Microsoft

Budowanie aplikacji biznesowych
przy użyciu
Windows® Presentation Foundation
i wzorca MVVM

Raffaele Garofalo

Przekład:
Jakub Niedźwiedź

APN Promise
Warszawa 2011

Budowanie aplikacji biznesowych przy użyciu Windows® Presentation Foundation i wzorca MVVM

Authorized Polish translation of English edition of
Building Enterprise Applications with Windows® Presentation Foundation and the Model View ViewModel Pattern
ISBN: 978-0-735-65092-3

APN PROMISE SA, ul. Kryniczna 2, 03-934 Warszawa
tel. +48 22 35 51 600, fax +48 22 35 51 699
e-mail: mspress@promise.pl

ISBN: 978-83-7541-078-5
Przekład: Jakub Niedźwiedź
Redakcja: Marek Włodarz
Korekta: Ewa Swędrowska
Skład i łamanie: MAWart Marek Włodarz

Spis treści

Wstęp

Windows Presentation Framework (WPF), Silverlight i Windows Phone 7 są najnowszymi technologiami budowania elastycznych interfejsów użytkownika (UI) dla aplikacji tworzonych przy użyciu narzędzi firmy Microsoft. Wszystkie te trzy technologie opierają się na języku znaczników XAML przy opisywaniu elementów i układu interfejsu użytkownika, a aplikacje dla wszystkich tych trzech platform można programować przy użyciu najczęściej stosowanych języków Microsoft .NET Framework: Visual C# lub Visual Basic .NET. Będąc programistą .NET planującym stworzenie nowej aplikacji biznesowej przy użyciu .NET Framework warto rozważyć zastosowanie którejś z tych technologii do utworzenia interfejsu użytkownika. Planując zbudowanie aplikacji w oparciu o jedną z tych technologii warto też poważnie rozważyć nauczenie się i zastosowanie wzorca prezentacyjnego Model View ViewModel (MVVM), który jest wzorcem projektowym stworzonym specjalnie dla tych technologii.

Tego właśnie dotyczy niniejsza książka. Można by się zastanawiać, po co kolejna książka dotycząca WPF? Po zapoznaniu się ze spisem treści można by z kolei pomyśleć, po co kolejna książka na temat wzorców projektowych?

Aby odpowiedzieć na te pytania, zacznę od stwierdzenia, że od dawna zauważyłem, iż programiści nie szukają „Biblii wzorców" ani „Biblii opisującej tworzenie układu aplikacji", chcieliby natomiast mieć dostęp do prostej książki prowadzącej ich przez etapy opracowywania rzeczywistej, ale prostej aplikacji, która *stosuje* i *wyjaśnia* wzorce – a jednocześnie może być ponownie wykorzystana w przyszłych projektach jako „szablon" dla kolejnych aplikacji.

WPF i Silverlight są młodymi technologiami i procentowy udział programistów przechodzących na te nowe sposoby projektowania interfejsu użytkownika jest nadal niewielki. Jest kilka powodów takiego stanu rzeczy. Po pierwsze, krzywa uczenia się jest dosyć stroma. Jeśli ktoś przyzwyczaił się do technologii Windows Forms, Java Swing lub Delphi, to sposób projektowania i organizowania struktury aplikacji przy użyciu XAML i WPF jest znacząco różny – w istocie nazwałbym go „rewolucyjnym".

W przeszłości korzystałem z dobrze znanych wzorców do budowania aplikacji, w tym z wzorca Model View Presenter w przypadku aplikacji Windows Forms oraz wzorca Model View Controller w przypadku aplikacji ASP.NET. Jednak w przypadku WPF te dwa podejścia są przestarzałe, ponieważ nie mogą korzystać z wydajnych mechanizmów zapewnianych przez XAML. Oczywiście nadal można korzystać z mechanizmów wiązania w WPF stosując wzorzec Model View Presenter, ale kosztem zbyt dużego wysiłku. Na szczęście wzorzec MVVM zapewnia dobrą alternatywę.

Firma Microsoft we współpracy z kilkoma architektami przetworzyła pierwotny model prezentacyjny, który wiele lat temu zaproponował Martin Fowler. Ta wersja (nazwana wzorcem Model View ViewModel) jest idealnym podejściem w przypadku WPF i Silverlight, ponieważ została zaprojektowana specjalnie dla tych technologii! Niestety podobnie jak XAML, MVVM jest dosyć nową technologią, więc obecnie nie ma zbyt wielu informacji na temat jej implementowania. Jest kilku blogerów stosujących podejście MVVM i piszących na ten temat; są też osoby związane z budowaniem narzędzi przeznaczonych dla MVVM. Niemal wszystko jest nadal w fazie eksperymentów i istnieje niewiele naprawdę konkretnych przykładów.

Stąd wziął się pomysł na książkę dotyczącą budowania aplikacji biznesowej przy użyciu MVVM. W miarę czytania tej książki zobaczymy przykłady, które pokażą, jak zbudować prosty program do obsługi relacji z klientami (CRM) przy pomocy WPF 4, Silverlight 4 i wzorca MVVM. Ta książka będzie przewodnikiem przez cały proces architektoniczny ilustrując odpowiednie podejście do zastosowania MVVM. Skorzystamy też z kilku innych nowych technologii dostarczanych przez platformę Microsoft .NET 4, takich jak Managed Extensions, Windows Workflow Foundation 4 i oczywiście Entity Framework.

Najpierw pokażemy narzędzia. Następnie przejdziemy do budowania aplikacji CRM zaczynając od modelu domenowego, stosując prostą technikę zachowywania danych w relacyjnej bazie danych przy pomocy dwóch najpopularniejszych systemów Object--Relational Mapper (O/RM) dostępnych dla .NET: Entity Framework i NHibernate. Następnie zobaczymy, jak dodać większą elastyczność przy pomocy platformy MEF.

Wreszcie nauczymy się stosować w tym modelu logikę biznesową i sprawdzanie poprawności danych w sposób spełniający wymagania wzorca MVVM. W tej fazie przyjrzymy się też Windows Workflow Foundation (WF) 4.0, stworzonemu przez Microsoft wydajnemu, nowemu silnikowi przepływów zadań oraz zbadamy kroki wymagane do zbudowania prostego silnika przepływów zadań.

Pozostałe rozdziały skupiają się na MVVM. Istnieją cztery główne pojęcia, które trzeba poznać, aby prawidłowo korzystać z MVVM: *polecenia*, *szablon*, *silnik wiązań*, oraz *orkiestracja* wszystkiego razem. Podczas tego procesu odwiedzimy wszystkie warstwy wymagane do ukończenia klasycznej aplikacji biznesowej, a co ważniejsze, będziemy w stanie ponownie wykorzystać opisane tu części jako szablony do budowania przyszłych aplikacji. Istnieją oczywiście pewne różnice pomiędzy WPF a Silverlight, więc niniejsza książka spróbuje omówić je w miarę możliwości.

Na koniec krótko przyjrzymy się dostępnym zestawom narzędzi MVVM, takim jak PRISM, który jest złożoną platformą aplikacyjną dla WPF i Silverlight. Pomoże nam to w określeniu, kiedy i jak powinniśmy korzystać z każdego z nich podczas procesu budowania niewielkiej i elastycznej platformy MVVM.

Kluczowym celem tej książki jest zapewnienie pełnego przewodnika opisującego krok po kroku korzystanie z WPF/Silverlight w połączeniu z MVVM przy tworzeniu ogólnego kodu, który będzie można ponownie wykorzystać w przyszłości.

Budowanie aplikacji biznesowych przy pomocy Windows Presentation Foundation i wzorca Model View ViewModel zapewnia nie tylko solidną analizę sposobu działania wzorca MVVM i jego zastosowania w połączeniu z WPF i Silverlight, ale także oferuje wyczerpujący przewodnik po budowaniu warstwowych aplikacji przy użyciu najczęściej stosowanych technik. Książka ta celowo nie pokazuje *całego* kodu związanego z danym projektem; skupia się natomiast bardziej na zasadach i wzorcach, które programiści powinni stosować przy tworzeniu dobrze zorganizowanych i łatwych w utrzymaniu aplikacji biznesowych.

Ta książka analizuje każdą warstwę składającą się na aplikację biznesową począwszy od modelu domenowego (znanego też jako warstwa biznesowa), poprzez warstwę danych (omawiając przy tym Entity Framework i NHibernate), a kończąc rozdziałem poświęconym regułom biznesowym i Windows Workflow Foundation. Oczywiście pojawi się też rozdział poświęcony wzorcowi MVVM.

Oprócz wzorców i praktyk wyjaśnionych w tej książce, rozdział 7 zawiera przydatną listę platform i wtyczek z otwartym kodem źródłowym stosowanych przez innych członków społeczności .NET do budowania aplikacji implementujących wzorzec MVVM w technologiach WPF lub Silverlight.

Kto powinien przeczytać tę książkę

Ta książka jest przeznaczona dla każdego programisty lub architekta oprogramowania .NET, który chce nauczyć się, jak budować aplikacje biznesowe korzystając z dobrze znanych biznesowych wzorców architektonicznych, w tym wzorca MVVM. Czytelnicy powinni mieć już solidnie opanowane podstawowe zagadnienia związane z programowaniem, znać ogólny cel i zastosowanie wzorców, a także oczywiście znać technologie WPF, Silverlight lub Windows Phone 7. Choć książka ta porusza wszystkie te zagadnienia, to nie próbuje nauczyć podstaw programowania lub zasad stosowania wzorców. Jest natomiast przeznaczona dla programistów i architektów, którzy budowali już aplikacje .NET i chcą zająć się projektowaniem i budowaniem biznesowych aplikacji przy pomocy .NET.

W szczególności książka ta przeznaczona jest dla programistów WPF lub Silverlight, którzy mają już doświadczenie w jednej lub obu tych technologiach, ale którzy nie wiedzą jeszcze, jak implementować wzorzec MVVM – albo dla programistów, którzy w jakimś stopniu zaznajomili się już z MVVM i chcą opanować techniki efektywnego zastosowania wzorca MVVM. W tym celu trzeba mieć pewną podstawową wiedzę na temat WPF i Silverlight; jeśli jej komuś brakuje, to przed przeczytaniem tej książki zalecam zapoznanie się z tematami poleceń wytyczanych, wiązania danych i stylów.

Założenia

Koncentrując się na wzorcach projektowych, architekturach oprogramowania oraz zwinnych technikach i metodologiach programowania (agile) książka ta zakłada u czytelnika podstawowe umiejętności tworzenia aplikacji WPF lub Silverlight przy pomocy .NET Framework i Visual Studio. Ponadto zakłada, że czytelnik tworzył już aplikacje łączące się z bazą danych i posiadające interaktywny interfejs użytkownika.

Cały kod przykładowy zapewniony w tej książce został utworzony przy pomocy języka Visual C# dostępnego w .NET Framework 4. Trzeba dobrze rozumieć język C#, aby śledzić i wykorzystywać ten kod. Książka szeroko wykorzystuje zarówno WPF, jak i Silverlight, więc należy mieć co najmniej podstawową wiedzę na temat tych dwóch technologii (a także solidne podstawy języka znaczników XAML – książka ta wykorzystuje w przykładach kod XAML).

Organizacja tej książki

Ta książka została opracowana w taki sposób, że każdy rozdział skupia się na określonym zagadnieniu. Pierwszy rozdział „Wprowadzenie do aplikacji biznesowych i wzorca Model View ViewModel" jest ogólnym wprowadzeniem do aplikacji biznesowych, ich składników i ich struktury. Rozdział 2 „Wzorce projektowe" pokazuje pełny przegląd wszystkich dobrze znanych wzorców projektowych i wzorców architektonicznych używanych do programowania aplikacji biznesowych, a zwłaszcza do projektowania luźno powiązanych składników. Rozdział 3 „Model domenowy" jest wprowadzeniem do modelu domenowego i projektowania sterowanego domeną (DDD – Domain-Driven Design). Ilustruje, jak osiągać cele projektowe DDD i jak unikać powszechnych błędów, które zwykle występują przy budowaniu aplikacji DDD. Rozdział 4 „Warstwa dostępu do danych" koncentruje się na warstwie dostępu do danych (DAL – Data Access Layer) i sposobach jej budowania przy użyciu systemu O/RM, takiego jak Entity Framework i/lub NHibernate. Rozdział 5 „Warstwa biznesowa" skupia swoją uwagę na projektowaniu i budowie warstwy logiki biznesowej (BLL – Business Logic Layer) w tym dogłębnie omawia reguły biznesowe, silniki reguł biznesowych i projektowanie architektury zorientowanej na usługi (SOA – Service-Oriented Architecture). Wreszcie rozdział 6, „Warstwa interfejsu użytkownika w MVVM" omawia dogłębnie MVVM, natomiast rozdział 7 „Platformy i zestawy narzędzi MVVM" wymienia dostępne platformy programowe i narzędzia, które mogą być przydatne podczas budowania aplikacji biznesowych przy pomocy MVVM.

Od czego najlepiej zacząć czytanie tej książki

Rozdziały tej książki omawiają różne aspekty budowania aplikacji biznesowej. Poza pierwszymi dwoma rozdziałami, które stanowią ogólny przegląd technik używanych w tej książce, każdy rozdział skupia się na określonej warstwie aplikacji biznesowej.

Poniższa tabela może pomóc w ustaleniu, gdzie najlepiej sięgnąć, jeśli ktoś planuje skupić się tylko na określonej warstwie.

Jeśli ktoś	To powinien:
Jest nowicjuszem w programowaniu aplikacji biznesowych i aplikacji wielowarstwowych	Przeczytać całą książkę i poeksperymentować z rozwiązaniami użytymi jako przykłady w każdym rozdziale.
Jest zaznajomiony z wzorcami projektowymi i architekturami oprogramowania, ale ich jeszcze dobrze nie opanował	Przejrzeć rozdziały 1 i 2, aby przypomnieć sobie podstawowe pojęcia. Następnie uważnie przeczytać pozostałe rozdziały stosując zasady napotkane w każdym rozdziale w swoich codziennych zadaniach.
Jest zainteresowany *jedynie* określoną warstwą, taką jak DAL lub BLL	Uważnie przeczytać określony rozdział, który omawia daną warstwę będącą przedmiotem zainteresowania. Aby jednak ustalić kontekst, należy też przejrzeć pozostałe rozdziały.
Jest zainteresowany jedynie wzorcem MVVM	Przeczytać rozdziały 1 i 2, aby utrwalić wiedzę na temat wzorców projektowych i wzorców prezentacyjnych, a następnie uważnie przeczytać rozdziały 6 i 7.

Konwencje i zasady w tej książce

Ta książka przedstawia informacje korzystając z konwencji zaprojektowanych tak, aby informacje te były czytelne i łatwe w odbiorze.

W większości przypadków książka ta zawiera osobne ćwiczenia dla programistów Visual Basic i programistów Visual C#. Można pominąć ćwiczenia, które nie dotyczą wybranego języka.

- Elementy w ramkach z etykietami, takimi jak „Uwaga”, zapewniają dodatkowe informacje lub alternatywne metody wykonania danego kroku.
- Tekst, który należy wpisać (oprócz bloków kodu), wyróżniony jest pogrubieniem.
- Znak plus (+) pomiędzy nazwami dwóch klawiszy oznacza, że trzeba nacisnąć te klawisze jednocześnie. Na przykład „Alt+Tab” oznacza, że trzeba przytrzymać wciśnięty klawisz Alt podczas naciskania klawisza Tab.
- Pionowa kreska pomiędzy dwoma lub więcej elementami menu (na przykład File | Close) oznacza, że należy wybrać pierwsze menu lub element menu, potem następne, i tak dalej.

Wymagania systemowe

Do pracy z kodem i przykładami z tej książki potrzebny będzie następujący sprzęt i oprogramowanie:

- Dowolny z następujących systemów operacyjnych: Windows XP z Service Pack 3 (oprócz Starter Edition), Windows Vista z Service Pack 2 (oprócz Starter Edition), Windows 7, Windows Server 2003 z Service Pack 2, Windows Server 2003 R2, Windows Server 2008 z Service Pack 2 lub Windows Server 2008 R2.
- Visual Studio 2010, dowolne wydanie (kilka dodatkowych elementów do pobrania osobno może być wymaganych w przypadku korzystania z produktów Express Edition).
- SQL Server 2008 Express Edition lub wyższa wersja (wydanie 2008 lub R2) z narzędziem SQL Server Management Studio 2008 Express lub wyższą wersją (zawarte w Visual Studio, wydania Express Edition wymagają oddzielnego pobrania).
- Procesor o prędkości 1,6 GHz lub większej (zalecane 2 GHz).
- 1 GB (system 32-bitowy) lub 2 GB (system 64-bitowy) pamięci RAM (należy dodać 512 MB w przypadku uruchamiania w maszynie wirtualnej lub w przypadku wydań SQL Server Express Edition; więcej w przypadku bardziej zaawansowanych wydań SQL Server).
- 3,5 GB dostępnego miejsca na dysku twardym.
- Napęd dysku twardego o prędkości 5400 RPM.
- Karta grafiki zgodna z DirectX 9 działająca z rozdzielczością ekranu 1024 × 768 lub wyższą.
- Napęd DVD-ROM (w przypadku instalowania Visual Studio z płyty DVD).
- Połączenie internetowe do pobierania oprogramowania lub przykładów kodu.

W zależności od konfiguracji Windows, konieczne mogą być uprawnienia administratora lokalnego do zainstalowania lub skonfigurowania produktów Visual Studio 2010 i SQL Server 2008.

Przykłady kodu

Większość rozdziałów w tej książce zawiera ćwiczenia, które pozwalają w interaktywny sposób wypróbować nowy materiał poznany w głównym tekście. Wszystkie projekty przykładowe w postaci przed wykonaniem i po wykonaniu ćwiczenia są dostępne do pobrania ze strony tej książki na witrynie wydawnictwa O'Reilly Media:

http://oreilly.com/catalog/9780735650923/

Wystarczy kliknąć łącze Examples na tej stronie. Gdy pojawi się lista plików, należy zlokalizować i pobrać plik MvvmCrm.zip.

Uwaga Aby skorzystać z przykładów kodu, w swoim systemie trzeba mieć zainstalowane programy Visual Studio 2010 i SQL Server 2008. Poniższe instrukcje wykorzystują SQL Server Management Studio 2008 do skonfigurowania przykładowej bazy danych używanej w przykładowych ćwiczeniach. Należy zainstalować najnowsze pakiety serwisowe dla każdego produktu, jeśli są dostępne.

Instalowanie przykładów kodu

Aby zainstalować przykłady kodu na swoim komputerze, należy:

1. Rozpakować plik MvvmCrm.zip pobrany ze strony *http://oreilly.com/catalog/9780735650923/*.
2. Przejrzeć wyświetlaną umowę licencyjną dla użytkownika końcowego. Zaakceptować warunki umowy, a następnie kliknąć Next.

Uwaga Jeśli umowa licencyjna się nie pojawi, to można uzyskać do niej dostęp z poziomu tej samej strony WWW, z której pobrano plik MvvmCrm.zip.

Korzystanie z przykładów kodu

Struktura rozwiązania Visual Studio dostarczanego z tą książką jest podzielona na sześć różnych projektów, w których każdy projekt składa się z pełnego kodu źródłowego dla związanego z nim rozdziału w książce. Cała aplikacja stanowi program CRM opracowany w WPF.

Podziękowania

Gdy ktoś jest jedynym autorem książki, to jest trwale związany z tym, co daje ona innym; w istocie jest to jeden z powodów, dla których wiele osób chce napisać książkę. Ale nawet wyłączny autor nie jest jedyną osobą odpowiedzialną za powstanie książki. Chciałbym podziękować wszystkim osobom, które pomogły mi w napisaniu i wydaniu tej książki, ponieważ bez nich pozostałaby ona jedynie pomysłem.

To moja pierwsza książka. Pisanie jej było dla mnie wspaniałą przygodą i mam nadzieję, że jest to początek czegoś nowego, do czego czuję się naturalnie predysponowany. Nie byłbym w stanie napisać tej książki bez ogromnej pomocy mojej wspaniałej żony Deborah. Pisanie książki wymaga czasu, a pracuję na pełny etat w firmie ubezpieczeniowej, więc kilka wolnych godzin spośród dni spędzonych na pisaniu książki i wyszukiwaniu dokumentacji (co zajęło pełne sześć miesięcy) zostało zabranych z naszego wspólnego czasu. Bez tak wspaniałej i wyrozumiałej żony prawdopodobnie nie byłbym w stanie poświęcić tego czasu. Wiele razy, gdy byłem bliski zrezygnowania z ukończenia tej książki – ze względu na skomplikowanie i ogrom informacji – stanowczo nakłaniała mnie do ukończenia tej pracy, jak doskonały menedżer projektu! Dziękuję, Debbie!

Chciałbym też podziękować Russellowi Jones'owi, mojemu redaktorowi i głównej osobie z wydawnictwa kontaktującej się ze mną w sprawie tej książki. Jest on jedyną osobą, która wierzyła we mnie od początku i zaangażowała się w przekonanie Microsoft Press do tego projektu. Zawsze będę mu za to wdzięczny. Pomógł mi też w ukończeniu tej pracy na czas i organizował cały projekt.

Na koniec chcę podziękować Davidowi Hillowi, który jest recenzentem technicznym tej książki i moim mentorem. David jest pracownikiem w zespole patterns & practices w firmie Microsoft. Jego nieocenione uwagi podczas pisania tej książki pomogły mi znacznie poprawić moje ogólne pojęcie na temat wzorców prezentacyjnych oraz poprawnie skonstruować architekturę aplikacji biznesowej. David jest elastyczny i skromny. Mam niezwykłe szczęście, że miałem okazję z nim pracować i mam nadzieję na kolejną współpracę w przyszłości.

Dziękuję Wam wszystkim!

Errata i wsparcie dla tej książki

Podjęliśmy wszelkie starania w celu zapewnienia poprawności tej książki i dołączonej do niej zawartości. Jeśli ktoś znajdzie błąd, prosimy o zgłoszenie go na naszej witrynie Microsoft Press w serwisie oreilly.com:

1. Przejdź do *http://microsoftpress.oreilly.com*.
2. W polu Search wpisz numer ISBN lub tytuł książki.
3. Wybierz książkę z wyników wyszukiwania.
4. Na stronie katalogowej książki, pod rysunkiem okładki pojawi się lista łączy.
5. Kliknij View/Submit Errata.

Dodatkowe informacje i usługi związane z książką można znaleźć na jej stronie katalogowej. Jeśli potrzebne jest dodatkowe wsparcie, wystarczy wysłać e-maila do Microsoft Press Book Support pod adres *mspinput@microsoft.com*.

Należy zwrócić uwagę, że wsparcie techniczne dla produktów nie jest oferowane pod powyższymi adresami.

Chcemy poznać opinie Czytelników

W wydawnictwie Microsoft Press zadowolenie Czytelnika jest naszym głównym priorytetem, a informacja zwrotna jest cennym zasobem. Swoje opinie na temat tej książki można zostawiać pod adresem:

http://www.microsoft.com/learning/booksurvey

Ankieta jest krótka i przeczytamy każdy zgłoszony komentarz i pomysł. Z góry dziękujemy za wszelkie uwagi!

Kontakt

Możemy pozostać w kontakcie! Jesteśmy dostępni w serwisie Twitter pod adresem *http://twitter.com/MicrosoftPress*.

ROZDZIAŁ 1

Wprowadzenie do aplikacji biznesowych i wzorca Model View ViewModel

Po zakończeniu tego rozdziału będziemy w stanie:

- Zidentyfikować aplikację biznesową.
- Wybrać odpowiednią technologię do utworzenia aplikacji biznesowej.

Wzorzec Model View ViewModel

Wzorzec Model View ViewModel (MVVM) został przedstawiony przez Johna Gossmana (architekta oprogramowania w firmie Microsoft w dziedzinie technologii Windows Presentation Foundation i Silverlight) na jego blogu w 2005 roku. MVVM jest wyspecjalizowaną odmianą wzorca Presentation Model (PM), który został przedstawiony w roku 2004 przez Martina Fowlera.

Jednym z głównych celów wzorca PM jest oddzielenie abstrakcyjnego widoku (View) – widocznego interfejsu użytkownika – od logiki prezentacyjnej, aby łatwiej było testować interfejs użytkownika. Dodatkowymi celami może być umożliwienie ponownego wykorzystywania logiki prezentacyjnej w różnych interfejsach użytkownika i różnych technologiach interfejsów użytkownika, co jest ograniczane przez powiązania pomiędzy interfejsem użytkownika a innym kodem oraz pozwala projektantom interfejsów użytkownika na pracę w bardziej niezależny sposób. MVVM jest wyspecjalizowaną interpretacją wzorca PM zaprojektowaną pod kątem wymagań Windows Presentation Foundation (WPF) i Silverlight.

Strukturalnie aplikacja MVVM składa się przede wszystkim z trzech głównych składników: modelu (*Model*), widoku (*View*) i modelu widoku (*ViewModel*).

- Model jest elementem, który reprezentuje jakieś pojęcie biznesowe; może to być cokolwiek od prostego przedstawienia klienta do skomplikowanego opisu handlu na giełdzie.
- Widok jest graficzną kontrolką lub zbiorem kontrolek odpowiedzialnych za przedstawienie danych modelu na ekranie. Widok może być oknem WPF, stroną Silverlight lub po prostu kontrolką szablonu danych w XAML.

- Model widoku odpowiada za „magię" działającą w tle. Model widoku zawiera logikę interfejsu użytkownika, polecenia, zdarzenia i odwołania do modelu. W MVVM model widoku nie odpowiada za aktualizowanie danych wyświetlanych w interfejsie użytkownika – dzięki świetnemu silnikowi wiązania danych zapewnianemu przez WPF i Silverlight model widoku nie musi tego robić. Dzieje się tak dlatego, że widok obserwuje model widoku, więc jak tylko model widoku ulega zmianie, to interfejs użytkownika się aktualizuje. Aby to było możliwe, model widoku musi implementować interfejs *INotifyPropertyChanged* i wyzwalać zdarzenie *PropertyChanged*.

Pierwotnie tylko technologia WPF była wystarczająco zaawansowana, żeby spełniać wymagania wzorca MVVM. W wersji Silverlight 2 mieliśmy opcję implementowania MVVM, ale było to trudniejsze niż implementowanie MVVM w WPF. Obecnie w wersji Silverlight 4 możemy stosować MVVM zarówno w WPF, jak i Silverlight w taki sam sposób korzystając z możliwości wiązania danych, poleceń, zachowań i szablonów danych.

Gdy stosujemy wzorzec MVVM, musimy specjalnie zadbać o model widoku. Ponieważ ma on tak dużo obowiązków, to łatwo jest tworzyć nieuporządkowane rozwiązania, w których będziemy ponownie pisać ten sam kod. Jednak przy zastosowaniu odpowiedniego podejścia wzorzec MVVM może nam zaoszczędzić czas i pomóc w utworzeniu interfejsu użytkownika, który będzie łatwy do testowania i utrzymania. Oczywiście w celu prawidłowego wykorzystania MVVM trzeba koniecznie opanować język XAML i jego strukturę przy budowaniu interfejsu użytkownika. Musimy też wiedzieć, jak działa silnik wiązania XAML, a także jaką strukturę mają obiekty i zachowania poleceń *(ICommand)* oraz szablony danych. Do skutecznego wykorzystania MVVM zarówno w WPF, jak i Silverlight, trzeba też znać różnice pomiędzy WPF a Silverlight.

Ta książka dogłębnie analizuje każdy składnik wzorca MVVM. Na końcu utworzymy prostą aplikację biznesową stosującą MVVM, która będzie mogła być wykorzystywana jako szablon dla dowolnych przyszłych aplikacji MVVM. Jednocześnie zbudujemy niewielką platformę narzędziową MVVM, która będzie działać jako automatyczny składnik do wykorzystania w aplikacjach WPF lub Silverlight upraszczający pisanie aplikacji MVVM. Platforma ta zapewni na przykład podstawową klasę ViewModel, przykładowego brokera komunikatów i inne funkcje wymagane w typowej aplikacji MVVM.

Aplikacje biznesowe

Z mojego doświadczenia najlepszym sposobem nauczenia się nowej technologii jest zbudowanie aplikacji krok po kroku. Aplikacja biznesowa stanowi najlepszy przykład z kilku powodów: jest odpowiednia dla elastycznej technologii interfejsu użytkownika występującej zarówno w WPF, jak i Silverlight; jest otwarta na zastosowanie wzorca MVVM; i jest to typowy rodzaj aplikacji, więc możemy ponownie wykorzystać te przykłady później dla prawdziwych celów biznesowych.

Uwaga Aplikacje biznesowe obsługują działania istotne dla przedsiębiorstwa takie, jak księgowość, zarządzanie łańcuchem dostaw lub planowanie zasobów. Aplikacje biznesowe są zwykle dużymi programami, które zawierają wiele zintegrowanych funkcjonalności i wiążą się z innymi aplikacjami oraz systemami zarządzania baz danych. Są też często nazywane aplikacjami dla przedsiębiorstw.

Aplikacja biznesowa może być dowolną aplikacją istotną dla prowadzenia biznesu: systemem zarządzania relacjami z klientami używanym w biurze, oprogramowaniem księgowym używanym przez departament finansowy do przygotowywania listy płac lub dowolnym innym typem aplikacji biznesowej, która spełnia określone wytyczne i ma określony styl interfejsu użytkownika. Jeśli by się nad tym zastanowić, to takie aplikacje doskonale pasują do pojęcia „szablonu".

Aplikacje biznesowe są najczęściej zamawianymi przez klientów, a przy tym najłatwiejszymi do zaprogramowania. Jednocześnie bywają najtrudniejszymi do zaprojektowania. Wynika to z tego, że chociaż ich *struktura* jest zwykle dość prosta i powtarzalna, to ich *wymagania* często zmieniają się podczas procesu tworzenia oraz w okresie ich użytkowania.

Coraz częściej aplikacje biznesowe wykorzystują interfejsy WWW, co sprawia, że stają się łatwiej dostępne poprzez przeglądarki, łatwiejsze we wdrażaniu i aktualizowaniu oraz pozwalają na realizację scenariuszy biznesowych, które wymagają dostępu do tych samych funkcji przez partnerów biznesowych i klientów. Wykorzystują też osobiste funkcje aplikacyjne, takie jak poczta elektroniczna i książki adresowe.

Aplikacja biznesowa często zmienia się w sposób przyrostowy podczas projektowania. Książka na temat zarządzania projektami, którą jakiś czas temu przeczytałem (dzięki swojemu szefowi), wspominała, że największe wydatki departamentów IT i producentów oprogramowania wiążą się z *utrzymywaniem* istniejącego oprogramowania. Zwykle osoby zaangażowane w projekt programistyczny dowolnego rodzaju uważają, że najdroższą częścią jest faza *programowania* prowadząca do pierwszej edycji programu, ale tak naprawdę dopiero po wydaniu pierwszej wersji pojawiają się prawdziwe kłopoty. Na przykład załóżmy, że tworzymy i sprzedajemy aplikację księgową, w której pierwotnie nie zaprojektowano obsługi wypłat dla pracowników. Po jakimś czasie klienci poprosili o tę nową „funkcję". Jeśli projekt nie jest wystarczająco

elastyczny na przyjmowanie nowych elementów i zmian, to prawdopodobnie stracimy klienta i aplikacja okaże się porażką.

Aplikacja biznesowa świetnie pasuje do technologii WPF/Silverlight i wzorca MVVM, ponieważ skupia się na wszystkich typowych problemach, które mały, średni lub duży zespół napotka podczas różnych faz procesu tworzenia programu, a które można rozwiązać korzystając z tych elastycznych technologii. Niestety książka nie nauczy nas wszystkiego, więc w tej książce nie dowiemy się, jak budować przemysłowe aplikacje CRM albo jak stosować metodologię Scrum w swoim zespole – ale dowiemy się, jak zbudować aplikację biznesową, która implementuje niewielki system do zarządzania klientami korzystając z najnowszych technologii Microsoft.

Wybór odpowiedniej technologii

Ponieważ możemy zbudować aplikację biznesową albo przy pomocy WPF, albo Silverlight, to musimy przeanalizować wymagania projektu, aby ustalić, która technologia jest najbardziej odpowiednia dla tej określonej aplikacji i z których narzędzi chcemy skorzystać do jej zbudowania. Aby odpowiedzieć na te pytania, najpierw zbadamy, jak wybrać technologię pomiędzy Silverlight a WPF, a następnie zbadamy narzędzia, które firma Microsoft obecnie oferuje do projektowania interfejsu użytkownika. Na koniec przejdziemy do analizy typowego układu graficznego aplikacji biznesowej i oczekiwań użytkowników wobec niego.

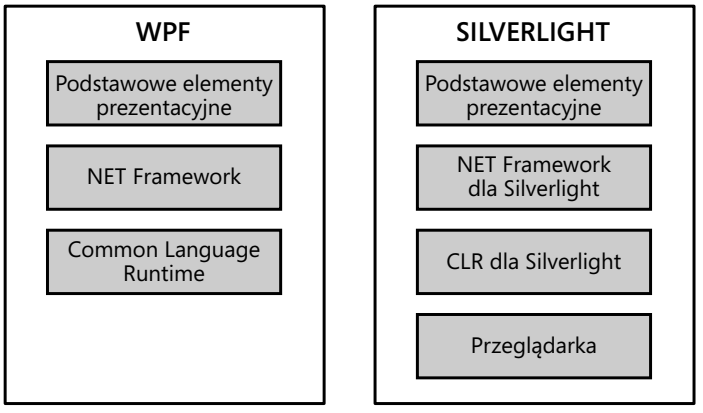
Silverlight czy WPF?

Silverlight i WPF opierają się na tej samej podstawowej technologii: Microsoft .NET Framework. W obu przypadkach budujemy interfejsy użytkownika przy pomocy języka XAML.

Technologia WPF jest następcą Windows Forms, więc została zaprojektowana tak, aby obejmowała pełen zestaw kontrolek interfejsu użytkownika i elementów multimedialnych, przy pomocy których możemy tworzyć bogate i interaktywne aplikacje klienckie dla systemu Windows. Silverlight jest międzyprzeglądarkową, międzyplatformową technologią, która obsługuje bogate aplikacje internetowe. Zasadniczo WPF służy do budowania aplikacji klienckich dla systemu Windows, a Silverlight służy do tworzenia bogatych aplikacji WWW, ale również przeglądarki mogą obsługiwać aplikacje WPF, a aplikacje Silverlight mogą być uruchamiane poza przeglądarką na pulpitach.

Istnieje kompatybilność pomiędzy Silverlight a WPF, ponieważ obie te technologie wykorzystują ten sam język opisu interfejsu użytkownika (XAML), ten sam zestaw składników interfejsu (choć Silverlight wykorzystuje tylko podzbiór tego zestawu), tę samą bazową bibliotekę klas .NET oraz środowisko CLR. Jedyną znaczącą różnicą jest tutaj to, że Silverlight obecnie wykorzystuje inną implementację .NET CLR.

Rysunek 1-1 wyświetla główne różnice pomiędzy tymi dwiema technologiami.



Rysunek 1-1 Przegląd architektury WPF i Silverlight

Ponieważ Silverlight skupia się na odbiorcach korzystających z wielu platform i przeglądarek, to firma Microsoft zmuszona była do zmniejszenia i ograniczenia środowiska uruchomieniowego dla tej technologii. Wniosek z tego jest taki, że najlepiej od początku planować aplikację pod kątem ostatecznej grupy docelowej dla naszej aplikacji biznesowej, ponieważ nie wszystkie funkcje WPF będą dostępne w Silverlight i znacznie trudniej jest przejść później z jednej technologii na drugą.

Oczywiście technologie WPF i Silverlight stają się coraz lepsze z każdym wydaniem, więc istnieje nadzieja, że w przyszłości otrzymamy zunifikowaną platformę, ale na razie ważne jest, aby pamiętać, że grupy docelowe dla tych dwóch technologii są nieco inne.

Uwaga Firma Wintellect we współpracy z firmą Microsoft wydała dokument dostępny pod adresem *http://wpfslguidance.codeplex.com*, który w pełni wyjaśnia różnice pomiędzy tymi dwiema technologiami. Dokument ten ma około 69 stron. Jak można się spodziewać, ta książka nie może obejmować wszystkich tych różnic; dlatego jedynie podkreśla najważniejsze z nich.

Pierwsza różnica dotyczy technologii istotnych dla implementacji MVVM. Silverlight nie implementuje wytyczanych poleceń, wyzwalaczy albo szablonów danych w taki sam sposób jak WPF. Dlatego, aby uzyskać takie samo (lub podobne) zachowanie, musimy implementować niestandardową funkcjonalność w Silverlight. Najpierw jednak słowo ostrzeżenia dotyczące wykorzystania wyzwalaczy w WPF i Silverlight przy implementowaniu wzorca MVVM: nie powinny być intensywnie używane, ponieważ łatwo może się zdarzyć, że będą zawierać logikę prezentacyjną, której nie będzie się dało testować. Logika ta nie jest dostępna w modelu widoku, ale jest udostępniana w widoku przez wyzwalacz.

Silverlight 4 zawiera bogaty zestaw kontrolek, stylów i szablonów, z których jednym jest ciekawy szablon aplikacji biznesowej. Z kolei WPF zawiera mniejszy zestaw narzędzi z kontrolkami.

Z której technologii powinniśmy więc skorzystać – Silverlight, czy WPF? Odpowiedź brzmi: należy dokonać wyboru w oparciu o typ budowanej aplikacji i najbardziej typową grupę docelową dla tej aplikacji. Jeśli na przykład zamierzamy stworzyć aplikację biznesową dla departamentu finansowego, która nie będzie używana poza firmą klienta, to WPF jest odpowiednią technologią. Z drugiej strony, jeśli mamy opracować aplikację CRM, która będzie używana przez klientów i menedżerów korzystających z różnych urządzeń, to lepiej umieścić aplikację w przeglądarce, a więc Silverlight byłby odpowiednią technologią.

Możemy łatwo zbudować dwie warstwy interfejsu użytkownika, jeśli prawidłowo korzystamy z wzorca MVVM: jedną warstwę dla WPF i jedną warstwę dla Silverlight. Obecnie wielu programistów stosuje to podejście z dwoma warstwami interfejsu użytkownika.

Ostateczna grupa docelowa i zadania naszej aplikacji stanowią klucze, które powinny wpływać na wybór zastosowanej technologii. Nie musimy się martwić w tym momencie różnicami w zestawie kontrolek lub interfejsie użytkownika; firma Microsoft wydała zestaw narzędzi projektowych (Microsoft Expression Studio), które mogą obsługiwać cały proces projektowania zarówno na potrzeby WPF, jak i Silverlight.

Narzędzia firmy Microsoft do budowania interfejsu użytkownika

Największym problemem dla programistów, którzy chcą przejść na WPF lub Silverlight jest krzywa uczenia się. Obie te technologie wykorzystują nową specyfikację języka opisu interfejsu użytkownika o nazwie XAML, który jest po prostu deklaratywnym językiem znaczników, podobnie jak HTML lub XML. Oczywiście nie jest łatwo korzystać z tego języka do budowy układów graficznych, gdy nie wiemy, jak działa silnik przetwarzający XAML. Podobnie nie jest łatwo zaimplementować pełne wsparcie narzędzi projektowych dla podejścia WYSIWYG. XAML jest bardzo elastycznym językiem znaczników z kilkoma ograniczeniami. Na przykład możemy umieścić kontrolkę *DataGrid* w przycisku typu Button – nawet jeśli nie miałoby to sensu z punktu widzenia użyteczności. Taka elastyczność może doprowadzać silniki graficzne do szału.

Żeby pomóc w rozwiązywaniu takich problemów, firma Microsoft wydała pakiet narzędzi graficznych o nazwie Expression Studio. Najnowsza wersja to Expression Studio 4, którą trzeba kupić oddzielnie (można również kupować pojedynczo każde z narzędzi dostępnych w pakiecie Expression). Ten pełen zestaw aplikacji dla projektantów WPF/Silverlight obsługuje cały proces projektowania aplikacji XAML, od początkowego symulowania interfejsu użytkownika po wszystkie elementy projektowe zawarte w finalnym produkcie. Niektóre z narzędzi w pakiecie Expression Studio, takie jak Expression Web, są przeznaczone specjalnie dla projektantów WWW. Program Expression Blend przeznaczony dla projektantów interfejsu użytkownika oddziela nie tylko kod proceduralny od znaczników, ale też oddziela zadania projektowe od zadań

programistycznych pozwalając programistom skupić się na pisaniu kodu biznesowego, a projektantom projektować funkcjonalny interfejs użytkownika bez konieczności znajomości C#, Visual Basic lub dowolnego innego języka .NET. MVVM jest kluczem do tego procesu współpracy pomiędzy projektantem a programistą. W istocie program Expression Blend jest dostarczany z określoną przestrzenią nazw, którą programiści mogą wykorzystać do utworzenia atrapy modelu widoku na potrzeby projektantów. Projektanci następnie mogą wiązać widok z tym odbiciem ostatecznego modelu widoku i projektować warstwę interfejsu użytkownika.

Pakiet Expression Studio w 60-dniowej wersji próbnej można pobrać pod adresem *http://www.microsoft.com/expression/*, kupić go przez sieć lub uzyskać poprzez subskrypcję MSDN.

Expression Blend

Expression Blend jest głównym produktem pakietu Expression Suite dla projektanta WPF/Silverlight. Pliki projektowe tego programu są w pełni kompatybilne z Microsoft Visual Studio. Możemy pracować nad projektem w programie Expression Blend, a następnie otwierać ten projekt w Visual Studio i vice versa. Ta dwukierunkowa kompatybilność ułatwia wykorzystanie programu Expression Blend do zaprojektowania szablonu i kontrolek naszej aplikacji biznesowej, a następnie przejście do Visual Studio w celu napisania kodu .NET. Mimo tej wygody, przechodzenie tam i z powrotem pomiędzy Expression Blend a Visual Studio nie jest obowiązkowe, ponieważ program Expression Blend może przetwarzać XAML oraz budować rozwiązania w językach C# i Visual Basic tak samo jak Visual Studio.

Korzystając z Expression Blend możemy projektować interfejs użytkownika w XAML, tworzyć bibliotekę kontrolek dla Silverlight lub WPF albo po prostu projektować i stosować niestandardowe style w naszej aplikacji XAML. Jedną z naprawdę mocnych funkcji Expression Blend jest możliwość tworzenia szablonów danych w czasie projektowania. Ta możliwość oznacza, że projektant graficzny nie potrzebuje „prawdziwej" bazy danych lub plików z danymi, żeby przedstawiać realistyczne wyniki w programie projektowym; Expression Blend pozwala nam łatwo skonfigurować szablon danych lub może wygenerować taki szablon. Końcowy wynik pojawia się w zintegrowanym środowisku projektowym i wygląda tak, jak wyniki, które moglibyśmy uzyskać przy użyciu danych rzeczywistych.

Najnowsza wersja Expression Blend 4 ma pełną obsługę WPF i Silverlight w czasie projektowania i znacznie ułatwia pracę projektanta. Ponadto Expression Blend udostępnia pakiet Behaviors SDK, który dodaje obsługę wzorca MVVM w czasie projektowania. Funkcja ta sprawia, że Expression Blend jest podstawowym narzędziem projektanta interfejsów użytkownika dla aplikacji wykorzystujących MVVM.

Na koniec, aby wspomnieć kilka nowych funkcji w najnowszej wersji Expression Blend, możemy łatwo budować i emulować aplikacje dla nowej platformy mobilnej Windows Phone 7; tworzyć wspaniałe przejścia i animacje dla swoich aplikacji