<http://msdn2.microsoft.com/en-us/library/bb404300.aspx>

Silverlight Technical Articles

Wprowadzenie do Silverlight

Laurence Moroney
Microsoft Corporation

Ostatnia aktualizacja: 30 kwietnia 2007 roku

Artykuł dotyczy następujących produktów i technologii:
   Microsoft Silverlight (technologia znana wcześniej pod nazwą kodową "WPF/E")
   Microsoft Expression Blend

**Streszczenie:**
Dokument zawiera ogólny opis technologii Silverlight i charakteryzuje jej miejsce wśród narzędzi programistycznych przeznaczonych do tworzenia następnej generacji aplikacji internetowych. Objętość dokumentu po wydrukowaniu — około 8 stron.

**Spis treści**

## Co to jest Silverlight?

Silverlight jest przeznaczoną do zastosowań internetowych technologią prezentacji, która z założenia ma działać na różnorodnych platformach systemowych. Technologia ta pozwala na tworzenie bogatych, atrakcyjnych wizualnie interfejsów użytkownika, które mogą pracować w dowolnym środowisku — w przeglądarkach internetowych, na różnych urządzeniach i w różnych systemach operacyjnych (np. Apple Macintosh). Podobnie jak w przypadku WPF (Windows Presentation Foundation) — technologii prezentacyjnej stworzonej dla Microsoft .NET Framework 3.0 — podstawą Silverlight jest język XAML (e**X**tensible **A**pplication **M**arkup **L**anguage).

Dokument zawiera wprowadzenie do Silverlight oraz wskazówki, jak używać zestawu narzędzi Microsoft, obejmującego Microsoft Expression Blend, Microsoft Visual Studio 2005 i XAML, do budowania bogatych graficznie witryn internetowych. Najpierw jednak przyjrzymy się wydarzeniom, które miały wpływ na powstanie technologii Silverlight i miejscu, jakie ta technologia zajmuje wśród innych narzędzi programistycznych.

## Ewolucja sposobów budowy witryn internetowych — przejście do Web.Next

Gdy Tim Berners-Lee z CERN wynalazł współczesny Internet, witryny internetowe były systemem, który umożliwiał przechowywanie dokumentów statycznych w podłączonym do sieci systemie i tworzenie połączeń pomiędzy tymi dokumentami. Przez następne lata wynalazek był rozwijany. Następnym logicznym krokiem były dokumenty „aktywne”, generowane za każdym razem na żądanie, zawierające informacje zależne od czasu ich generacji lub od użytkownika pobierającego dokument. Umożliwiły to technologie takie jak CGI. Z upływem czasu możliwość generowania dokumentów internetowych stała się powszechna, nastąpił dalszy rozwój technologii CGI, a także Java, ASP i ASP.NET.

Technologia ASP.NET była kamieniem milowym na drodze umożliwiania programistom szybkiego tworzenia wysokiej jakości aplikacji internetowych z wykorzystaniem technik tworzenia aplikacji serwerowych i najlepszych w branży narzędzi z rodziny Visual Studio.

Największym utrudnieniem w tworzeniu aplikacji internetowych był interfejs użytkownika — ograniczenia techniczne uniemożliwiały tworzenie bogatych interfejsów użytkownika, które mogłyby choć w części dorównać interfejsom aplikacji klienckich działających w oparciu o dane dostępne lokalnie.

Obiekt **XMLHttpRequest**, udostępniony w 2000 roku przez firmę Microsoft jako moduł przeglądarki Internet Explorer 5, stał się podstawą technologii AJAX (Asynchronous JavaScript and XML — asynchroniczny JavaScript i XML), która umożliwiła bardziej dynamiczne reagowanie aplikacji internetowych na działania użytkownika czy odświeżanie niewielkich fragmentów strony internetowej bez potrzeby ponownego pobierania treści całej strony. Innowacyjne rozwiązania oparte na AJAX, takie jak serwis mapowy Windows Live Local, o kolejny krok skróciły dystans dzielący aplikacje internetowe od aplikacji działających lokalnie na komputerze użytkownika.

Silverlight to kolejny etap ewolucji prowadzący do umożliwienia projektantom i programistom aplikacji udostępniania swoim klientom bogatych interfejsów użytkownika. Projektant może po prostu zaprojektować interfejs i zapisać pracę w formacie umożliwiającym bezpośrednie wykorzystanie interfejsu w Internecie. Do tej pory projektant tworzył witrynę internetową i interfejs użytkownika z wykorzystaniem narzędzi umożliwiających budowanie rozbudowanych projektów, ale programista musiał dopasować projekt do możliwości zapewnianych przez przeglądarkę internetową. W przypadku Silverlight projektanci mogą zbudować interfejs użytkownika i zapisać go w postaci XAML. Programista może następnie z wykorzystaniem biblioteki uruchomieniowej Silverlight osadzić kod XAML bezpośrednio w stronie internetowej. Projektanci i programiści mogą teraz jeszcze ściślej współpracować w zakresie tworzenia zaawansowanych interfejsów użytkownika.

Ponieważ XAML oparty jest na XML, jest formatem tekstowym, stanowiącym łatwą do sprawdzenia przez zaporę internetową definicję rozbudowanego interfejsu użytkownika. Chociaż istnieją inne technologie, które można wykorzystać do budowy interfejsów użytkownika bogatszych niż te tworzone wyłącznie za pomocą DHTML, CSS i JavaScript — na przykład aplety Java, kontrolki ActiveX czy pliki Flash — wszystkie są oparte na binarnych formatach danych. Sprawdzenie bezpieczeństwa plików binarnych jest trudne. Trudne jest też aktualizowanie aplikacji — wprowadzenie jakichkolwiek zmian wymaga przeinstalowania całej aplikacji, nie jest procesem przyjaznym dla użytkownika i może prowadzić do zastojów w aktualizowaniu stron. W przypadku wykorzystania technologii Silverlight, wprowadzenie zmiany w interfejsie użytkownika sprowadza się do zapisania nowego pliku XAML na serwerze. Gdy użytkownik po raz kolejny otworzy stronę, interfejs użytkownika zostanie zaktualizowany bez żadnej reinstalacji.

Sercem technologii Silverlight jest moduł rozszerzający funkcjonalność przeglądarki o możliwość renderowania kodu XAML i wyświetlania wyników w oknie przeglądarki. Moduł ten jest niewielki (wielkość pliku do pobrania to około 2 MB) i może zostać zainstalowany gdy użytkownik otworzy witrynę zawierającą treść opartą na Silverlight. Struktura strony XAML jest dostępna z zewnątrz modułu z poziomu kodu JavaScript, możliwa jest więc interakcja z treścią strony, w której osadzono moduł. Programiści mogą wykorzystać JavaScript do tworzenia procedur obsługi zdarzeń lub modyfikowania zawartości strony XAML.

Dość już tych rozważań teoretycznych! Przejdźmy do części praktycznej i przyjrzymy się naszemu pierwszemu projektowi Silverlight.

## Utworzenie prostej aplikacji Silverlight

Rozpoczniemy od utworzenia bardzo prostej aplikacji XAML w Silverlight za pomocą narzędzia Microsoft Expression Blend. Aby utworzyć w Blend nową aplikację Silverlight, należy z menu **File** wybrać opcję **New project**. Pojawi się okno nowego projektu (patrz ilustracja 1).



**Ilustracja 1. Utworzenie nowego projektu Silverlight w narzędziu Expression Blend**

Kliknięcie przycisku **OK** spowoduje utworzenie nowego projektu. Projekt ten będzie zawierał domyślną stronę HTML, plik z kodem schowanym (code-behind) JavaScript dla strony HTML, dokument XAML, plik z kodem schowanym JavaScript dla dokumentu XAML oraz plik Silverlight.js.

Plik Silverlight.js zawiera kod niezbędny do pobrania kontrolki Silverlight i utworzenia jej instancji. Plik ten dystrybuowany jest jako część zestawu Silverlight SDK.

Plik Default.html jest standardową stroną HTML. Zawiera trzy referencje do plików z kodem JavaScript: Silverlight.js, Default.html.js (zawierającego specyficzny dla aplikacji kod, niezbędny do utworzenia instancji kontrolki Silverlight) oraz Scene.xaml.js (zawierającego procedury obsługi zdarzeń zdefiniowanych w kodzie XAML).

Taka struktura plików ma na celu odseparowanie strony (default.html) od kodu tworzącego instancję kontrolki (default.html.js), projektu (Scene.xaml) i kodu obsługi zdarzeń (Scene.xaml.js). Ponownie odkładamy teorię na bok i przechodzimy do tworzenia prostej aplikacji.

## Utworzenie interfejsu użytkownika dla odtwarzacza wideo

Dodajmy do projektu jakiś plik wideo. W tym celu kliknij prawym przyciskiem myszy plik projektu w oknie Project Files (okno to znajduje się w prawej górnej części ekranu), a następnie wybierz polecenie **Add Existing Item...**.

Po dodaniu do projektu pliku WMV, plik ten pojawi się w oknie Project Explorer, a do sceny zostanie dodany nowy element Media.



**Ilustracja 2. Dodanie elementu Media do sceny XAML**

Możesz teraz uruchomić projekt. Spowoduje to uruchomienie przeglądarki, która odtworzy plik wideo.

Można wyłączyć automatyczne odtwarzanie pliku — wystarczy odpowiednio zmodyfikować kod XAML. W prawej części okna z projektem XAML widoczne są dwie zakładki: **Design** oraz **XAML**. Gdy klikniemy zakładkę XAML, okno powinno przełączyć się w widok kodu (patrz ilustracja 3). Zmodyfikuj kod XAML elementu **MediaElement**, dodając atrybut **AutoPlay=False**.



**Ilustracja 3. Edycja kodu XAML w edytorze XAML**

Jeśli teraz uruchomisz aplikację, kontrolka Silverlight wyświetli pierwszą klatkę pliku wideo, ale nie odtworzy go.

**Dodanie kontrolek do odtwarzacza wideo**

Dodaj do aplikacji dwa bloki tekstowe, nadaj im etykiety **Play** oraz **Stop** i nazwij je odpowiednio: **txtPlay** oraz **txtStop**. Gdy to zrobisz, kod XAML będzie wyglądał podobnie, jak ten przedstawiony poniżej:

Skopiuj kod

<Canvas

 xmlns="http://schemas.microsoft.com/client/2007"

 xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

 Width="640" Height="480"

 Background="White"

 >

<MediaElement AutoPlay="False" x:Name="Movie\_wmv" Width="320" Height="240" Canvas.Left="128" Canvas.Top="56" Source="Movie.wmv" Stretch="Fill"/>

<TextBlock x:Name="txtPlay" Width="72" Height="24" Canvas.Left="136" Canvas.Top="336" Text="Play" TextWrapping="Wrap"/>

 <TextBlock x:Name="txtStop" Width="80" Height="24" Canvas.Left="136" Canvas.Top="368" Text="Stop" TextWrapping="Wrap"/>

</Canvas>

Teraz dodaj do tych bloków tekstowych deklaracje procedur obsługi zdarzeń. Aby zadeklarować procedurę obsługi zdarzenia kliknięcia myszą, należy użyć atrybutu **MouseLeftButtonDown**. Dla bloku tekstowego **txtPlay** zadeklaruj procedurę **DoPlay**, a dla bloku **txtStop** procedurę **DoStop**. Kod XAML z deklaracjami procedur obsługi zdarzeń powinien wyglądać następująco:

Skopiuj kod

<TextBlock x:Name="txtPlay" Width="72" Height="24" Canvas.Left="136"

 Canvas.Top="336" Text="Play" TextWrapping="Wrap"

 **MouseLeftButtonDown="javascript:DoPlay"**/>

<TextBlock x:Name="txtStop" Width="80" Height="24" Canvas.Left="136"

 Canvas.Top="368" Text="Stop" TextWrapping="Wrap"

 **MouseLeftButtonDown="javascript:DoStop"**/>

Teraz, gdy użytkownik aplikacji kliknie jeden z bloków tekstowych, aplikacja zgłosi wystąpienie zdarzenia, które można przechwycić i obsłużyć za pomocą funkcji JavaScript.

**Obsługa zdarzeń w JavaScript**

Wraz z szablonem został utworzony plik Scene.xaml.js, który można wykorzystać do przechwytywania i obsługi zdarzeń za pomocą JavaScript. W poprzedniej sekcji zadeklarowaliśmy w kodzie XAML procedury obsługi zdarzeń **DoPlay** i **DoStop**, a teraz możemy je zaimplementować. Kod procedur przedstawiono poniżej:

Skopiuj kod

function DoPlay(sender, eventArgs)

{

 var theHost = document.getElementById("SilverlightControl");

 var theMedia = theHost.content.findName("Movie\_wmv");

 theMedia.Play();

}

function DoStop(sender, eventArgs)

{

 var theHost = document.getElementById("SilverlightControl");

 var theMedia = theHost.content.findName("Movie\_wmv");

 theMedia.Stop();

}

W tym przykładzie kontrolka Silverlight nosi nazwę SilverlightControl. Referencja do tej kontrolki przechowywana jest w zmiennej JavaScript o nazwie theHost. Zmienna służy do odnalezienia elementu **Media**, który w tym przykładzie ma nazwę Movie\_wmv. Element został utworzony automatycznie w momencie, gdy do projektu dodaliśmy plik z filmem. Nazwa elementu generowana jest na podstawie nazwy pliku z filmem — jeśli dodaliśmy do projektu plik Movie.wmv, element został nazwany Movie\_wmv. Jeśli został użyty plik o innej nazwie, nazwa elementu także jest inna.

Element media udostępnia metody **Play** i **Stop**, które pozwalają na uruchamianie i zatrzymywanie odtwarzania filmu.

Skoro w funkcji uzyskaliśmy już referencję do elementu media, możemy użyć tych metod do uruchamiania i zatrzymywania odtwarzania naszego filmu. Końcową postać naszej przykładowej aplikacji można obejrzeć na ilustracji 4.



**Ilustracja 4. Działająca aplikacja**

Zbudowałeś właśnie swoją pierwszą aplikację Silverlight! Więcej materiałów na temat Silverlight możesz znaleźć w nowym [centrum dla programistów Silverlight](http://msdn.microsoft.com/silverlight/) oraz w witrynie [www.silverlight.net](http://www.silverlight.net).

## W jaki sposób uruchamiana jest kontrolka Silverlight?

Strona HTML wywołuje funkcję **createSilverlight()**, która jest zdefiniowana w kodzie schowanym dla strony HTML (w pliku Default.html.js).

Skopiuj kod

Sys.Silverlight.createObjectEx({

 source: "Scene.xaml",

 parentElement: document.getElementById("SilverlightControlHost"),

 id: "SilverlightControl",

 properties: {

 width: "100%",

 height: "100%",

 version: "0.9"

 },

 events: {

 onLoad: Sys.Silverlight.createDelegate(scene, scene.handleLoad)

 }

 });

Funkcja ma wiele parametrów wywołania — wskazanie kodu XAML, który ma zostać wyświetlony, określenie wyglądu kontrolki Silverlight oraz deklaracje procedur obsługi zdarzeń onLoad i onError.

Parametr **source:** pozwala wskazać kod XAML, który ma zostać wyświetlony wewnątrz kontrolki Silverlight. Może to być plik zewnętrzny (jak w naszym przykładzie) lub znajdujący się w kodzie strony nazwany element **<script>**.

Kontrolka Silverlight na stronie HTML powinna być umieszczona wewnątrz nazwanego elementu **<div>**. Nazwa tego elementu powinna zostać przekazana w parametrze **parentElement:** funkcji inicjującej kontrolkę Silverlight.

Identyfikator kontrolki określany jest za pomocą parametru **id:**.

Atrybuty fizyczne kontrolki, takie jak wysokość i szerokość oraz wersja, są przekazywane w postaci tablicy w parametrze **properties:**. Listę wszystkich właściwości można znaleźć w [dokumentacji Silverlight SDK](http://msdn2.microsoft.com/en-us/library/bb188743.aspx).

## Podsumowanie

W dokumencie zamieszczono ogólny opis technologii Silverlight i scharakteryzowano jej miejsce wśród narzędzi programistycznych przeznaczonych do tworzenia następnej generacji aplikacji internetowych. Pokazaliśmy zastosowanie XAML jako wspólnego języka pozwalającego zapisać wizję projektanta, dodać logikę utworzoną przez programistę i dostarczyć gotową aplikację do użytkownika. Przedstawiliśmy narzędzie Expression Blend i sposób wykorzystania go do budowy interfejsu użytkownika strony internetowej oraz sposób programowania interfejsu użytkownika za pomocą języka JavaScript.

Pokazane w tym artykule możliwości Silverlight to tylko wierzchołek góry lodowej. Technologia ta ma bardzo szeroką funkcjonalność, pozwalającą już dziś tworzyć aplikacje dla Internetu następnej generacji. Wygląda na to, że tworzenie takich aplikacji będzie bardzo przyjemne… A więc do dzieła!