka C++ má, najmä v priamej komparácii s prípravou štandardných programov s vizuálnym rozhraním, svoje špecifiká (tie bližšie rozoberieme v jednej z nasledujúcich kapitol tejto knihy). V tejto chvíli podotknime, že v záujme tvorby WinRT-programov sa musia vývojári v jazyku C++ oboznámiť nielen s novými obsahovo centrickými dizajnerskými princípmi, ale aj s jazykom XAML a v neposlednej miere tiež s novou platformou WinRT.

3. **Vývoj natívnych programov v štýle WinRT použitím webových technológií (HTML5 + CSS3 + JavaScript).** Weboví programátori môžu opätovne využiť svoje technické know-how pri vývoji natívnych programov, ktoré budú konformné s novým rozhraním systému Windows 8. Základy tejto vývojovej paradigmy spočívajú na deklaratívnom jazyku HTML5, kaskádových štýloch CSS3 a skriptovacím programovacím jazyku JavaScript. Zatiaľ čo vizuálne motívy programu v štýle WinRT navrhujeme pomocou konštrukcií HTML5 a CSS3, algoritmy definujúce vzorce správania programu implementujeme pomocou jazyka JavaScript.

Pravdu povediac, pri porovnaní metodík vývoja bežnej webovej aplikácie a natívneho programu v štýle WinRT nájdeme viacero spoločných rysov. Napríklad, aj WinRT-program môže dynamicky získavať dáta z webových služieb a spracúvať ich v podobe notifikácií. Alebo, aj WinRT-program dokáže komunikovať v modeli klient-server s inými WinRT-programami či vzdialenými webovými servermi a procesmi, ktoré na týchto serveroch bežia. Na druhej strane, programovanie aplikácií v štýle WinRT má svoje špecifiká, s ktorými sa spravidla nestretávame pri budovaní štandardných webových programov.

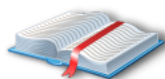
Snáď najväčšou devízou programov v štýle WinRT je ich dotykové ovládanie. Poznamenajme, že dotykové ovládanie nie je v kontexte WinRT-programov ponímané ako doplnkový, ale priam ako fundamentálny spôsob interakcie človeka s počítačom. Druhým pozitívom WinRT-programov je ich silná väzba na platformu WinRT. Algoritmy formujúce aplikačnú logiku programu môžu využívať schopností platformy WinRT aj prostredníctvom jazyka JavaScript (v skutočnosti je platforma WinRT prístupná pre ktorýkoľvek z podporovaných programovacích jazykov, vrátane Visual Basicu, C# a C++). V neposlednom rade je vývojárom v jazyku JavaScript k dispozícii knižnica Windows Library for JavaScript (známa aj ako WinJS). Knižnica WinJS obsahuje súpravu CSS3 štýlov a programových súborov jazyka JavaScript. Tieto združujú triedy, ktorých objekty sa môžu vyskytovať vo vizuálnych motívoch rozhraní programov v štýle WinRT. Knižnica

WinJS ponúka veľa užitočných objektov (ako napríklad **DatePicker**, **FlipView** či **ListView**), ktoré nie sú súčasťou HTML5.

4. **Vývoj natívnych programov v štýle WinRT použitím programovacieho jazyka C++ a platformy Microsoft DirectX.** Hoci štandardný WinRT-program vieme vyvinúť v kombinácii programovacieho jazyka C++ a platformy WinRT, existuje aj ďalšia paradigma, ktorá sa koncentruje na vývoj programov v štýle WinRT použitím jazyka C++ a platformy Microsoft DirectX. Ako všetci pokročilí programátori dobre vedia, platforma DirectX sa používa pri vývoji robustných počítačových hier a na efekty intenzívnych multimediálnych aplikácií využívajúcich pokročilé techniky v oblasti rovinnej (2D) a priestorovej (3D) počítačovej grafiky. V prípade, keď je vašim cieľom konštrukcia počítačovej hry alebo inej na grafiku náročnej aplikácie pre operačný systém Windows 8 a jeho nové vizuálne rozhranie, odporúčame vám aplikovať paradigmu C++ a DirectX.

Je však nutné podotknúť, že vývoj počítačových hier v štýle WinRT je náročnou programátorskou disciplínou. K základným prerekvizitám programátora ašpirujúceho na zvládnutie tejto úlohy patria znalosti z nasledujúcich oblastí:

- Pokročilé programovacie techniky v jazyku C++.
- Pokročilé objektové programovanie v prostredí operačného systému Windows 8.
- Pokročilé matematické znalosti princípov vykresľovania scén a manipulácie s objektmi v 2D a 3D priestore.
- Vybrané znalosti z oblasti počítačovej grafiky (práca s bitovými mapami, textúrami, drôtovými modelmi a ich pohybmi, respektíve transformáciami ich pozícií v priestore).

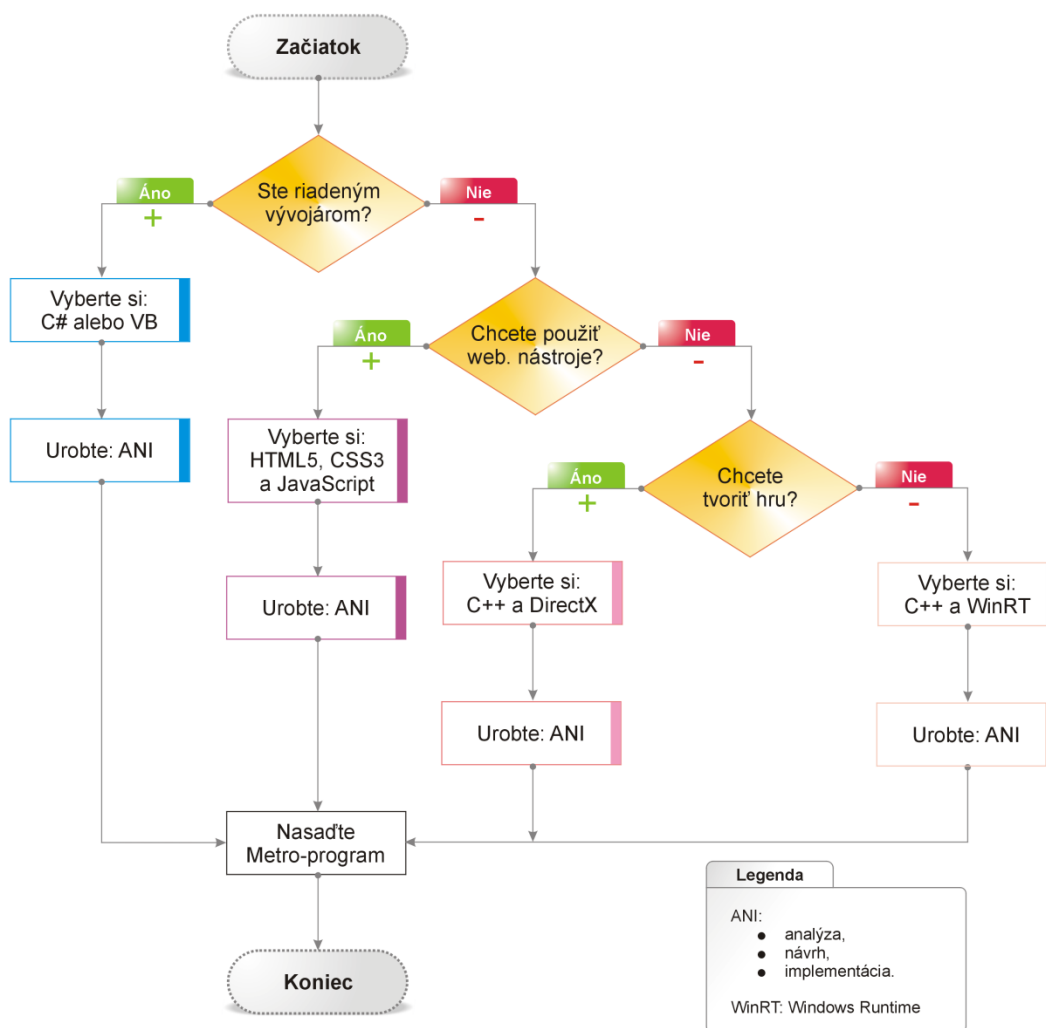


Poznámka: Pri vývoji natívnych programov v štýle WinRT pomocou platformy DirectX sa využíva najaktuálnejšia verzia tejto platformy, ktorou bola v čase tvorby tejto publikácie verzia DirectX 11.1.

Platforma DirectX zahŕňa viacero knižníc, ktoré bližšie charakterizuje tab. 2.1.

Tab. 2.1: Prehľad a charakteristika knižníc tvoriacich platformu Microsoft DirectX	
Knižnica	Charakteristika
Direct3D	Knižnica je určená na hardvérovo akcelerované vykresľovanie trojrozmernej (3D) počítačovej grafiky.
Direct2D	Knižnica slúži na hardvérovo akcelerované vykresľovanie rastrovej a vektorovej dvojrozmernej (2D) grafiky.
DirectXMath	Knižnica obsahuje funkcie na podporu realizácie rutinných matematických operácií, ktoré súvisia s grafickými primitívami vyskytujúcimi sa v 2D- a 3D-počítačovej grafike. Ide najmä o softvérovú podporu vykonávania vektorových a maticových transformácií.
DirectWrite	Knižnica sa využíva predovšetkým na hardvérovo akcelerované vykresľovanie textových regiónov a motívov vizuálneho rozhrania programu.
XAudio2	Knižnica implementuje na nízkej úrovni algoritmy, ktoré uskutočňujú spracovanie audio signálov pre potreby dodania zvukových efektov a melódií do programu.
XInput	Knižnica podporuje prácu s periférnymi zariadeniami, ktoré sú určené na hranie hier.

Na obr. 2.1 uvádzame vývojový diagram, ktorý vám pomôže vybrať konkrétnu paradigmu, technológie a nástroje na vývoj programov v štýle WinRT presne podľa vášho odborného a profesionálneho profilu.



Obr. 2.1: Selekcia vývojovej paradigmy, spriaznených technológií a nástrojov



Kapitola 3

Životný cyklus programu v štýle WinRT

3 Životný cyklus programu v štýle WinRT

V tejto kapitole preskúmame životný cyklus programu v štýle WinRT. Analýzu pritom upriamime na dva variantné pohľady skúmania životného cyklu WinRT-programu:

1. Životný cyklus vývoja programu v štýle WinRT.
2. Životný cyklus spracovania programu v štýle WinRT.

3.1 Životný cyklus vývoja programu v štýle WinRT

Napriek tomu, že vývoj programu, ktorý je konformný s novým vizuálnym rozhraním operačného systému Microsoft Windows 8, je špecifický, stále možno vývojový cyklus opísať štandardnou metodikou vývoja počítačového softvéru. Táto metodika je zložená z nasledujúcich vývojových etáp:

1. Analýza požiadaviek na WinRT-program.
2. Návrh softvérovej architektúry WinRT-programu.
3. Implementácia navrhutej softvérovej architektúry WinRT-programu so zvolenými programovacím jazykmi, nástrojmi, platformami a technológiami.
4. Testovanie WinRT-programu.
5. Nasadenie a predaj WinRT-programu.

V nasledujúcich častiach podkapitol vyložíme všetky etapy životného cyklu vývoja programu v štýle WinRT.

3.1.1 Analýza požiadaviek na program v štýle WinRT

V prvej etape vývojového procesu sa sústredíme na analytické riešenie viacerých problémových domén. Jadrom tejto činnosti je špecifikácia funkčných a nefunkčných

požiadaviek na budúci WinRT-program. V analytickej fáze je našim cieľom spracovať jasnú charakteristiku základných črt budúceho softvérového produktu:

1. **Účel existencie WinRT-programu.** V tomto smere hľadáme odpovede na tieto otázky: prečo chceme program vôbec zhotoviť, na čo bude program slúžiť a aké potreby používateľov bude program uspokojovať. Napríklad, keď budeme chcieť vytvoriť tréningového manažéra, je zrejmé, že primárnym cieľom sa stane snaha o počítačom riadené spracovanie tréningových aktivít používateľa. Kým doteraz používateľ uchovával svoje tréningové dáta manuálne, nový program mu poskytne komplexné možnosti nielen pri archivácii vstupných dát, ale tiež pri generovaní výkonnostných štatistik a tréningových plánov, a to presne podľa typológie používateľa a konfigurácie jeho športového profilu.
2. **Vymedzenie procesov, ktoré bude WinRT-program automatizovať.** Úlohy, ktoré program vykonáva, verne mapujú reálne procesy, v ktorých sa ocitajú objekty, s ktorými prichádzajú používatelia tohto programu do vzájomných interakčných väzieb. Softvérové inžinierstvo definuje viacero techník na účinné analyzovanie procesov, determinovanie aktérov týchto procesov a klasifikovanie väzieb nielen medzi aktérmi, ale aj procesmi, v ktorých sa tieto entity v reálnom svete vyskytujú. Keďže program v štýle WinRT je objektovým programom, najjednoduchším spôsobom je vymedziť dve základné množiny: množinu objektov a množinu procesov, v ktorých sa tieto objekty v čase vyskytujú. Napríklad, v prípade tréningového manažéra patria do množiny analyzovaných objektov typy športov realizovaných pri tréningových aktivitách (beh, bicyklovanie, plávanie, tenis atď.), zatiaľ čo procesmi sú jednotlivé športové tréningy s konkrétnymi dátami (povedzme, že bežec začal svoj tréning na dráhe dnes o desiatej hodine, pričom zabehol 5 kiloWinRTv s priemerným bežeckým tempom 5 minút a 45 sekúnd na kilometer bežeckej trasy).
3. **Výber vizuálnej koncepcie WinRT-programu.** Hoci programy v štýle WinRT sa od iných štandardných programov určených pre rozhranie Aero odlišujú už na prvý pohľad, môžeme ich ďalej diferencovať podľa implementovanej vizuálnej

konceptie. Pri navrhovaní grafického rozhrania nového WinRT-programu môžu vývojári využívať viaceré navigačné vzory. Tie sú nápomocné pri budovaní programov, ktorých interná vizuálna koncepcia je vertikálna (skladajúca sa z niekoľkých hierarchicky rozmiestnených vrstiev) alebo horizontálna (tvorená jednou vrstvou). Nech už si vyberieme ktorúkoľvek vizuálnu koncepciu, vždy pracujeme tak, aby sme boli konformní s mottom „rýchlosť a plynulosť so zacielením na obsah“.

4. **Voľba cieľového publika používateľov WinRT-programu.** V informatike existujú desiatky rôznych typov informačných systémov, programov a softvérových aplikácií. Každý z týchto produktov je určený pre špecifický cieľový segment používateľov. Tréningový manažér je program v štýle WinRT, ktorý je osožný predovšetkým pre amatérskych a profesionálnych športovcov, rovnako ako aj pre ľudí vyznávajúcich aktívny životný štýl. Všetky spomenuté osoby dovedna tvoria cieľový segment používateľov tréningového manažéra.
5. **Konkurenčná analýza existujúcich WinRT-programov, ktoré patria do rovnakej kategórie typového aplikačného softvéru.** Používatelia, softvérové firmy a počítačové programy – to všetko sú kategórie, ktoré majú fixnú povahu: na tomto svete je ich vždy konečný počet. Po tom, čo sme položili základnú koncepciu nového WinRT-programu a určili cieľový segment jeho používateľov, je nanajvýš vhodné zoznámiť sa s tými programami, ktoré vystupujú voči nášmu programu ako priami konkurenti. Termínom „priamy konkurent“ chápeme program rovnakého zamerania a so zacielením na identický segment používateľov. Zmyslom konkurenčnej analýzy je zostaviť prehľad programov iných softvérových spoločností a ohodnotiť tieto programy podľa vopred determinovaných kritérií aj s váhami ich dôležitosti. Napokon získame súpis silných a slabých stránok konkurenčných programov. Podľa neho môžeme potom doplniť, respektíve rozšíriť množinu funkčných a nefunkčných požiadaviek na budúci program v štýle WinRT. To nám umožní vytvoriť program s viacerými konkurenčnými výhodami.

3.1.2 Návrh softvérovej architektúry programu v štýle WinRT

Ako sme už uviedli, pri vyvíjaní programov v štýle WinRT aplikujeme objektový paradigmu. Každý WinRT-program je platným objektovým programom, čo znamená, že je zložený z množiny objektov, ktoré disponujú svojimi stavmi (a prechodmi medzi nimi) a ktoré medzi sebou komunikujú pomocou zasielania a prijímania správ. Softvéroví architekti pri návrhu architektúry objektového programu zhotovujú dokumenty, odrážajúce variantné pohľady na architektúru tohto programu. Pre potreby tejto publikácie budeme bližšie charakterizovať dva pohľady na WinRT-program: statický pohľad a dynamický pohľad.

Statický pohľad na WinRT-program vyjadruje hierarchiu a vzťahy medzi triedami (alebo aj inými objektovými typmi), ktoré formujú základnú objektovú architektúru tohto programu. Architekti teda navrhujú triedy, definujú funkcionality inštancií týchto tried, analyzujú a nadväzujú väzby medzi triedami (buď na základe agregácie a kompozície, alebo dedičnosti), rozhodujú o výbere a implementácii rozhraní, detegujú pozitívne miesta na začlenenie polymorfizmu a uskutočňujú ďalšie činnosti. Na konci týchto aktivít je hotový statický diagram tried, ktorý predstavuje statický pohľad na objektovú architektúru programu v štýle WinRT.

Dynamických pohľadov na objektovú architektúru programu v štýle WinRT môže existovať viacero. Atribút dynamiky naznačuje skutočnosť, že budeme sledovať prácu objektov programu v čase, prípadne prácu používateľa s programom, a to rovnako v istom časovom intervale. V prvom prípade softvéroví architekti analyzujú a navrhujú interakcie medzi objektmi programu a definujú zmeny stavov týchto objektov. S každým objektom je asociovaný stavový stroj, ktorý na požiadanie vygeneruje stavový diagram objektu so všetkými jeho prípustnými stavmi. Objekt sa počas svojho života dostáva do rôznych stavov. Transport z jedného stavu objektu do iného stavu je spravidla uskutočnený na základe správy, ktorú objekt prijme a spracuje (pôvodcom tejto správy môže byť iný objekt v blízkom okolí skúmaného objektu, alebo aj sám používateľ). Keď nanesieme zmeny stavov objektu na časovú krivku, získame dynamický pohľad na správanie objektu v čase. Ak túto akciu realizujeme pre všetky objekty programu v štýle

WinRT, získame štýl správania programu. Samozrejme, spôsob, akým sa program správa, je závislý od požiadaviek používateľa. Tak sa dostávame k druhému dynamickému pohľadu na WinRT-program, ktorý generuje používateľský profil. Program v štýle WinRT má mnoho nových vlastností, ktoré ovplyvňujú aj prácu používateľa s ním. Napríklad, vďaka dynamickým dlaždiciam môže program informovať používateľa o dôležitých správach bez toho, aby musel používateľ tento program aktivovať. Pochopiteľne, program stále beží v pozadí, reaguje na správy, uskutočňuje výpočty či realizuje ďalšie akcie. V tomto smere sa použitie programu v štýle WinRT ponáša na prácu programov známych z prostredia mobilných telefónov a komunikátorov.

Softvéroví architekti majú k dispozícii nástroje, ktoré im umožnia naprojektovať celú architektúru WinRT-programu. Keď je našou intenciou zhotoviť architektúru programu, potrebujeme v zásade dve základné veci: štandardizovaný jazyk a s ním spojenú metodiku. Najpopulárnejším jazykom na tvorbu architektonických riešení softvérových produktov je v súčasnosti jazyk UML (angl. *Unified Modeling Language*). Ten poskytuje grafické objekty, notácie, štýly a techniky na prípravu statického a dynamického pohľadu na budúci program v štýle WinRT. Sprievodnou metodikou je obvykle agilná metodika ako napríklad SCRUM, ktorá vraví, ako efektívne zvládnuť všetky štádia návrhu počítačového softvéru.

Finálnym produktom softvérových architektov je exaktná dokumentácia, ktorá nielenže opisuje, ako bude nový program v štýle WinRT vyzerat', ale aj ako sa bude správať a ako bude reagovať na požiadavky používateľa.

3.1.3 Implementácia navrhutej softvérovej architektúry programu v štýle WinRT

Keď je softvérová architektúra WinRT-programu dokončená, je odovzdaná tímu vývojárov, ktorí sú zodpovední za jej implementáciu. V tejto etape ide o naprogramovanie celého programu. Pokiaľ je program robustný, možno jeho vývoj (s diferenciáciou na spriaznené množiny komponentov) delegovať na viaceré skupiny vývojového tímu. Programátori

vyvíjajú nový WinRT-program vo vybratom programovacom jazyku a so zvolenými nástrojmi, platformami a technológiami. V druhej kapitole tejto knihy sme podrobne rozobrali viaceré paradigmy a technológie, ktoré môžu vývojári uplatniť pri programovaní počítačovej aplikácie pre nové Windows-rozhnanie. Na tomto mieste teda konštatujeme, že WinRT-programy môžeme vytvárať buď ako:

- riadené – v programovacích jazykoch Visual Basic 2012 a C# 5.0,
- natívne – v programovacom jazyku C++ s podporou platformy WinRT,
- natívne – v programovacom jazyku C++ s podporou platformy DirectX,
- natívne – v programovacom jazyku JavaScript s podporou webových technológií (HTML5 a CSS3).

Vývojový proces WinRT-programu disponuje niekoľkými špecifikami, ktoré programátori dosiaľ nepovažovali za úplne bežné. Jedným z dôkazov predchádzajúceho tvrdenia, ktorý sa objavuje pri tvorbe programov pre nové rozhranie systému Windows 8, je kladenie veľkého dôrazu na asynchrónne programovanie. Zmyslom asynchrónneho programovania je maximalizácia elasticity programu na pokyny používateľa. Ak je asynchronizmus implementovaný do WinRT-programu správne, nikdy sa nemôže stať, že by tento program prestal spracúvať vstupy používateľa, alebo že by sa javil ako nečinný. Softvéroví inžinieri spoločnosti Microsoft vravia, že každá výpočtová operácia, ktorej realizácia presiahne 50 milisekúnd, by mala byť naprogramovaná ako asynchrónna. Dokonca, v básovej knižnici tried (BCL) sú mnohé metódy už „z výroby“ dodávané ako asynchrónne, pričom ich synchrónne ekvivalenty ani neexistujú. Pozitívom pre riadených programátorov je plná natívna podpora asynchrónneho programovania v jazykoch Visual Basic 2012 (pomocou kľúčových slov **Async** a **Await**) a C# 5.0 (pomocou kľúčových slov **async** a **await**).

V nasledujúcich kapitolách sa zoznámime s praktickou tvorbou riadených a natívnych programov v štýle WinRT, pričom budeme používať jazyky Visual Basic 2012, C# 5.0 a C++ (podľa štandardu C++11).

3.1.4 Testovanie programu v štýle WinRT

Programy v štýle WinRT sú nasadzované do elektronického obchodu Windows. Avšak skôr, ako sa môže WinRT-program úspešne dostať do tohto elektronického obchodu, musí prejsť testovacím a certifikačným procesom. Pre zdarné uskutočnenie validačného procesu musia byť splnené tieto náležitosti:

- Na počítači je nainštalovaný operačný systém Windows 8.
- Na počítači je nainštalovaná súprava Windows App Certification Kit, ktorá je súčasťou súpravy Windows Software Development Kit (SDK) pre programy v štýle WinRT.
- Počítač a vývojár na ňom pracujúci musia vlastniť platnú vývojársku licenciu.
- WinRT-program, ktorý má podstúpiť testovanie a certifikáciu, musí byť skompilovaný a spojený do priamo spustiteľného súboru.

Program Windows App Certification Kit vykonáva nasledujúce testy:

- Kontrola správnosti formátovania obsahu manifestu programu v štýle WinRT.
- Kontrola integrity zdrojov uvedených v manifeste programu v štýle WinRT.
- Kontrola spoľahlivosti programu v štýle WinRT (s detekciou kolíznych stavov uviaznutia či úplného zlyhania fungovania programu).
- Kontrola ostrého zostavenia programu v štýle WinRT.
- Kontrola správneho kódovania obsahov súborov tvoriacich zostavenie programu v štýle WinRT.
- Kontrola výkonnosti programu v štýle WinRT (je zameraná na diagnostiku toho, ako rýchlo sa program spúšťa či ako promptne dochádza k jeho suspendácii a následnému oživeniu).
- Kontrola použitia len kompatibilných API v kódovej báze programu v štýle WinRT.
- Kontrola bezpečnosti programu v štýle WinRT (pomocou nástrojov BinScope Binary Analyzer a Attack Surface Analyzer).

3.1.5 Nasadenie a predaj programu v štýle WinRT

Programy v štýle WinRT budú po svojom nasadení prístupné cez elektronický obchod Windows, a to buď ako voľne prístupné programy, alebo ako platené programy. Individuálny vývojár alebo vývojársky tím bude musieť mať v elektronickom obchode predplatený svoj účet. Výška predplatného by sa mala pochybovať v desiatkach eur na dobu jedného roka. Vývojári platených programov budú zarábať na každej jednej predanej licencii týchto softvérových produktov. Z celkových tržieb za predaj programu v štýle WinRT sa odpočíta poplatok za zaradenie programu do katalógu obchodu Windows a tiež aj miestne dane. Čistá čiastka za predaj programu bude nakoniec poukázaná na bankový účet individuálneho vývojára respektíve softvérovej firmy.

3.2 Životný cyklus spracovania programu v štýle WinRT

Používatelia získavajú programy v štýle WinRT z elektronického obchodu Windows. Je dôležité vedieť, že distribučná jednotka WinRT-programu obsahuje manifest a súbory, ktoré spoločne vytvárajú zostavenie tohto programu. Distribučná jednotka (alebo tiež aplikačný balíček) má súborovú koncovku .appx. Každý WinRT-program obsahuje svoj jedinečný manifest, ktorý uchováva informácie o kolekcii súborov, z ktorých sa tento program skladá. Manifest tiež stanovuje, či program v štýle WinRT požaduje prístup k chráneným zdrojom počítačového systému, prípadne k periférnym zariadeniam, ktoré sú k tomuto systému pripojené.

Na nasledujúcich riadkoch budeme skúmať životný cyklus spracovania programu v štýle WinRT. Spracovaním v tomto kontexte myslíme priamy beh WinRT-programu v prostredí operačného systému Windows 8.

Životný cyklus spracovania programu v štýle WinRT sa skladá z týchto etáp:

1. **Aktivácia programu v štýle WinRT.** Aktivácia je prvá akcia v životnom cykle WinRT-programu. Program je aktivovaný vždy, keď dôjde k jeho spusteniu používateľom. Pri spustení program zobrazuje uvítaciu obrazovku (angl. *splash screen*). Hoci s princípmi uvítacích obrazoviek sme konformní už dlhší čas, až pri stavbe programov v štýle WinRT sa táto softvérová črta dostáva skutočne do popredia. Koniec koncov, začlenenie uvítacej obrazovky prináša viaceré pozitívne efekty. Napríklad, počas zobrazenia uvítacej obrazovky môže program vykonať všetky inicializačné akcie. K tým patrí najmä príprava zobrazenia grafického rozhrania programu – inicializácia objektov a registrácia spracovateľov udalostí, na ktoré sú tieto objekty citlivé. Podobne, uvítacia obrazovka poskytuje informácie o názve programu, autorských právach či o platnosti zakúpenej licencie. V každom prípade je užitočné ponechať uvítaciu obrazovku viditeľnú len na nevyhnutne dlhý čas. Smernice spoločnosti Microsoft vravia, že spustenie programu v štýle WinRT a spracovanie jeho aktivácie sú akcie, ktoré by nemali presiahnuť časový interval 5 sekúnd. V momente, keď je aktivácia programu dokončená, program beží a jeho stav sa mení z **NotRunning** na **Running**.
2. **Beh programu v štýle WinRT.** Pseudostrokový kód (v prípade riadených WinRT-programov) alebo strokový kód (v prípade natívnych WinRT-programov) je spracúvaný tak dlho, pokiaľ sa program nachádza v stave behu (**Running**). Bežiaci program realizuje automatizáciu všetkých naprogramovaných činností, pričom v prípade potreby aktívne komunikuje s používateľom. Beh programu môže byť prerušený (alebo tiež suspendovaný), a to napríklad vtedy, keď používateľ opustí program a prepne na iný program. Za týchto okolností sa program, ktorý doteraz pôsobil v popredí, odsúva do pozadia. Systém Windows 8 sa správa rozumne, pretože zmena stavu bežiaceho programu (**Running**) na stav suspendovaného programu (**Suspended**) sa uskutoční až po istej chvíli (presnejšie, po 10 sekundách). Vysvetlenie je jasné: systém Windows 8 inteligentne čaká, pretože existuje šanca, že používateľ sa z iného programu prepne veľmi rýchlo zase na

náš program. Ak by sa toto naozaj stalo, program je stále ponímaný ako bežiaci a nedochádza k jeho suspendácii.

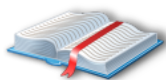
Pre vývojárov je signifikantná možnosť zachytenia udalosti **Suspending**, ktorá je generovaná tesne pred tým, ako je program suspendovaný. Je veľmi užitočné navrhnúť program tak, aby dokázal na udalosť **Suspending** reagovať archiváciou dát – a to jednak dát samotného programu a takisto aj dát, ktoré boli pri doterajšom behu programu vytvorené používateľom. Okrem dátovej perzistencie sa odporúča uvoľniť všetky výhradne pridelené systémové zdroje, ako sú napríklad manipulátory k súborom. To je priateľské riešenie, pretože chceme, aby po suspendácii nášho programu mohli s týmito zdrojmi pracovať aj iné programy bez vzniku kolíznych stavov.



Dôležité: Archivácia dát, dealokácia zdrojov a všetky ďalšie úkony, ktoré budú programovou cestou vykonané v spracovateľovi udalosti **Suspending**, nesmú trvať viac ako 5 sekúnd. Po prekročení tejto časovej hranice totiž systém Windows 8 nadobudne presvedčenie, že program prestal reagovať a explicitne ukončí jeho spracovanie.

Systém Windows 8 disponuje plnou podporou preemptívneho viacúlohového spracovania, takže je samozrejmé, že umožňuje koexistenciu viacerých programov v štýle WinRT. Predpokladajme, že v systéme beží niekoľko programov, no používateľ pracuje len s jedným z nich. V takejto situácii je rozumné všetky ostatné programy prepnúť zo stavu behu do stavu suspendácie. Stratégiou systému Windows 8 je ponechať suspendovaný taký počet programov, aký umožní alokovať vyhradená časť fyzickej pamäti počítača. Pri kritickom využití systémových pamäťových prostriedkov sa však môže stať, že operačný systém bude musieť beh niektorých WinRT-programov ukončiť. Istým problémom však je, že systém Windows 8 nijakým spôsobom neinformuje programy o tom, že budú v najbližšom čase ukončené. Preto jediným variantom uchovania aktuálneho stavu programu je vykonať perzistentné operácie tesne pred jeho suspendáciou.

3. **Obnovenie programu v štýle WinRT.** Keď je program suspendovaný a používateľ sa naň prepne, dochádza k jeho obnoveniu. To znamená, že stav programu sa mení z pozastaveného (**Suspended**) na bežiaci (**Running**). Do úvahy je vzatý aj aktuálny progres programu. Z toho vyplýva, že po obnovení svojho stavu program pokračuje v realizovaní výpočtových procesov na tom mieste, kde skončil vtedy, keď bol suspendovaný. Analogicky, ak program vykonal archiváciu svojich dát pred suspendáciou, tieto dáta sú pri obnovení načítané a použité na uvedenie objektov programu do aktuálneho a okamžite použiteľného stavu. Pri obnovení programu je generovaná udalosť **Resuming**, ktorú ošetruje spracovateľ s naprogramovanou aplikačnou logikou.



Poznámka: Existuje šanca, že program v štýle WinRT zotrvá v režime suspendácie niekoľko hodín, ba až dní. V pozastavenom stave program neprijíma žiadne udalosti zo sieťového prúdu, takže na ne ani nemôže reagovať. Ak je WinRT-program závislý od pripojenia na internet, je potrebné po jeho obnovení opätovne nadviazať spojenie so sieťou a uskutočniť načítanie aktuálnych dát z príslušných sieťových dátových zdrojov.

4. **Uzatvorenie programu v štýle WinRT.** Novinkou v rozhraní WinRT operačného systému Windows 8 je absencia povinnosti používateľov explicitne uzatvárať WinRT-programy. Lepším riešením je ponechať riadenie konečnej fázy životného cyklu programu v štýle WinRT na samotný operačný systém.



Tip: Na druhej strane, používateľ môže i naďalej program priamo ukončiť aktiváciou klávesovej skratky ALT+F4, respektíve vykonaním príslušného dotykového gesta. Vo vizuálnom rozhraní programu sa ale nesmú vyskytovať žiadne ovládacie prvky na jeho explicitné ukončenie, pretože inak by program neprešiel certifikačným a schvaľovacím procesom (a nemohol by byť distribuovaný do elektronického obchodu Windows).

Ukončenie WinRT-programu sa v skutočnosti odohráva v dvoch etapách. Najskôr je program suspendovaný, na čo môže vývojár reagovať spracovateľom udalosti **Suspending** a uchovať požadované dáta programu. Potom je beh programu

ukončený, čo reflektuje zmena jeho stavovej charakteristiky na stav **NotRunning**. V prípade potreby môžu vývojári realizovať rôzne reťazce akcií pred ukončením programu na základe toho, či bol program ukončený používateľom, alebo operačným systémom. Je dobré mať na pamäti, že WinRT-program by sa nikdy nemal sám uzatvárať programovou cestou. Ak by program takúto akciu vykonal, systém Windows 8 ho bude považovať za havarovaný.

Analyzujme ešte ďalšie dva stavy, v ktorých sa program v štýle WinRT môže ocitnúť:

1. **Havária programu v štýle WinRT.** Keď program havaruje, obvykle stratí možnosť vedenia priameho komunikačného dialógu s používateľom. Regule systému Windows 8 predpisujú, že v prípade výskytu takýchto javov musí byť program konformný s politikou riešenia havarijných stavov. Nuž a tá vraví, že je nutné sa urýchlene dostať na východiskové miesto operačného systému, ktorým je pás dlaždíc. Dáta o kolízii programu sú zoskupené a zaslané spoločnosti Microsoft, ktorá ich ďalej postúpi tvorcom programu s odporúčaním na odstránenie detegovaného kolízneho stavu. Po svojom opätovnom spustení sa havarovaný program ocitá v inicializačnom stave.
2. **Odinštalovanie programu v štýle WinRT.** Používateľ môže program odinštalovať z operačného systému Windows 8. Vo chvíli, keď sa tak stane, je zo systému odstránený nielen samotný program, ale aj jeho pracovné dáta. Na druhej strane, používateľské dáta (teda dáta používateľa vytvorené činnosťou programu) zostávajú v operačnom systéme prítomné aj naďalej.



Kapitola 4

Vývoj prvého programu v štýle WinRT v jazykoch C# a Visual Basic

4 Vývoj prvého programu v štýle WinRT v jazykoch C# a Visual Basic

V tejto kapitole sa zoznámime s tvorbou prvého programu v štýle WinRT použitím programovacích jazykov C# a Visual Basic. Program bude slúžiť na výpočet ejekčnej frakcie srdca pacienta. Celý vývojový proces budeme viesť štýlom „krok za krokom“, pričom detailne vysvetlíme všetky etapy tohto procesu.

4.1 Charakteristika prvého programu v štýle WinRT

Náš nový WinRT-program bude obsahovať len jednu stránku, ktorá umožní používateľovi vypočítať hodnotu ejekčnej frakcie jeho srdca. V kardiológii je ejekčná frakcia srdca pacienta jedným z hlavných ukazovateľov správnej činnosti srdca. Ejekčná frakcia totiž vraví, ako dobre si pacientovo srdce plní funkciu prečerpávania krvi. Hodnotu ejekčnej frakcie vypočítame podľa nasledujúceho vzorca:

$$E_f = \frac{ObjemKrv_{i_1} - ObjemKrv_{i_2}}{ObjemKrv_{i_1}} * 100 [\%]$$

kde:

- E_f je ukazovateľ ejekčnej frakcie.
- $ObjemKrv_{i_1}$ je objem krvi v ľavej srdcovej komore pred kontrakciou srdcového svalu.
- $ObjemKrv_{i_2}$ je objem krvi v ľavej srdcovej komore po kontrakcii srdcového svalu.

Keďže chceme získať hodnotu ejekčnej frakcie v percentách, zlomkom generovaný výpočet násobíme hodnotou 100.

Kardiológovia vedia, že normálna hodnota ejekčnej frakcie srdca pacienta sa pohybuje v intervale <55 %, 75 %>.

Ukážme si praktický výpočet ejekčnej frakcie srdca na konkrétnom pacientovi. Predpokladajme, že v ľavej srdcovej komore pacienta sa pred kontrakciou srdcového svalu nachádza 110 ml krvi. Po kontrakcii srdcového svalu je 70 ml krvi vypudených do krvného obehu. To znamená, že aktuálne sa v ľavej srdcovej komore nachádza 40 ml krvi. Výpočet ejekčnej frakcie srdca pacienta je potom nasledujúci:

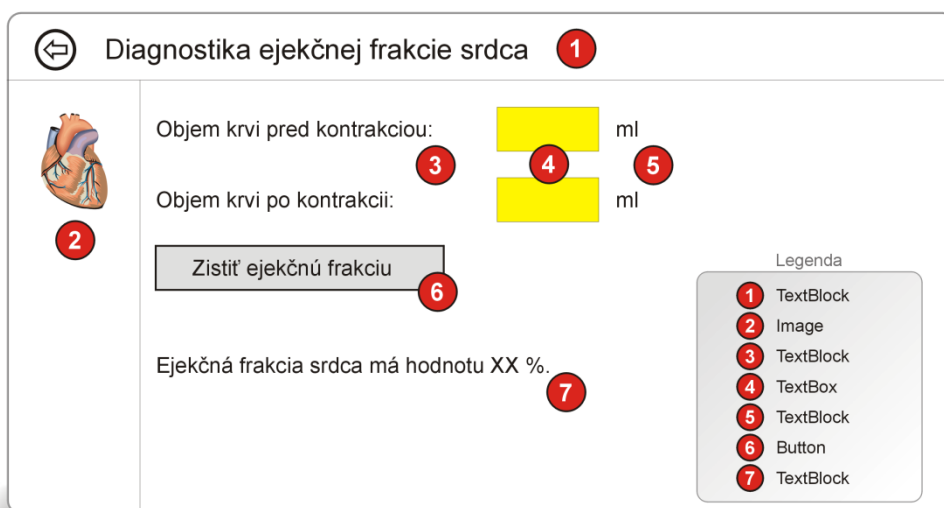
$$E_f = \frac{110 - 40}{110} * 100 = \frac{70}{110} * 100 = 63,63 \%$$

Ako zisťujeme, pacientovo srdce disponuje ejekčnou frakciou s hodnotou približne 64 %.

4.2 Náčrt vizuálneho rozhrania prvého programu v štýle WinRT

Vizuálne rozhranie každého riadeného programu v štýle WinRT navrhujeme pomocou deklaratívneho jazyka XAML, a to buď priamou tvorbou fragmentov tohto jazyka, alebo za asistencie návrhárskych nástrojov, ktoré kódové fragmenty jazyka XAML generujú automaticky. Poznamenajme, že vývojári môžu s výhodou využiť služby vizuálnych návrhárov, ktorí sú zakomponovaní jednak do prostredia produktu Visual Studio 2012 a rovnako aj do prostredia produktu Blend for Visual Studio 2012.

Proces návrhu vizuálneho rozhrania programu v štýle WinRT na výpočet ejekčnej frakcie srdca začneme prvotným náčrtom hlavnej obrazovky programu (obr. 4.1).



Obr. 4.1: Nákres vizuálneho rozhrania programu v štýle WinRT na výpočet ejekčnej frakcie srdca pacienta

Ako môžeme vyčítať z nákresu vizuálneho rozhrania WinRT-programu, pri jeho konštrukcii budeme počítat s nasledujúcimi ovládacími prvkami:

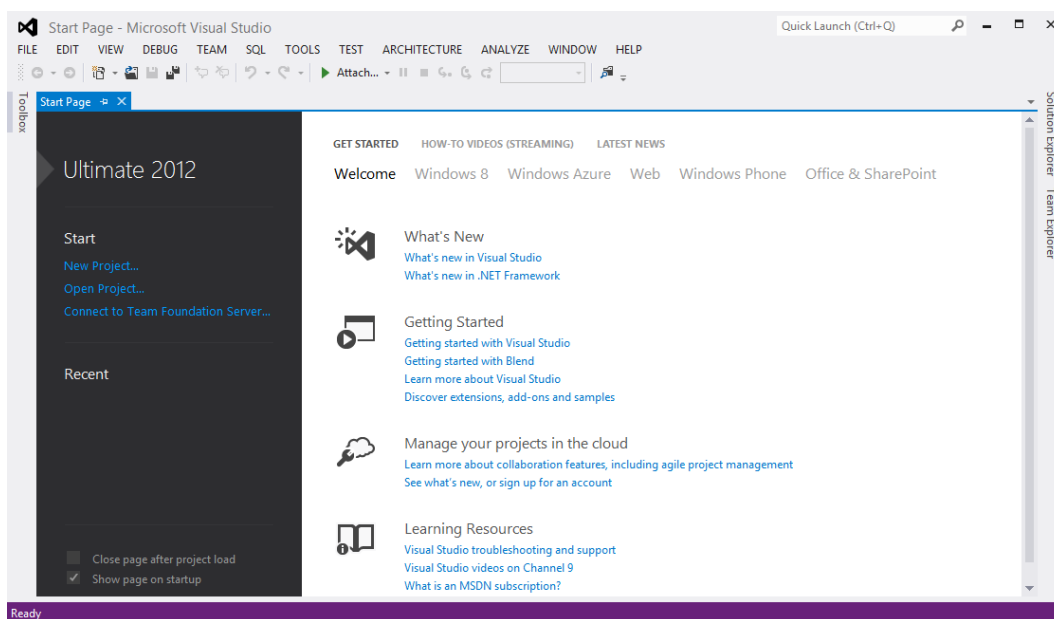
- **Ovládací prvok TextBlock.** Dovedna šesť inštancií tohto ovládacieho prvku použijeme na účel zobrazenia titulného textu programu (1) a takisto aj pri tvorbe textových návestí (3, 5, 7). V tomto kontexte chápeme textové návestia ako regióny neaktívneho textu slúžiace na prezentáciu požadovaných textových správ pre používateľa.
- **Ovládací prvok Image.** Jednu inštanciu tohto ovládacieho prvku upotrebíme na zobrazenie rastrovej podoby srdca (2).
- **Ovládací prvok TextBox.** Dve inštanacie tohto ovládacieho prvku nám umožnia získať od používateľa vstupné dáta (4). Ako vieme, na výpočet ejekčnej frakcie potrebujeme údaje o objemoch krvi v ľavej srdcovej komore pred a po kontrakcii srdcového svalu.

- **Ovládací prvok Button.** Inštancia tohto ovládacieho prvku reprezentuje tlačidlo, na ktoré používateľ klikne ihneď potom, ako zadá do príslušných textových polí konkrétne vstupné dáta. Tlačidlo (6) je aktívny objekt generujúci udalosť **Click**. Spracovateľ tejto udalosti uskutoční algoritmus, ktorý vypočíta hodnotu ukazovateľa ejekčnej frakcie a zobrazí ju v textovom návěstí (7).

4.3 Založenie projektu nového programu v štýle WinRT v jazykoch C# a Visual Basic

Projekt nového programu v štýle WinRT zostrojíme takto:

1. Spustíme produkt Visual Studio 2012.
2. Na úvodnej obrazovke **Start Page** vyberieme voľbu **New Project** (obr. 4.2).

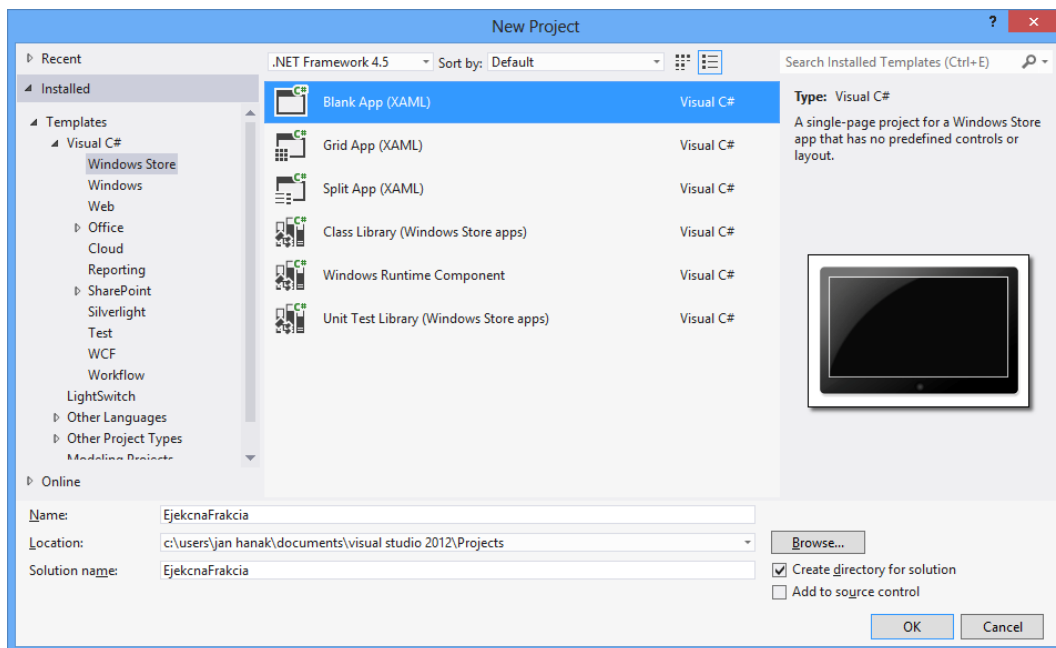


Obr. 4.2: Založenie projektu nového programu v štýle WinRT (1. fáza)

3. V dialógovom okne **New Project** zvolíme najskôr projektovú šablónu **Blank App (XAML)**. Podľa jazykovej preferencie vývojára sa líšia aj úplné cesty vedúce k tejto projektovej šablóne:

- Úplná cesta pre jazyk C#: **Installed** → **Templates** → **Visual C#** → **Windows Store** → **Blank App (XAML)**.
- Úplná cesta pre jazyk Visual Basic: **Installed** → **Templates** → **Visual Basic** → **Windows Store** → **Blank App (XAML)**.

Do textového poľa **Name** zadáme názov programu „EjkcnaFrakcia“. Rovnako sa uistíme, že je aktívna voľba vytvorenia samostatného riešenia pre novo zakladaný projekt (**Create directory for solution**). Finálnu podobu dialógového okna **New Project** pre jazyk C# uvádza obr. 4.3.

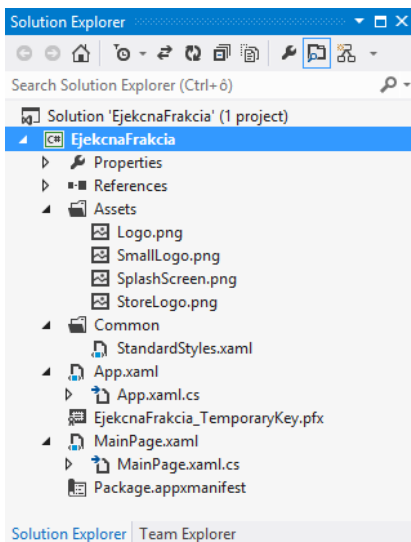


Obr. 4.3: Výber projektovej šablóny a konfigurácia základných nastavení projektu

4. Po klepnutí na tlačidlo **OK** sprievodca naštartuje proces generovania nového projektu prázdneho programu v štýle WinRT. Len čo je nový projekt vytvorený, v prostredí Visual Studio 2012 sa otvorí editor zdrojového kódu s hlavným aplikačným zdrojovým súborom App.xaml.cs (respektíve App.xaml.vb). Každý riadený WinRT-program obsahuje viacero dôležitých projektových súčastí, ktoré preskúmame v nasledujúcej podkapitole.

4.4 Analýza projektových súčastí programu v štýle WinRT

Keď sprievodca zhotoví projekt, presunieme sa do podokna **Solution Explorer**, ktoré zobrazuje všetky samočinne vygenerované projektové súčasti (obr. 4.4).



Obr. 4.4: Kolekcia súčastí nového projektu v podokne **Solution Explorer**

V tab. 4.1 predstavujeme bližší opis základných súčastí nového projektu, ktorý vznikol podľa šablóny **Blank App (XAML)**.

Tab. 4.1: Prehľad a charakteristika projektových súčastí

Názov súčasti	Súbor	Charakteristika
Manifest	Package.appxmanifest	<p>Manifest jednoznačne identifikuje riadené zostavenie programu v štýle WinRT. Obsahuje základné metadáta, ku ktorým patrí identifikátor programu, vstupný bod programu a jeho opis, rovnako ako aj zoznam súborov, ktoré sa v riadenom zostavení programu v štýle WinRT vyskytujú. Okrem toho však manifest združuje veľa ďalších informácií, ktoré súvisia nielen so vzhladom programu, ale aj s jeho použitím, schopnosťami, deklaráciami a konfiguráciou jeho distribučných nastavení.</p> <p>Tip: Hoci je súbor s manifestom XML-súborom, vývojové prostredie Visual Studio 2012 integruje vizuálneho návrhára manifestov (Manifest Designer), vďaka ktorému môžu vývojári vykonávať zmeny v manifeste programu vizuálnou cestou, a tak nie sú odkázaní na priamu úpravu príslušného XML-kódu.</p>
Priečink Assets	Priečink tvoria grafické súbory: Logo.png, SmallLogo.png, SplashScreen.png a StoreLogo.png	V priečinku zdrojov sa nachádzajú grafické súbory, ktoré reprezentujú vizuály pre logo programu (v normálnej a zmenšenej verzii), logo pre uvítaciu obrazovku programu a ikonu programu, ktorá sa bude objavovať v elektronickom obchode Windows.
Priečink Common	Priečink obsahuje súbor: StandardStyles.xml	V priečinku Common je situovaný XML-súbor, ktorý deklaruje štýly pre rýchlejší vývoj programov v štýle WinRT. V skutočnosti sú deklarované štýly vyžadované väčšinou projektových šablón prostredia Visual Studio 2012.

(Tab. 4.1 pokračuje na ďalšej strane knihy)

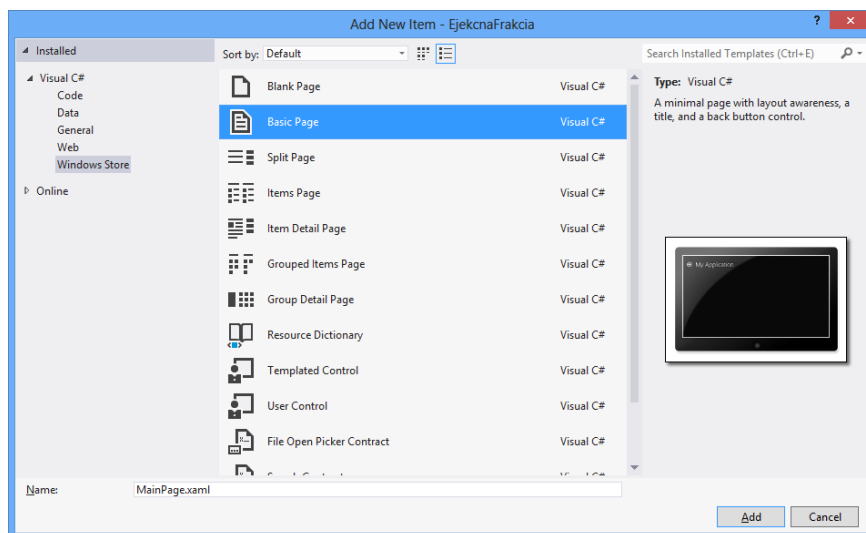
Tab. 4.1: Prehľad a charakteristika projektových súčastí (pokračovanie)

Názov súčasti	Súbor	Charakteristika
XAML-súbor a zdrojový súbor programu	App.xaml, App.xaml.cs (respektíve App.xaml.vb)	Súbor App.xaml je prispôsobený na úschovu deklarácií základných grafických prvkov programu v štýle WinRT. Na druhej strane pôsobia zdrojové súbory (s koncovkami .cs alebo .vb), ktoré deklarujú hlavnú triedu WinRT-programu s názvom App . Technicky je trieda App priamou podtriedou bázevej triedy Application a jej deklarácia v príslušnom zdrojovom súbore je parciálna. V tele triedy App sa okrem definície verejného a bezparametrického konštruktora vyskytujú aj implicitne zhotovení spracovateľa udalostí OnLaunched a OnSuspending .
XAML-súbor a zdrojový súbor hlavnej stránky programu	MainPage.xaml, MainPage.xaml.cs (respektíve MainPage.xaml.vb)	Súbor MainPage.xaml deklaruje triedu Page so základným usporiadaním pre mriežku (Grid). Obsah tohto XAML-súboru je štandardne viditeľný v dvoch zobrazovacích režimoch: jednak v režime návrhu (záložka Design) a rovnako aj v režime kódu (záložka XAML). Po použití projektovej šablóny Blank App (XAML) je hlavná stránka programu prázdna. Návrhári používateľského rozhrania ju však môžu zaplniť inštanciami vybraných ovládacích prvkov. Proces návrhu vizuálneho rozhrania programu v štýle WinRT sa môže odohrávať tiež dvojako: buď vizuálnym návrhom, alebo priamym zadaním fragmentov jazyka XAML (s podporou senzitívnej technológie IntelliSense). Zdrojový súbor hlavnej stránky zavádza parciálnu deklaráciu triedy MainPage (táto trieda je podtriedou bázevej triedy Page). Trieda MainPage definuje verejný bezparametrický konštruktor a spracovateľa udalosti OnNavigatedTo .

4.5 Vývoj grafického rozhrania prvého programu v štýle WinRT

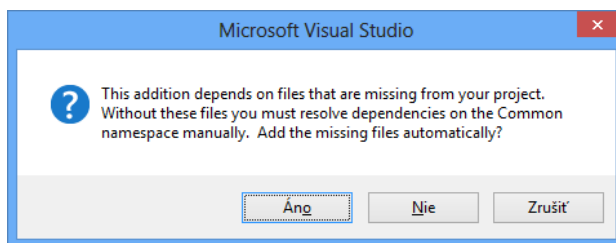
Hlavná stránka programu v štýle WinRT (**Blank Page**), ktorú získame po použití projektovej šablóny **Blank Page (XAML)**, neposkytuje ani základnú funkcionality. Samozrejme, ak by sme chceli, mohli by sme túto funkcionality a sprievodné pracovné triedy vytvoriť sami, no nazdávame sa, že lepším riešením je nahradiť aktuálnu hlavnú stránku novou hlavnou stránkou s už implementovanou základnou funkcionality (**Basic Page**). Preto uskutočníme tieto akcie:

1. V podokne **Solution Explorer** klikneme na položku MainPage.xaml pravým tlačidlom myši, čím prístupníme lokálnu ponuku.
2. Z lokálnej ponuky vyberieme príkaz **Delete**. Prostredie Visual Studio 2012 zobrazí potvrdzovacie dialógové okno. My klepneme na tlačidlo **OK**, a tak vyjadríme svoj súhlas s odstránením existujúcej hlavnej stránky.
3. Otvoríme ponuku **Project** a aktivujeme príkaz **Add New Item**.
4. V stromovej štruktúre **Intalled** → **Visual C#** (respektíve **Installed** → **Visual Basic**) označíme voľbu **Windows Store**. Na opačnej strane sa zobrazia všetky šablóny projektových položiek, ktoré môžu byť začlenené do programov v štýle WinRT.
5. Jednou z týchto položiek je aj položka **Basic Page**, ktorú vyberieme.
6. Zmeníme identifikátor novej projektovej položky v textovom poli **Name**. Implicitný názov BasicPage1.xaml upravíme na MainPage.xaml (obr. 4.5).



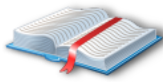
Obr. 4.5: Selekcia projektovej súčasti pre základnú stránku (**Basic Page**)

7. Vykonané zmeny uložíme stisnutím tlačidla **Add**.
8. Prostredie Visual Studio 2012 zobrazí dialógové okno s informáciou, že pridanie základnej stránky (**Basic Page**) je závislé od pridania asociovaných súborov (obr. 4.6).



Obr. 4.6: Súhlas s pridaním asociovaných súborov

Keďže chceme tieto súbory automaticky pridať a rovnako aj ihneď správne vyriešiť požadované závislosti, klepneme v dialógovom okne na tlačidlo **Yes**.

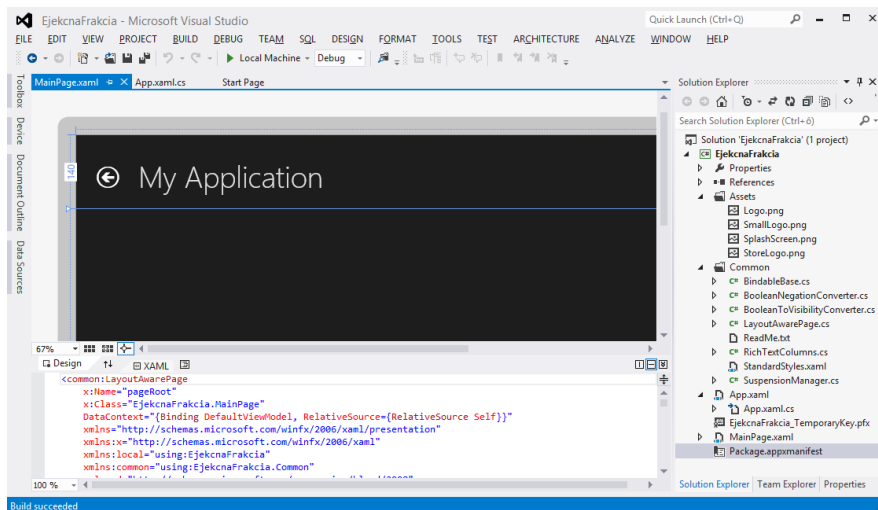


Poznámka: Po dokončení akcie môžeme preskúmať priečinok **Common** v podokne **Solution Explorer**. Ako zistíme, získali sme viacero nových zdrojových súborov (napríklad `BindableBase.cs`, `LayoutAwarePage.cs` či `SuspensionManager.cs`), ktoré robia spoločnosť doteraz osamotenému súboru `StandardStyles.xaml`.

Zmeny grafického rozhrania programu v štýle WinRT nie sú viditeľné okamžite po pridaní novej základnej stránky. Je totiž nutné uskutočniť zostavenie projektu, a preto vyberieme ponuku **Build** a klepneme na položku **Build Solution** (alebo využijeme klávesovú skratku F6). Keď sa zdrojové súbory projektu preložia, v spodnom stavovom pruhu uvidíme správu „Build succeeded“. V tej istej chvíli sa aktualizuje aj plocha vizuálneho návrhára, ktorá znázorňuje vzhľad základnej stránky nášho prvého WinRT-programu (obr. 4.7).



Tip: Novinkou stavového pruhu prostredia Visual Studio 2012 je jeho schopnosť sémantickej kolorizácie podľa druhu realizovanej operácie. Povedané menej formálne, stavový pruh vyjadruje druh spracúvanej operácie aj výberom príslušnej farby. Napríklad, za bežných okolností je tento pruh tyrkysový, no pri preklade sa jeho podkladová farba zmení na tmavomodrú. Naopak, operácia načítavania projektových súčastí je reprezentovaná oranžovou farbou.



Obr. 4.7: Vzhľad základnej stránky WinRT-programu

Základná stránka WinRT-programu je vďaka použitiu projektovej súčasti **Basic Page** vyspelejšia, ako bola predtým. Obsahuje tlačidlo návratu a hlavný text (zatiaľ implicitne nastavený na hodnotu „My Application“). Najmä na počítačoch s menšími displejmi sa môže stať, že zobrazovacia plocha vizuálneho návrhára nebude schopná zobraziť hlavnú stránku programu v celej veľkosti. Vtedy odporúčame použiť lupu a oddialiť zameranie tak, aby sme videli celú hlavnú stránku. (Napríklad, na obr. 4.7 je veľkosť zobrazenia nastavená na hodnotu 50 %.)

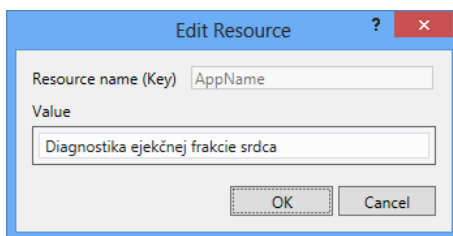
4.5.1 Úprava vzhľadu hlavnej stránky prvého programu v štýle WinRT

Prvým krokom, ktorý uskutočníme, je zmena hlavného textu programu. Hlavný text je uchovaný v inštancii ovládacieho prvku **TextBlock**. Najprv tento objekt vyberieme, a potom sa presunieme do podokna jeho vlastností (**Properties**).



Tip: Ak podokno **Properties** nevidíme, klávesovou skratkou F4 ho rýchlo zobrazíme.

Tu zmeníme identifikátor objektu (vlastnosť **Name**) z pôvodnej hodnoty **pageTitle** na hodnotu **txbTitulnyText**. Do vlastnosti **Text** zase uložíme textový reťazec „Diagnostika ejekčnej frakcie srdca“. To spravíme tak, že pri textovom poli, ktoré je spojené s vlastnosťou **Text**, klikneme na zelený štvorec (■). V momente sa objaví kontextová ponuka, v ktorej aktivujeme príkaz **Edit Resource**. V rovnomenom dialógovom okne potom do textového poľa **Value** uložíme novú hodnotu pre hlavný text programu a klikneme na tlačidlo **OK** (obr. 4.8).



Obr. 4.8: Zmena hlavného textu programu

Ďalej budeme postupovať podľa nákresu vizuálneho rozhrania nášho prvého programu v štýle WinRT, ktorý sme uviedli na obr. 4.1 v podkapitole 4.2 *Náčrt vizuálneho rozhrania prvého programu v štýle WinRT*. To znamená, že na mriežku hlavnej stránky programu nanesieme inštancie všetkých ovládacích prvkov, ktoré budeme potrebovať na získanie vstupných dát, ich spracovanie a vypísanie výstupných dát. Tab. 4.2 sumarizuje vytvorené a nakonfigurované objekty grafického rozhrania programu. Ovládacie prvky selektujeme z podokna **Toolbox**.

Tab. 4.2: Sumárny prehľad objektov vizuálneho rozhrania prvého WinRT-programu

Objekt	Vlastnosť objektu	Hodnota vlastnosti objektu
Image (Objekt slúži na zobrazenie obrázka srdca)	Name	imgSrdce
	Source	Assets/grafika_srdce.png
	Stretch	Uniform
	Width	102
	Height	138
	Margin (Left)	49
	Margin (Top)	20

(Tab. 4.2 pokračuje na ďalšej strane knihy)

Tab. 4.2: Sumárny prehľad objektov vizuálneho rozhrania prvého WinRT-programu (pokračovanie)

Objekt	Vlastnosť objektu	Hodnota vlastnosti objektu
TextBlock (Objekt slúži na zobrazenie textového návestia)	Name	txbObjemKrvi1
	Text	Objem krvi pred kontrakciou:
	Width	275
	Height	27
	Margin (Left)	179
	Margin (Top)	20
	Text (FontSize)	20 px
TextBlock (Objekt slúži na zobrazenie textového návestia)	Name	txbObjemKrvi2
	Text	Objem krvi po kontrakcii:
	Width	275
	Height	27
	Margin (Left)	179
	Margin (Top)	80
	Text (FontSize)	20 px

(Tab. 4.2 pokračuje na ďalšej strane knihy)

Tab. 4.2: Sumárny prehľad objektov vizuálneho rozhrania prvého WinRT-programu (pokračovanie)

Objekt	Vlastnosť objektu	Hodnota vlastnosti objektu
TextBox (Objekt slúži na zadanie objemu krvi v ľavej srdcovej komore pred kontrakciou srdcového svalu)	Name	txtObjemKrvi1
	Text	120
	Width	41
	Height	37
	Margin (Left)	479
	Margin (Top)	20
	Text (FontSize)	20 px
TextBox (Objekt slúži na zadanie objemu krvi v ľavej srdcovej komore po kontrakcii srdcového svalu)	Name	txtObjemKrvi2
	Text	50
	Width	41
	Height	37
	Margin (Left)	479
	Margin (Top)	80
	Text (FontSize)	20 px

(Tab. 4.2 pokračuje na ďalšej strane knihy)

Tab. 4.2: Sumárny prehľad objektov vizuálneho rozhrania prvého WinRT-programu (pokračovanie)

Objekt	Vlastnosť objektu	Hodnota vlastnosti objektu
TextBlock (Objekt slúži na zobrazenie textového návestia)	Name	txbObjem1_ml
	Text	ml
	Width	64
	Height	27
	Margin (Left)	559
	Margin (Top)	20
	Text (FontSize)	20 px
TextBlock (Objekt slúži na zobrazenie textového návestia)	Name	txbObjem2_ml
	Text	ml
	Width	64
	Height	27
	Margin (Left)	559
	Margin (Top)	80

(Tab. 4.2 pokračuje na ďalšej strane knihy)

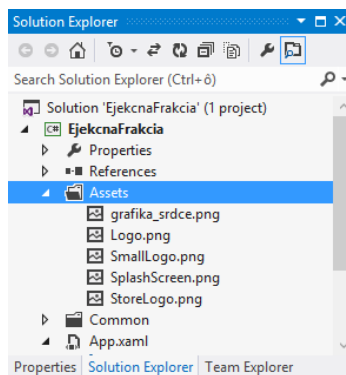
Tab. 4.2: Sumárny prehľad objektov vizuálneho rozhrania prvého WinRT-programu (pokračovanie)

Objekt	Vlastnosť objektu	Hodnota vlastnosti objektu
Button (Objekt slúži ako tlačidlo na spustenie výpočtu ejekčnej frakcie)	Name	btnZistitEF
	Content	Zistiť ejekčnú frakciu
	Width	275
	Height	52
	Margin (Left)	179
	Margin (Top)	219
	Text (FontSize)	14.667
TextBlock (Objekt slúži na vypísanie hodnoty ejekčnej frakcie)	Name	txbEF
	Text	Prázdny reťazec
	Width	444
	Height	33
	Margin (Left)	179
	Margin (Top)	319
	Text (FontSize)	20 px

Návodný postup, ktorý dokumentuje tab. 4.2, nás spoľahlivo dovedie k finálnej podobe vizuálneho rozhrania programu v štýle WinRT. Objekt ovládacieho prvku **Image** používame na zobrazenie ilustračnej schémy ľudského srdca. Pre potreby tohto programu

sme vytvorili vektorový obraz srdca, ktorý sme následne rasterizovali. Dôležité je podotknúť, že obrázok srdca je uložený v súbore `grafika_srdce.png` a tento súbor je umiestnený medzi grafickými zdrojmi programu. Zaradenie grafického súboru do zdrojov programu v štýle WinRT uskutočníme podľa nasledujúceho postupu:

1. V podokne **Solution Explorer** klepneme pravým tlačidlom myši na priečinok so zdrojmi (**Assets**).
2. V kontextovej ponuke ukážeme na položku **Add** a potom zvolíme príkaz **Existing Item**.
3. V rovnomennom dialógovom okne vyhladáme požadovaný grafický súbor a klikneme na tlačidlo **Add**.
4. Týmto spôsobom sa kolekcia zdrojov programu rozšíri o novo dodaný grafický súbor (obr. 4.9).



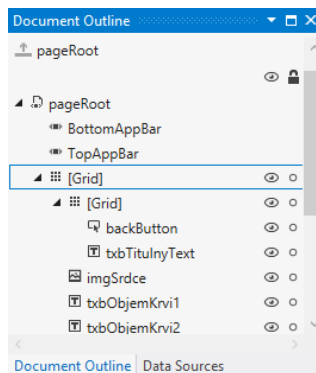
Obr. 4.9: Začlenenie grafického súboru do kolekcie zdrojov WinRT-programu

Ihneď potom, čo sme úspešne vykonali import nového grafického zdroja programu, môžeme tento zdroj priradiť k vlastnosti **Source** objektu ovládacieho prvku **Image**.



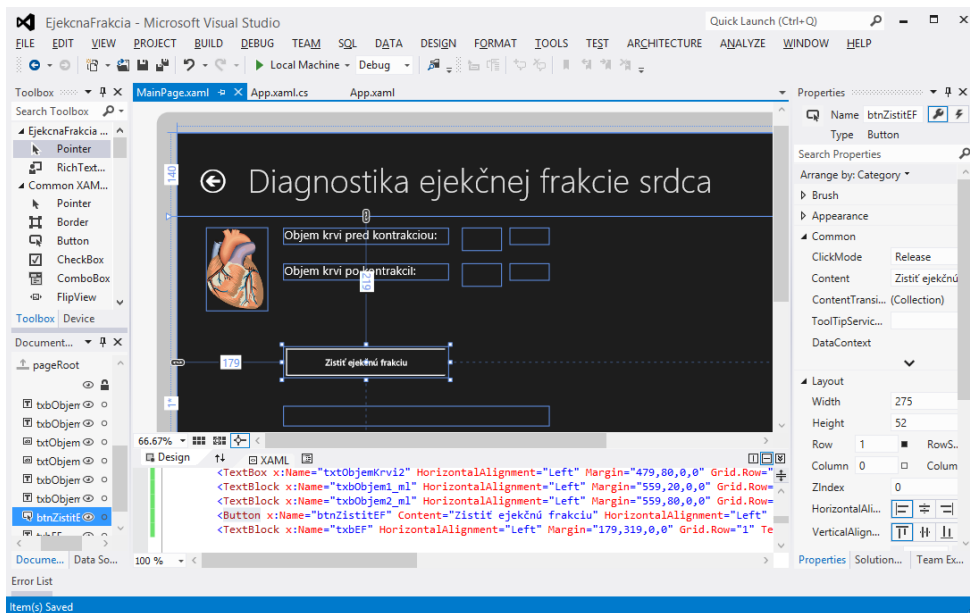
Tip: V procese vizuálneho programovania nám veľmi pomôže podokno **Document Outline**, ktoré znázorňuje prehľadnú stromovú štruktúru stránky a objektov na nej sídliačich (tejto vizuálnej kompozícii objektov často vravíme osnova). Ak toto podokno nevidíme, zobrazíme ho touto cestou: ponuka **View** → položka **Other**

Windows → príkaz **Document Outline**. Na obr. 4.10 sa môžeme zoznámiť s obsahom podokna **Document Outline** pre vizuálnu kompozíciu prvého programu v štýle WinRT.



Obr. 4.10: Podokno **Document Outline** s osnovou objektov vizuálneho rozhrania

Na obr. 4.11 uvádzame kompletný návrh vizuálneho rozhrania nového programu v štýle WinRT.



Obr. 4.11: Finálny vzhľad navrhnutého vizuálneho rozhrania programu v štýle WinRT

4.6 Implementácia logiky prvého programu v štýle WinRT

Program je určený na diagnostiku ejekčnej frakcie srdca pacienta. Keď používateľ programu zadá vstupné dáta a klepne na tlačidlo, program vypočíta ukazovateľ ejekčnej frakcie a zobrazí ho.

Logika programu, ktorá zisťuje hodnotu ejekčnej frakcie, bude implementovaná v spracovateľovi udalosti **Click** tlačidla.

Tohto spracovateľa najrýchlejšie zostavíme poklepaním na tlačidlo. Existuje však aj iná cesta:

1. Uistíme sa, že vo vizuálnom návrhárovi je vybratý objekt tlačidla.
2. Presunieme sa do podokna **Properties** a klikneme na tlačidlo s ikonou blesku (⚡).
3. Objaví sa kolekcia udalostí, na ktoré je zvolený objekt citlivý.
4. Vyhľadáme udalosť **Click** a do jej textového poľa zapíšeme názov metódy, ktorá bude vystupovať ako spracovateľ tejto udalosti.



Tip: Ak do textového poľa udalosti nezapíšeme žiaden názov pre jeho spracovateľa, ale len poklepeme na toto textové pole, editor zdrojového kódu zostaví spracovateľa udalosti s implicitným názvom. Ako vieme, implicitný názov spracovateľa akejkoľvek udalosti pozostáva z identifikátora objektu, symbolu podčiarkovníka (_) a identifikátora špecifikovanej udalosti. To pre tlačidlo s názvom **btnZistiteF** znamená automatické zostrojenie spracovateľa udalosti s identifikátorom **btnZistiteF_Click**. Dodajme, že bez ohľadu na použitý (či už implicitný, alebo explicitný) identifikátor pre spracovateľa udalosti **Click** tlačidla, je tento spracovateľ vždy reprezentovaný parametrickou metódou so súkromnými prístupovými právami.

5. Vizuálny návrhár sa spojí s editorom zdrojového kódu, ktorý vygeneruje nového spracovateľa pre zvolenú udalosť príslušného objektu.

Zdrojový kód spracovateľa udalosti **btnZistitEF_Click** s implementovanou logikou vyzerá v jazyku C# takto:

```
private void btnZistitEF_Click(object sender, RoutedEventArgs e)
{
    // Definície lokálnych premenných.
    float objemKrviPredKontrakciou, objemKrviPoKontrakcii;
    float ejecnaFrakcia;
    // Inicializácia lokálnych premenných vstupnými dátami.
    objemKrviPredKontrakciou = Convert.ToSingle(txtObjemKrvi1.Text);
    objemKrviPoKontrakcii = Convert.ToSingle(txtObjemKrvi2.Text);
    // Výpočet ukazovateľa ejekčnej frakcie.
    ejecnaFrakcia = (objemKrviPredKontrakciou - objemKrviPoKontrakcii)
        / objemKrviPredKontrakciou * 100;
    // Zobrazenie hodnoty ejekčnej frakcie na výstupe.
    txbEF.Text = "Ejekčná frakcia srdca má hodnotu " +
        ejecnaFrakcia.ToString("0.00") + " %.";
}
```

Zdrojový kód spracovateľa udalosti **btnZistitEF_Click** s implementovanou logikou vyzerá v jazyku Visual Basic takto:

```
Private Sub btnZistitEF_Click(sender As Object, e As RoutedEventArgs) _
Handles btnZistitEF.Click
    'Definície lokálnych premenných.
    Dim objemKrviPredKontrakciou, objemKrviPoKontrakcii As Single
    Dim ejecnaFrakcia As Single
    'Inicializácia lokálnych premenných vstupnými dátami.
    objemKrviPredKontrakciou = Convert.ToSingle(txtObjemKrvi1.Text)
    objemKrviPoKontrakcii = Convert.ToSingle(txtObjemKrvi2.Text)
    'Výpočet ukazovateľa ejekčnej frakcie.
    ejecnaFrakcia = (objemKrviPredKontrakciou - objemKrviPoKontrakcii) / _
        objemKrviPredKontrakciou * 100
    'Zobrazenie hodnoty ejekčnej frakcie na výstupe.
    txbEF.Text = "Ejekčná frakcia srdca má hodnotu " &
        ejecnaFrakcia.ToString("0.00") & " %."
End Sub
```

Komentár k zdrojovému kódu programu v štýle WinRT: V tele spracovateľa najskôr definujeme lokálne premenné reálneho dátového typu **float** (respektíve **Single**). Len čo sú lokálne premenné alokované, realizujeme ich inicializáciu. Pritom predpokladáme, že v čase spracovania tohto kódu budú požadované vstupné dáta vždy k dispozícii.

Vstupnými dátami sú objemy krvi (merané v ml) v ľavej srdcovej komore pred a po kontrakcii srdcového svalu.

Pripomeňme, že pri stavbe vizuálneho rozhrania programu sme do textových polí určených na úschovu vstupných dát napevno priradili hodnoty 120 a 50. Z toho vyplýva, že tieto textové polia budú po spustení programu obsahovať práve tieto vstupné dáta a ak používateľ nezadá iné vstupné dáta, bude ukazovateľ ejekčnej frakcie vypočítaný z implicitne dodaných vstupných dát. Takýto prístup sa snaží minimalizovať riziko vzniku kolíznych stavov vznikajúcich pri absencii vstupných dát. Parciálny algoritmus výpočtu ukazovateľa ejekčnej frakcie je dobre štruktúrovaný, pričom verne kopíruje vopred charakterizovaný matematický predpis. Po stanovení ejekčnej frakcie srdca pacienta je tento údaj znázornený vo výstupnom textovom návěstí (s presnosťou na dve desatinné miesta). Samozrejme, vzhľadom na opakovateľnosť algoritmu programu môže používateľ ľubovoľne meniť vstupné dáta a vyčíslívať ejekčné frakcie pre srdcia viacerých pacientov.

4.7 Konfigurácia grafických zdrojov prvého programu v štýle WinRT

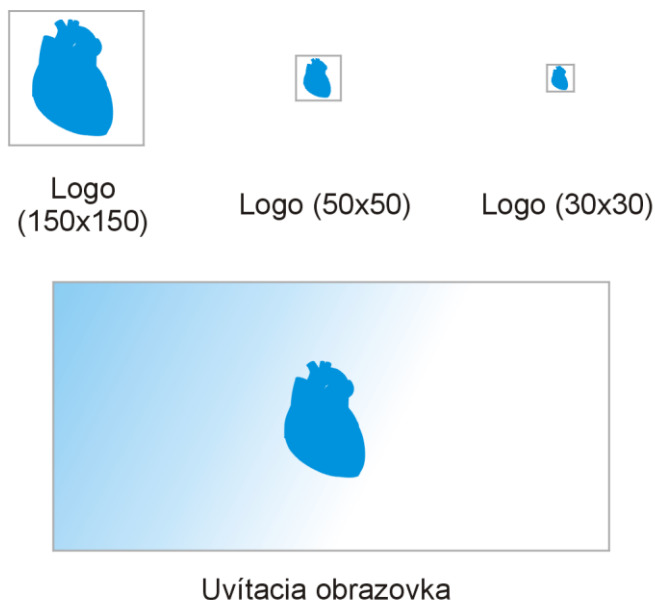
Grafické zdroje, ktoré sú predmetom nášho ďalšieho skúmania, sú tieto:

- **Logo programu v štýle WinRT.** Toto logo musí mať rozmery 150x150 obrazových bodov. Grafický súbor uchovávajúci toto logo má názov Logo.png.
- **Logo programu v štýle WinRT pre potreby elektronického obchodu Windows.** Toto logo musí mať rozmery 50x50 obrazových bodov. Grafický súbor uchovávajúci toto logo má názov StoreLogo.png.
- **Malé logo programu v štýle WinRT.** Toto logo musí mať rozmery 30x30 obrazových bodov. Grafický súbor uchovávajúci toto logo má názov SmallLogo.png.

- **Uvítacia obrazovka programu v štýle WinRT.** Táto obrazovka musí mať rozmery 620x300 obrazových bodov. Grafický súbor uchovávajúci toto logo má názov SplashScreen.png.

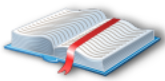
Všetky spomenuté grafické súbory sú grafickými zdrojmi programu v štýle WinRT. Tieto grafické zdroje sa nachádzajú v priečinku EjecnaFrakcia\EjecnaFrakcia\Assets.

Pre náš prvý program v štýle WinRT sme pripravili logá vo všetkých potrebných rozmeroch a rovnako aj štýlovú uvítaciu obrazovku (obr. 4.12).

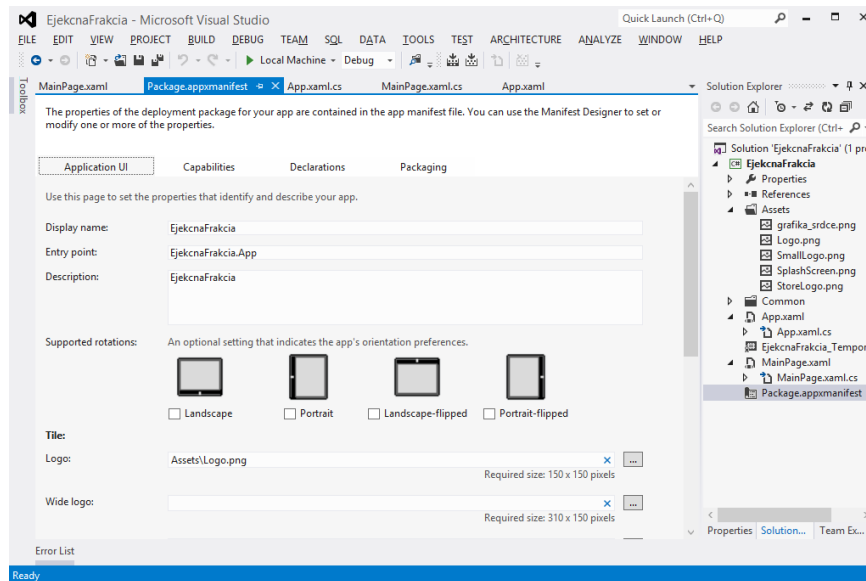


Obr. 4.12: Navrhnuté grafické zdroje prvého WinRT-programu

Aby všetko fungovalo správne, musíme upraviť vzhľad všetkých grafických zdrojov, ktoré sú umiestnené v priečinku **Assets**.



Poznámka: Cesty ku grafickým zdrojom programu v štýle WinRT môžeme konfigurovať z prostredia nástroja **Manifest Designer**. Tento nástroj sa spustí vždy, keď v podokne **Solution Explorer** klikneme na položku manifestu s názvom **Package.appxmanifest**. Po spustení návrhára manifestu najskôr skontrolujeme, či je aktívna karta **Application UI**. Potom môžeme preskúmať voľby **Logo**, **Small logo** či **Splash Screen**. Ak si neprajeme na dlaždici WinRT-programu zobrazovať názov programu, zvolíme možnosť **No Logos** pri voľbe **Show name**.

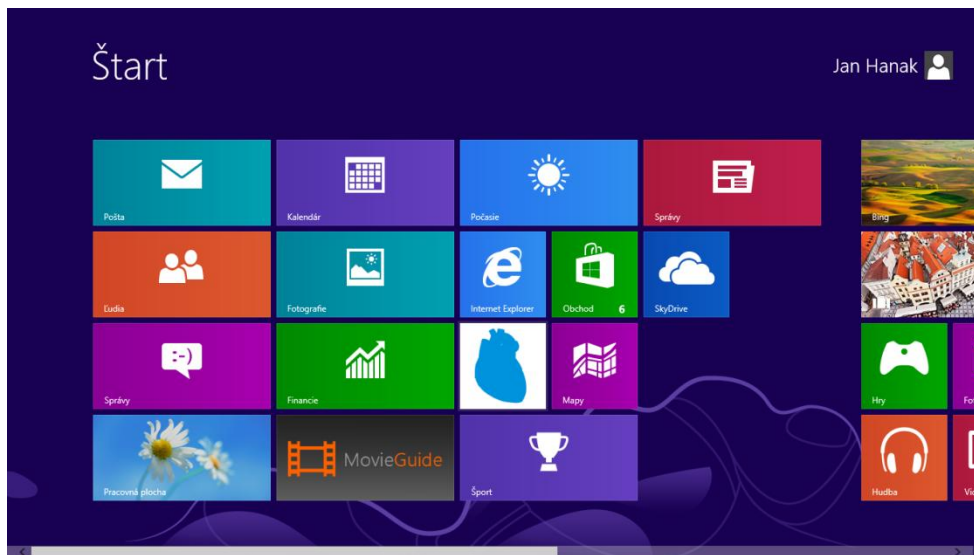


Obr. 4.13: Možnosti nástroja **Manifest Designer**

4.8 Zostavenie a spustenie programu štýle WinRT v systéme Windows 8

Zostavenie programu v štýle WinRT a jeho spustenie vykonáme výberom voľby **Start Debugging** z ponuky **Debug** (prípadne využijeme klávesovú skratku F5). Ak vyvíjame program na počítači s operačným systémom Windows 8, tak automaticky po svojom zostavení je tento program nainštalovaný do kolekcie existujúcich programov systému

Windows 8. Túto skutočnosť postrehneme veľmi ľahko, pretože dlaždica s logom programu v štýle WinRT sa vyskytne na páse dlaždíc (obr. 4.14).



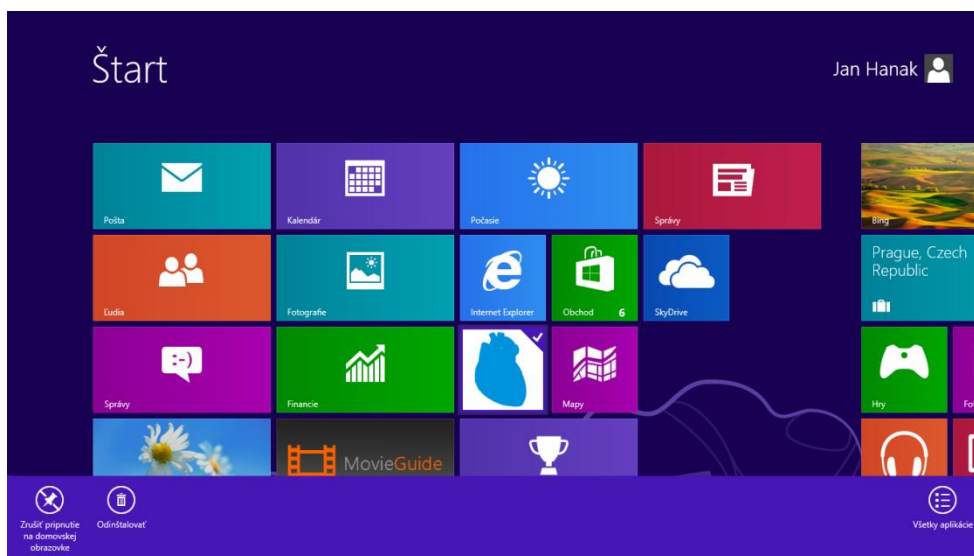
Obr. 4.14: Nový program (s logom modrého srdca) v štýle WinRT je umiestnený na páse dlaždíc

Pri zostavovaní a spúšťaní WinRT-programov priamo v systéme Windows 8 uplatňuje prostredie Visual Studio 2012 režim **Local Machine**. Ako sme už opísali, tento režim zahrnuje okamžitú inštaláciu nového programu v štýle WinRT do operačného systému. Takto nainštalovaný program zostáva prístupný (prostredníctvom svojej dlaždice na páse dlaždíc) aj po ukončení ladiaceho procesu, respektíve po uzatvorení vývojového prostredia Visual Studio 2012.

Keď dokončíme testovanie programu, do vývojového prostredia Visual Studio 2012 sa prenesieme klávesovou skratkou ALT+TAB. Beh spusteného programu ukončíme aktiváciou voľby **Stop Debugging** z ponuky **Debug** (ekvivalentne nám poslúži aj klávesová skratka SHIFT+F5).

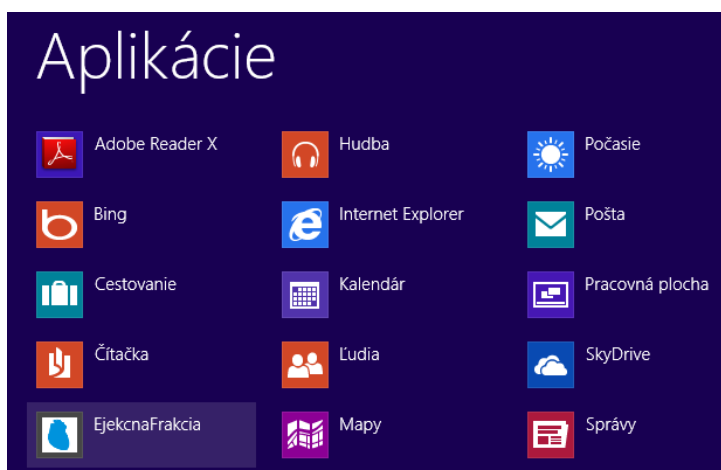
Keď na dlaždicu nainštalovaného WinRT-programu klikneme pravým tlačidlom myši, objaví sa spodný panel s dvomi možnosťami:

1. Zrušiť pripnutie programu na domovskej stránke **Štart**.
2. Odinštalovať program.



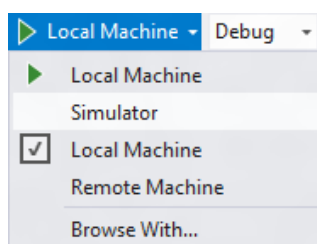
Obr. 4.15: Prehľad aktívnych operácií asociovaných s dlaždicou nového programu

Ak zrušíme pripnutie programu, jeho dlaždica zmizne z pásu dlaždíc, avšak program je stále prístupný z aplikačnej sekcie. Aplikačnú sekciu otvoríme klepnutím na tlačidlo **Vyhľadávanie** v postrannom paneli s kľúčovými tlačidlami systému Windows 8 (obr. 4.16).



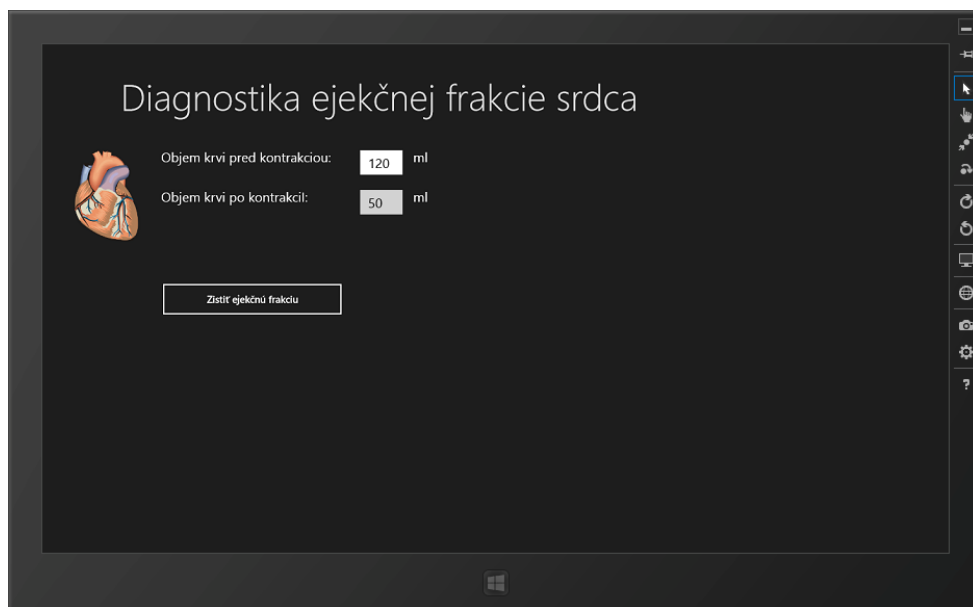
Obr. 4.16: Aplikačná sekcia systému Windows 8 s novým programom v štýle WinRT

Pre úplnosť dodajme, že program v štýle WinRT môžu vývojári testovať aj v prostredí softvérového simulátora. Tento simulátor je v skutočnosti virtuálnym emulátorom zariadenia s dotykovou obrazovkou a s nainštalovaným operačným systémom Windows 8. Simulátor zapneme selekciou voľby **Simulator** zo štandardného panela prostredia Visual Studio 2012 (obr. 4.17).



Obr. 4.17: Prepnutie testovacieho režimu na simulátor

Zostavenie a spustenie programu v štýle WinRT v prostredí simulátora je veľmi podobné štandardnému spracovaniu tohto programu na fyzickom počítači so systémom Windows 8. Beh nášho programu v prostredí simulátora dokumentuje obr. 4.18.



Obr. 4.18: Spracovanie programu v štýle WinRT v prostredí simulátora



Kapitola 5

Vývoj prvého programu v štýle WinRT v jazyku C++

5 Vývoj prvého programu v štýle WinRT v jazyku C++

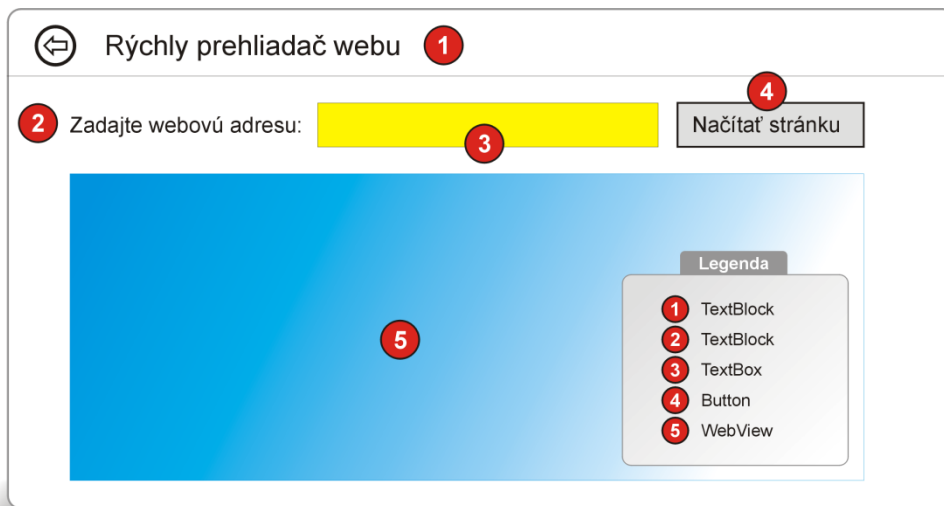
Táto kapitola vysvetľuje vývojový proces prvého programu v štýle WinRT v jazyku C++. Vytvorený program bude pôsobiť ako rýchly prehliadač webových stránok s vlastným vizuálnym rozhraním. Pri vývoji programu budeme používať jazyk C++/CX, teda jazyk, ktorý kombinuje natívne C++ s (rovnako natívnymi) komponentovými rozšíreniami (angl. *Component Extensions*) v zacielení na platformu WinRT.

5.1 Charakteristika prvého programu v štýle WinRT

Program, ktorý vyvineme, umožní používateľom zadávať adresy webových stránok a zobrazovať ich obsahy. Pôjde teda o jednoduchý webový prehliadač s minimalistickým grafickým rozhraním a s primárnou koncentráciou na požadovaný elektronický obsah.

5.2 Náčrt vizuálneho rozhrania prvého programu v štýle WinRT

Vizuálne rozhranie programu v štýle WinRT budeme deklarovať pomocou fragmentov jazyka XAML. Prostredie produktu Visual C++ 2012 obsahuje (podobne ako prostredia produktov Visual Basic 2012 a Visual C# 2012) samostatného vizuálneho návrhára pre XAML grafické rozhrania. Tvorcovia grafických rozhraní a programátori môžu použiť tohto návrhára na modelovanie základných i pokročilých grafických elementov stránok programu v štýle WinRT. Náš prvý WinRT-program zhotovený v jazyku C++ bude obsahovať len jednu stránku, na ktorú umiestnime inštancie všetkých potrebných ovládacích prvkov. Prvotný náčrt vizuálneho rozhrania prvého programu v štýle WinRT uvádzame na obr. 5.1.



Obr. 5.1: Nákres vizuálneho rozhrania rýchleho webového prehliadača v štýle WinRT

Z ilustračnej schémy je zrejmé, že grafické používateľské rozhranie programu v štýle WinRT bude tvorené inštanciami nasledujúcich ovládacích prvkov:

- **Ovládací prvok TextBlock.** Na stránke WinRT-programu sa vyskytujú 2 objekty tohto ovládacieho prvku: jeden funguje ako hlavný text programu (1), zatiaľ čo druhý poskytuje inštrukcie používateľovi (2).
- **Ovládací prvok TextBox.** Objekt tohto ovládacieho prvku pracuje ako textové pole (3), do ktorého používateľ zadá jednoznačnú adresu požadovanej webovej stránky. Ak nebude určené inak, textové pole programu bude implicitne uchovávať webovú adresu slovenského sídla spoločnosti Microsoft (<http://www.microsoft.sk>).
- **Ovládací prvok Button.** Tlačidlo (4) slúži na spracovanie vstupných údajov od používateľa a na vykonanie operácie načítania cieľovej webovej stránky.

- **Ovládací prvok WebView.** Objekt tohto ovládacieho prvku je aktívnym zobrazovačom webového obsahu (5), ktorý je získaný z cieľového webového zdroja.

5.3 Založenie projektu nového programu v štýle WinRT v jazyku C++

Projekt pre nový program v štýle WinRT založíme v jazyku C++ takto:

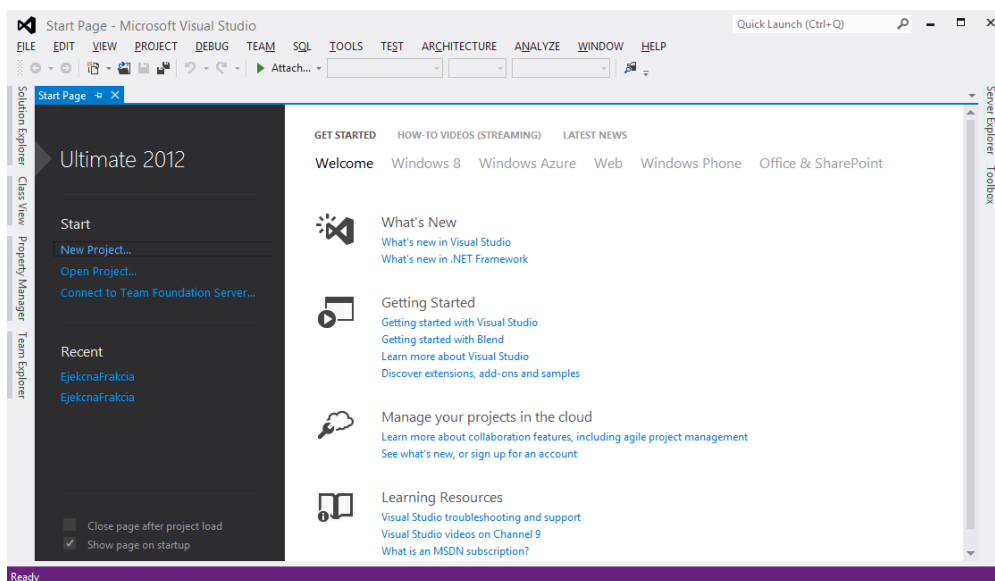


Dôležité: V nasledujúcom návodnom postupe predpokladáme, že vzhľad vývojového prostredia Visual Studio 2012 je nakonfigurovaný tak, aby zodpovedal profilu vývojára v jazyku C++. Profil vývojára v C++ nastavíme v prostredí Visual Studio 2012 takto:

1. Otvoríme ponuku **Tools** a vyberieme položku **Import and Export Settings**.
2. Spustí sa rovnomenný sprievodca. V jeho prvom kroku zvolíme možnosť **Reset all settings** a klikneme na tlačidlo **Next**.
3. V druhom kroku uskutočníme selekciu voľby **No, just reset settings, overwriting my current settings** a potvrdíme tlačidlom **Next**.
4. V treťom kroku označíme položku **Visual C++ Development Settings**, čím sprievodcovi prikážeme, aby prichystal vývojové prostredie na proces tvorby riadených a natívnych programov v jazyku C++.
5. Klepnutím na tlačidlo **Finish** spustíme konfiguračný proces vývojového prostredia Visual Studio 2012.

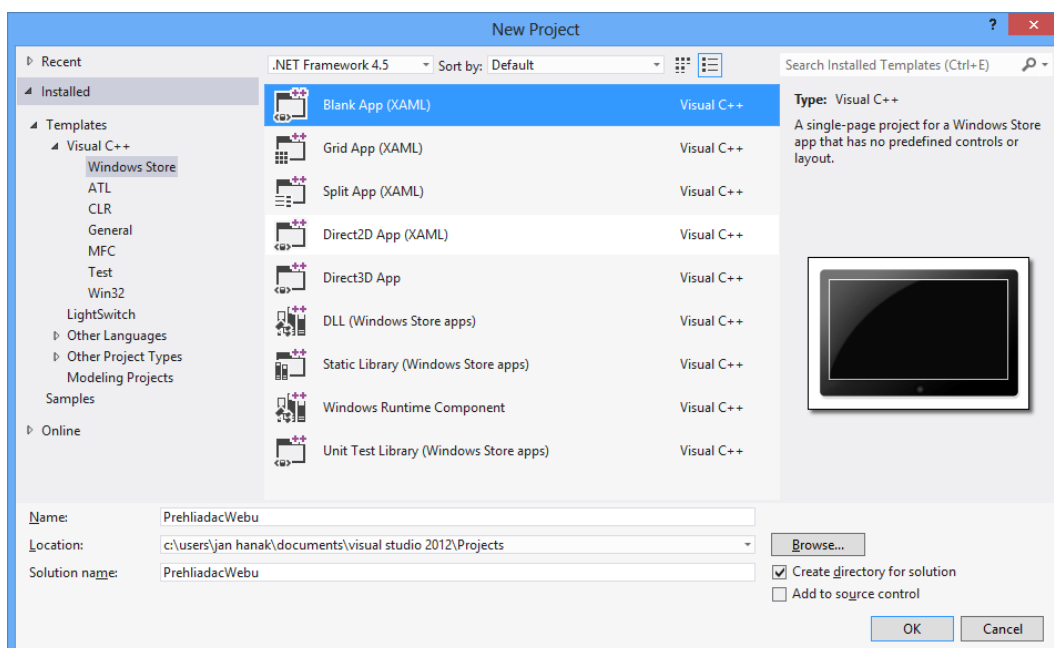
Prostredie Visual Studio 2012 môže byť variabilne konfigurované, a to presne podľa profesionálneho profilu cieľového vývojára. Je dobré vedieť, že v kolekcii vopred pripravených vývojárskych profilov sú uložené profily pre vývojárov pracujúcich v jazykoch Visual Basic, C#, C++ a F#, ďalej tiež profily pre databázových profesionálov a tvorcov webových aplikácií.

1. Na úvodnej obrazovke **Start Page** klikneme na príkaz **New Project** (obr. 5.2).



Obr. 5.2: Založenie projektu nového programu v štýle WinRT v jazyku C++

2. V dialógovom okne **New Project** uskutočníme v sekcii **Installed** → **Templates** → **Visual C++** → **Windows Store** výber projektovej šablóny **Blank App (XAML)**.
3. Do textového poľa **Name** zapíšeme názov pre nový projekt: „PrehliadacWebu“. V tomto momente by mal dialóg **New Project** vyzerat' ako jeho ekvivalent na obr. 5.3.
4. Proces generovania nového projektu a jeho súčastí odštartujeme klepnutím na tlačidlo **OK**.



Obr. 5.3: Výber projektovej šablóny **Blank App (XAML)** a základná konfigurácia projektu

V nasledujúcej podkapitole sa zoznámime so základnými súčastami novo vytvoreného projektu programu v štýle WinRT v jazyku C++.

5.4 Analýza projektových súčastí programu v štýle WinRT

Nový projekt, ktorý vznikol použitím projektovej šablóny **Blank App (XAML)**, obsahuje súčasti, ktoré bližšie opisujeme v tab. 5.1.

Tab. 5.1: Prehľad a charakteristika projektových súčastí WinRT-programu v jazyku C++

Názov súčasti	Súbor	Charakteristika
Manifest	Package.appxmanifest	<p>Manifest jednoznačne identifikuje natívne zostavenie programu v štýle WinRT. Obsahuje základné metadáta, ku ktorým patrí identifikátor programu, vstupný bod programu a jeho opis, rovnako ako aj zoznam súborov, ktoré sa v natívnom zostavení programu v štýle WinRT vyskytujú. Okrem toho však manifest združuje veľa ďalších informácií, ktoré súvisia nielen so vzťahom programu, ale aj s jeho použitím, schopnosťami, deklaráciami a konfiguráciou jeho distribučných nastavení.</p> <p>Tip: Hoci je súbor s manifestom XML-súborom, vývojové prostredie Visual Studio 2012 integruje vizuálneho návrhára manifestov (Manifest Designer), vďaka ktorému môžu vývojári vykonávať zmeny v manifeste programu vizuálnou cestou, a tak nie sú odkázaní na priamu úpravu príslušného XML-kódu.</p>
Priečink Assets	Priečink tvoria grafické súbory: Logo.png, SmallLogo.png, SplashScreen.png a StoreLogo.png	V priečinku zdrojov sa nachádzajú grafické súbory, ktoré reprezentujú vizuály pre logo programu (v normálnej a zmenšenej verzii), logo pre uvítaciu obrazovku programu a ikonu programu, ktorá sa bude objavovať v elektronickom obchode Windows.
Priečink Common	Priečink obsahuje súbor: StandardStyles.xml	V priečinku Common je situovaný XML-súbor, ktorý deklaruje štýly pre rýchlejší vývoj programov v štýle WinRT. V skutočnosti sú deklarované štýly vyžadované väčšinou projektových šablón prostredia Visual Studio 2012.
Priečink External Dependencies	Štandardné a vygenerované hlavičkové súbory	Priečink obsahuje množinu hlavičkových súborov štandardnej knižnice jazyka C++ a rovnako sa v ňom nachádza aj množina novo vygenerovaných hlavičkových súborov (ako napríklad App.g.h či MainPage.g.h).

(Tab. 5.1 pokračuje na ďalšej strane knihy)

Tab. 5.1: Prehľad a charakteristika projektových súčastí WinRT-programu v jazyku C++ (pokračovanie)

Názov súčasti	Súbor	Charakteristika
XAML-súbor, hlavičkový súbor (.h) programu a implementačný súbor (.cpp) programu	App.xaml, App.xaml.cpp a App.xaml.h	Súbor App.xaml je prispôsobený na úschovu deklarácií základných grafických prvkov programu v štýle WinRT. Na druhej strane pôsobí hlavičkový súbor (.h) a zdrojový súbor (.cpp) programu, ktoré deklarujú a definujú hlavnú triedu WinRT-programu s názvom App . Technicky je trieda App odkazovou (ref), zapečatenou (sealed) a parciálnou (partial) podtriedou bázevej triedy Application z menného priestoru Windows::UI::Xaml . V tele triedy App sa okrem definície verejného a bezparametrického konštruktora vyskytujú aj implicitne zhotovení spracovateľa udalostí OnLaunched a OnSuspending .
XAML-súbor, hlavičkový súbor (.h) a implementačný súbor (.cpp) hlavnej stránky programu	MainPage.xaml, MainPage.xaml.cpp a MainPage.xaml.h	<p>Súbor MainPage.xaml deklaruje triedu Page so základným usporiadaním pre mriežku (Grid). Obsah tohto XAML-súboru je štandardne viditeľný v dvoch zobrazovacích režimoch: jednak v režime návrhu (záložka Design) a rovnako aj v režime kódu (záložka XAML). Po použití projektovej šablóny Blank App (XAML) je hlavná stránka programu prázdna. Návrhári používateľského rozhrania ju však môžu zaplniť inštanciami vybraných ovládacích prvkov. Proces návrhu vizuálneho rozhrania programu v štýle WinRT sa môže odohrávať tiež dvojako: buď vizuálnym návrhom, alebo priamym zadaním fragmentov jazyka XAML (s podporou senzitívnej technológie IntelliSense).</p> <p>Hlavičkový súbor hlavnej stránky programu zavádza parciálnu (partial) deklaráciu odkazovej (ref) triedy MainPage. Táto trieda je podtriedou bázevej triedy Page z menného priestoru Windows::UI::Xaml::Controls.</p> <p>Trieda MainPage definuje verejný bezparametrický konštruktor a spracovateľa udalosti OnNavigatedTo.</p>

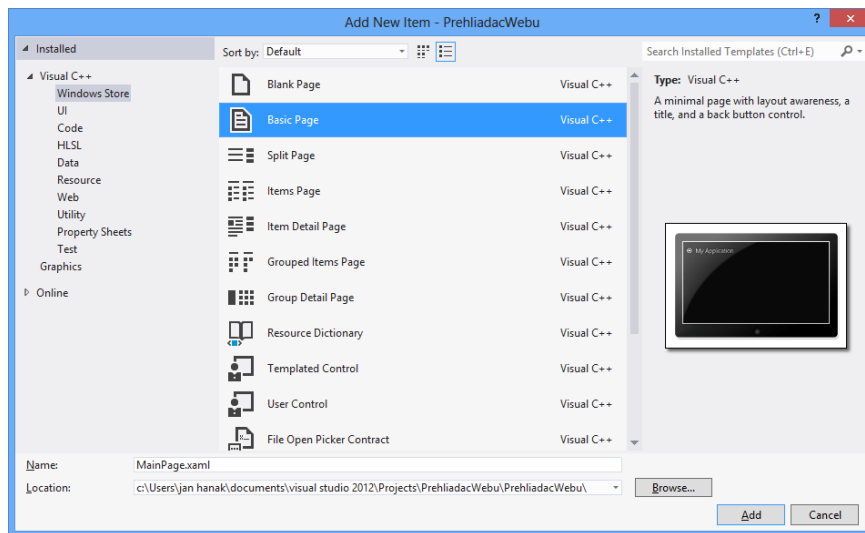
Pre vývojárov v jazyku C++ sú dôležité aj syntaktické skelety súborov App.g.h a MainPage.g.h. Na rozdiel od už spomenutých hlavičkových a implementačných súborov, súbory App.g.h a MainPage.g.h nie sú určené na explicitnú modifikáciu vývojárom. Súbory App.g.h a MainPage.g.h nie sú v podokne **Solution Explorer** implicitne viditeľné – keď ich chceme prehliadať, musíme najskôr zapnúť zobrazovanie všetkých súborov (**Show All Files**) a potom rozvinúť priečinok **Debug**. Po kliknutí na ikonu súboru App.g.h, respektíve MainPage.g.h, sa v editore zdrojového kódu objavia syntaktické skelety týchto súborov.

5.5 Vývoj grafického rozhrania prvého programu v štýle WinRT

Hoci by sme mohli upraviť prázdnu stránku WinRT-programu, lepším riešením je nahradiť túto prázdnu stránku novou stránkou so základnou funkcionalitou. Preto urobíme toto:

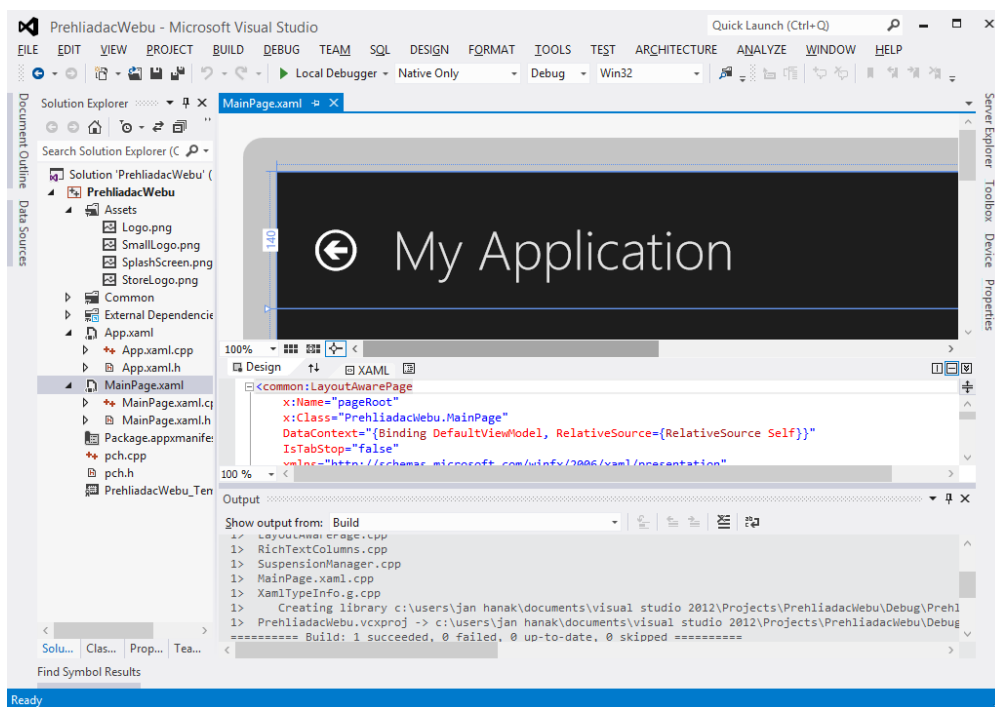
1. V podokne **Solution Explorer** klikneme na položku MainPage.xaml pravým tlačidlom myši a z kontextovej ponuky vyberieme príkaz **Remove**.
2. Objaví sa potvrdzovacie dialógové okno, v ktorom klikneme na tlačidlo **Delete**. Táto akcia spôsobí trvalé vymazanie súborov MainPage.xaml, MainPage.xaml.cpp a MainPage.cpp.
3. Otvoríme ponuku **Project** a aktivujeme príkaz **Add New Item**.
4. V stromovej štruktúre naľavo označíme možnosť **Visual C++** → **Windows Store**.
5. Z kolekcie šablón stránok vyberieme stránku so základnou funkcionalitou **Basic Page**.

6. Súboru novo pridávanej základnej stránky, ktorá sa stane hlavnou stránkou vyvíjaného programu v štýle WinRT, prisúdime identifikátor MainPage.xaml (obr. 5.4).



Obr. 5.4: Pridanie základnej stránky (**Basic Page**) do projektu WinRT-programu

7. Stisneme tlačidlo **Add**, čím pridáme základnú stránku **Basic Page** do projektu programu v štýle WinRT.
8. Prostredie Visual Studio 2012 zobrazí dialógové okno s požiadavkou na import potrebných externých projektových súčastí. My klikneme na tlačidlo **Yes**, čím odsúhlasíme spracovanie importných operácií.
9. Zostavíme projekt voľbou **Build** → **Build Solution** (alebo rýchlejšie stlačíme kláves F7).
10. Po zostavení projektu uvidíme v XAML-návrhárovi grafický model hlavnej stránky nášho programu v štýle WinRT (obr. 5.5).



Obr. 5.5: Hlavná stránka programu v štýle WinRT so základnou funkcionalitou

Základná stránka WinRT-programu je vďaka použitiu projektovej súčasti **Basic Page** vyspelejšia, ako bola predtým. Obsahuje tlačidlo návratu a hlavný text (zatiaľ implicitne nastavený na hodnotu „My Application“).

5.5.1 Úprava vzhľadu hlavnej stránky programu v prostredí produktu Blend for Visual Studio 2012

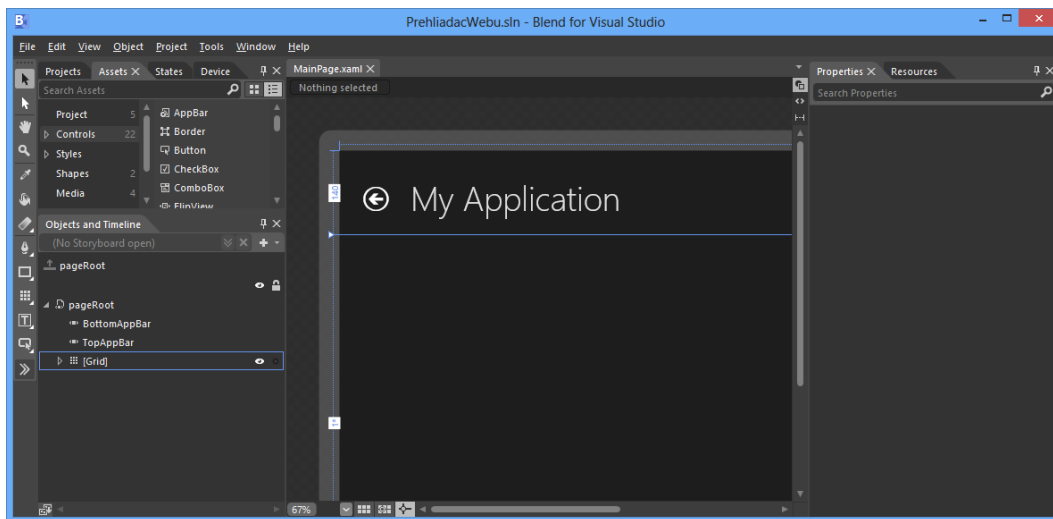
Grafickú podobu hlavnej stránky programu v štýle WinRT môžeme upravovať tromi spôsobmi:

- Pomocou XAML-návrhára, ktorý je zabudovaný do prostredia Visual Studio 2012.

- Pomocou samostatne pôsobiaceho grafického editora Blend for Visual Studio 2012.
- Pomocou externého grafického nástroja, ktorý je schopný generovať a editovať XAML-kód projektu WinRT-programu.

V tejto časti podkapitoly si ukážeme, ako na účel úpravy vizuálneho rozhrania programu v štýle WinRT použiť produkt Blend for Visual Studio 2012. Prosím, riadte sa nasledujúcimi inštrukciami:

1. Začneme tak, že v podokne **Solution Explorer** klikneme pravým tlačidlom myši na položku MainPage.xaml a z lokálnej ponuky vyberieme príkaz **Open in Blend**.
2. Spustí sa editor Blend for Visual Studio 2012, ktorý automaticky načíta všetky súčasti projektu WinRT-programu. Hlavná stránka programu sa rozprestrie na návrhárskom plátne (obr. 5.6).

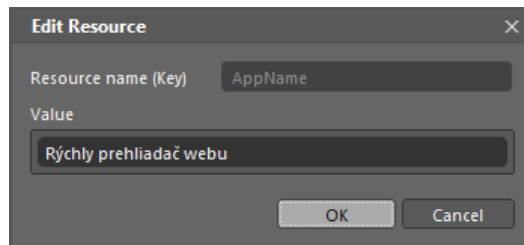


Obr. 5.6: Hlavná stránka WinRT-programu v editore Blend for Visual Studio 2012

3. V podokne **Objects and Timeline** (ktoré je implicitne ukotvené naľavo) sa nachádza osnova grafických elementov tvoriacich hlavnú stránku programu.

Koreňovým elementom tejto osnovy je prvok **pageRoot**. Keďže naším prvým krokom je zmena hlavného textu programu, označíme element **pageTitle**. Úplná cesta k spomínanému elementu znie: **pageRoot** → **Grid** → **Grid** → **pageTitle**.

4. Hlavný text programu je uchovaný v objekte ovládacieho prvku **TextBlock**. Najskôr zmeníme vlastnosť **Name** tohto objektu na hodnotu **txbTitulnyText**. (Keď realizáciu tejto zmeny potvrdíme aktiváciou klávesu Enter, zmení sa identifikátor objektu aj v osnove podokna **Objects and Timeline**.)
5. Hlavný text „Rýchly prehliadač webu“ uložíme do vlastnosti **Text** objektu prvku **TextBlock**. Vlastnosť **Text** je zaradená v sekcii **Common**. Po kliknutí na pôvodnú hodnotu tejto vlastnosti („My Application“) sa rozvinie lokálna ponuka, z ktorej vyberieme príkaz **Edit Resource**. V zobrazenom dialógu potom vykonáme zmenu hodnoty (**Value**) tejto vlastnosti podľa zadania (obr. 5.7).



Obr. 5.7: Úprava titulného textu programu v štýle WinRT

6. Po stisnutí tlačidla **OK** sa zmení hlavný text WinRT-programu tak, ako sme chceli.

Ďalej budeme postupovať podľa nákresu vizuálneho rozhrania nášho prvého WinRT-programu v jazyku C++, ktorý sme uviedli na obr. 5.1 v podkapitole 5.2 *Náčrt vizuálneho rozhrania prvého programu v štýle WinRT*. To znamená, že na mriežku hlavnej stránky programu nanesieme inštancie všetkých ovládacích prvkov, ktoré budeme potrebovať na získanie adresy webovej stránky a zobrazenie jej obsahu. Tab. 5.2 sumarizuje vytvorené a nakonfigurované objekty grafického rozhrania programu.



Tip: Ovládacie prvky volíme z podokna **Assets**. Odporúčame zapnúť voľbu zobrazenia všetkých dostupných ovládacích prvkov. Túto voľbu aktivujeme tak, že v ľavom paneli podokna **Assets** klepneme najskôr na položku **Controls**, a potom aktivujeme položku **All**. Vlastnosti už zostrojených objektov príslušných ovládacích prvkov modifikujeme v podokne **Properties** (toto podokno je implicitne ukotvené pri pravej strane hlavného okna nástroja Blend for Visual Studio 2012).

Tab. 5.2: Sumárny prehľad objektov vizuálneho rozhrania WinRT-programu v jazyku C++		
Objekt	Vlastnosť objektu	Hodnota vlastnosti objektu
TextBlock (Objekt slúži na zobrazenie textového návestia)	Name	txbWebovaAdresa
	Text	Zadajte webovú adresu:
	Width	210
	Height	24
	Margin (Left)	84
	Margin (Top)	54
	Text (FontSize)	14 pt

(Tab. 5.2 pokračuje na ďalšej strane knihy)

Tab. 5.2: Sumárny prehľad objektov vizuálneho rozhrania WinRT-programu v jazyku C++ (pokračovanie)

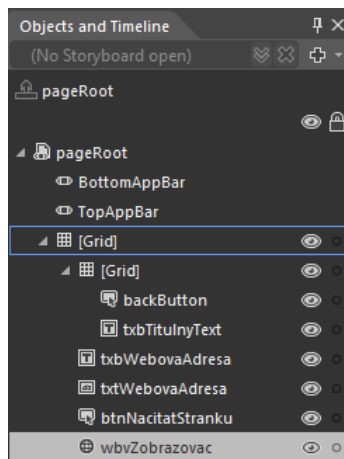
Objekt	Vlastnosť objektu	Hodnota vlastnosti objektu
TextBox (Objekt slúži na absorpciu adresy webovej stránky)	Name	txtWebovaAdresa
	Text	http://www.microsoft.sk
	Width	412
	Height	42
	Margin (Left)	321
	Margin (Top)	44
	Text (FontSize)	14 pt
Button (Objekt je tlačidlo, po klepnutí naň sa začne proces načítavania webovej stránky)	Name	btnNacitatStranku
	Content	Načítať stránku
	Width	209
	Height	42
	Margin (Left)	768
	Margin (Top)	44
	Text (FontSize)	14 pt

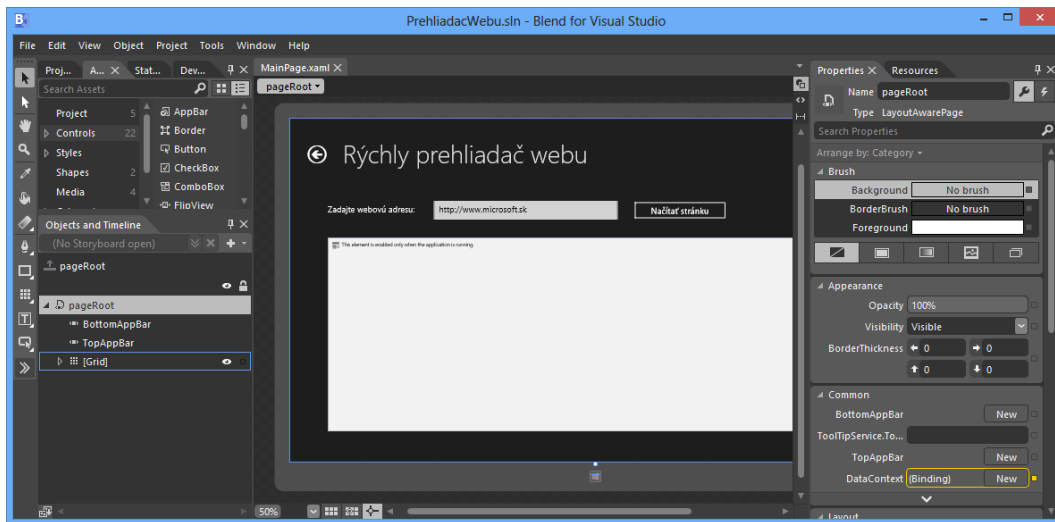
(Tab. 5.2 pokračuje na ďalšej strane knihy)

Tab. 5.2: Sumárny prehľad objektov vizuálneho rozhrania WinRT-programu v jazyku C++ (pokračovanie)

Objekt	Vlastnosť objektu	Hodnota vlastnosti objektu
WebView (Objekt hostujúci a zobrazujúci HTML- obsah)	Name	wbvZobrazovac
	Width	1213
	Height	433
	Margin (Left)	84
	Margin (Top)	127

Po pridaní všetkých objektov na mriežku hlavnej stránky WinRT-programu vyzerá podokno **Objects and Timeline** ako na obr. 5.8. Finálny vzhľad návrhárskeho plátna programu Blend for Visual Studio 2012 znázorňuje zase obr. 5.9.

Obr. 5.8: Finálna podoba podokna **Objects and Timeline**



Obr. 5.9: Finálna podoba návrhárskeho plátna programu Blend for Visual Studio 2012

Nakoniec vykonané zmeny uložíme z prostredia programu Blend for Visual Studio 2012 (**File** → **Save All**). Vývojové prostredie Visual Studio 2012 deteguje externú zmenu súboru hlavnej stránky WinRT-programu (MainPage.xaml), na ktorú reaguje zobrazením dialógového okna s odporúčaním na opätovné načítanie tejto stránky. To je presne to, čo chceme, a preto v dialógovom okne zvolíme tlačidlo **Yes to All**.

5.6 Implementácia logiky prvého programu v štýle WinRT

Logika programu bude implementovaná v spracovateľovi udalosti **Click** tlačidla na načítanie obsahu webovej stránky. Tohto spracovateľa najrýchlejšie vygenerujeme poklepaním na tlačidlo vo vizuálnom návrhovi prostredia Visual Studio 2012.



Poznámka: Štandardný postup na samočinné vygenerovanie spracovateľa ľubovoľnej udalosti pre ktorýkoľvek objekt je v prostredí Visual C++ 2012 nasledujúci:

1. Vo vizuálnom návrhovi preneseť zameranie na cieľový objekt.

2. Presunieme sa do podokna **Properties** a aktivujeme tlačidlo s ikonou blesku (⚡).
3. V kolekcií zobrazených udalostí vyhľadáme požadovanú udalosť.
4. Do textového poľa, ktoré je prepojené s požadovanou udalosťou, zadáme názov jej spracovateľa. Na zostavenie implicitne pomenovaného spracovateľa udalosti stačí, keď dvakrát klikneme do asociovaného textového poľa.

Predostretý algoritmus tvorby spracovateľov pre udalosti objektov je platný tak pre inštancie ovládacích prvkov (s vlastnou vizuálnou reprezentáciou za behu programu), ako aj pre inštancie komponentov (bez vlastnej vizuálnej reprezentácie za behu programu).

Spracovateľ udalosti **Click** tlačidla **btnNacitatStranku** je situovaný v zdrojovom súbore MainPage.xaml.cpp a jeho syntaktický obraz je takýto:

```
void PrehliadacWebu::MainPage::btnNacitatStranku_Click(Platform::Object^ sender,
Windows::UI::Xaml::RoutedEventArgs^ e)
{
    // Definícia adresy webovej stránky ako zdroja URI.
    Uri ^webovaStranka = ref new Uri(txtWebovaAdresa->Text);
    // Načítanie obsahu webovej stránky do zobrazovača.
    wbvZobrazovac->Navigate(webovaStranka);
}
```

Komentár k zdrojovému kódu jazyka C++: Najskôr vytvárame inštanciu triedy **Uri** (z menného priestoru **Windows::Foundation**). Preťaženému konštruktoru tejto triedy odovzdávame webovú adresu, ktorú zadá používateľ (pripomeňme, že ak nie je určené inak, bude implicitnou webovou adresou sídlo spoločnosti Microsoft). Parametrický konštruktor preto vytvorí zdroj URI z textového reťazca, ktorý je uchovaný v textovom poli **txtWebovaAdresa**.



Dôležité: Hoci skrátené charakterizujeme uvedený zdrojový kód ako kód jazyka C++, v skutočnosti je tento zdrojový kód zapísaný v jazyku C++/CX (teda v natívnom C++ so začlenenými komponentovými rozšíreniami). Programové konštrukcie, operátory a kľúčové slová, ktoré jazyk C++/CX prináša, sa vo veľkej miere podobajú na syntaktické ekvivalenty jazyka C++/CLI. Ak ste niekedy pracovali v jazyku C++/CLI, rýchlo pochopíte aj sémantiku zdrojového kódu komponentových rozšírení jazyka C++. Radi by sme však poukázali na skutočnosť, že zatiaľ čo programy napísané

v jazyku C++/CLI mohli byť riadené, zmiešané či natívne, aplikácie vytvorené v jazyku C++/CX sú vždy rýdzo natívne.

Pri definovaní lokálnej odkazovej premennej **webovaStranka** používame operátor striedky (^). Na operátor ^ môžeme v jazyku C++/CX nahliadať ako na ekvivalent operátora * v jazyku C++. Nový objekt triedy **Uri** zakladáme pomocou operátora **new**, s ktorým je kontextovo spojený modifikátor **ref**. Inštančný výraz **ref new Uri(...)** znamená pre prekladač okamžitú alokáciu objektu špecifikovanej triedy a jeho inicializáciu parametrickým konštruktorom. Odkaz na zrodený objekt je uskladnený v odkazovej premennej **webovaStranka**.

Na interpretáciu HTML-obsahu cieľovej webovej stránky používame špeciálny zobrazovač – objekt ovládacieho prvku **WebView**. Keď v súvislosti s týmto objektom zavoláme jeho parametrickú metódu **Navigate** s jednoznačne determinovaným zdrojom URI, zobrazovač vykreslí HTML-kód cieľovej webovej stránky.

5.7 Konfigurácia grafických zdrojov prvého programu v štýle WinRT

Grafické zdroje WinRT-programu sú prístupné z priečinka **Assets** podokna **Solution Explorer**. Fyzicky sa tieto grafické zdroje nachádzajú v priečinku **PrehliadacWebu\PrehliadacWebu\Assets**.

Na nasledujúcich riadkoch podáme ich stručnú charakteristiku:

- **Logo programu v štýle WinRT.** Toto logo musí mať rozmery 150x150 obrazových bodov. Grafický súbor uchovávajúci toto logo má názov **Logo.png**.
- **Logo programu v štýle WinRT pre potreby elektronického obchodu Windows.** Toto logo musí mať rozmery 50x50 obrazových bodov. Grafický súbor uchovávajúci toto logo má názov **StoreLogo.png**.

- **Malé logo programu v štýle WinRT.** Toto logo musí mať rozmery 30x30 obrazových bodov. Grafický súbor uchovávajúci toto logo má názov SmallLogo.png.
- **Uvítacia obrazovka programu v štýle WinRT.** Táto obrazovka musí mať rozmery 620x300 obrazových bodov. Grafický súbor uchovávajúci toto logo má názov SplashScreen.png.

Pre prvý program v štýle WinRT v jazyku C++ sme pripravili logá vo všetkých potrebných rozmeroch a rovnako aj uvítaciu obrazovku (obr. 5.10).



Logo
(150x150)



Logo (50x50)



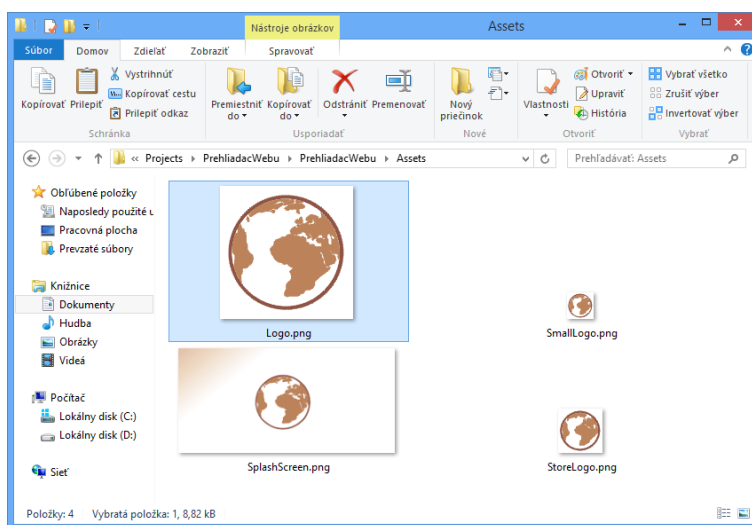
Logo (30x30)



Uvítacia obrazovka

Obr. 5.10: Kolekcia navrhnutých grafických zdrojov pre WinRT-program

Aby všetko fungovalo správne, musíme upraviť vzhľad všetkých grafických zdrojov, ktoré sú umiestnené v priečinku **Assets** (obr. 5.11).



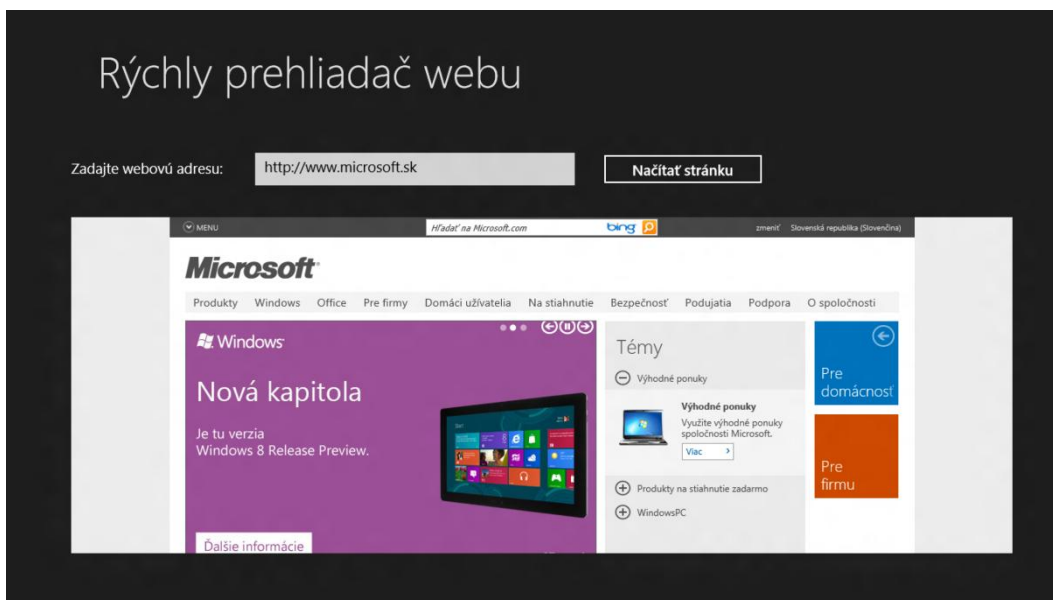
Obr. 5.11: Hotové grafické zdroje WinRT-programu

Ak si neprajeme na dlaždici WinRT-programu zobrazovať názov programu, zvolíme možnosť **No Logos** pri voľbe **Show name** v nástroji **Manifest Designer**.

5.8 Zostavenie a spustenie programu v štýle WinRT v systéme Windows 8

Projekt programu v štýle WinRT zostavíme spustením príkazu **Build Solution** z ponuky **Build** (ekvivalentná klávesová skratka F7). Nasadenie WinRT-programu sa uskutoční vo chvíli, keď z ponuky **Debug** vyberieme voľbu **Start Debugging** (respektíve stlačíme kláves F5).

Program v štýle WinRT, ktorý sme vyvinuli v jazyku C++, môžeme v akcii vidieť na obr. 5.12.

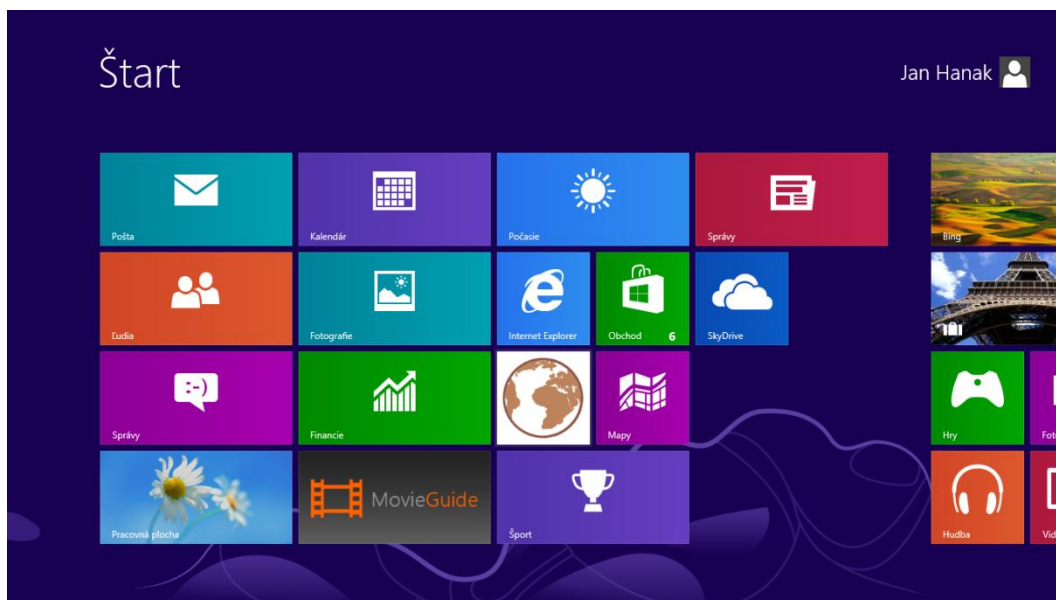


Obr. 5.12: Rýchly prehliadač webu ako WinRT-program jazyka C++

Teraz je ten správny čas na otestovanie správneho fungovania WinRT-programu. Vzhľadom na to, že textové pole s adresou cieľovej webovej stránky je vyplnené ihneď po spustení programu, môžeme rovno klepnúť na tlačidlo **Načítať stránku**. V závislosti od rýchlosti nainštalovaného internetového pripojenia bude HTML-obshav úvodnej stránky spoločnosti Microsoft zobrazený s menšou alebo väčšou latenciou. S programom môžeme experimentovať: jednoducho zadajme do adresného panela adresy ďalších webových sídiel a pozorujme, ako sa mení obsah zabudovaného webového zobrazovača.

Späť do vývojového prostredia Visual Studio 2012 sa dostaneme prostredníctvom klávesovej skratky ALT+TAB. Beh relácie programu v štýle WinRT ukončíme aktiváciou príkazu **Stop Debugging** z ponuky **Debug** (alebo aplikovaním klávesovej skratky SHIFT+F5).

Nový WinRT-program má svoju dlaždicu na páse dlaždíc ponuky **Štart** (obr. 5.13).



Obr. 5.13: Dlaždica nového WinRT-programu v ponuke **Štart**

Keď na dlaždicu programu klepneme pravým tlačidlom myši, objaví sa spodný panel s možnosťami na zrušenie pripnutia tohto programu a jeho odinštalovanie.

Záver

V tejto príručke sme vás zoznámili so základným fungovaním operačného systému Microsoft Windows 8 a s procesom vytvárania prvých programov, ktoré sú úplne kompatibilné s novým vizuálnym rozhraním. Rovnako sme vyložili problematiku životných cyklov WinRT-programov a prakticky sme zostavili prvé riadené a natívne programy v štýle WinRT pomocou programovacích jazykov Visual Basic, C# a C++.

Samozrejme, sme si vedomí skutočnosti, že táto publikácia predstavuje len prvý krok vo vzdelávaní vývojárov, ktorí chcú tvoriť softvérové produkty pre prostredie WinRT systému Windows 8. Teraz, keď ste preštudovali kompletnú obsahovú náplň tejto knihy, radi by sme rozšírili vaše obzory a poskytli vám zbierku ďalších hodnotných informačných zdrojov, z ktorých môžete čerpať nové vedomosti a praktické skúsenosti.

Uvádzame kolekciu nových informačných zdrojov pre vývojárov v prostredí WinRT:

1. Domovská stránka pre vývojárov programov v štýle WinRT:
<http://msdn.microsoft.com/en-us/windows/apps/br229512>.
2. Praktické ukážky programov v štýle WinRT na voľné prevzatie:
<http://code.msdn.microsoft.com/windowsapps/>.
3. Komunitné zdroje a blogy pre vývojárov programov v štýle WinRT:
<http://msdn.microsoft.com/en-US/windows/apps/br229515>.
4. Prehľad návrhárskych princípov využiteľných pri vývoji programov v štýle WinRT:
<http://msdn.microsoft.com/en-US/library/windows/apps/hh779072>.
5. Kolekcia rozhraní API na podporu vývoja programov v štýle WinRT v jazykoch Visual Basic, C#, C++ a JavaScript:
<http://msdn.microsoft.com/library/windows/apps/br211369>.

Prajeme vám veľa úspechov pri vývoji nových programov v štýle WinRT!

Autor a realizačný tím spoločnosti Microsoft

O autorovi

Ing. Ján Hanák, PhD., MVP, vyštudoval Ekonomickú univerzitu v Bratislave. Tu, na Katedre aplikovanej informatiky Fakulty hospodárskej informatiky (KAI FHI), pracuje ako vysokoškolský pedagóg. Prednáša a vedie semináre týkajúce sa programovania a vývoja počítačového softvéru v programovacích jazykoch C, C++ a C#. K jeho favoritom patrí tiež Visual Basic, F# a C++/CLI.

Je renomovaným autorom odbornej počítačovej literatúry. V jeho portfóliu môžete nájsť nasledujúce knižné tituly:

1. **Vývoj moderných WinRT-programov pre systém Windows 8.** Bratislava: Microsoft Slovakia, 2012.
2. **Základy databázového vývoja v prostredí Visual Studio LightSwitch 2011.** Bratislava: Microsoft Slovakia, 2012.
3. **Visual Basic 2010 - praktické príklady.** Kralice na Hané: Computer Media, 2012.
4. **Moderné paralelné programovanie.** Bratislava: Vydavateľstvo EKONÓM, 2011.
5. **Programování v jazyce Visual Basic 2010.** Kralice na Hané: Computer Media, 2011.
6. **Softvérové technológie na platforme Microsoft .NET.** Bratislava: Eurokódex, 2011.
7. **Rýchly vývoj aplikácií v jazyku Visual Basic 2010 pre systém Windows 7.** Bratislava: Microsoft Slovakia, 2011.
8. **Programování v jazyce C.** Kralice na Hané: Computer Media, 2011.
9. **Ako sa stať softvérovým vývojárom.** Bratislava: Microsoft Slovakia, 2010.
10. **C++: Akademický výučbový kurz.** Bratislava: Vydavateľstvo EKONÓM, 2010.
11. **Inovácie v jazykoch Visual Basic 2010, C# 4.0 a C++.** Brno: Artax, 2010.
12. **Programování v jazyce C.** Kralice na Hané: Computer Media, 2010.
13. **Visual Basic 2010 – Hotové riešenia.** Bratislava: Microsoft Slovakia, 2010.
14. **C#: Akademický výučbový kurz, 2. aktualizované a rozšírené vydanie.** Bratislava: Vydavateľstvo EKONÓM, 2010.
15. **Praktické objektové programování v jazyce C# 4.0.** Brno: Artax, 2009.

16. **Praktické paralelné programovanie v jazykoch C# 4.0 a C++**. Brno: Artax, 2009.
17. **C++/CLI - Praktické príklady**. Brno: Artax, 2009.
18. **C# 3.0 - Programování na platformě .NET 3.5**. Brno: Zoner Press, 2009.
19. **C++/CLI - Začínáme programovat**. Brno: Artax, 2009.
20. **C#: Akademický výučbový kurz**. Bratislava: Vydavateľstvo EKONÓM, 2009.
21. **Základy paralelného programovania v jazyku C# 3.0**. Brno: Artax, 2009.
22. **Objektovo orientované programovanie v jazyku C# 3.0**. Brno: Artax, 2008.
23. **Inovácie v jazyku Visual Basic 2008**. Praha: Microsoft, 2008.
24. **Visual Basic 2008: Grafické transformácie a ich optimalizácie**. Bratislava: Microsoft Slovakia, 2008.
25. **Programovanie B – Zbierka prednášok (Učebná pomôcka na programovanie v jazyku C++)**. Bratislava: Vydavateľstvo EKONÓM, 2008.
26. **Programovanie A – Zbierka prednášok (Učebná pomôcka na programovanie v jazyku C)**. Bratislava: Vydavateľstvo EKONÓM, 2008.
27. **Expanzívne šablóny: Príručka pre tvorbu "code snippets" pre Visual Studio**. Bratislava: Microsoft Slovakia, 2008.
28. **Kryptografia: Príručka pre praktické odskúšanie symetrického šifrovania v .NET Framework-u**. Bratislava: Microsoft Slovakia, 2007.
29. **Príručka pre praktické odskúšanie vývoja nad Windows Mobile 6.0**. Bratislava: Microsoft Slovakia, 2007.
30. **Príručka pre praktické odskúšanie vývoja nad DirectX**. Bratislava: Microsoft Slovakia, 2007.
31. **Príručka pre praktické odskúšanie automatizácie aplikácií Microsoft Office 2007**. Bratislava: Microsoft Slovakia, 2007.
32. **Visual Basic 2005 pro pokročilé**. Brno: Zoner Press, 2006.
33. **C# – praktické příklady**. Praha: Grada Publishing, 2006.
34. **Programujeme v jazycích C++ s Managed Extensions a C++/CLI**. Praha: Microsoft, 2006.
35. **Přecházíme z jazyka Visual Basic 6.0 na jazyk Visual Basic 2005**. Praha: Microsoft, 2005.

36. **Visual Basic .NET 2003 – Začínáme programovat.** Praha: Grada Publishing, 2004.

V rokoch 2006 – 2012 bol jeho prínos vývojárskym komunitám ocenený celosvetovými vývojárskymi titulmi **Microsoft Most Valuable Professional (MVP)** s kompetenciou **Visual Developer – Visual C++**.



Ing. Ján Hanák, PhD., MVP, pôsobí ako vysokoškolský pedagóg, spisovateľ počítačovej literatúry, softvérový vývojár a evanjelista moderných IT technológií. Na svojom konte má 36 odborných prác, ku ktorým patria vysokoškolské učebnice, vedecké monografie, skriptá a príručky venované problematike vývoja počítačového softvéru v programovacích jazykoch C, C++, C#, Visual Basic a C++/CLI. V rokoch 2006 – 2012 bol jeho prínos vývojárskymi komunitám ocenený celosvetovo uznávanými vývojárskymi titulmi spoločnosti Microsoft s názvom Most Valuable Professional (MVP) s kompetenciou Visual Developer – Visual C++.