Microsoft®
Silverlight™

Ľuboslav Lacko

Silverlight 3.0

Úvod do vývoja aplikácií na platforme Silverlight 3

Microsoft®

Silverlight 3

Ľuboslav Lacko

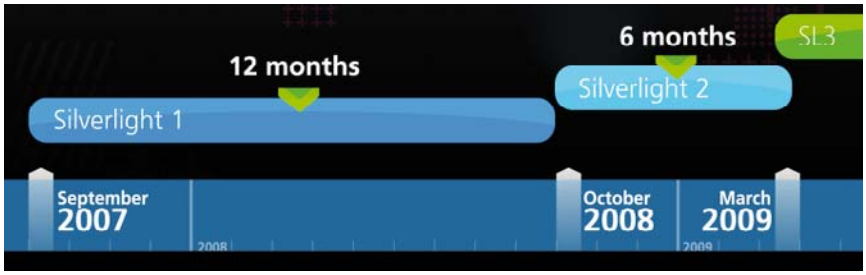
Obsah

Stručný pohľad do histórie platformy Silverlight

K výraznému zvýšeniu úrovne prezentačnej vrstvy a interaktivity webových aplikácií môže prispieť aj technológia Silverlight z dielne spoločnosti Microsoft. Silverlight rozširuje prezentačnú úroveň prehľadávača webového obsahu o nové možnosti s využitím vektorovej grafiky a multimédií. Fyzicky je to plug-in do prehľadávača webového obsahu.

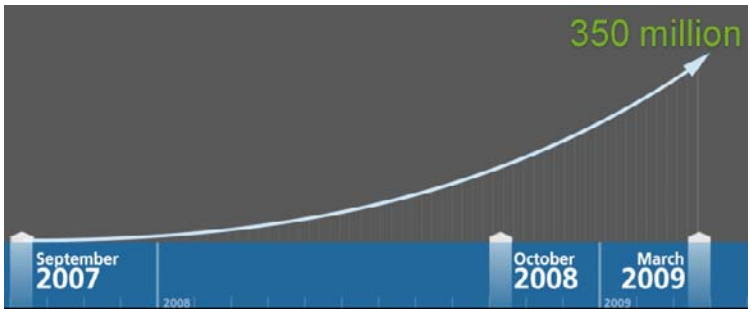
Zjednodušene by sa Silverlight aplikácia dala prirovnať k interaktívnemu zobrazovaciemu a pracovnému priestoru, ktorý je zobrazený u klienta v okne prehľadávača webového obsahu, alebo v samostatnom okne, pričom aplikačná logika beží na serveri.

História platformy Silverlight je pomerne krátka, začala sa písať v septembri 2007, kedy spoločnosť Microsoft zároveň s finálnou verziou Silverlight 1.0 predstavila aj alfa verziu 1.1. Táto verzia bola následne, približne o rok premenovaná na Silverlight 2.0. Zatiaľ čo Silverlight 1.0 využíval ako programovací jazyk iba JavaScript, vo verzii 2.0 už bolo možné v plnej miere využívať .NET jazyky. Predtým, v procese vývoja mala táto technológia označenie Windows Presentation Foundation/Everywhere, čo naznačuje, zámer spoločnosti Microsoft preniesť čo možno najviac čŕt nového prezentačného rozhrania WPF(Windows Presentation Foundation), ktoré je súčasťou pla Framework od verzie 3.0.



Časová mapa histórie produktu Silverlight

Za necelé dva roky svojej histórie vzrástol počet inštalácií na klientskych počítačoch až na 350 miliónov, pričom krivka rastu je exponenciálna. Technológiu Silverlight využíva viac než 300 000 vývojárov a dizajnérov. Spoločnosť Microsoft má v súvislosti s touto technológiou 200 partnerov v 30 krajinách a využíva ju aj v takmer 200 svojich produktoch a webových projektoch. V globálnom meradle je k dispozícii viac než 10 000 aplikácií.



Exponenciálny nárast počtu používateľov produktu Silverlight sa vyšplhal až k 350 miliónom

Spoločnosť Microsoft využíva technológiu Silverlight vo vyše 200 svojich produktoch a weboch

V celosvetovom meradle sa technológia Silverlight využíva vo viac ako 10 000 aplikáciách

Pre ilustráciu možností prezentácie videa cez Silverlight uvedieme niekoľko čísel týkajúcich sa televíznych prenosov NBC z olympijských hier 2008 v Pekingu:

- 1,3 miliardy zobrazených stránok,
- 52,1 milióna unikátnych návštevníkov,
- 75,5 miliónov pozretých videoprenosov,
- 9,9 miliónov hodín videa (1 126 rokov),
- 27 minút priemerný čas pozerania,
- 35 miliónov prístupov z mobilných zariadení,
- 130 000 peak streams,
- 3,4 petabytov preneseného videa.

Platforma Silverlight je teda vo verzii 3.0 a v rovnakej verzii je aj táto publikácia. Aby mohli s publikáciou pracovať aj začiatočníci a mali podrobné a kompletné návody na základné typy aplikácií, niektoré základné témy sa mierne prekrývajú s predchádzajúcou publikáciou. Nakoľko Silverlight 2 je podmnožinou trojky, niektoré témy, napríklad databázové Silverlight aplikácie využívajúce LINQ, aplikácie využívajúce webové služby a podobne nájdete v predchádzajúcej verzii publikácie.

Novinky vo verzii 3.0

Od 10. 7. 2009 je k dispozícii verzia 3.0, ktorá prináša niektoré významné novinky a vylepšenia. Najskôr urobíme stručný prehľad noviniek a potom budú v jednotlivých kapitolách podrobne predstavené. Jednou z najvýznamnejších noviniek je možnosť **behu aplikácie na klientskom počítači „mimo prehľadávač webového obsahu".** V originálnej terminológii sa táto črta nazýva **„Out of Browser"** (OOB). Silverlight aplikácia sa najskôr nainštaluje do lokálneho operačného systému, kde na rozdiel od bežných aplikácií beží v izolovanom priestore „sandboxe", takže sa používatelia nemusia obávať dôsledkov prípadného škodlivého kódu. Lokálna inštalácia sa z pohľadu koncového používateľa správa ako desktopová – je reprezentovaná zástupcom na ploche, alebo v ponuke menu Štart a beží bez nutnosti inštalácie akéhokoľvek ďalšieho softvéru a to aj pri dočasnom odpojení od Internetu.

Silverlight 3 je plnohodnotná platforma pre RIA (Rich Internet Applications) aplikácie, či už pre zábavu, alebo podporu biznisu. K tomu prispieva aj **rozšírená dátová podpora a podpora pre business objekty**. Tieto je možné jednoducho triediť, filtrovať a stránkovať, presne tak ako sú používatelia zvyknutí pri klasických aplikáciách. Rovnako ako v ASP.NET, aj na platforme Silverlight 3 je možné validovať vstupné údaje zadané používateľmi a vizuálne ho informovať o jeho chybách. Na strane klienta je k dispozícii nový objekt CollectionView a **množina operácií pre prácu s údajmi na serveri**. Serverová stránka sa potom realizuje cez .NET RIA Services.

Pre prepojenie údajov na vizuálne objekty slúži **vylepšený databinding**. **ElementName binding** umožňuje prepojenie viacerých prvkov navzájom a to priamo v XAML kóde. V zborníku nájdete príklad pre prepojenie potenciometra a grafického objektu, pričom posúvaním potenciometra sa mení niektorý parameter grafického objektu. RelativeSource binding umožňuje prepojenie prvku „samého na seba", alebo s údajmi šablóny, ak je jej súčasťou. Nový ovládací prvok typu **dátový formulár** podporuje validáciu, aktualizáciu a stránkovanie údajov.

K rozšíreniu platformy Silverlight 3 nepochybne prispeje aj jej otvorenosť. Všetky ovládacie prvky boli uvoľnené aj so zdrojovými kódmi ako projekt **Silverlight Toolkit**.

Čo nenájdu vyhľadávače, to ako keby na Internete ani neexistovalo. Jedným z čoraz významnejších faktorov úspechu v prakticky každej oblasti podnikania je zobrazenie prepojení na webový obsah príslušnej firmy na čelných pozíciách významných vyhľadávačov. Algoritmy vyhľadávania a utrieďovania výsledkov vyhľadávania sú jedným z najcennejších know-how každého vyhľadávača a prísne sa utajujú. Preto získavajú na význame rôzne metódy spätného inžinierstva, poodhaľujúce okľukou činnosť týchto algoritmov. Na ich základe sa vypracovávajú odporúčania pre tvorcov webových stránok. Táto problematika sa zvykne označovať skratkou **SEO** (Search Engine Optimization, alebo po našom optimalizácia pre vyhľadávače), čo je metodika odporúčaní pre tvorbu webových stránok, aby sa tieto zobrazovali vo výsledkoch vyhľadávania v najpoužívanejších internetových vyhľadávačoch na popredných pozíciách, čo priláka na tieto stránky čo najviac návštevníkov. Vyhľadávacie algoritmy mali problém z RIA technológiami, preto je veľmi dôležité, aby podporovali SEO. Silverlight 3 dokáže previesť obsah, ktorý sa generuje z databáz na ľahko indexovateľný HTML kód. Podporuje aj bookmarky vo vnútri aplikácie. V anglickej dokumentácii sa táto vlastnosť nazýva **Deep Linking**.

Pre dobrý dojem z plynulosti zobrazovania je dôležitá **podpora hardvérovej akcelerácie**, čo prenesie časť záťaže z CPU na GPU. Pre dobrý grafický dojem sú určené aj **3D transformácie**, ktoré umožňujú umiestniť rovinu v ktorej sú ovládacie prvky ľubovoľne do priestoru. K dispozícii sú aj **grafické efekty na úrovni pixlov**, napríklad tieňovanie alebo rozostrenie.

Nakoľko Silverlight primárne vykresľuje grafiku vektorovo, je pre zvýšenie výkonu graficky bohatej aplikácie k dispozícii možnosť **bitmapovej cache**. Grafický obsah, ktorý nemusí meniť veľkosť je možné z vektorovej formy previesť na bitovú mapu a tú umiestniť do lokálnej vyrovnávacej pamäte. Typickým príkladom použitia

je pozadie aplikácie. S jednotlivými pixlami a rámcami je možné pracovať na úrovni triedy WritableBitmap. Dá sa použiť na snímanie videa, dátovú vizualizáciu, alebo algoritmické generovanie grafického obsahu.

Pomocou technológie **Deep Zoom** je možné vykresľovať aj obrázky s vysokým rozlíšením. Príklad pre zobrazenie fotky v rôznom rozlíšení nájdete v zborníku.

Aplikácie môžu meniť vzhľad dynamicky pomocou grafických tém. Pre animácie reálneho pohybu podľa fyzikálnych zákonov je k dispozícii nová črta **Animation Easing**. V zborníku je príklad animácie skákajúcej guľôčky.

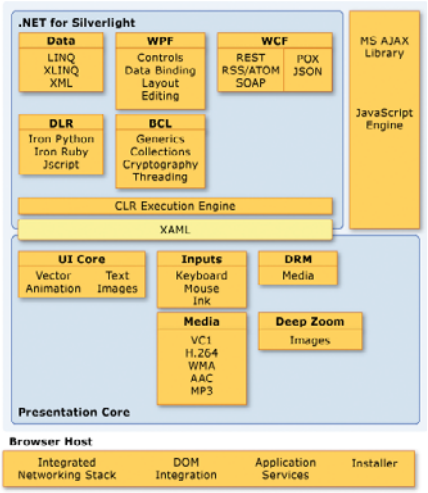
Od grafiky, cez novinky v animácii sme sa prepracovali až k podpore pre **video a audio vo vysokej kvalite**. Vo verzii Silverlight 3 je k dispozícii podpora živého aj on-demand **streamovania v úplnom HD (720+) rozlíšení** kombinované s technológiou **Smooth Streaming**. To prispieva k plynulosti prehrávania obrazu bez rušivých „trhavých" úsekov. Okrem kodeku VC-1/WMA je k dispozícii aj podpora pre MPEG4 vo formáte H.264 s ACC pre kvalitné audio. Pri vykresľovaní HD videa sa využíva aj GPU akcelerácia. Silverlight 3.0 podporuje aj využívanie kodekov tretích strán. Pomocou otvorenej Raw AV Pipeline je možné pre dekódovanie využiť externý komponent a obsah sa následne vykreslí v Silverlight okne.

Pre zrýchlenie komunikácie sa využíva binárne XML, ktoré obsahuje údaje v komprimovanej podobe.

Popis architektúry

Z hľadiska implementácie je technológia Silverlight multiplatformový plug-in do webového prehľadávača. Architektúra platformy Silverlight je rozdelená na dve základné časti: Prezentačná vrstva obsahuje komponenty a služby orientované na generovanie používateľského rozhrania a interakciu s používateľom. Používateľské rozhranie môže využívať renderovanie vektorovej a bitmapovej grafiky, textový výstup, animácie a prezentáciu multimediálneho obsahu v rôznych formátoch. Interakcia s užívateľom zahŕňa obsluhu udalostí generovaných používateľom pomocou myši a klávesnice. Spodnou vrstvou je inštalačný a aktualizačný komponent pre internetový prehľadávač.

Na obrázku architektúry si môžete všimnúť moduly prezentačného jadra **UI Core, Inputs, Media, Deep Zoom** a **DRM**, moduly platformy .NET Framework pre Silverlight Data, WPF (Windows Presentation Foundation) WCF (Window Communication Foundation), DLR, BCL a CLR (Common Language Runtime).



Architektúra platformy Silverlight

Programovacie modely

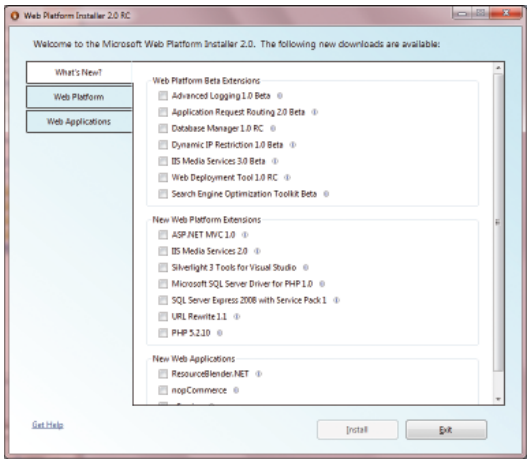
Silverlight aplikácie využívajú riadený (angl. managed) kód, ktorý beží na úrovni vrstvy CLR. Je možné využívať kompilované programovacie jazyky Visual Basic a C# cez Managed API, alebo dynamické jazyky, ako napríklad IronPython a IronRuby. Pre tieto jazyky je k Silverlight Dynamic Languages SDK. Kvôli spätnej kompatibilite so Silverlight 1.0, ktorý nevyužíval .NET jazyky, ale len java Script je k dispozícii JavaScript API for Silverlight. Pri využívaní tohto modelu je kód interpretovaný na úrovni prehľadávača webového obsahu.

Čo budete potrebovať

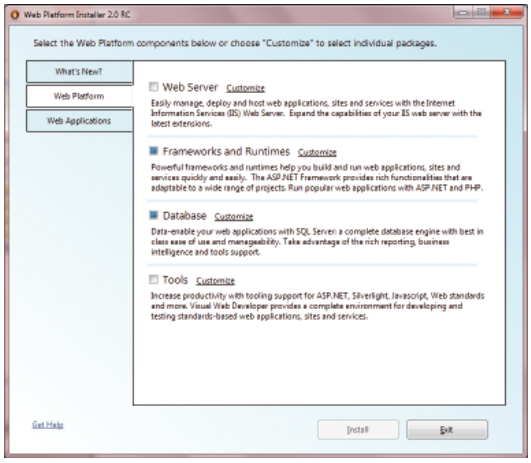
Vstupný bod pre zoznamovanie sa s technológiou Silverlight 3 je **http://www.seethelight.com/**. Pre zaujímavosť, do oficiálneho uvedenia finálnej verzie táto stránka ukazovala počet minút do príchodu Silverlight 3. Všetko potrebné pre vývoj aplikácií na platforme Silverlight nájdete na webe na adrese **http://silverlight.net/GetStarted/.**

Podľa novej stratégie spoločnosti Microsoft sú nástroje pre vývoj webových aplikácií sústredené v komplexnom balíku **Microsoft Web Platform**. Môžete si vybrať, ktoré nástroje si nainštalujete, prípadne nainštalovať **Visual Web Developer 2008 Express Edition** ako separátny produkt a doplniť ho o rozširujúci balík **Silverlight 3 Tools for Visual Studio**.

Úvodná stránka produktu Microsoft Web Platform



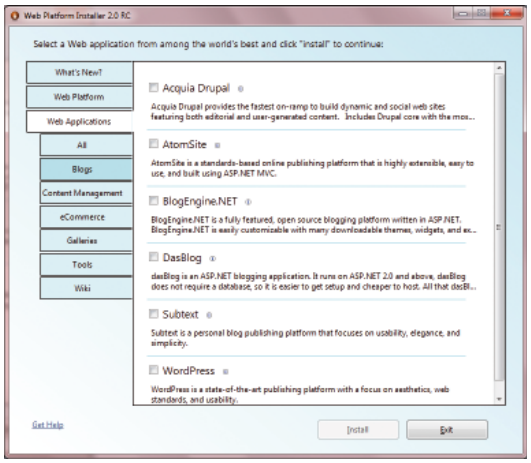
Microsoft Web Platform – prehľad noviniek



Microsoft Web Platform – záložka Webb Platform

Visual Web Developer Express Edition a Silverlight 3 Tools nainštalujete v položke Tools.

V záložke **Web Application** sú prototypy a hotové riešenia najpoužívanejších typov webových aplikácií, ako sú napríklad blogy, komponenty sociálnych sietí, portály a podobne.



Microsoft Web Platform – záložka Web Application

Pre vizuálny návrh prezentačného rozhrania, teda XAML stránok je ideálnym nástrojom **Microsoft Expression Blend 3**. V dobe písania tejto publikácie bol k dispozícii vo verzii **Microsoft Expression Blend 3 + SketchFlow RC**. Pre prípravu obrázkov v rôznych zobrazeniach prostredníctvom technológie Deep Zoom je potrebné nainštalovať nástroj **Deep Zoom Composer**.

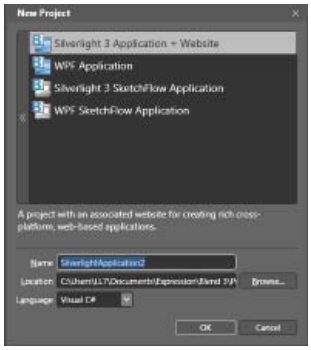
Microsoft Expression Blend 3

Microsoft Expression Blend 3 je flexibilné a produktívne grafické vývojové prostredie. Pomáha pri tvorbe moderných a vizuálne prepracovaných aplikácií s interaktívnou podporou 3D zobrazovania a prehrávania multimédií. Umožňuje vytvorenie a úpravy prezentačnej vrstvy webových aplikácií. Využíva nový druh značkovacieho jazyka XAML.

Poznámka: *Táto verzia umožňuje otvárať aj projekty pre Silverlight 2.0, ktoré automaticky inovuje na verziu 3.0. Spätný krok nie je možný, preto je potrebné otvárať Silverlight 2.0 projekty v staršej verzii.*

Microsoft Expression Blend 3 umožňuje vytvorenie a úpravy prezentačnej vrstvy webových aplikácií aj Windows WPF aplikácií s plnou podporou Silverlight 3.0. Dialóg pre vytvorenie nového projektu obsahuje šablóny pre:

- Silverlight 3 Application + Website,
- WPF Application,
- Silverlight 3 SketchFlow Application,
- WPF SketchFlow Application.

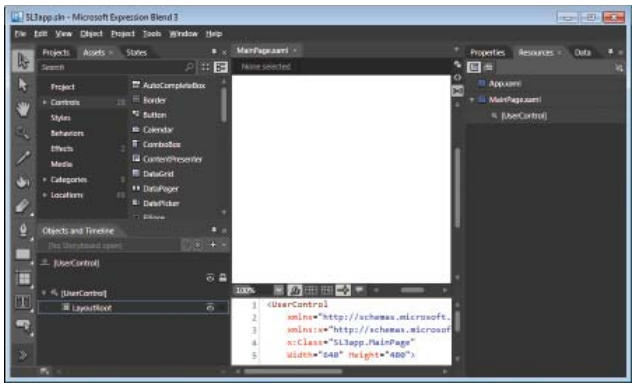


Dialóg pre vytvorenie nového projektu v prostredí Microsoft Expression Blend 3

Silverlight 3 podporuje programovanie aplikačnej logiky v .NET jazykoch. Implicitne sú v prostredí Expression Blend 3 k dispozícii programovacie jazyky:

- Visual C#,
- Visual Basic.

Azda najrýchlejšie bude zoznámiť sa s vývojovým prostredím Microsoft Expression Blend 3 „za behu" na praktickom príklade. Vytvorte novú aplikáciu typu Silverlight 3 a vhodne ju pomenujte. Aplikáciu umiestnite do príslušného priečinka v ktorom plánujete vytvárať projekty tohto typu.



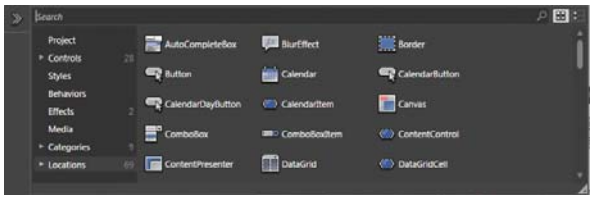
Pracovná plocha v prostredí Expression Blend 3 pre vývoj Silverlight 3.0 aplikácie

V strede pracovnej plochy návrhového prostredia je situovaná oblasť „Art Board". Umožňuje zobraziť grafické návrhové zobrazenie, prípadne návrh reprezentovaný v XAML jazyku, alebo v režime „Split" obidva režimy. Režim zobrazenia sa prepína pomocou ikoniek v pravej hornej časti stredného okna. V ľavej časti je v záložke „Projects" zobrazený zoznam súborov z ktorých pozostáva projekt. Grafický návrh prezentačnej vrstvy aplikácie, ktorá obsahuje napríklad definície vizuálnych prvkov, pozadia a podobne je uložený v súbore MainControl.xaml. V novom projekte obsahuje tento súbor definíciu prázdnej pracovnej plochy kde je prvok <Grid>:

```
<UserControl
       xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
       xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
       x:Class="SL3app.MainPage"
       Width="640" Height="480">

       <Grid x:Name="LayoutRoot" Background="White"/>
</UserControl>
```

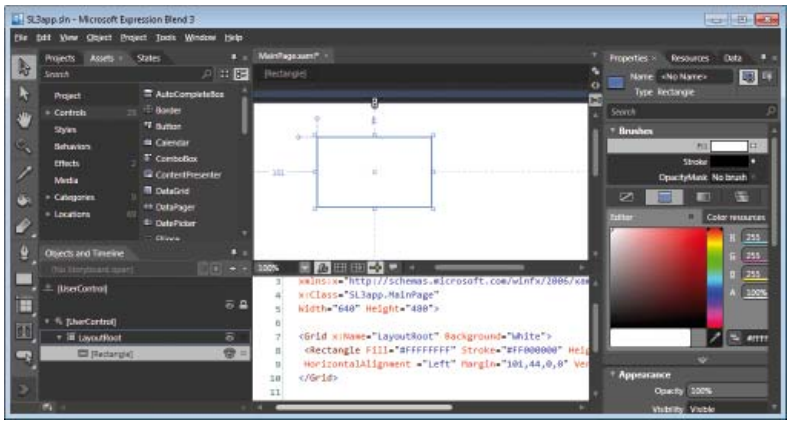
Návrh prezentačnej vrstvy

Úplne vľavo pri ľavom okraji okna aplikácie Expression Blend 3 je úzka lišta s ikonami nástrojov pre vizuálny návrh prezentačnej vrstvy. Pomocou nich môžete na pracovnej ploche vytvárať prvky vektorovej grafiky a ovládacie prvky. Posledná ikona v tvare dvojitej šípky sprístupňuje knižnicu prvkov Asset Library.



Knižnica prvkov

Možnosti pri návrhu budú ukázané na najjednoduchšom dvojrozmernom prvku **Rectangle**, teda obdĺžniku. V prvom kroku metódou „drag and drop" umiestnite obdĺžnik na požadované miesto pracovnej plochy a upravte jeho rozmery.



Vizuálny návrh grafického prvku (obdĺžnika)

V XAML kóde sa pridanie obdĺžnika prejaví takto:

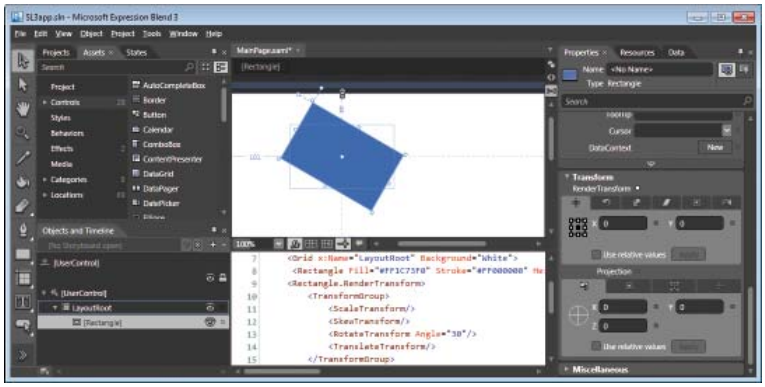
```
<Grid x:Name="LayoutRoot" Background="White">
  <Rectangle Fill="#FFFFFFFF" Stroke="#FF000000" Height="91"
  HorizontalAlignment="Left" Margin="101,44,0,0" VerticalAlignment="Top" Width="149"/>
</Grid>
```

Všimnite si záložku **Properties** v pravom bočnom okne. V nej môžete nastavovať parametre prvku, ktorý je vybraný a vyznačený v grafickom návrhovom okne. K dispozícii je široká paleta možností, napríklad pre návrh farebného dizajnu. Ak napríklad chcete zobraziť obdĺžnik vyplnený farebným gradientom, vytvoríte ho pomocou príslušných návrhových prvkov v tejto záložke. V príklade sme zmenili farbu výplne na svetlo modrú:

```
<Rectangle Fill="#FF1C73F0" Stroke="#FF000000" Height="91"
   HorizontalAlignment="Left" Margin="101,44,0,0"
  VerticalAlignment="Top" Width="149" OpacityMask="#FF000000"/>
```

Transformácia

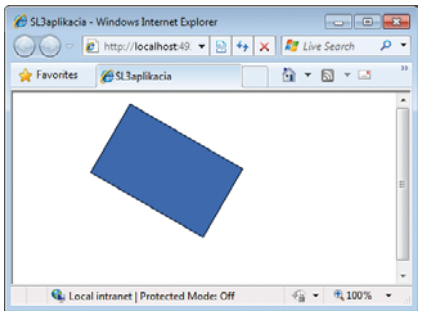
Pri podrobnejšom skúmaní záložky **Properties** zistíte, že je rozdelená na niekoľko podzložiek. Predchádzajúce zmeny farebného návrhu sa vykonávali v záložke **Brushes**. Ak je potrebná geometrická transformácia objektu, napríklad pootočenie, alebo skosenie využijete záložku Transform. Objekt môžete pootočiť aj priamo uchopením za vrchol a následným pootočením.



Geometrická transformácia grafického prvku

```
<Rectangle Fill="#FF1C73F0" Stroke="#FF000000" Height="91" HorizontalAlignment="Left"
Margin="101,44,0,0" VerticalAlignment="Top" Width="149" OpacityMask="#FF000000"
RenderTransformOrigin="0.5,0.5">
	<Rectangle.RenderTransform>
		<TransformGroup>
			<ScaleTransform/>
			<SkewTransform/>
			<RotateTransform Angle="30"/>
			<TranslateTransform/>
		</TransformGroup>
	</Rectangle.RenderTransform>
</Rectangle>
```

Projekt môžete zostaviť a spustiť pomocou menu **Project- Run Project**.



Spustenie projektu

Všimnite si v kontextovom menu, že projekt môžete otvoriť aj vo vývojovom prostredí Visual Studio 2008, samozrejme len v prípade ak je toto prostredie nainštalované a doplnené o Silverlight 3 a pokračovať vo vývoji v tomto prostredí. V nasledujúcej kapitole je popísané vytvorenie projektu v prostredí Visual Studio 2008. Ak ste robili nejaký cvičný projekt v prostredí Expression Blend 3, môžete ho pomocou položky v kontextovom menu otvoriť v prostredí Visual Studio a zoznámiť sa s možnosťami vývoja aplikačného kódu v tomto prostredí.

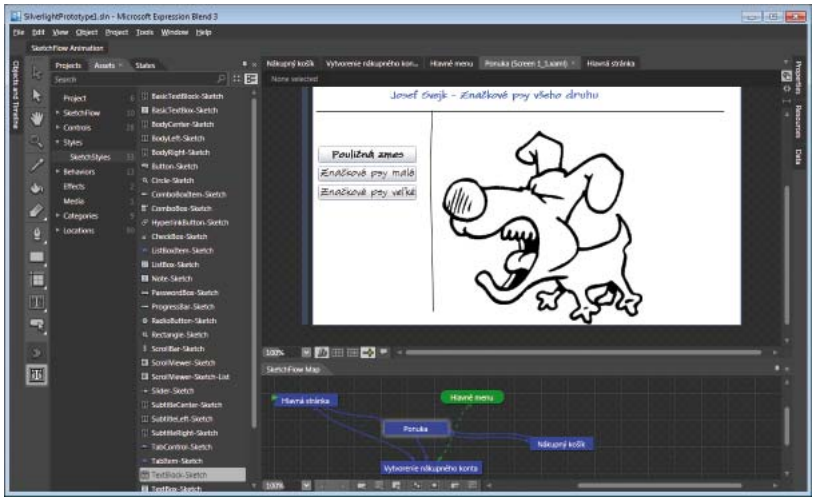
Sketch Flow

Aktuálne dostupná verzia Expression Blend 3 je doplnená aj o funkcionalitu Sketch Flow. Umožňuje vytvárať a modelovať prototypy stránok a postupnosť navigácie po jednotlivých stránkach aplikácie. Ide o modelovací nástroj na pretavenie vízie o dizajne a fungovaní aplikácie do konkrétneho projektu.

Vytvorte si cvičný projekt typu Silverlight 3 SketchFlow Application a vymyslite si nejaký námet. V spodnej časti sa v rámci konceptuálneho a logického modelovania vytvára diagram stránok a navigačných prvkov, ktoré budú tvoriť aplikáciu a prepojenia medzi nimi. Napríklad začnete hlavnou stránkou alebo prihlasovacou stránkou a následne v grafickej forme rozvíjate štruktúru aplikácie. Všimnite si ikonky pod okienkami diagramu, z nich môžete metódou „drag and drop" vytvárať nadväzné stránky. Zároveň s vytvorením prvku diagramu predstavujúceho stránku sa vytvorí aj stránka samotná.

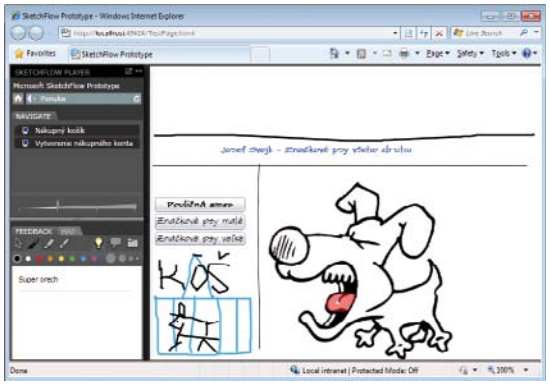
V hornej časti je možné najskôr naskicovať dizajn a následne ho dotvárať reálnymi grafickými prvkami. Pre náčrty sú k dispozícii špeciálne graficky stvárnené prvky, ktoré sú v priečinku SketchStyles (viď obrázok). Dizajn týchto prvkov simuluje náčrt voľnou rukou.

Dizajn celého projektu je možné vyexportovať do textového editora Word. Všetky stránky budú prehľadne uložené v jednom dokumente. Možno si pri neskoršej údržbe starších projektov lámete hlavu nad tým, ako to bolo urobené a ľutujete, že ste si nerobili aspoň poznámky, ak už nie dokumentáciu. Sketch Flow vám nielen pomôže pri dizajne a návrhu, ale za vás vytvorí aj dokumentáciu.



Projekt typu Sketch Flow

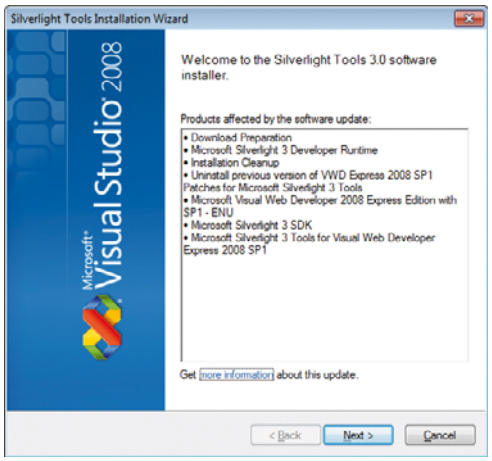
Po spustení aplikácie môžu členovia návrhárskeho tímu návrh pripomienkovať – ako inak, graficky. Pre možnosť dokresľovania feedbacku je pred spustením aplikácie potrebné na úrovni modulu Silverlight Plugin povoliť parameter **Enable Application Storage**. Konfigurácia sa nastavuje pomocou programu Silverlight.Configuration.exe, ktorý je pri implicitnej inštalácii umiestnený v priečinku c:\Program Files\Microsoft Silverlight\..cislo verzie.. Feedback je možné uložiť.



Spustenie a test projektu typu Sketch Flow

Vývoj Silverlight 3.0 projektov v prostredí Visual Studio 2008 (alebo Visual Web Developer 2008 Express Edition)

Pre vývoj Silverlight 3 aplikácií v prostredí Visual Studio 2008, prípadne Visual Web Developer 2008 Express Edition je potrebné ich najskôr (ak už nemáte) doplniť o balíček SP1 a následne nainštalovať doplnok Silverlight Tools 3.0 pre Visual Studio 2008.

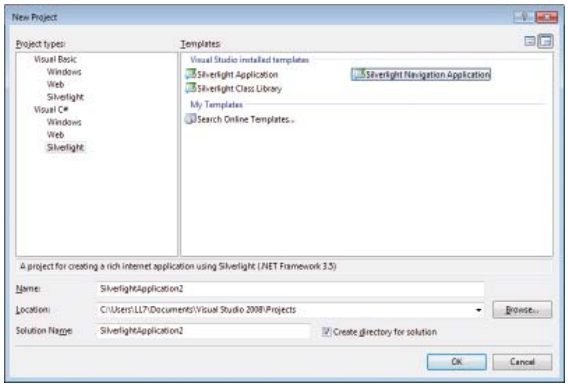


Inštalácia Silverlight Tools 3.0 pre Visual Studio 2008

Po nainštalovaní doplnku Silverlight Tools 3.0 pre Microsoft Visual Studio 2008 sú k dispozícii tri šablóny pre vývoj Silverlight aplikácií:

- Silverlight Application,
- Silverlight Class Library,
- Silverlight Navigation Application.

Šablóna projektu Silverlight Navigation Application sa od Silverlight Application líši tým, že projekt obsahuje aj hlavnú stránku aplikácie s navigačnými prvkami, ktorá umožňuje jednoduché a intuitívne prepínanie podstránok.

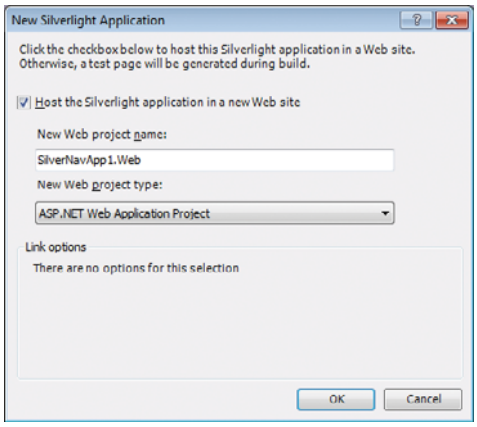


Ponúkané typy Silverlight projektov

Úvodná Silverlight 3.0 aplikácia v prostredí Visual Studio 2008

Vo väčšine publikácií venovaných začiatočníkom je aplikácia typu „Hello World", ktorá po stlačení nejakého aktívneho prvku vypíše na obrazovku nejaký text, najčastejšie pozdrav. Aby sa začiatočníci pri čítaní tohto zborníku necítili ukrátení, bude aj tu jednoduchý príklad tohto typu aplikácie, no aby boli demonštrované vizuálne možnosti a interaktivita technológie Silverlight 3.0 bude projekt po stlačení tlačidla zobrazovať zadané meno a dátum vybraný pomocou grafického interaktívneho prvku typu kalendár.

Vytvorte nový projekt typu **Silverlight Application**, napríklad s názvom HelloSL3. Po výbere typu aplikácie nasleduje otázka ohľadne spôsobu spúšťania Silverlight aplikácie. Nakoľko ide o webovú aplikáciu odporúčame ponechať implicitnú možnosť pridať hosťovaciu stránku do projektu ASP.NET aplikácie.

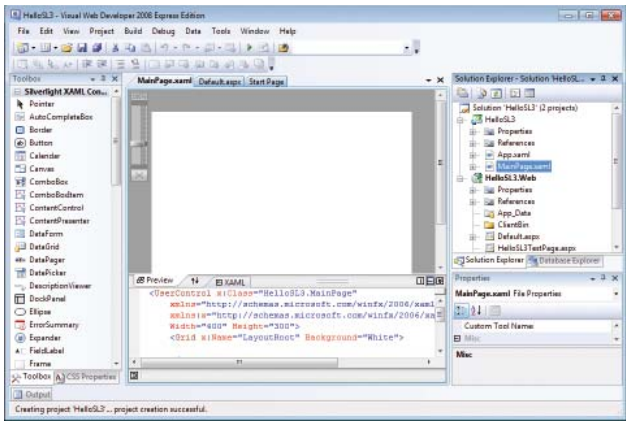


Výber spôsobu hosťovania Silverlight aplikácie

Po voľbe základných parametrov, teda názvu projektu, jeho umiestnenia a programovacieho jazyka, modul vývojového prostredia nazývaný „Sprievodca vytvorením aplikácie" vytvorí prázdnu šablónu aplikácie, čiže ľudovo povedané aplikáciu, ktorá zatiaľ nič nerobí a nič nezobrazuje. V prípade šablóny **Silverlight Navigation Application** sa vytvorí aj hlavná stránka aplikácie s navigačnými prvkami.

Zoznámenie sa s návrhovým prostredím

Po vytvorení projektu je pracovná plocha rozdelená na oblasť pre návrh v grafickom prostredí v hornej časti a okno pre zobrazenie XAML kódu v spodnej časti. Vpravo sú okná **Solution Explorer a Properties**. Vľavo môžete zobraziť okno **Toolbox** s ponukou prvkov pre grafický návrh.



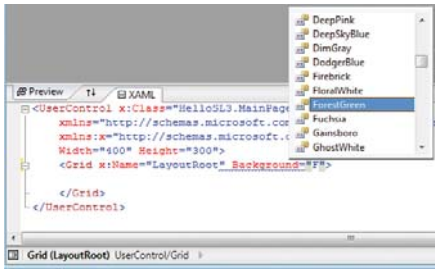
Prostredie pre vývoj Silverlight aplikácie so zobrazeným oknom Toolbox (Visual Web Developer 2008 Express Edition)

UPOZORNENIE: V aktuálnej verzii Silverlight Tools 3.0 pre Visual Studio 2008 nie je k dispozícii grafický náhľad XAML stránok. Táto funkcionalita bude naplno implementovaná v ďalšej verzii Visual Studio 2010, možno na určitej úrovni funkcionality aj v ďalšej verzii Silverlight Tools 3.0 pre Visual Studio 2008.

XAML kód v súbore **MainPage.xaml** obsahuje definíciu pracovnej plochy:

```
<UserControl x:Class="HelloSL3.MainPage"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    Width="400" Height="300">
    <Grid x:Name="LayoutRoot" Background="White">

    </Grid>
</UserControl>
```

V stránke je element **Grid**, ktorý umožňuje rozmiestňovanie prvkov na relatívne alebo v prípade potreby aj absolútne pozície a má podobné vlastnosti ako tabuľka na HTML stránke. Ak by ste sa v tejto verzii pokúsili pridať vizuálny prvok z okna Toolbox priamo na návrhovú plochu, neuspeli by ste. Prvky sa pridávajú priamo do XAML kódu. Pri zadávaní vlastností prvku je výhodné využiť črtu **Intellisense**, ktorá vám formou interaktívneho pomocníka ponúka názvy parametrov a kostru syntaxe pre ich zápis. Tento pomocník funguje nielen pre XAML a kľúčové slová a objekty programovacích jazykov, ale aj pre HTML dokumenty, ASP.NET tagy a podobne. Ako prvý pokus môžete zmeniť farbu pozadia.



Využitie Intellisense pre pridávanie a nastavovanie parametrov

Aby sa vám s prvkom Grid pri rozmiestňovaní vizuálnych prvkov lepšie pracovalo môžete si ho vhodne rozdeliť na riadky a stĺpce a pomocou nastavenia parametra ShowGridLines="True" nastaviť zobrazovanie mriežky:

```
<Grid x:Name="LayoutRoot" Background="FloralWhite" ShowGridLines="True">
        <Grid.RowDefinitions>
            <RowDefinition Height="40"/>
            <RowDefinition Height="220"/>
            <RowDefinition Height="40"/>
        </Grid.RowDefinitions>
        <Grid.ColumnDefinitions>
            <ColumnDefinition Width="75" />
            <ColumnDefinition Width="325"/>
        </Grid.ColumnDefinitions>

    </Grid>
```

Prvé spustenie aplikácie

Po ukončení prvej etapy návrhu prezentačnej a aplikačnej vrstvy, v tomto príklade je vlastne len zobrazená mriežka tabuľky, nastala príležitosť pre prvé spustenie aplikácie. Pre tento účel slúži buď tlačidlo nástrojovej lišty v tvare zelenej šípky, alebo položka menu **Debug | Start**, prípadne klávesová skratka **F5**. O preklade a zostavení aplikácie získate prehľadný výpis v okne „Output".

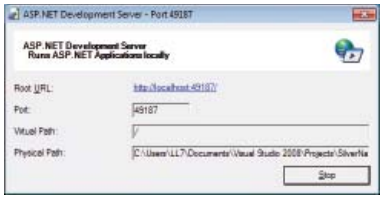
XAML kód bol preložený do „assembly" súboru. Tento súbor má príponu „XAP" a nachádza sa v podpriečinku projektu...\ Bin.

Poznámka: *Ak si „binárny" XAP súbor pozriete binárnym editorom alebo prehľadávačom, zistíte, že prvé dva znaky „PK" avizujú, že ide o ZIP archív. Ak si teda súbor s príponou XAP skopírujete do pomocného priečinka a premenujete jeho príponu na „ZIP", môžeme z tohto archívu vybaliť dva súbory: AppManifest.xaml a SilverApp.dll.*

Pri prvom spustení aplikácie vás vývojové prostredie upozorní, že v konfiguračnom súbore pre webové aplikácie nie je povolené ladenie. Ak chcete ladenie povoliť, čo samozrejme odporúčame, vytvorí vývojové prostredie lokálny súbor web.config, platný pre konkrétnu aplikáciu, v ktorom bude ladenie povolené.

Dialóg pre povolenie ladenia hosťujúcej ASP.NET aplikácie

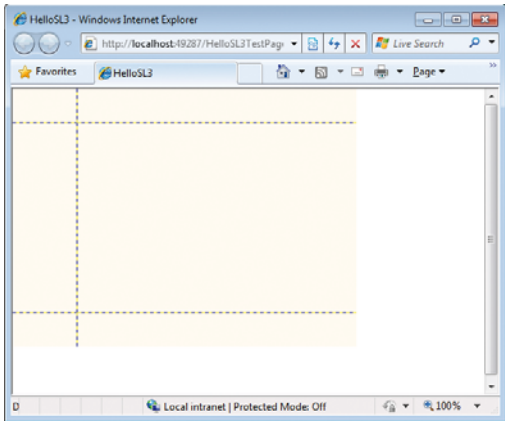
Po spustení aplikácie bude aktivovaný samostatný webový listener, ktorý je integrálnou súčasťou vývojového prostredia. Jeho činnosť je indikovaná ikonou v pravom dolnom rohu obrazovky operačného systému (v blízkosti ikony času a ikony prepínania lokálnej klávesnice). Aktivovaním tejto ikony môžete kedykoľvek zistiť podrobnosti o parametroch tohto lokálneho webového servera. V informačnom dialógu sa dozvieme aký port, fyzický a virtuálny priečinok tento server používa a na akej URL adrese je prístupná konkrétna aplikácia.



Listener pre spustenie hosťujúcej ASP.NET aplikácie

Následne po spustení lokálneho webového listenera bude v jeho réžii spustená aplikácia samotná. URL adresu aplikácie je možné vložiť do iných prehľadávačov (Firefox, Chrome...).

Ak máte implicitne nastavený prehľadávač, budete v závislosti od jeho verzie požiadaní, aby ste prípadne povolili spúšťanie a zobrazovanie intranetových aplikácií.



Spustenie aplikácie

Návrh formulára pre interakciu s používateľom

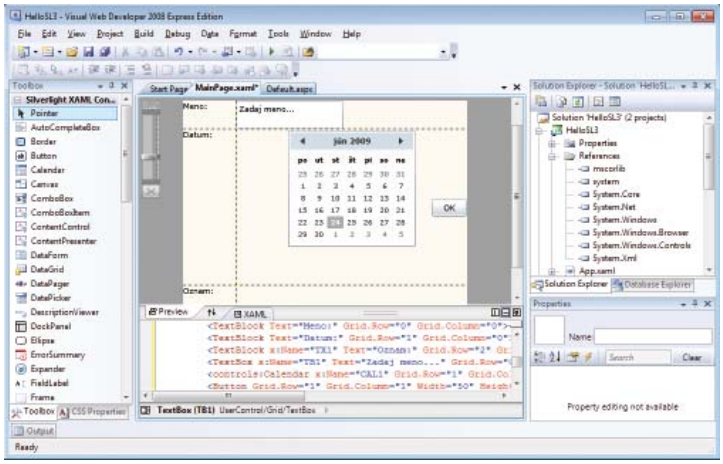
Formulár bude obsahovať textové nápisy v prvkoch **TextBlock**, pole typu **TextBox** pre vstup textu a grafický prvok **Calendar**. Pre každý prvok je potrebné definovať jeho umiestnenie v rámci mriežky Grid, ktorá bude zabezpečovať relatívnu polohu prvkov voči sebe aj pri prípadnej zmene veľkosti okna. Postup návrhu je zrejmý z obrázku a výpisu
XAML kódu.

Ešte raz zdôrazňujeme, že v prostredí Visual Studio 2008 prvky nepridávate na návrhovú plochu, ale na vhodné miesto XAML kódu do vnútra tagu <Grid>.

Aby bolo možné s obsahom prvkov pracovať v kóde, je potrebné prvky identifikovať podľa názvu, preto je ich potrebné v XAML kóde pomenovať. Pomenovať je potrebné prvky pre zadanie mena, kalendárový prvok a prvok pre vypísanie oznamu:

```
<Grid x:Name="LayoutRoot" Background="FloralWhite" ShowGridLines="True">
  <Grid.RowDefinitions>
      <RowDefinition Height="40"/>
      <RowDefinition Height="220"/>
```

```
      <RowDefinition Height="40"/>
  </Grid.RowDefinitions>
  <Grid.ColumnDefinitions>
      <ColumnDefinition Width="75" />
      <ColumnDefinition Width="325"/>
  </Grid.ColumnDefinitions>
  <TextBlock Text="Meno:" Grid.Row="0" Grid.Column="0"></TextBlock>
  <TextBlock Text="Datum:" Grid.Row="1" Grid.Column="0"></TextBlock>
  <TextBlock x:Name="TX1" Text="Oznam:" Grid.Row="2" Grid.Column="0"
Grid.ColumnSpan="2"></TextBlock>
        <TextBox x:Name="TB1" Text="Zadaj meno..." Grid.Row="0" Grid.Column="1"
Width="150" HorizontalAlignment="Left"></TextBox>
        <controls:Calendar x:Name="CAL1"Grid.Row="1"
Grid.Column="1"></controls:Calendar>
        <Button Grid.Row="1" Grid.Column="1" Width="50" Height="25"
HorizontalAlignment="Right" Content="OK"></Button>
</Grid>
```

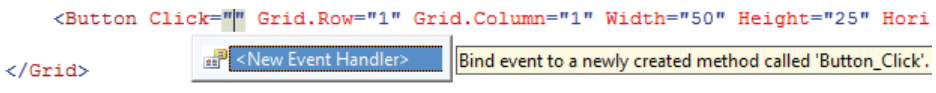


Návrh formulára aplikácie

Kód pre obsluhu udalostí

Každú aplikáciu tvorí aplikačná a prezentačná vrstva. Úlohou aplikačnej vrstvy je pracovať s údajmi a premennými a pripraviť údaje, ktoré sa zobrazia používateľovi. Úlohou prezentačnej vrstvy je výpis týchto údajov vo vhodnej forme. Keby ste aplikáciu v tejto fáze spustili, zobrazil by sa formulár pre zadávanie a zobrazovanie údajov, no po zatlačení tlačidla by sa nič nestalo. Aplikačný kód vlastne vdýchne aplikácii život.

Pre tlačidlo pridajte parameter Click. Po pridaní tohto parametra sa pridá do XAML kódu text Click =“”. Medzi úvodzovky bude automaticky umiestnená položka kontextového menu pre vytvorenie obslužnej procedúry udalosti kliknutia na tlačidlo.



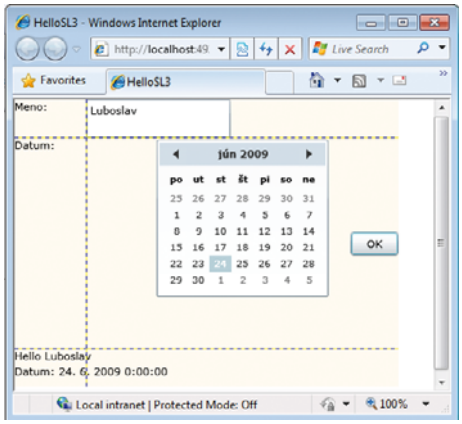
Pridanie odkazu na obslužný kód udalosti prvku

Kliknutím na túto položku bude vytvorené telo obslužnej procedúry, čiže jej prázdny kód. V súbore **MainPage.xaml.cs** bude kostra kódu procedúry pre obsluhu udalosti kliknutia na tlačidlo:

```
namespace HelloSL3
{
    public partial class MainPage : UserControl
    {
        public MainPage()
        {
            InitializeComponent();
        }

        private void Button_Click(object sender, RoutedEventArgs e)
        {

        }
    }
}
```

Kód po stlačení tlačidla vypíše text zadaný v poli meno a dátum vybraný cez kalendárový prvok. V kóde je ošetrená aj situácia ak dátum nie je vybraný:

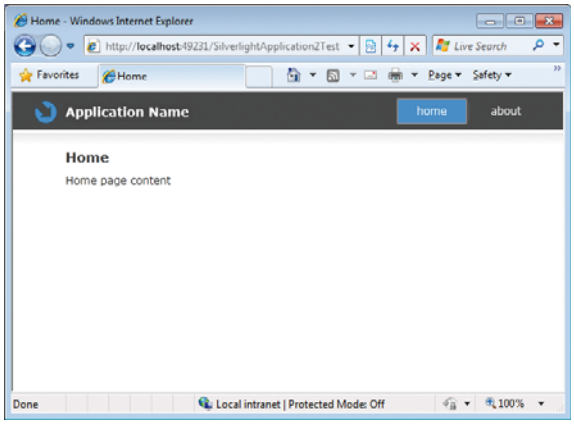
```
private void Button_Click(object sender, RoutedEventArgs e)
{
    string dateString;
    if (CAL1.SelectedDate == null)
    {
        dateString = "<datum nie je vybraty...>";
    }
    else
    {
        dateString = CAL1.SelectedDate.ToString();
    }
    TX1.Text = "Hello " + TB1.Text + "\n" + "Datum: " + dateString;

}
```



Spustenie aplikácie

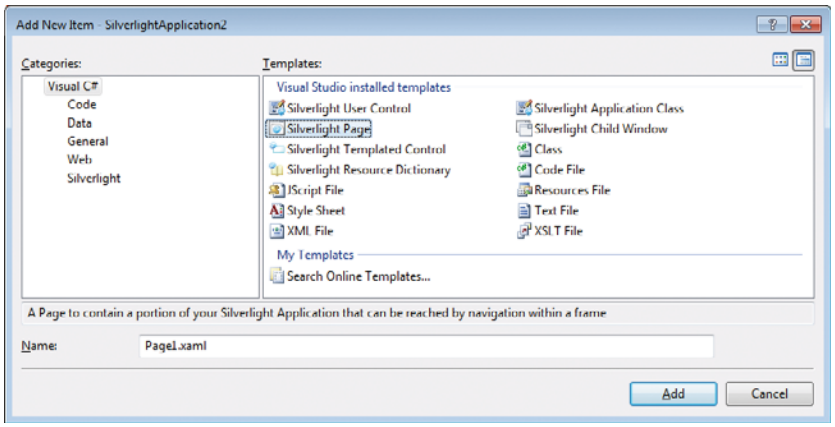
Projekt typu Silverlight Navigation Application

Projekt typu Silverlight Navigation Application obsahuje navigačnú stránku, pomocou ktorej je možné sa prepínať na ďalšie stránky projektu.



Silverlight aplikácia s navigačnou stránkou (Visual Web Developer 2008 Express Edition)

Novú stránku pridáte do projektu cez položku **Add New Item**, aplikovanú na priečinok **Views** a v rovnomennom dialógu vyberiete položku **Silverlight Page**.

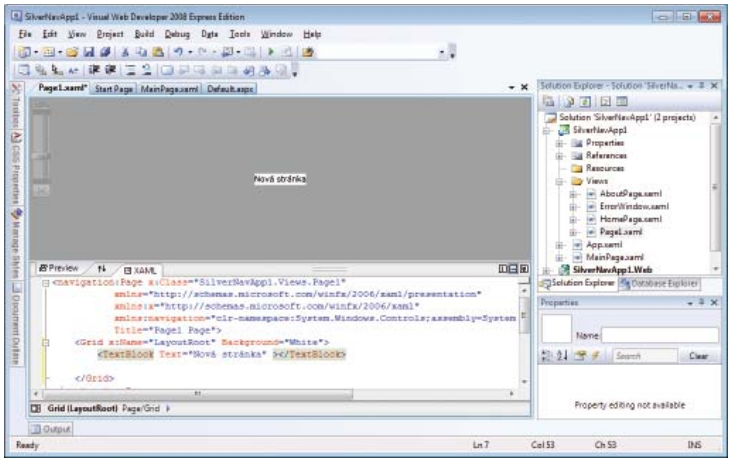


Možnosti pridania novej entity do Silverlight aplikácie

Do kódu stránky pridajte nejaký grafický prvok, alebo text, aby ste ju po spustení aplikácie vedeli identifikovať:

```
<navigation:Page x:Class="SilverlightApplication2.Page1"
           xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
           xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
           xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
           xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
           mc:Ignorable="d"
           xmlns:navigation="clr-
namespace:System.Windows.Controls;assembly=System.Windows.Controls.Navigation"
           d:DesignWidth="640" d:DesignHeight="480"
           Title="Page1 Page">
```

```
    <Grid x:Name="LayoutRoot">
      <TextBlock Text="Nová stránka" ></TextBlock>
    </Grid>
</navigation:Page>
```

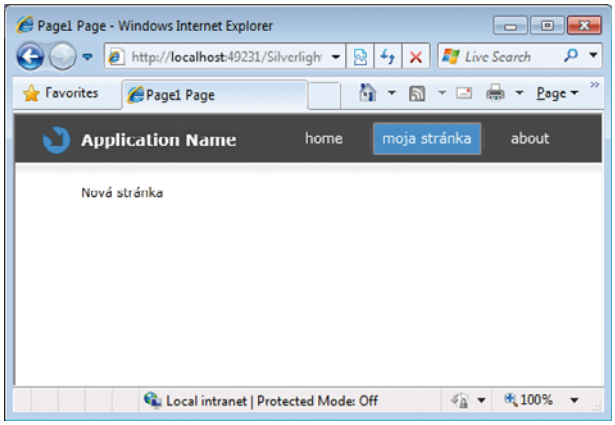


Nová XAML stránka

Aby sa stránka sprístupnila, je potrebné pridať prepojenie na hlavnú stránku Silverlight aplikácie MainPage. XAML. V našom prípade bol pridaný objekt HyperlinkButton a oddeľovací obdĺžnik:

```
<Grid x:Name="LayoutRoot" Style="{StaticResource LayoutRootGridStyle}">
   <Border x:Name="ContentBorder" Style="{StaticResource ContentBorderStyle}">
   <navigation:Frame x:Name="ContentFrame" Style="{StaticResource ContentFrameStyle}"
      Source="/Home" Navigated="ContentFrame_Navigated"
      NavigationFailed="ContentFrame_NavigationFailed">
                <navigation:Frame.UriMapper>
                  <uriMapper:UriMapper>
                    <uriMapper:UriMapping Uri="" MappedUri="/Views/Home.xaml"/>
                    <uriMapper:UriMapping Uri="/{pageName}"
                     MappedUri="/Views/{pageName}.xaml"/>
                  </uriMapper:UriMapper>
                </navigation:Frame.UriMapper>
            </navigation:Frame>
        </Border>

  <Grid x:Name="NavigationGrid" Style="{StaticResource NavigationGridStyle}">
    <Border x:Name="BrandingBorder" Style="{StaticResource BrandingBorderStyle}">
      <StackPanel x:Name="BrandingStackPanel" Style="{StaticResource
      BrandingStackPanelStyle}">
         <ContentControl Style="{StaticResource LogoIcon}"/>
         <TextBlock x:Name="ApplicationNameTextBlock"
            Style="{StaticResource ApplicationNameStyle}"
            Text="Application Name"/>
      </StackPanel>
      </Border>
     <Border x:Name="LinksBorder" Style="{StaticResource LinksBorderStyle}">
       <StackPanel x:Name="LinksStackPanel"
        Style="{StaticResource LinksStackPanelStyle}">
```

```
      <HyperlinkButton x:Name="Link1" Style="{StaticResource LinkStyle}"
         NavigateUri="/Home" TargetName="ContentFrame" Content="home"/>

     <Rectangle x:Name="Divider1" Style="{StaticResource DividerStyle}"/>
     <HyperlinkButton x:Name="Link3" Style="{StaticResource LinkStyle}"
         NavigateUri="/Page1" TargetName="ContentFrame" Content="moja stránka"/>
     <Rectangle x:Name="Divider2" Style="{StaticResource DividerStyle}"/>
     <HyperlinkButton x:Name="Link2" Style="{StaticResource LinkStyle}"
        NavigateUri="/About" TargetName="ContentFrame" Content="about"/>
   </StackPanel>
 </Border>
</Grid>
</Grid>
```



Nová XAML stránka

Ak chcete využiť túto šablónu projektu , jedným z prvých krokov bude aj zmena názvu aplikácie. Je potrebné zmeniť atribút Text objektu **ApplicationNameTextBlock** z **Application Name** na vlastný názov. Moja super aplikácia.

Komplexný dizajn aplikácie je v súbore **Styles.xaml**. Tento súbor sa nachádza v priečinku **Assets**. Výpis je skrátený. Môžete skúsiť experimentovať so zmenou farieb, štýlu a podobne:

```
<ResourceDictionary
  xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
  xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
  xmlns:navigation="clr-
namespace:System.Windows.Controls;assembly=System.Windows.Controls.Navigation">

  <!-- ******MAIN PAGE STYLES****** -->
  <!-- *************************** -->
  <!-- Primary Color Brushes -->
  <SolidColorBrush x:Key="NavigationBackgroundColorBrush" Color="#FF484848"/>
  <SolidColorBrush x:Key="NavigationForegroundColorBrush" Color="#FFFFFFFF"/>
  <SolidColorBrush x:Key="HighLightColorBrush" Color="#FF0097FC"/>
  <SolidColorBrush x:Key="HoverHyperlinkForegroundColorBrush" Color="#FFEBF7FF"/>
  <SolidColorBrush x:Key="HoverHyperLinkBackgroundColorBrush" Color="#FF747474"/>
  <SolidColorBrush x:Key="BodyTextColorBrush" Color="#FF313131"/>
```

```
  <!-- LayoutRoot Grid Style -->
  <Style x:Key="LayoutRootGridStyle" TargetType="Grid">
    <Setter Property="Background" Value="#FFFFFFFF"/>
  </Style>

  <!-- Content Border Style -->
  <Style x:Key="ContentBorderStyle" TargetType="Border">
    <Setter Property="Background">
      <Setter.Value>
        <LinearGradientBrush EndPoint="0.5,0.045" StartPoint="0.5,0">
          <GradientStop Color="#6FCCCCCC"/>
          <GradientStop Color="#00CCCCCC" Offset="1"/>
        </LinearGradientBrush>
      </Setter.Value>
    </Setter>
    <Setter Property="BorderBrush" Value="#FFFFFFFF"/>
    <Setter Property="BorderThickness" Value="0,3,0,0"/>
    <Setter Property="Margin" Value="0,42,0,0"/>
    <Setter Property="VerticalAlignment" Value="Stretch"/>
    <Setter Property="HorizontalAlignment" Value="Stretch"/>
  </Style>

  <!-- Content Frame Style -->
  <Style x:Key="ContentFrameStyle" TargetType="navigation:Frame">
    <Setter Property="Background" Value="Transparent"/>
    <Setter Property="BorderBrush" Value="Transparent"/>
    <Setter Property="Padding" Value="58,15,58,15"/>
    <Setter Property="VerticalContentAlignment" Value="Stretch"/>
    <Setter Property="HorizontalContentAlignment" Value="Stretch"/>
  </Style>
...
```

Vývoj toho istého projektu v prostredí Expression Blend 3 a Visual Studio 2008

Moderná interaktívna webová aplikácia vzniká za intenzívnej spolupráce dizajnérov prezentačnej vrstvy aplikácie a vývojárov aplikačnej logiky. Keď návrhár predloží definitívny návrh ako by mala aplikácia vyzerať, začnú vývojári pracovať na implementácii tohto návrhu.

Vývojár sa pri implementácii návrhov dizajnéra riadi možnosťami prezentačnej vrstvy a návrhového prostredia, takže výsledná podoba niektorých ovládacích prvkov je iná než bola pôvodne navrhnutá. Naproti tomu ak je dizajnérov návrh vytvorený v takom prostredí a formáte, ktorý je podporovaný aj vývojovými prostrediami, je zaručené, že výsledná podoba aplikácie bude presne zodpovedať návrhu dizajnéra. V praxi to funguje tak, že dizajnér odovzdá svoj návrh vývojárskemu tímu v jazyku XAML a vývojársky tím do návrhu naprogramuje aplikačnú logiku.

Všimnite si, že kontextové menu XAML stránok obsahuje aj položku **Open in Expression Blend**...

Zmeny vykonané návrhovom prostredí Expression Blend 3 sú po upozornení následne akceptované aj v prostredí Visual Studio 2008 a naopak.

Objekty pre vytvorenie prezentačnej vrstvy

Po pokusoch s obdĺžnikom v úvodnom príklade nastal čas na podrobnejšie zoznámenie sa s hierarchiou objektov pre tvorbu prezentačnej vrstvy a používateľského rozhrania.

Kontajnery na zapuzdrovanie prvkov

Na zapuzdrovanie prvkov sa využívajú kontajnerové objekty:

- Grid,
- Canvas,
- Stack Panel.

Tieto objekty zapuzdrujú prvky nielen z pohľadu objektovo orientovaného programovania ale aj doslova vizuálne, to znamená, že napríklad udržiavajú ich absolútnu aj relatívnu polohu aj pri zmene rozmerov okna.

Objekt Grid

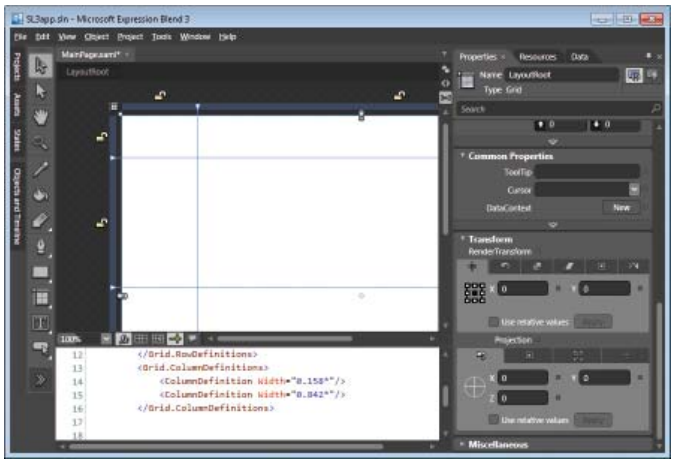
Znovu pripomenieme XAML kód prázdnej stránky vygenerovanej sprievodcom:

```
<UserControl
       xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
       xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
       x:Class="SL3app.MainPage"
       Width="640" Height="480">

       <Grid x:Name="LayoutRoot" Background="White"/>
</UserControl>
```

Už táto kostra stránky obsahuje kontajnerový prvok Grid. Prečo?
-Určite ste si všimli, že väčšina webových aplikácií má pracovnú plochu rozvrhnutú na niekoľko samostatných oblastí. Môže to byť realizované napríklad pomocou HTML tabuľky. Táto tabuľka vlastne udržiava prvky v nej na absolútnych, alebo relatívnych pozíciách.

Ak v prostredí Expression Blend 3 kliknete na malý štvorček v pravom hornom rohu orámovania bielej návrhovej plochy, môžete pomocou posúvania a umiestňovania pravítok rozdeliť pracovnú plochu na niekoľko oblastí a to vodorovne alebo zvisle.

Ukážeme príklad tabuľky, so záhlavím a riadkami, aké sa často používajú na webových stránkach. Všimnite si, že rozmery, ktoré sa budú prispôsobovať napríklad veľkosti stránky sú definované pomocou znaku „*“ (hviezdička).



Interaktívny návrh mriežky tabuľky

V našom prípade bol výsledkom vizuálneho návrhu kód:

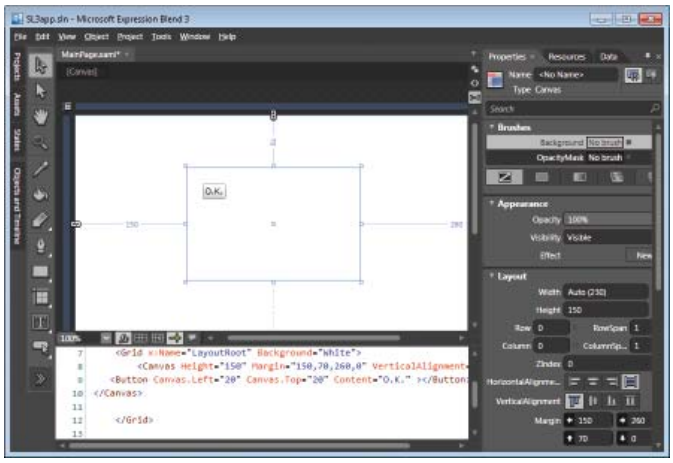
```
<Grid x:Name="LayoutRoot" Background="White">
            <Grid.RowDefinitions>
                  <RowDefinition Height="0.117*"/>
                  <RowDefinition Height="0.36*"/>
                  <RowDefinition Height="0.523*"/>
            </Grid.RowDefinitions>
            <Grid.ColumnDefinitions>
                  <ColumnDefinition Width="0.158*"/>
                  <ColumnDefinition Width="0.842*"/>
            </Grid.ColumnDefinitions>
  </Grid>
```

Prepínaním ikony v ľavom hornom rohu sa dá meniť absolútne a relatívne rozmiestnenie buniek. Pomocou ikon visacích zámkov môžete „zamknúť" fixné nastavenie rozmerov vybranej oblasti, je potrebné nastaviť parameter ShowGridLines="true". Pri absolútnom rozmiestnení prvý riadok napríklad <RowDefinition Height="10"/> alebo prvý stĺpec napríklad <ColumnDefinition Width="10"/> udávajú okraj.

Objekt Canvas

Silverlight aplikácie využívajú objektový model pracovnej plochy s názvom Canvas. Preklad tohto pojmu do slovenčiny je pomerne výstižný, je to virtuálne maliarske, prípadne premietacie plátno, ktoré reprezentuje viditeľnú prezentačnú vrstvu Silverlight aplikácie. Na túto plochu sa potom umiestňujú grafické objekty. Pri grafických objektoch môžete stanoviť ich polohu. Vo fragmente kódu je definícia polohy tlačidla voči ploche Canvas:

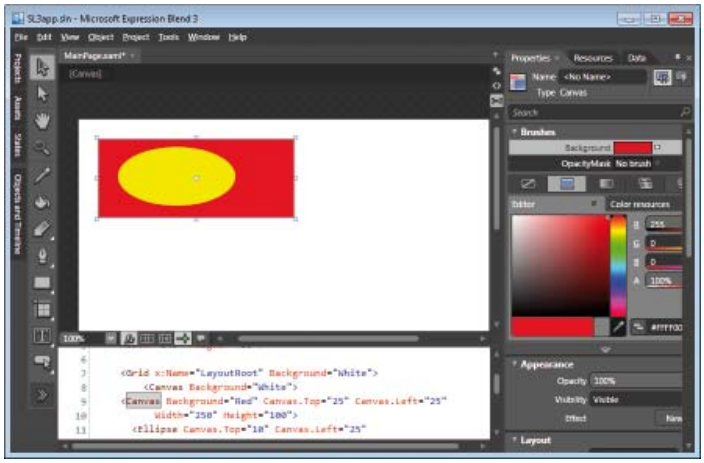
```
<Canvas Height="150" Margin="150,70,260,0" VerticalAlignment="Top">
   <Button Canvas.Left="20" Canvas.Top="20" Content="O.K." ></Button>
</Canvas>
```



Relatívna poloha plochy geometrického obrazca voči ploche Canvas

Všimnite si aj v kóde aj na obrázku, že poloha objektu **Canvas** je vymedzená voči kontejneru Grid, že XAML aplikácia môže obsahovať viac plôch „Canvas", pričom tieto môžu byť rôzne vnorené. V príklade je vo vnútri hlavnej (bielej) plochy vnorená ďalšia, červená vnorená plocha Canvas. Vo vnútri vnorenej plochy je zobrazená geometrická plocha (elipsa), pričom poloha elipsy je zadaná ako relatívna voči vnútornej vnorenej ploche „Canvas":

```
<Canvas Background="White">
       <Canvas Background="Red" Canvas.Top="25" Canvas.Left="25"
           Width="250" Height="100">
       <Ellipse Canvas.Top="10" Canvas.Left="25"
                Width="150" Height="75" Fill="Yellow" />
  </Canvas>
</Canvas>
```

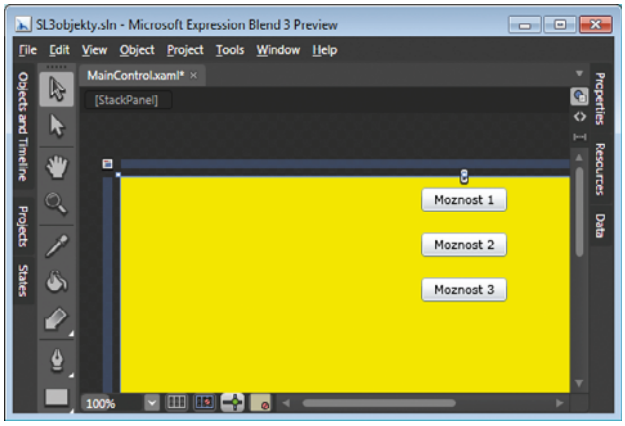


Vnorenie objektov Canvas

Objekt Stack Panel

Objekt Stack Panel slúži na rozmiestnenie objektov nad sebou, alebo vedľa seba. Implicitná orientácia usporiadania je vertikálna. Príklad ukazuje umiestnenie troch tlačidiel:

```
<StackPanel Background="Yellow">
   <Button  Content="Moznost 1" Width="80" Margin="10"/>
   <Button  Content="Moznost 2" Width="80" Margin="10"/>
   <Button  Content="Moznost 3" Width="80" Margin="10"/>
</StackPanel>
```



Usporiadanie objektov pomocou Stack Panelu

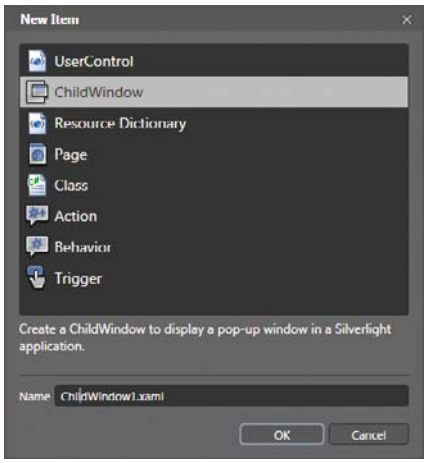
Ak zmeníte orientáciu na horizontálnu, zmení sa usporiadanie objektov zapuzdrených v Stack Paneli

Široká ponuka komponentov

Silverlight 3 obsahuje viac než 60 prispôsobiteľných ovládacích prvkov typu „out-of-the-box ". Náhodne spomenieme nové prvky, napríklad ChildWindow, TreeView a DataGrid.

Dialógové okno

Jedným z prvkov stierania rozdielu medzi klasickými a webovými aplikáciami s interaktívnym používateľským rozhraním je aj ialógové okno, ktoré určite dôverne poznáte z klasických aplikácií.



Pridanie novej entity, v tomto prípade ChildWindow

Do projektu bude pridaná šablóna okna obsahujúca dve tlačidlá:

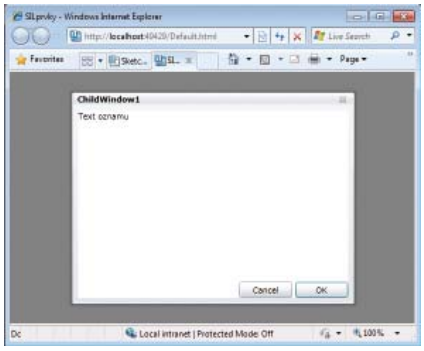
```
<Grid x:Name="LayoutRoot" Background="White">
       <Grid.RowDefinitions>
              <RowDefinition />
              <RowDefinition Height="Auto" />
       </Grid.RowDefinitions>
  <TextBlock Text="Text oznamu"></TextBlock>
<Button x:Name="OKButton" Content="OK" Click="OKButton_Click" Width="75"
      Height="23" HorizontalAlignment="Right" Grid.Row="1" />
<Button x:Name="CancelButton" Content="Cancel" Click="CancelButton_Click"
      Width="75" Height="23" HorizontalAlignment="Right"  Margin="0,0,79,0"
      Grid.Row="1" />
</Grid>
```

Pripravené sú aj šablóny obslužného kódu tlačidiel:

```
private void OKButton_Click(object sender, RoutedEventArgs e)
{
    this.DialogResult = true;
}

private void CancelButton_Click(object sender, RoutedEventArgs e)
{
    this.DialogResult = false;
}
```

Z hlavnej stránky môžete dialógové okno aktivovať napríklad tlačidlom s obslužným kódom:

```
private void Button_Click(object sender, System.Windows.RoutedEventArgs e)
{
       ChildWindow1 dlg = new ChildWindow1();
       dlg.Show();
}
```

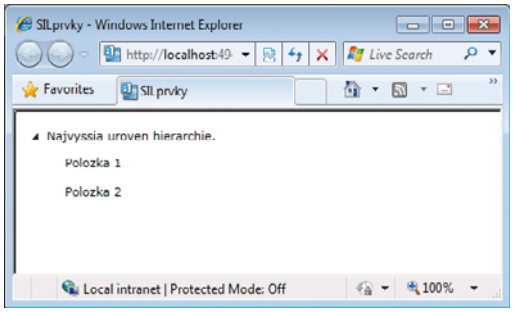


Aplikácia využívajúca ChildWindow

TreeView pre zobrazenie hierarchickej štruktúry

Aplikácie nadväzujú na procesy a organizačné štruktúry z reálneho sveta. Jedný m z aspektov reálneho života – o podnikovom prostredí ani nehovoriac – je **hierarchická štruktúra**. Hierarchicky sú zoradené organizačné zložky, zamestnanci do kategórií a podkategórií sú rozdelené produkty. Jednou z možností pre zobrazenie hierarchických štruktúr v Silverlight aplikáciách je objekt TreeView. Všimnite si, že do záhlavia XAML dokumentu bol pridaný namespace System.Windows.Controls:

```
<UserControl
       xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
       xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
       xmlns:controls="clr-
namespace:System.Windows.Controls;assembly=System.Windows.Controls"
       x:Class="SILprvky.MainPage"
       Width="640" Height="480">

       <Grid x:Name="LayoutRoot" Background="White">
    <controls:TreeView>
    <controls:TreeViewItem Header="Najvyssia uroven hierarchie.">
       <controls:TreeViewItem.Items>
         <controls:TreeViewItem>
             <controls:TreeViewItem.Header>
                <TextBlock Text="Polozka 1" Margin="2"/>
             </controls:TreeViewItem.Header>
          </controls:TreeViewItem>

               <controls:TreeViewItem>
              <controls:TreeViewItem.Header>
                   <TextBlock Text="Polozka 2" Margin="2"/>
              </controls:TreeViewItem.Header>
          </controls:TreeViewItem>
       </controls:TreeViewItem.Items>
     </controls:TreeViewItem>
   </controls:TreeView>
</Grid>
</UserControl>
```



Aplikácia využívajúca TreeView

DataGrid

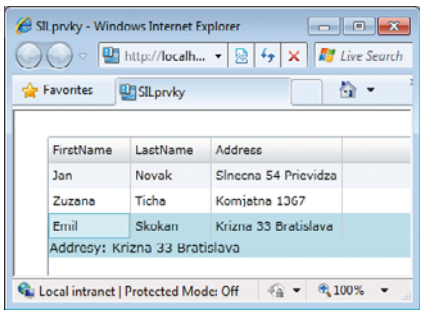
Pre komfortné a interaktívne zobrazenie záznamov, či už z relačných databáz, XML súborov, alebo údajov zapuzdrených v objektoch je určený prvok DataGrid. Tento prvok podporuje aj zobrazovanie hierarchickej štruktúry údajov. V príklade budú údaje poskytnuté triedou Customer, ktorá obsahuje údaje o zákazníkoch.

XAML kód:

```
<UserControl
xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
xmlns:data="clr-namespace:System.Windows.Controls;
      assembly=System.Windows.Controls.Data"
xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
x:Class="SILprvky.MainPage"
Width="640" Height="480" mc:Ignorable="d">

<Grid x:Name="LayoutRoot" Background="White">
<data:DataGrid x:Name="dataGrid1"
       Margin="31,24,45,225"
       RowDetailsVisibilityMode="VisibleWhenSelected"
       d:LayoutOverrides="VerticalAlignment" >
       <data:DataGrid.RowDetailsTemplate>
             <DataTemplate>
                   <StackPanel Orientation="Horizontal">
                         <TextBlock FontSize="12" Text="Addresy: " />
             <TextBlock FontSize="12" Text="{Binding Address}"/>
                   </StackPanel>
             </DataTemplate>
  </data:DataGrid.RowDetailsTemplate>
</data:DataGrid>

</Grid>
</UserControl>
```

Údaje budú poskytnuté triedou Customer:

```
using System.Collections.Generic;

namespace SILprvky
{
       public partial class MainPage : UserControl
       {
              public MainPage()
              {
                     InitializeComponent();
                     dataGrid1.ItemsSource = Customer.GetSampleCustomerList();
              }
}

public class Customer
{
    public String FirstName { get; set; }
    public String LastName { get; set; }
    public String Address { get; set; }

    public Customer(String firstName, String lastName, String address)
    {
        this.FirstName = firstName;
        this.LastName = lastName;
        this.Address = address;
     }

    public static List<Customer> GetSampleCustomerList()
    {
        return new List<Customer>(new Customer[3]
       {
            new Customer("Jan", "Novak",
                "Slnecna 54 Prievidza"),
            new Customer("Zuzana", "Ticha",
                "Komjatna 1367"),
            new Customer("Emil", "Skokan",
                "Krizna 33 Bratislava"),
         });
    }
}
```

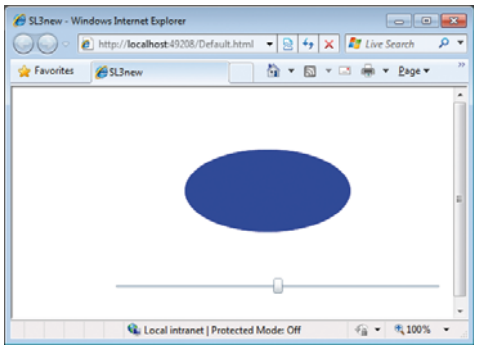


Aplikácia využívajúca DataGrid

Zviazanie elementov (databinding)

Na platforme Silverlight 3 je možné nielen napojiť elementy na zdroj údajov, ale prostredníctvom databindingu ich prepojiť navzájom. Logicky vzaté, má zmysel prepojiť prvok, ktorý generuje údaje na prvok, ktorý údaje „konzumuje", teda sa podľa nich správa. V príklade bude takýmto prvkom geometrický obrazec – elipsa, ktorá bude mať parameter, dĺžku jednej zo svojich osí, naviazaný na prvok, pomocou ktorého bude možné meniť hodnoty tohto parametra:

```
<Grid x:Name="LayoutRoot" Background="White">
  <Slider Width="400" Minimum="0" Maximum="400" Value="200" x:Name="reg"/>
  <Ellipse Height="100" Fill="Blue" Width="{Binding ElementName=reg, Path=Value}"
   Margin ="200,75,224,0" VerticalAlignment="Top" d:LayoutOverrides="Height"/>
</Grid>
```



Zviazanie elementov

Väzba medzi prvkami je v tomto prípade jednosmerná, teda Slider ovplyvňuje grafický prvok – elipsu. Opačná väzba nie je definovaná, nakoniec nie je ani logická, nakoľko elipsa je pasívny grafický prvok. Ako ďalší námet pre zviazanie elementov si môžete vyskúšať zviazanie prvku Slider s prvkom TextBlock, v ktorom sa bude zobrazovať hodnota nastavená pomocou prvku.

Validácia údajov

Na formuláre v interaktívnych webových aplikáciách sú kladené určité požiadavky, hlavne vzhľadom na komfort používateľov. Hlavne v aplikáciách kde „ide o biznis" je dôležité, aby údaje zadané používateľom (údaje, rodné číslo, číslo objednávky, číslo kreditnej karty...) boli zadané úplne presne a správne. Od bežného klienta, napríklad návštevníka internetového obchodu, samozrejme nemôžete chcieť, aby poznal všetky detaily vašej aplikácie. Pre vývojára je „samozrejmé", že sa bude používať desatinná bodka, matematicky „odchovaný" klient neprogramátor tam celkom pochopiteľne zadá čiarku, problémy sú zo zadávaním formátu dátumu a času a podobne. Preto musíte používateľovi pomôcť nielen vhodným a jednoznačným návrhom formulára pre zadávanie údajov, ale aj zadávané údaje kontrolovať a používateľa usmerňovať tak, aby výsledkom jeho snaženia bola úspešná transakcia.

V príklade bude ukázaný formulár pre zadanie dvoch údajov – mena a osobného čísla:

```
<StackPanel x:Name="LayoutRoot" Background="White">
    <TextBox Margin="15" Width="200"
        Text="{Binding Meno,Mode=TwoWay,ValidatesOnExceptions=True}" />
    <TextBox Margin="15" Width="200"
        Text="{Binding OsCislo,Mode=TwoWay,ValidatesOnExceptions=True}"/>
       <ListBox x:Name="lstErrors"  ItemsSource="{Binding}">
        <ListBox.ItemTemplate>
         <DataTemplate>
             <TextBlock Text="{Binding Exception.Message}" />
         </DataTemplate>
        </ListBox.ItemTemplate>
      </ListBox>
      <Button Width="200" Content="Potvrd" Click="Button_Click"  />
</StackPanel>
```

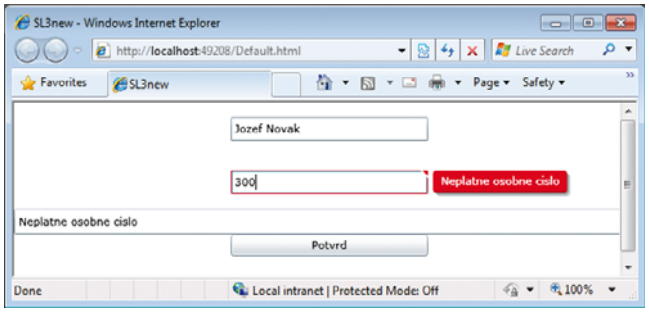
V prípade mena sa bude kontrolovať, či bol tento údaj zadaný v požadovanej minimálnej a maximálnej dĺžke reťazca, v prípade osobného čísla sa kontroluje, či je osobné číslo v požadovanom intervale hodnôt:

```
using System.Collections.Generic;
using System.Collections.ObjectModel;

namespace SL3new
{
   public partial class MainControl : UserControl
   {
       public MainControl()
       {
              // Required to initialize variables
              InitializeComponent();
              this.Loaded += OnLoaded;
       }

        void OnLoaded(object sender, RoutedEventArgs e)
    {
      this.DataContext = new Pracovnik("Jozef Novak", 100);
      lstErrors.DataContext = this;
    }


    private void Button_Click(object sender, System.Windows.RoutedEventArgs e)
    {
      List<ValidationError> errors = new List<ValidationError>();

      foreach (UIElement ui in LayoutRoot.Children)
      {
        FrameworkElement fe = ui as FrameworkElement;

        if (fe != null)
        {
          foreach (ValidationError ve in Validation.GetErrors(fe))
          {
            errors.Add(ve);
          }
        }
      }
```

```
      lstErrors.DataContext = errors;
    }
}

  public class InvalidDataException : Exception
  {
    public InvalidDataException(string msg) : base(msg)
    {
    }
  }
```

Validované prvky sú napojené na objekt, v tomto prípade triedu Pracovník, kde sú zahrnuté aj validačné pravidlá:

```
 public class Pracovnik
 {
    string meno;
    int osCislo;

       public Pracovnik(string meno, int osCislo)
    {
      this.meno = meno;
      this.osCislo = osCislo;
    }

       public string Meno
    {
      get {return (meno);}
      set {KontrolaDlzky(value, 3, 15); meno = value;}
    }


       public int OsCislo
    {
      get { return (osCislo); }
      set
      {
        if ((value < 100) || (value > 199))
        {
          throw new InvalidDataException("Neplatne osobne cislo");
        }
        osCislo = value;
      }
    }
```



Validácia zadávaných údajov

Animácia

Po zvládnutí základných grafických prvkov nastal čas presunúť sa na vyššiu métu a „rozpohybovať" scénu. Animácia významným spôsobom oživí intranetovú aplikáciu. Pohybujúce sa veci na seba upriamujú pozornosť a mnohé postupy, princípy a podobne sa dajú najlepšie vysvetliť na názorných animovaných ukážkach.

Nakoľko kontajnerový prvok Grid, ktorý je na stránku umiestnený pri vytváraní Silverlight 3 projektu sa hodí skôr na udržiavanie relatívnej polohy prvkov voči sebe, v tejto skupine aplikácii bude nahradený kontajnerovým prvkom Canvas, alebo StackPanel, čiže zjednodušene povedané, animácia sa bude odohrávať na „premietacom" plátne.

Double Animation

Vytvorte objekt, ktorý chcete animovať, v našom prípade vyfarbený kruh, teda geometrický objekt Ellipse s rovnakými osami:

```
<Canvas x:Name="LayoutRoot" Background="White">
   <Ellipse x:Name="Kruh" Width="100" Height="100" Fill="Blue" />
</Canvas>
```

Najskôr bude predstavený objekt **DoubleAnimation**, ktorý umožňuje robiť animáciu tak, že sa mení hodnota parametrov typu double, teda desatinných čísel s dvojnásobnou presnosťou. Pre objekt Ellipse môžete skúsiť meniť priesvitnosť, teda parameter Opacity.

Znova zdôrazňujeme, že pomocou Double Animation je možné meniť len hodnoty parametrov typu Double, napríklad rozmery obdĺžnika, uhol pootočenia pri transformácii a podobne. Nie je možné meniť hodnoty parametrov celočíselného typu.

Pri animácii je potrebné definovať počiatočnú a cieľovú hodnotu meneného parametra a čas, za ktorý sa zmena vykoná. Môžete nastaviť opakovanie deja, prípadne nastaviť, aby opakovanie prebiehalo striedavo reverzne, v tomto prípade sa plocha najskôr zosvetlí a potom nastane reverzný dej, kedy plocha stmavne:

```
<Storyboard>
  <DoubleAnimation
    Storyboard.TargetName="Kruh"
    From="1.0" To="0.0" Duration="0:0:1"
    AutoReverse="True" RepeatBehavior="Forever" />
</Storyboard>
```

Animovaný dej bude zapuzdrovať objekt **Storyboard**, ktorý umožní aj riadiť časový priebeh animácie. Kompletný kód pre animáciu zmeny priesvitnosti bude:

```
<Canvas x:Name="LayoutRoot" Background="White">
  <Canvas.Resources>
    <Storyboard x:Name="sbPohyb">
      <DoubleAnimation
```

```
        Storyboard.TargetName="Obdlznik"
        Storyboard.TargetProperty="Opacity"
        From="1.0" To="0.0" Duration="0:0:3"
        AutoReverse="True" RepeatBehavior="Forever" />
    </Storyboard>
  </Canvas.Resources>
     <Rectangle x:Name="Obdlznik" Width="150" Height="100" Fill="Blue" />
</Canvas>
```

Spustenie animácie

Ak by ste spustili aplikáciu v tomto okamihu, kruh by sa síce vykreslil, ale nebude sa nič meniť, nakoľko animácia zatiaľ nebola nijako aktivovaná. Môžete pridať kód pre spustenie animácie, hneď po spustení do konštruktora Main Control:

```
public partial class MainControl : UserControl
{
       public MainControl()
       {
              InitializeComponent();
              sbPohyb.Begin();
       }
}
```

alebo ako reakciu na nejakú udalosť, napríklad na kliknutie myšou na grafický objekt:

```
<Ellipse x:Name="Kruh" Width="100" Height="100" Fill="Blue"
MouseLeftButtonDown="Mouse_Clicked"/>
```

Obslužný kód udalosti bude:

```
private void Mouse_Clicked(object sender, MouseEventArgs e)
{
    sbPohyb.Begin();
}
```

Color Animation

Veľmi často sa vyskytuje požiadavka na zmenu farby. Pre tento účel nie je možné využiť Double Animation, nakoľko hodnoty parametrov charakterizujúce farby sú celé čísla. Preto je pre zmenu farby k dispozícii špecializovaný objekt pre animáciu Color Animation:

```
<StackPanel x:Name="LayoutRoot" >
    <StackPanel.Resources>
        <Storyboard x:Name="sbAnimacia">
            <ColorAnimation Storyboard.TargetName="caFarba"
             Storyboard.TargetProperty="Color"
             From="Blue" To="Yellow" Duration="0:0:5" />
        </Storyboard>
    </StackPanel.Resources>
```

```
    <StackPanel.Background>
        <SolidColorBrush x:Name="caFarba" Color="Blue" />
    </StackPanel.Background>
</StackPanel>
```

Nezabudnite pridať kód pre spustenie animácie:

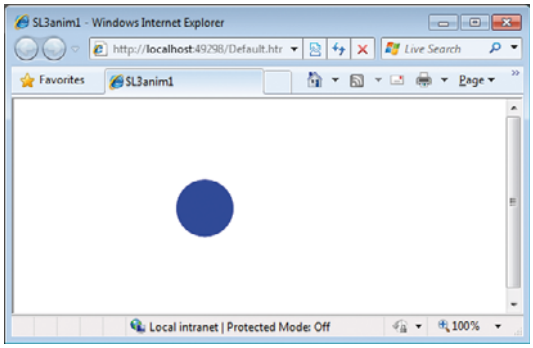
```
public MainControl()
{
       InitializeComponent();
       sbAnimacia.Begin();
}
```

Point Animation

Takáto animácia asi pre efektné stránky nepostačí. Síce sa tam niečo menilo, v tomto prípade priesvitnosť, no nebol tam žiadny pohyb. Preto nastal čas predstaviť ďalší objekt **PointAnimation**, ktorý umožní meniť súradnice bodu. Pre elipsu budeme meniť polohu stredu a tým pohybovať aj celou elipsou.

Aby bolo možné využívať zmenu parametrov objektu typu Point, elipsu bude potrebné vykresliť nie ako jednoduchý obrazec, ale ako EllipseGeometry:

```
<Canvas x:Name="LayoutRoot" Background="White">
  <Canvas.Resources>
    <Storyboard x:Name="sbPohyb">
      <PointAnimation Storyboard.TargetName="Kruh"
         Storyboard.TargetProperty="Center"
          From="50,50" To="400,200" Duration="0:0:5"
          AutoReverse="True" RepeatBehavior="Forever" />
    </Storyboard>
  </Canvas.Resources>
 <Path Fill="Blue">
    <Path.Data>
        <EllipseGeometry x:Name="Kruh"Center="50,50" RadiusX="30" RadiusY="30" />
    </Path.Data>
  </Path>
</Canvas>
```



Príklad pre Point Animation

Riadenie priebehu animácie

Zatiaľ sa animácia len spustí, ale nijako priebežne neriadi. Pre riadenie animácie sú v širokom spektre aplikácií obvyklé tlačidlá, alebo inak graficky stvárnené ovládacie prvky s významom základných riadiacich funkcií animácie: „ŠTART", „STOP" a „PAUZA".

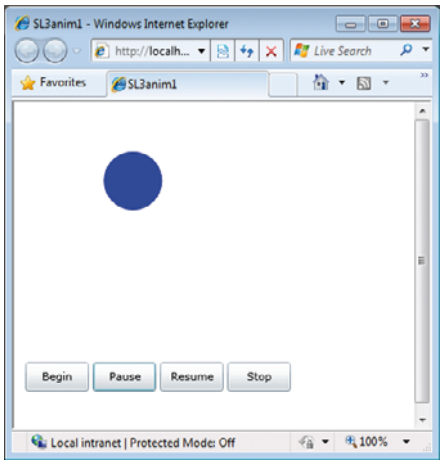
Do XAML kódu, je potrebné pridať jednotlivé tlačidlá a referencie na obsluhu udalosti stlačenia ľavého tlačidla myši:

```
<StackPanel Orientation="Horizontal" Canvas.Left="10" Canvas.Top="265">
  <Button Click="Animation_Begin"
    Width="65" Height="30" Margin="2" Content="Begin" />

  <Button Click="Animation_Pause"
    Width="65" Height="30" Margin="2" Content="Pause" />

  <Button Click="Animation_Resume"
    Width="65" Height="30" Margin="2" Content="Resume" />

  <Button Click="Animation_Stop"
    Width="65" Height="30" Margin="2" Content="Stop" />
</StackPanel>
```

Pre jednotlivé tlačidlá sa aktivujú obslužné procedúry:

```
private void Animation_Begin(object sender, RoutedEventArgs e)
{
  sbPohyb.Begin();
}

private void Animation_Pause(object sender, RoutedEventArgs e)
{
   sbPohyb.Pause();
}

private void Animation_Resume(object sender, RoutedEventArgs e)
{
  sbPohyb.Resume();
}

private void Animation_Stop(object sender, RoutedEventArgs e)
{
   sbPohyb.Stop();
}
```

Po spustení aplikácie môžeme ovládanie animácie vyskúšať.



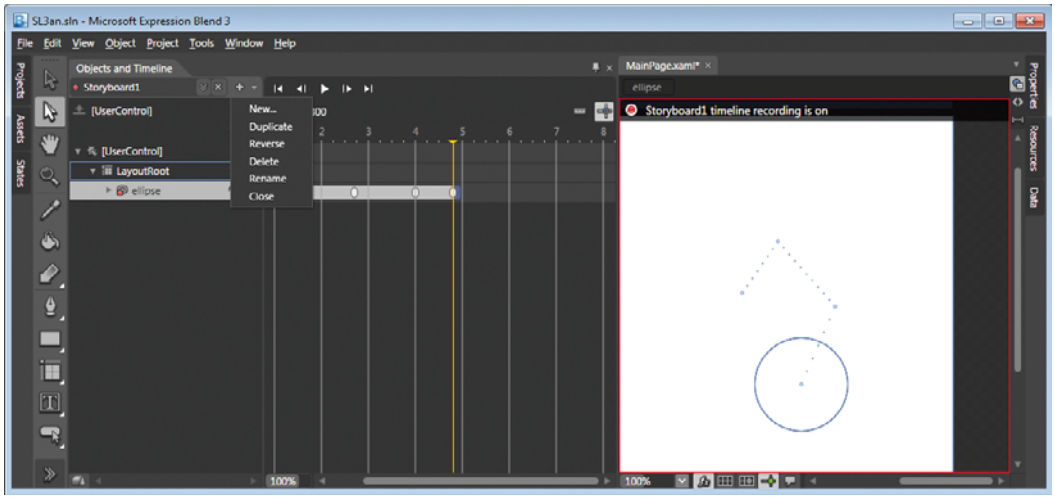
Riadenie animácie pomocou tlačidiel

Vytvorenie animácie v grafickom návrhovom prostredí

V predchádzajúcom príklade bola animácia vytvorená priamo v XAML kóde. Pre zložitejšie animácie je výhodné využiť vizuálne návrhové možnosti prostredia Expression Blend.

V časti pracovnej plochy v záložke označenej **Object and Timeline** vytvorte nový objekt typu **Storyboard**.

„Nahrávanie" postupu animácie funguje jednoducho a intuitívne. Zobrazí sa časová os a Canvas sa prepne do režimu nahrávania. Všimnite si, že v tomto režime je canvas ohraničený červeným rámikom a v ľavom hornom rohu je červený krúžok ako symbol režimu nahrávania. Úseky na časovej osi sú označené časovou stupnicou. Presuňte kurzor (zvislú žltú čiaru) do ďalšieho časového úseku a pomocou vzťažných bodov zmeňte scénu.



Nahrávanie animácie na časovej osi. Všimnite si menu pre vytvorenie nového objektu

Tieto zmeny sa automaticky zaznamenajú v XAML kóde (náš ilustračný výpis je skrátený):

```
<Storyboard x:Name="Storyboard1">
<DoubleAnimationUsingKeyFrames BeginTime="00:00:00" Storyboard.TargetName="path"
  Storyboard.TargetProperty="(UIElement.RenderTransform).(TransformGroup.Children)
  [3].(TranslateTransform.X)">
     <EasingDoubleKeyFrame KeyTime="00:00:01.8000000" Value="39"/>
     <EasingDoubleKeyFrame KeyTime="00:00:02.7000000" Value="33"/>
     <EasingDoubleKeyFrame KeyTime="00:00:03.6000000" Value="94"/>
     <EasingDoubleKeyFrame KeyTime="00:00:04.6000000" Value="100"/>
</DoubleAnimationUsingKeyFrames>
```

Aby sa animácia cyklicky opakovala, môžeme zmeniť vlastnosť „Repeat behavior" . Variácií a možností na tému animácia je v prostredí Silverlight tak veľa, že máme pre vás len jedno odporúčanie – vyskúšať si to na vlastnom projekte.

Animation Easing

Pre niektoré scény je dôležité ich reálne rozpohybovanie. Aby pohyb objektov verne odrážal reálne prostredie a v ňom platiace fyzikálne zákony, môžete využiť funkciu Easing, ktorá riadi animáciu pomocou matematických rovníc popisujúcich fyzikálne zákony. Funkcií pre Easing je veľké množstvo (), na praktickom príklade ukážeme funkciu BounceEase, napodobňujúcu skákanie pružného objektu pri odraze od podložky, čiže zjednodušene povedané, ak pustíte pingpongovú loptičku na betón a ona začne skákať:

```
<Canvas x:Name="LayoutRoot" Background="White">
  <Canvas.Resources>
     <Storyboard x:Name="pohyb">
       <DoubleAnimation From="0" To="200" Duration="0:0:5"
           Storyboard.TargetName="kruh"
           Storyboard.TargetProperty="(Canvas.Top)">
           <DoubleAnimation.EasingFunction>
            <BounceEase EasingMode="EaseOut" Bounces="10" Bounciness="1"></BounceEase>
           </DoubleAnimation.EasingFunction>
           </DoubleAnimation>
         </Storyboard>
      </Canvas.Resources>
    <Ellipse Name="kruh" Width="50" Height="50" Fill="Blue" Canvas.Top="0"
       Canvas.Left="0"  ></Ellipse>
    </Canvas>
```

Nezabudnite pridať kód pre spustenie animácie, buď hneď po spustení, alebo ako reakciu na nejakú udalosť, napríklad na kliknutie myšou na grafický objekt:

```
public partial class MainControl : UserControl
{
       public MainControl()
       {
              InitializeComponent();
              pohyb.Begin();
       }
}
```

Po spustení aplikácie sa zobrazí skákajúca guľôčka, pričom podobne ako v reálnom svete môžete meniť napríklad vlastnosti (pružnosť) guľôčky a podobne a sledovať ako sa zmení pohyb animovaného objektu.

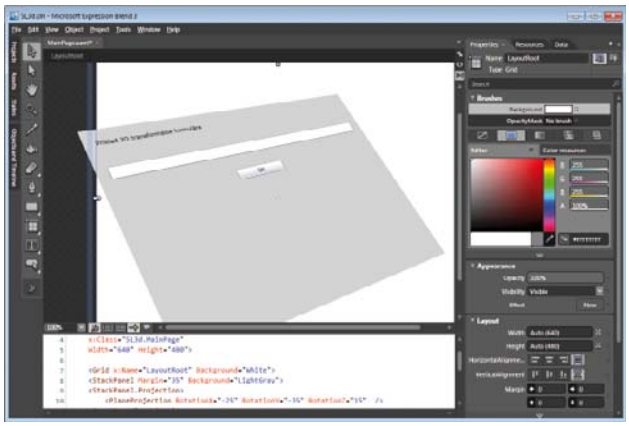
3D Efekty

Nová verzia prostredia Silverlight 3 priniesla aj nové možnosti pre vylepšenie a skvalitnenie vzhľadu prezentačnej vrstvy aplikácie a možností jej interakcie s používateľom.

3D perspektíva umožní rozmiestňovať dvojrozmerné objekty do 3D priestoru, napríklad virtuálne plátno na ktorom sa zobrazuje text, celé formuláre, obrázky, prípadne video sa namapuje na stenu v priestore pootočenej kocky a podobne. 3D perspektíva umožňuje rotáciu podľa všetkých troch osí X, Y, Z , pričom v týchto osiach je možné zvoliť aj stred otáčania a ofsety.

Ukážeme príklad aplikácie 3D efektu PlaneProjection na formulár zapuzdrený v kontajnerovom objekte StackPanel:

```
<Grid x:Name="LayoutRoot" Background="White">
       <StackPanel Margin="35" Background="LightGray">
    <StackPanel.Projection>
        <PlaneProjection RotationX="-25" RotationY="-35" RotationZ="15"  />
    </StackPanel.Projection>
    <TextBlock Margin="20">Príklad 3D transformácie formulára</TextBlock>
    <TextBox Margin="20"></TextBox>
    <Button Margin="10" Content="OK" Width="100" />
    </StackPanel>
</Grid>
```



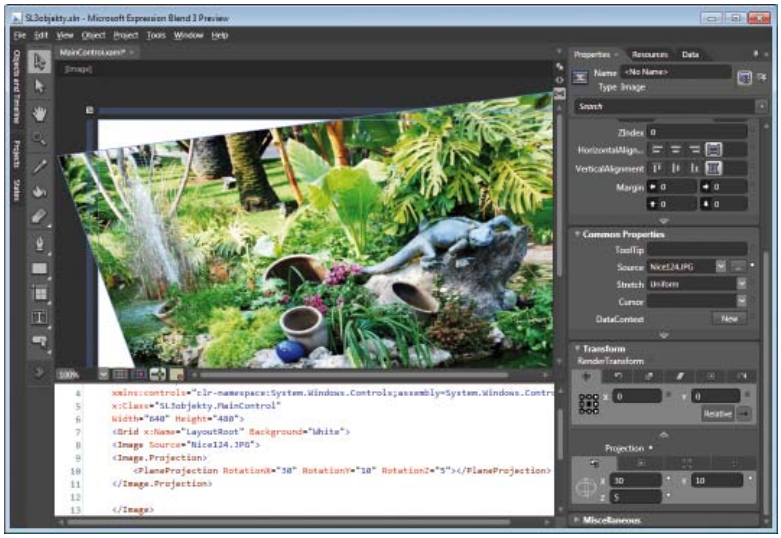
Príklad 3D zobrazenia formulára

Podobne môžete skúsiť zobraziť v priestore aj obrázok. Základný kód pre planárne zobrazenie obrázku využíva prvok Image:

```
<Image Source="Nice124.JPG">
</Image>
```

Vo verzii 3.0 je možné nastaviť projekciu, napríklad pootočiť obrázok v požadovaných osiach:

```
<Image Source="Nice124.JPG">
    <Image.Projection>
        <PlaneProjection RotationX="30" RotationY="10" RotationZ="10">
        </PlaneProjection>
    </Image.Projection>
</Image>
```



Zobrazenie obrázku v perspektíve

Podobne je možné zobrazovať aj video:

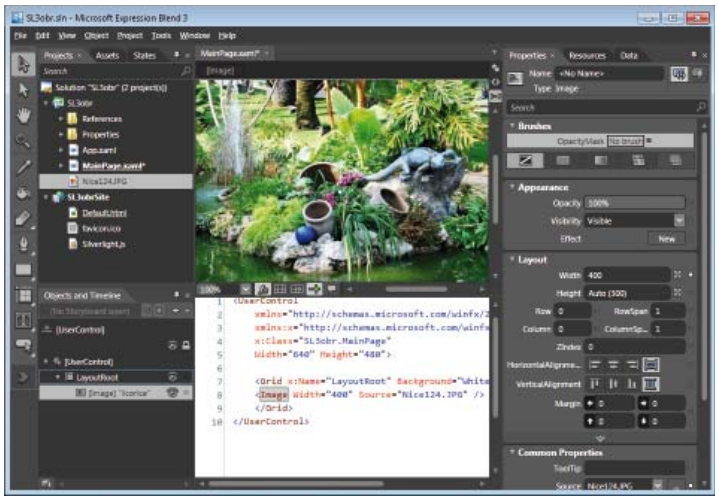
```
 <Grid x:Name="LayoutRoot" Background="White">
    <MediaElement x:Name="me"  Stretch="None" Source="Windows7VHDBoot.wmv">
    </MediaElement>
</Grid>
```

Práca s obrázkami

Najjednoduchším multimediálnym prvkom je obrázok. Pre jeho zobrazenie slúži XAML tag:

```
<Image Source="Nice124.JPG">
</Image>
```

Parameter Source udáva URL adresu obrázku. V príklade je súbor s obrázkom umiestnený priamo do priečinka Silverlight aplikácie.



Zobrazenie obrázku

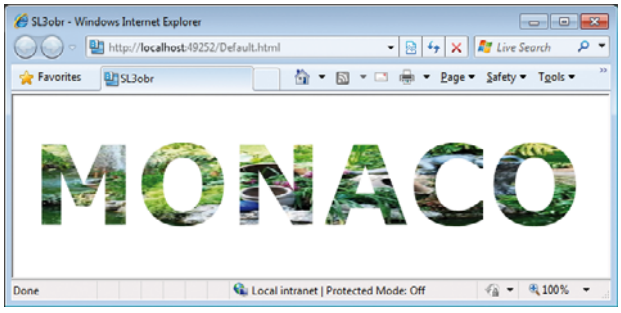
Pomocou Objektu Clip je možné obrázok vhodne „zarámovať", teda umiestniť do geometrického objektu:

```
<Image Source="Nice124.jpg"
    Width="200" Height="150">
    <Image.Clip>
      <EllipseGeometry RadiusX="100" RadiusY="75" Center="100,75"/>
    </Image.Clip>
  </Image>
```

Môžete skúsiť využiť vlastnosť Opacity pre neostrý okraj.

Obrázok je možné využiť aj ako podklad pre vyplnenie, podobne ako štetec. Pomocou objektu ImageBrush je možné vytvoriť nadpis, ktorý nebude vyplnený farbou, ale motívom obrázku:

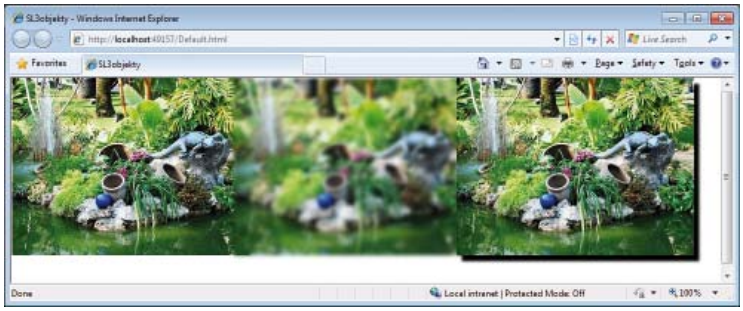
```
<Grid x:Name="LayoutRoot" Background="White">
    <TextBlock Margin="20" FontFamily="Verdana" FontSize="120"
        FontWeight="Bold">
        MONACO
        <TextBlock.Foreground>
           <ImageBrush ImageSource="Nice124.JPG"/>
        </TextBlock.Foreground>
    </TextBlock>
</Grid>
```



Využitie obrázku pre vyfarbenie objektu

Môžete taktiež vyskúšať rôzne Pixel Shaders efekty:

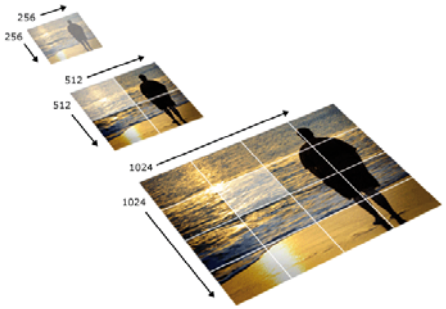
```
<Canvas x:Name="LayoutRoot" Background="White">
<Image x:Name="img1" Source="Nice124.JPG" Canvas.Left="1"></Image>
<Image x:Name="img2" Source="Nice124.JPG" Canvas.Left="300">
  <Image.Effect>
     <BlurEffect Radius="8"></BlurEffect>
  </Image.Effect>
</Image>
<Image x:Name="img3" Source="Nice124.JPG" Canvas.Left="600">
<Image.Effect>
   <DropShadowEffect ShadowDepth="30"></DropShadowEffect>
</Image.Effect>
</Image>
</Canvas>
```



Pixel Shaders

Deep Zoom

Pre zobrazenie obrázkov využívajúcich Deep Zoom je potrebné vytvoriť súbor obrázkov, takzvanú pyramídu v rôznom zväčšení. Spravidla najnižším rozlíšením je 256 x 256 pixlov. Pre tento účel je potrebné nainštalovať aplikáciu Deep Zoom Composer.



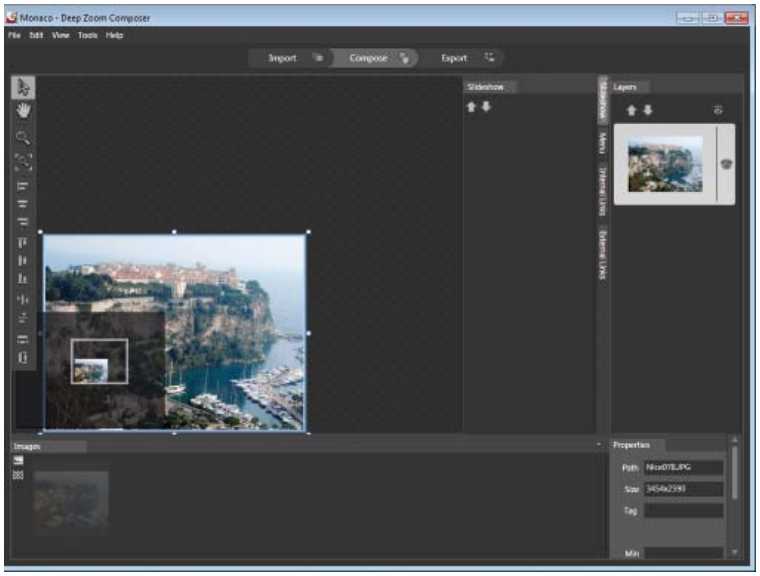
Princíp rozkladu obrázku pomocou aplikácie Deep Zoom Composer

Ukážeme príklad zobrazenia obrázku v rôznom stupni rozlíšenia. V aplikácii Deep Zoom Composer si po vytvorení nového projektu všimnite v hornej časti jednoduchú nástrojovú lištu s tromi tlačidlami:

- Import,
- Compose,
- Export.

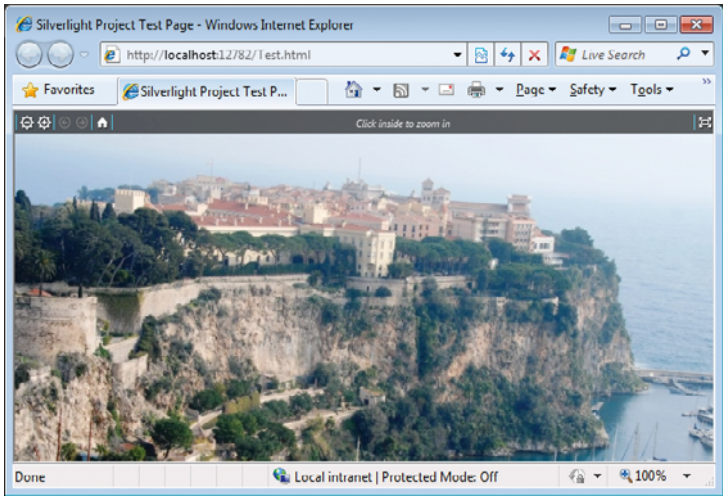
Pomocou týchto tlačidiel sa prepína režim pracovnej obrazovky aplikácie. Najskôr v záložke **Import** pridajte obrázok v najvyššom požadovanom rozlíšení pomocou tlačidla **Add image**.

Po prepnutí do režimu **Compose** umiestnite obrázok na pracovnú plochu. V originálnej terminológii sa táto návrhová plocha nazýva artboard. Zvoľte veľkosť obrázku.



Deep Zoom Composer

Dej pokračuje v poslednom priečinku **Compose**. V záložke Custom zvoľte typ exportu **Silverlight Deep Zoom** a aktivujte tlačidlo Export. Výstup a zväčšovanie si môžete vyskúšať po ukončení exportu v testovacej aplikácii.



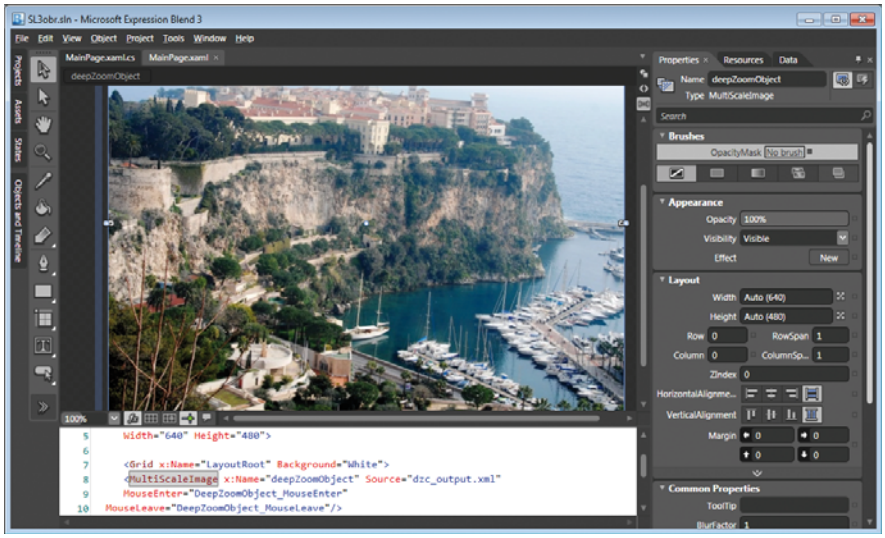
Prezeranie výstupu z aplikácie Deep Zoom Composer v testovacej aplikácii

Vytvorte novú aplikáciu a výstup exportu z aplikácie Deep Zoom Composer umiestnite do jeho priečinka. Do projektu pridajte XAML tag:

```
<MultiScaleImage x:Name="deepZoomObject" Source="dzc_output.xml" />
```

Ak chcete zväčšovanie a zmenšovanie ovládať, je potrebné pridať udalosti a ich obslužný kód:

```
<MultiScaleImage x:Name="deepZoomObject" Source="dzc_output.xml"
       MouseEnter="DeepZoomObject_MouseEnter"
 MouseLeave="DeepZoomObject_MouseLeave"/>
```



Element MultiScaleImage

Obslužný kód udalostí:

```
private void DeepZoomObject_MouseEnter(object sender, MouseEventArgs e)
{
      this.deepZoomObject.ZoomAboutLogicalPoint(3, .5, .5);
}

private void DeepZoomObject_MouseLeave(object sender, MouseEventArgs e)
{
   double zoom = 1;
   zoom = zoom / 3;
   this.deepZoomObject.ZoomAboutLogicalPoint(zoom, .5, .5);
}
```

Možnosti tejto črty odporúčame vyskúšať napríklad na **http://memorabilia.hardrock.com/** alebo na hlavnej stránke projektu Silverlight.

Využitie grafického procesora

Aj tie najlacnejšie klientske počítače dnes disponujú pomerne výkonným grafickým adaptérom, ktorý pri väčšine webových aplikácií doslova zaháľa a o grafické zobrazenie sa stará CPU. Silverlight 3 umožňuje pre graficky náročné operácie využiť aj výkon grafickej karty. Skúste vytvoriť graficky náročnú aplikáciu, najlepšie s nejakou animáciou a pozrite si ako zaťažuje procesor. Následne nájdite HTML stránku hosťovacej webovej aplikácie, ktorá je súčasťou Silverlight projektu a zamerajte pozornosť na sekciu <Object>, ktorá obsahuje nastavenie parametrov:

```
<object data="data:application/x-silverlight," type="application/x-silverlight-2"
width="100%" height="100%">
   <param name="source" value="ClientBin/SILprvky.xap"/>
   <param name="onerror" value="onSilverlightError" />
   <param name="background" value="white" />
   <param name="minRuntimeVersion" value="3.0.40624.0" />
   <param name="autoUpgrade" value="true" />
   ...
</object>
```

Do sekcie vložte parameter povoľujúci využitie GPU:

```
<param name="EnableGPUAcceleration" value="true" />
```

Teraz môžete porovnať, nakoľko sa do hry zapojil grafický akcelerátor a ako sa tým odľahčilo procesoru.

Prehrávanie multimediálnych súborov

Nástup televízie s vysokým rozlíšením, si vyžiadal nasadiť účinnejšie kompresné metódy, aby sa existujúca kapacita prenosových pásiem lepšie využila. Preto Silverlight 3 podporuje **komprimačný algoritmu videa H.264, MPEG-4 AVC**, pri ktorom je komprimované video prenášané v transportnom toku (kontajneri) MPEG-2. Kodeky H.264 umožnia oproti MPEG-2 znížiť nároky na kapacitu pri prenose videa dva až trikrát. Podporované sú aj formáty videa, ktoré využívajú **YouTube, iPhone a Flash**. Grafický výkon aplikácií môže podstatne stúpnuť nakoľko Silverlight 3 podporuje **GPU akceleráciu**, teda akceleráciu na úrovni grafickej karty. Vyšší grafický výkon potom umožní aj najnáročnejšie aplikácie spúšťať v celoobrazovkovom režime.

V našom príklade sme využili súbor „Wildlife.wmv“, ktorý je implicitne umiestnený do knižnice dokumentov, konkrétne videí v operačnom systéme Microsoft Windows 7. Súbor prekopírujte napríklad do priečinka aplikácie. V záložke okna „Project“ v pravej časti obrazovky aktivujte kontextové menu a vyberte položku „Add Existing Item...“. Takto pridáte video súbor do projektu. Súbor sa zobrazí v okne „Project“. Dvojitým kliknutím na ikonu súboru, alebo presunom symbolu videa na plochu aplikácie sa pridá do XAML kódu objekt MediaElement pre prehrávanie videa:

```
<MediaElement x:Name="Wildlife_wmv" Margin="318,239,-958,-479"
  Source="/Wildlife.wmv" Stretch="Fill"/>
```

Všimnite si, že pri tomto návrhovom úkone bol vytvorený prvok typu **MediaElement** s názvom „Wildlife_wmv“. Názov bol odvodený od mena multimediálneho súboru, ktorý sa bude prostredníctvom prvku MediaElement prehrávať. Tento názov potom budete používať v kóde, keď budete chcieť doplniť ovládanie prehrávania.

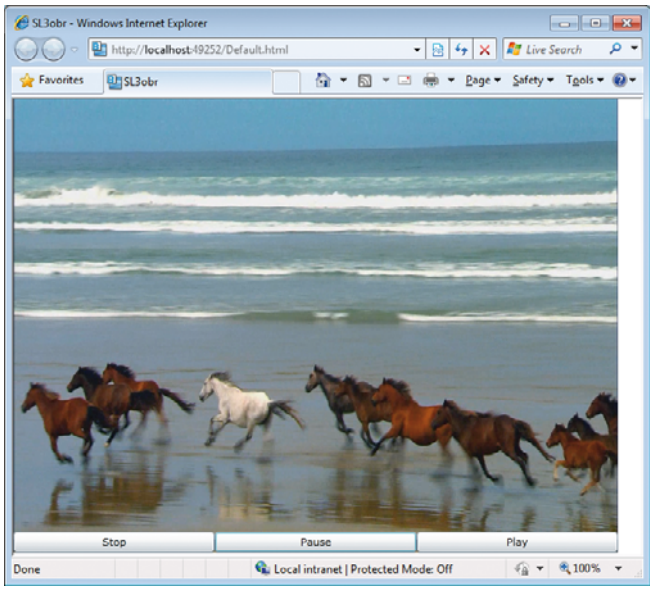
Teraz môžeme klávesom „F5“ projekt spustiť a otestovať. Po otvorení prehľadávača sa automaticky začne prehrávať videosúbor, zatiaľ bez možnosti ovládania. Preto je v ďalšej fáze príkladu potrebné do projektu pridať aspoň základné ovládacie prvky, pre zastavenie a pre opätovné spustenie prehrávania:

```
<Grid x:Name="LayoutRoot" Background="White">
       <Grid.ColumnDefinitions>
          <ColumnDefinition Width="*" />
          <ColumnDefinition Width="*" />
```

```
            <ColumnDefinition Width="*"/>
        </Grid.ColumnDefinitions>

          <Grid.RowDefinitions>
         <RowDefinition Height="*" />
          <RowDefinition Height="Auto" />
    </Grid.RowDefinitions>

    <MediaElement x:Name="media" Source="wildlife.wmv" Width="300" Height="300"
                  Grid.Column="0" Grid.Row="0" Grid.ColumnSpan="3" />
    <Button Click="StopMedia"
     Grid.Column="0" Grid.Row="1" Content="Stop" />
    <Button Click="PauseMedia"
     Grid.Column="1" Grid.Row="1" Content="Pause" />
    <Button Click="PlayMedia"
     Grid.Column="2" Grid.Row="1" Content="Play" />
</Grid>
```

Obslužný kód udalostí:

```
private void StopMedia(object sender, RoutedEventArgs e)
{
    Wildlife_wmv.Stop();
}
private void PauseMedia(object sender, RoutedEventArgs e)
{
    Wildlife_wmv.Pause();
}
private void PlayMedia(object sender, RoutedEventArgs e)
{
    Wildlife_wmv.Play();
}
```



Prehrávanie videa s tlačidlovým ovládaním

Špeciálne režimy zobrazovania

Prepnutie zobrazovania na celú obrazovku

Systém okien pre viacero súčasne bežiacich aplikácií je bežný režim pre väčšinu bežných úkonov a prác na počítači. Pre niektoré aplikácie, napríklad také, ktoré zobrazujú veľké množstvo údajov, alebo grafiku, ktorá zaberá celú plochu môže byť výhodné zobrazovať v režime celej obrazovky (full screen). Do tohto režimu nie je možné sa prepnúť po štarte aplikácie, ale až ako odozvu na nejakú používateľovu aktivitu, takže nič v štýle automaticky spúšťaných reklám na celú obrazovku nehrozí. Režim zobrazovania sa nastavuje pomocou vlastnosti IsFullScreen. Môžete to urobiť napríklad ako obsluhu udalosti stlačenia tlačidla:

```
private void btCelaObrazovka_Click(object sender, System.Windows.RoutedEventArgs e)
{
       // TODO: Add event handler implementation here.
       App.Current.Host.Content.IsFullScreen = true;
}
```

Po prepnutí do celoobrazovkového režimu sa zobrazí upozornenie, že pomocou klávesu ESC sa aplikácia vráti do pôvodného zobrazovacieho režimu v okne.

Nakoľko usporiadanie ovládacích prvkov závisí od veľkosti okna aplikácie, môže byť užitočné, aby sa aplikácia o prepnutí do celoobrazovkového režimu „dozvedela" a bolo možné napríklad inak usporiadať ovládacie prvky. Pre tento účel slúži udalosť FullScreenChanged:

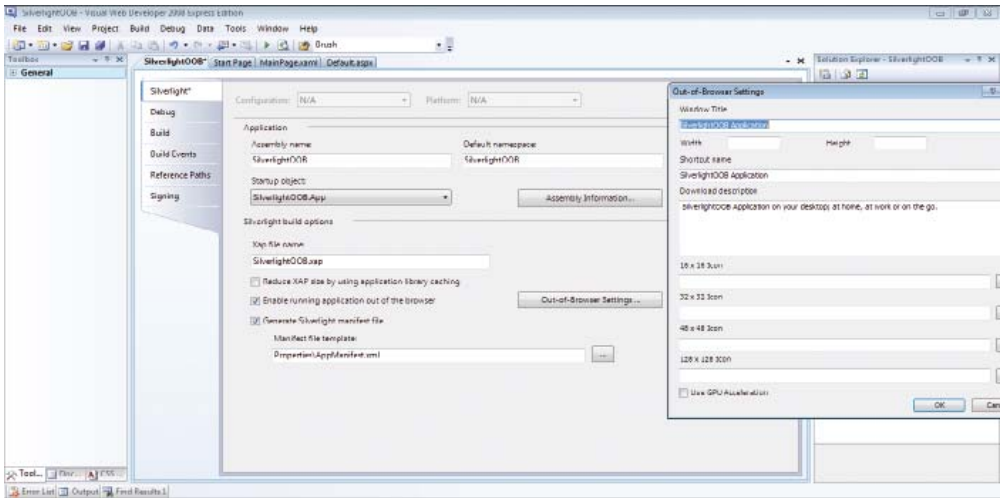
```
private void btCelaObrazovka_Click(object sender, System.Windows.RoutedEventArgs e)
{
   App.Current.Host.Content.FullScreenChanged += new
EventHandler(App_FullScreenChanged);
    App.Current.Host.Content.IsFullScreen = true;
}

void App_FullScreenChanged(object sender, EventArgs e)
{
    if (App.Current.Host.Content.IsFullScreen)
    {...}
    else
    {...}
}
```

Aplikácia typu Out – of – Browser

Nasadenie aplikácie na lokálny počítač prebehne jedným kliknutím na položku ponukového menu. Nakoľko OOB Silverlight aplikácia beží v izolovanom sandboxe, nie sú potrebné administrátorské oprávnenia pre inštaláciu. Pri každom štarte aplikácie sa v prípade internetovej konektivity skontroluje verzia a vykoná sa aktualizácia, ak je potrebná. Údaje je možné ukladať buď na server, alebo ak nie je aktuálne konektivita, dajú sa dočasne uložiť v Isolated Storage. Nakoľko Silverlight 3 obsahuje rozšírenú dátovú podporu aj podporu behu aplikácie mimo prehľadávač, je to ideálny framework aj pre zložitejšie LOB (Line of Business) aplikácie.

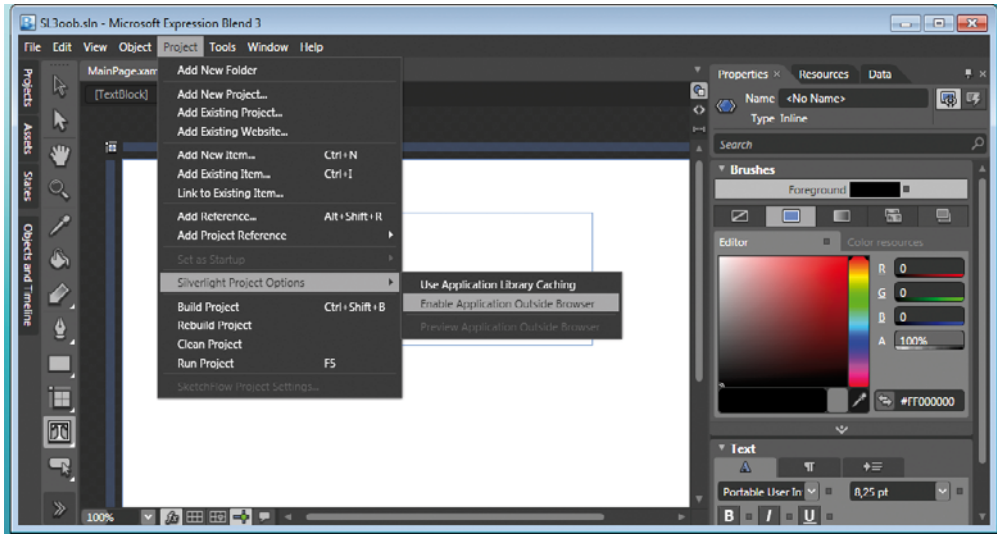
Po vytvorení projektu je potrebné v prostredí Visual Studio nastaviť OOB v menu Project - [Názov aplikácie] Properties. V záložke dialógového okna je potrebné označiť voľbu „**Enable running**

application out of browser". Pomocou tlačidla **Out of Browser Settings...** je možné nastaviť ikony a texty v záhlaví okna aplikácie.



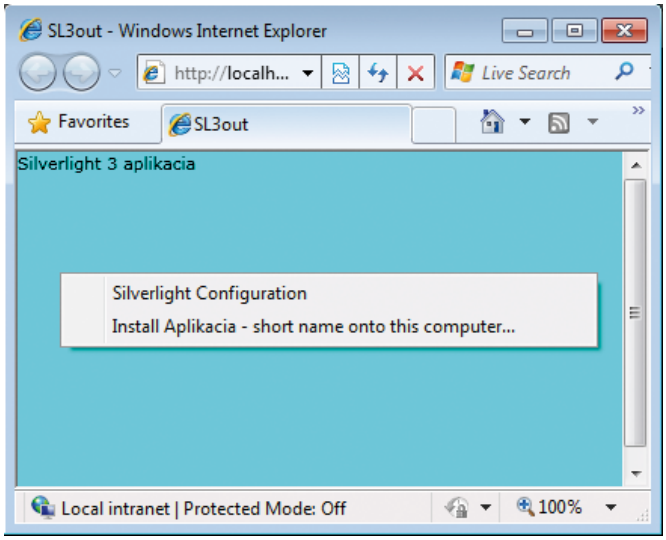
Úpravy parametrov aplikácie v prostredí Visual Studio, aby mohla bežať bez prehľadávača

Ak vyvíjate aplikáciu v prostredí Expression Blend, v menu **Project** aktivujte položky **Silverlight Project Options** a **Enable Application Outside Browser**.

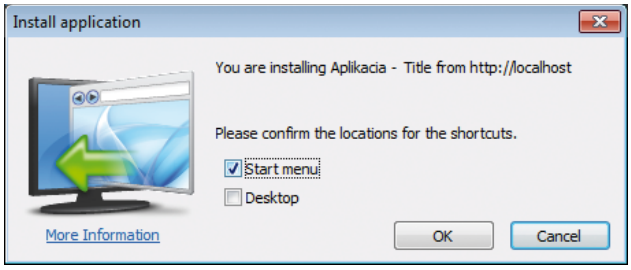


Úpravy parametrov aplikácie v prostredí Expression Blend, aby mohla bežať bez prehľadávača

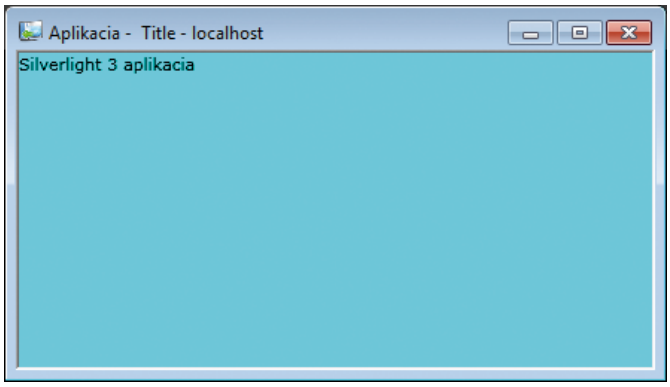
Po spustení aplikácie pribudne do kontextového menu voľba pre inštaláciu aplikácie tak, aby mohla bežať mimo prostredie prehľadávača. Aplikáciu potom spustíte štandardným spôsobom, teda buď pomocou Štart menu operačného systému Windows, alebo pomocou ikony na ploche.



Menu pre inštaláciu aplikácie



Dialóg pre inštaláciu aplikácie



Beh Silverlight 3 aplikácie v okne

Pri opätovnej aktivácii kontextového menu v okne prehľadávača bude toto menu obsahovať ponuku na odinštalovanie aplikácie.

Uloženie obsahu do súboru na lokálnom PC

Aby bolo možné ukladať časti kontextu Silverlight aplikácie do súboru na lokálnom disku, pribudol vo verzii Silverlight 3 dialóg typu „Save File". Projekt pre otestovanie tejto funkcionality bude obsahovať pole typu text Box pre zadanie textu, ktorý má byť uložený do súboru a tlačidlo pre aktiváciu ukladania:

```
<Grid x:Name="LayoutRoot" Background="Cyan">
       <TextBox x:Name="tbText" Height="25" Text="Vstup textu..."></TextBox>
       <Button x:Name="btUloz" Height="35" Margin="295,0,133,164"
        VerticalAlignment="Bottom" Content="Uloz text" Click="btUloz_Click" />
</Grid>
```

Obsluha udalosti stlačenia tlačidla bude pracovať s objektom **SaveFileDialog**:

```
private void btUloz_Click(object sender, System.Windows.RoutedEventArgs e)
{
       SaveFileDialog sfdUloz = new SaveFileDialog();
       bool? sf = sfdUloz.ShowDialog();
       if (sf == true)
       {
          using (Stream fs = (Stream)sfdUloz.OpenFile())
          {
             byte[] info = (new UTF8Encoding(true)).GetBytes(tbText.Text);
             fs.Write(info, 0, info.Length);
             fs.Close();
           }
         }
   }
```

Do kódu je potrebné pridať referencie na namespace:

```
using System.IO;
using System.Text;
```

V prípade vopred známeho typu súboru je možné pre objekt SaveFileDialog nastaviť filter:

```
aveFileDialog saveDialog = new SaveFileDialog();
saveDialog.DefaultExt = ".txt";
saveDialog.Filter = "Text File|*.txt|All Files|*.*";
```

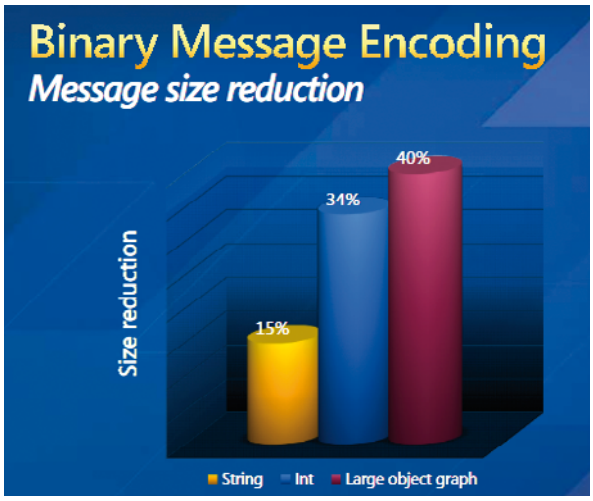
V predchádzajúcom príklade bol príklad uloženia textu do súboru, pričom text bol vygenerovaný Silverlight aplikáciou, presnejšie zadaný používateľom a spracovaný Silverlight aplikáciou. V druhom typickom scenári bude do súboru uložený obsah prevzatý z webu, v tomto prípade obrázok. Ako obrázok môžete využiť Silverlight logo z URL adresy
http://silverlight.net/Themes/silverlight/images/logo.jpg.

Upozornenie: Úloha sa zdá byť na prvý pohľad jednoduchá, stačí vytvoriť a aktivovať dialóg, vybrať meno súboru a uložiť ako binárny obsah. No binárny obsah je možné ukladať až vtedy, keď sa z webu načíta, čiže procedúra pre ukladanie obrázka musí byť aktivovaná až po jeho načítaní. Objekt **SaveFileDialog** je skonštruovaný na úrovni hlavnej triedy **MainControl :UserControl**, aby ho mohli využívať obidve procedúry. Procedúra **Ukladanie** sa aktivuje až po načítaní obsahu z webu:

```
using System.IO;
using System.Text;
using System.Net;

namespace SL3out
{
   public partial class MainControl : UserControl
   {

       SaveFileDialog sfd = new SaveFileDialog();


       public MainControl()
       {
              // Required to initialize variables
              InitializeComponent();
       }

       private void btUloz_Click(object sender, System.Windows.RoutedEventArgs e)
              {
                     sfd.DefaultExt = ".jpg";
            sfd.Filter = "JPG File|*.jpg|All Files|*.*";

            bool? open = sfd.ShowDialog();

            if (open.HasValue && open.Value)
            {
                Uri ur = new
               Uri("http://silverlight.net/Themes/silverlight/images/logo.jpg");
                WebClient wc = new WebClient();
                wc.OpenReadAsync(ur);
               wc.OpenReadCompleted += new OpenReadCompletedEventHandler(Ukladanie);
            }
        }

        void Ukladanie(object sender, OpenReadCompletedEventArgs e)
        {
            if (!e.Cancelled)
            {
                using (Stream fs = sfd.OpenFile())
                {
                    int length = Convert.ToInt32(e.Result.Length);
                    byte[] byteResult = new byte[length];
                    e.Result.Read(byteResult, 0, length);
                    fs.Write(byteResult, 0, byteResult.Length);
                    fs.Close();
                }
            }
        }
    }
}
```

Ďalšie technologické novinky

Silverlight 3 prináša veľa noviniek aj v oblasti sieťovej komunikácie a webových služieb. K zrýchleniu komunikácie prispeje nový **Binary Message Encoder**. Umožňuje zmenšiť veľkosť posielaných balíkov. O úspore kapacity pre jednotlivé typy správ svedčí graf na obrázku.



Redukcia veľkosti pre jednotlivé typy správ

Ak v Silverlight aplikácii využívajúcej webovú službu vybudovanej na platforme Silverlight 2 došlo k problému v komunikácii služby, tieto výnimky neboli šírené do Silverlight aplikácie a teda nebolo ich tam možné ošetriť. Prejavili sa len ako nič nehovoriaca všeobecná výnimka CommunicationException. V novej verzii Silverlight 3 sú k dispozícii výnimky **FaultException a FaultException<ExceptionDetail>**, ktoré nesú podrobné informácie o 0príčine zlyhania. Výnimky typu WCF error faults sú teraz šírené naprieč celým komunikačným reťazcom vrátane Silverlight 3 aplikácie.

Zjednodušilo sa aj používanie **Server-side push duplexu** a v novej verzii je k dispozícii aj **Binary XML serialization**.

Pre vytvorenie klientskej aplikácie využívajúcej WCF Duplex Service je potrebné do klientskej Silverlight aplikácie pridať referenciu na príslušnú službu (kontextové menu Add reference) a vytvoriť Call Back metódy:

```
sing System.Windows.Controls;
using System.ServiceModel;
using System.ServiceModel.Channels;
using SL3DuplexClient.SL3DuplexService;
using System;

namespace SL3DuplexClient
{
    public partial class MainPage : UserControl
    {
        public MainPage()
        {
            InitializeComponent();
```

```
            EndpointAddress address = new
            EndpointAddress("http://localhost:19021/DuplexService.svc");

            CustomBinding binding = new CustomBinding(
                new PollingDuplexBindingElement(),
                new BinaryMessageEncodingBindingElement(),
                new HttpTransportBindingElement());

            DuplexServiceClient proxy = new DuplexServiceClient(binding, address);
            proxy.ReceiveReceived += new
            EventHandler<ReceiveReceivedEventArgs>(proxy_ReceiveReceived);
            proxy.OrderAsync("Widget", 3);
            reply.Text = "Sent order of 3 Widgets." + Environment.NewLine;

        }

        void proxy_ReceiveReceived(object sender, ReceiveReceivedEventArgs e)
        {
            if (e.Error == null)
            {
                reply.Text += "Service reports Widget order is " + e.order.Status +
                "." + Environment.NewLine;

                if (e.order.Status == OrderStatus.Completed)
                {
                    reply.Text += "Here is the completed order:" +
                    Environment.NewLine;

                    foreach (string order in e.order.Payload)
                    {
                        reply.Text += order + Environment.NewLine;
                    }
                }

            }
        }

    }

}
```

Pre zobrazenie textu je na XAML stránke jediný prvok typu Text Block. Vytvorenie WCF Duplex Services služby je mimo tému tejto publikácie, popis nájdete na msdn **http://msdn.microsoft.com/en-us/library/dd470106(VS.95).aspx** a hotový kód celého riešenia serverovej aj klientskej strany si môžete prevziať z blogu **http://www.davidezordan.net/blog/?p=935**

Microsoft