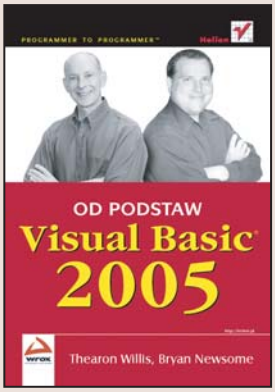
Visual Basic 2005. Od podstaw



Autorzy: Thearon Willis, Bryan Newsome
Tłumaczenie: Tomasz Walczak
ISBN: 83-246-0366-2
Tytuł oryginału: Beginning Visual Basic 2005
Format: B5, stron: 828

Rozpocznij przygodę z programowaniem w Visual Basicu

- Poznaj podstawy języka Visual Basic
- Napisz aplikacje dla systemu Windows i urządzeń przenośnych
- Stwórz usługi sieciowe i aplikacje WWW

Visual Basic jest jednym z najpopularniejszych języków programowania, stosunkowo łatwym do opanowania i bardzo uniwersalnym. Za jego pomocą można tworzyć aplikacje dla systemu Windows, aplikacje sieciowe i programy dla urządzeń mobilnych. Integracja Visual Basica z oferowanymi przez platformę .NET klasami bazowymi otworzyła przed programistami nowe możliwości. Obecnie jest to w pełni obiektowy język programowania umożliwiający wykorzystanie ogromnego potencjału tkwiącego w nowoczesnych platformach programistycznych.

Dzięki książce „Visual Basic 2005. Od podstaw" poznasz wszystkie zagadnienia niezbędne do tworzenia własnych aplikacji w tym języku. Dowiesz się, na czym polega programowanie obiektowe, jakie są główne elementy języka Visual Basic i jak stosować je w praktyce. Nauczysz się tworzyć okna dialogowe, menu i formularze, łączyć aplikacje z bazami danych, budować aplikacje WWW i usługi sieciowe. Poznasz wszystkie zastosowania Visual Basica.

- Instalacja Visual Basica 2005
- Podstawowe elementy i struktury języka
- Tworzenie okien dialogowych i formularzy
- Wyszukiwanie i usuwanie błędów
- Zasady programowania obiektowego
- Tworzenie własnych kontrolek
- Dostęp do baz danych za pomocą ADO.NET
- Aplikacje WWW
- Korzystanie z plików XML

Zostań profesjonalnym programistą Visual Basic

Wydawnictwo Helion
ul. Chopina 6
44-100 Gliwice
tel. (32)230-98-63
e-mail: helion@helion.pl

Spis treści

1

Wprowadzenie do języka Visual Basic 2005

Zadaniem tej książki jest umożliwienie szybkiego opanowania języka Visual Basic 2005 nawet tym osobom, które nigdy wcześniej nie programowały. Nowe informacje wprowadzane są stopniowo, na podstawie wcześniejszych rozdziałów. Weź więc głęboki wdech, powoli wypuść powietrze i pomyśl, że możesz to zrobić. Bez problemu! Naprawdę!

Programowanie w dużym stopniu przypomina uczenie dziecka wiązania sznurowadeł. Trudno cokolwiek osiągnąć, dopóki nie opanuje się poprawnego sposobu przekazywania instrukcji. Visual Basic 2005 to język, w którym można przekazać komputerowi instrukcje dotyczące wykonywania pewnych operacji. Ale podobnie jak dziecko, komputer zrozumie je tylko wtedy, gdy będą wystarczająco jasne. Jeśli nigdy nie pisałeś programów, może Ci się to wydawać trudne — i czasem rzeczywiście tak jest. Jednak Visual Basic 2005 to prosty język służący do wyjaśniania pewnych skomplikowanych rzeczy. Choć zrozumienie funkcjonowania programów na szczegółowym poziomie może być przydatne, Visual Basic 2005 zwalnia programistów z konieczności zajmowania się żmudnymi zawiłościami pisania programów dla systemu Windows. Pozwala to skoncentrować się na rozwiązywaniu ciekawszych problemów.

Visual Basic 2005 pomaga tworzyć programy działające w systemie operacyjnym Windows. Jeśli czytasz tę książkę, prawdopodobnie czujesz potrzebę lub chęć pisania programów tego typu. Nawet jeśli nigdy nie napisałeś programu komputerowego, wykonując „Spróbuj sam" przedstawione w tej książce, szybko poznasz różne aspekty języka Visual Basic 2005, jak również jego podstawę w postaci platformy .NET. Zobaczysz, że programowanie nie jest nawet w części tak trudne, jak to sobie wyobrażałeś. Zanim się o tym przekonasz, nauczysz się tworzyć różne typy programów. Ponadto — jak wskazuje nazwa .NET — języka Visual Basic 2005 można użyć do tworzenia aplikacji na urządzenia przenośne, na przykład komputery kieszonkowe czy telefony Smartphone. Jednak poznając nową technologię, trzeba najpierw nauczyć się chodzić, a dopiero potem biegać, dlatego początkowe rozdziały opisują podstawowe aplikacje dla systemu Windows, a dopiero w dalszej części książki przedstawione są aplikacje działające na innych platformach.

Ten rozdział przedstawia następujące zagadnienia:

- Instalację języka Visual Basic 2005.
- Przegląd zintegrowanego środowiska programistycznego (IDE) języka Visual Basic 2005.
- Tworzenie prostych programów dla systemu Windows.
- Używanie zintegrowanego systemu pomocy.

Programowanie dla systemu Windows i dla systemu DOS

Programy dla systemu Windows różnią się znacznie od swych wiekowych przodków — programów dla systemu MS-DOS. Przebieg działania programów dla systemu DOS był dość ściśle określony od początku do końca. Choć nie musi to oznaczać zmniejszenia funkcjonalności aplikacji, ogranicza sposób korzystania z niej przez użytkownika. Program dla systemu DOS jest jak przechodzenie przez korytarz. Aby dojść do końca, trzeba iść w jego kierunku, pokonując wszystkie napotkane przeszkody. Taki program pozwala otwierać po drodze jedynie określone drzwi.

Z drugiej strony system Windows zapoczątkował erę *programów sterowanych zdarzeniami*. *Zdarzenia* w tym przypadku obejmują na przykład kliknięcie przycisku, zmianę rozmiaru okna lub zmianę tekstu w polu tekstowym. Kod pisany przez programistę reaguje na takie zdarzenia. Wracając do przykładu z korytarzem — programy dla systemu Windows pozwalają znaleźć się na końcu korytarza w wyniku kliknięcia tego końca. Dzięki temu można pominąć całą długość korytarza. Jeśli znajdziesz się na końcu i stwierdzisz, że wcale nie chciałeś tam trafić, możesz łatwo przenieść się w nowe miejsce bez konieczności powrotu do punktu wyjścia. Program reaguje na Twoje ruchy i podejmuje odpowiednie działania, aby wykonać polecenia.

Kolejną wielką zaletą programów dla systemu Windows jest *niezależność od sprzętu*. To system Windows odpowiada za komunikację ze sprzętem, dzięki czemu nie musi tego robić programista. Nie trzeba znać zasad działania każdej drukarki laserowej na rynku, aby napisać program do drukowania. Nie trzeba uczyć się schematów kart graficznych, aby napisać grę. Windows opakowuje te operacje, udostępniając uniwersalne procedury komunikowania się ze sterownikami pisanymi przez producentów sprzętu. Prawdopodobnie jest to główny powód popularności systemu Windows. Te uniwersalne procedury to tak zwany interfejs programowania aplikacji (ang. *Application Interface Programming* — API).

Przed wprowadzeniem języka Visual Basic 1.0 w 1991 roku programiści musieli dobrze znać języki C i C++, jak również elementy składające się na sam system Windows, czyli API. Ta złożoność oznaczała, że jedynie zaangażowani i odpowiednio wykształceni programiści potrafili tworzyć programy działające w systemie Windows. Język Visual Basic stanowił wielką zmianę i szacuje się, że obecnie liczba wierszy kodu produkcyjnego w języku Visual Basic jest porównywalna w liczbą wierszy kodu we wszystkich innych językach.

Visual Basic zmienił oblicze programowania dla systemu Windows, znosząc konieczność pisania skomplikowanego kodu do obsługi interfejsu użytkownika. Możliwość *wizualnego* tworzenia własnego interfejsu pozwoliła programistom skoncentrować się na rozwiązywaniu problemów biznesowych. Po narysowaniu interfejsu użytkownika programista może dodać kod reagujący na zdarzenia.

Ponadto język Visual Basic od samego początku cechował się *rozszerzalnością*. Niezależni producenci szybko dostrzegli możliwość sprzedaży modułów ułatwiających pracę programistom. Te moduły, czyli *kontrolki*, początkowo nosiły nazwę VBXs, pochodzącą od rozszerzenia plików modułów. Przed wprowadzeniem języka Visual Basic 5.0, jeśli programiście nie podobał się sposób działania przycisku, mógł kupić inną wersję przycisku lub napisać własną, jednak te kontrolki musiały być napisane w językach C lub C++. Jednymi z pierwszych dostępnych kontrolek były narzędzia umożliwiające dostęp do baz danych. W języku Visual Basic 5.0 pojawiła się technologia *ActiveX*, która umożliwiała programistom tworzenie własnych *kontrolek ActiveX*.

Wprowadzenie przez Microsoft języka Visual Basic 3.0 to kolejny przełom w świecie programowania. Od tej pory możliwe było tworzenie aplikacji bazodanowych bezpośrednio dostępnych dla użytkownika (tak zwane *aplikacje frontonowe*) przy użyciu samego języka Visual Basic. Nie trzeba było posługiwać się dodatkowymi kontrolkami. Było to możliwe dzięki wprowadzeniu obiektów dostępu do danych (ang. *Data Access Objects* — DAO), które umożliwiały programistom manipulowanie danymi równie łatwo jak interfejsem użytkownika.

Wersje 4.0 i 5.0 języka Visual Basic zwiększały możliwości wersji 3.0, umożliwiając programistom pisanie programów dla nowej wersji systemu Windows — Windows 95. Co najbardziej istotne, nowe wersje języka ułatwiały także pisanie kodu nadającego się do powtórnego wykorzystania przez programistów piszących w innych językach programowania. Wersja 6.0 umożliwiła nowy sposób dostępu do baz danych dzięki modelowi ADO (ang. *ActiveX Data Objects*). Możliwości ADO zostały udostępnione przez Microsoft w celu umożliwienia dostępu do baz danych programistom aplikacji internetowych używającym języka ASP. Wszystkie te usprawnienia języka Visual Basic zapewniły mu dominującą pozycję w świecie programowania. Visual Basic pozwala programistom pisać stabilne i łatwe w pielęgnacji aplikacje w rekordowo krótkim czasie.

W lutym 2002 roku pojawił się Visual Basic .NET, w którym pozbyto się większości ograniczeń wcześniejszych wersji tego języka. W przeszłości Visual Basic krytykowano i wyśmiewano jako „zabawkowy" język, ponieważ nie udostępniał wszystkich właściwości cechujących bardziej złożone języki, na przykład języki C++ czy Java. W Visual Basic .NET Microsoft usunął te ograniczenia i przekształcił ten język w niezwykle wydajne narzędzie programistyczne. Visual Basic 2005 to kontynuacja tego trendu. Choć w języku Visual Basic 2005 nie wprowadzono tak znaczących zmian jak w Visual Basic .NET, zawiera on wystarczająco wiele usprawnień w języku i środowisku programistycznym, aby stanowił wartościową aktualizację. Visual Basic 2005 jest doskonałym wyborem dla programistów na wszystkich poziomach zaawansowania.

Instalacja Visual Basic 2005

Możesz używać Visual Basic 2005 w kilku postaciach:

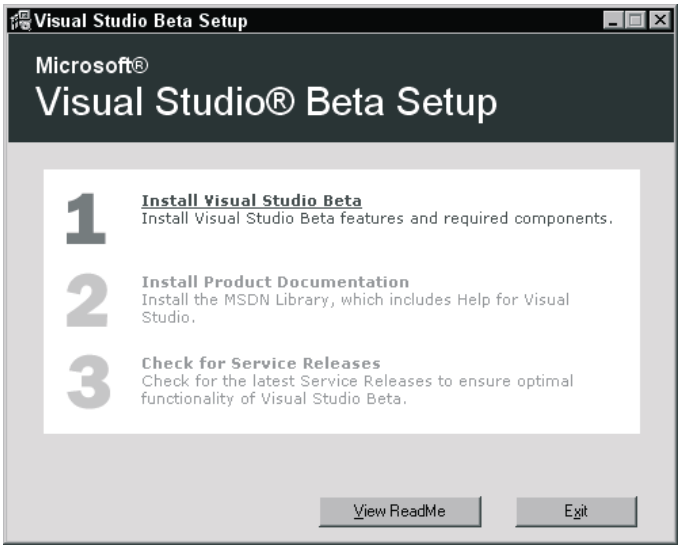
- Jako części środowiska Visual Studio 2005, które jest zestawem narzędzi i obejmuje także języki C# (wymawiaj jako „si-szarp"), J# (wymawiaj jako „dżej-szarp") i Visual C++. Linia produktów Visual Studio 2005 obejmuje wersje Visual Studio Standard Edition, Visual Studio Professional Edition, Visual Studio Tools for Office oraz Visual Studio Team System. Kolejne wersje udostępniają coraz więcej narzędzi do pisania dużych aplikacji i zarządzania ich tworzeniem.
- Jako wersji Express Edition, która w porównaniu z Visual Studio 2005 udostępnia ograniczony zestaw narzędzi i cech.

Obie wersje pozwalają tworzyć aplikacje dla systemu Windows. Proces instalacji jest prosty. Instalator środowiska Visual Studio potrafi nawet dokładnie określić, czego potrzebuje komputer, aby możliwa była instalacja tego środowiska.

Opisy w „Spróbuj sam" zakładają, że masz dostęp do środowiska Visual Studio 2005 Architect Edition. Większość opcji instalacji jest oczywista i w większości środowisk można zainstalować środowisko Visual Studio, posługując się domyślnymi ustawieniami. Niezależnie od instalowanej wersji instalacja zgodnie z domyślnymi ustawieniami powinna przebiegać bez zakłóceń.

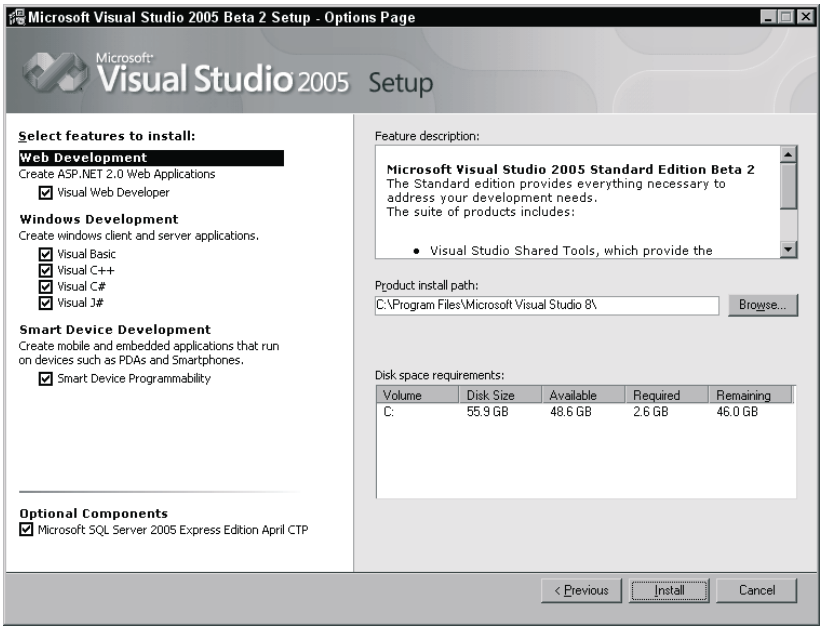
Spróbuj sam Instalacja Visual Basic 2005

1. Płyta z Visual Studio 2005 ma opcję autouruchamiania, jednak jeśli po włożeniu płyty nie pojawi się ekran instalacyjny, należy wpisać instrukcję `setup.exe` z poziomu głównego katalogu płyty. W tym celu należy otworzyć menu *Start* systemu Windows (zwykle znajduje się na dole ekranu) i wybrać opcję *Uruchom*. Następnie należy wpisać w polu *Otwórz* instrukcję `d:\setup.exe`, jeśli `d` to litera napędu CD-ROM. Po uruchomieniu programu instalacyjnego powinno pojawić się okno przedstawione na rysunku 1.1.
2. To okno dialogowe przedstawia kolejność instalowania elementów środowiska. Do poprawnego działania środowisko Visual Basic 2005 wymaga instalacji kilku aktualizacji, na przykład pakietu Service Pack 1 w przypadku systemu Windows XP. Program instalacyjny poinformuje Cię o brakujących aktualizacjach. Przed kontynuowaniem instalacji środowiska Visual Studio 2005 należy zainstalować potrzebne aktualizacje. Etap 1 instaluje środowisko Visual Studio, należy więc nacisnąć odnośnik *Install Visual Studio*.
3. Po zaakceptowaniu licencji użytkownika należy kliknąć przycisk *Continue* i przejść do kolejnego etapu.
4. Podobnie jak w przypadku większości programów instalacyjnych, pojawi się lista elementów środowiska, które można zainstalować (rysunek 1.2). Możesz zdecydować się na instalację jedynie potrzebnych składników. Na przykład, jeśli na dysku jest mało wolnego miejsca i nie planujesz w najbliższej przyszłości używać języka Visual



Rysunek 1.1. *Ekran powitalny instalatora środowiska Visual Studio 2005*

C++ 2005, możesz usunąć ten język z listy instalowanych składników. Możesz także określić lokalizację elementów, choć najłatwiej jest zostawić ustawienia domyślne. Wszystkie opcje pominięte w trakcie początkowej instalacji możesz dodać później, kiedy zmienią się Twoje potrzeby lub zainteresowania. Jeśli planujesz tworzyć aplikacje bazodanowe (opisane w rozdziale 16.), powinieneś zainstalować bazę danych SQL Server 2005 Express, która znajduje się na końcu listy.



Rysunek 1.2. *Instalacja składników środowiska Visual Studio 2005*

Każdy ze składników opisują informacje trojakiego rodzaju:

- Pole *Feature description* przedstawia zarys zastosowań i funkcji danego składnika.
- Pole *Product install path* pokazuje, gdzie dany składnik zostanie zainstalowany.
- Ostatnie pole, *Disk space requirements*, informuje o ilości pamięci na dysku zajmowanej przez wszystkie wybrane składniki.

Kiedy uruchamiasz Visual Basic 2005, wiele informacji przenoszonych jest z dysku do pamięci, a także z pamięci na dysk. Dlatego ważne jest, aby pozostawić odpowiednią ilość wolnego miejsca na dysku. Nie ma dokładnych reguł opisujących ilość potrzebnej pamięci, jednak dysk mający mniej niż 100 MB wolnej przestrzeni można uznać za pełny.

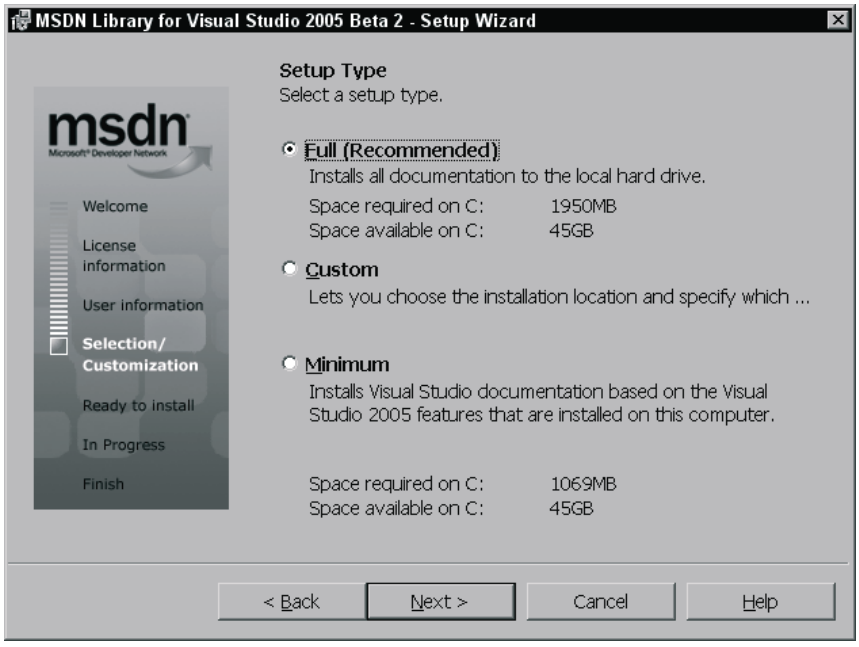
5. Po wybraniu wszystkich potrzebnych składników możesz kliknąć przycisk *Install*. Rozpocznie się instalacja, podczas której możesz spokojnie usiąść i zrelaksować się. Czas instalowania środowiska zależy od liczby wybranych składników oraz używanego komputera. Na przykład instalacja pełnego środowiska na komputerze z procesorem 2.4 GHz, 512 MB pamięci i systemem Windows XP Professional zajmuje około 20 minut.
6. Po zakończeniu instalacji zobaczysz okno dialogowe informujące o ukończeniu tego zadania.

   Na tym etapie możesz także zobaczyć informacje o problemach, które pojawiły się w czasie instalacji. Masz także okazję zobaczyć dziennik instalacji. Ten dziennik zawiera listę wszystkich operacji, które miały miejsce w czasie instalacji. O ile w trakcie instalacji nie wystąpiły jakieś błędy, można zrezygnować z oglądania dziennika. Na tym etapie instalacja środowiska Visual Studio 2005 jest już prawie zakończona. Możesz kliknąć przycisk *Done* i przejść do instalacji dokumentacji.
7. Instalacja biblioteki MSDN jest prosta, a następne punkty opisują jej etapy. Pierwszy ekran to ekran powitalny. Możesz kliknąć *Next*, aby przejść do następnego etapu.
8. Teraz możesz wybrać składniki dokumentacji, które chcesz zainstalować. Przedstawia to rysunek 1.3. Po wybraniu składników kliknij przycisk *Next*, aby rozpocząć proces instalacji.

Jeśli masz dużo wolnego miejsca na dysku, najlepiej jest zainstalować pełną dokumentację. W ten sposób uzyskasz dostęp do pełnej biblioteki, co jest istotne, jeśli w czasie instalacji wybierzesz ograniczony zestaw opcji, a później będziesz chciał dodać nowe właściwości.

9. Po zainstalowaniu dokumentacji MSDN ponownie zobaczysz ekran początkowy z dostępną opcją *Service Releases*.

Dobrym pomysłem jest wybranie opcji Service Releases, co powoduje automatyczne sprawdzanie aktualizacji środowiska. Microsoft włożył dużo pracy w udostępnianie aktualizacji oprogramowania poprzez internet. Te aktualizacje mogą obejmować wiele rzeczy, od dodatkowej dokumentacji po pakiety naprawiające błędy. Możesz wybrać instalację



Rysunek 1.3. *Instalacja systemu pomocy*

aktualizacji za pomocą płyty CD z pakietem Service Pack lub poprzez internet. Oczywiście aktualizacje poprzez internet wymagają aktywnego połączenia. Ponieważ aktualizacje mogą być dość duże, zaleca się używanie szybkiego połączenia.

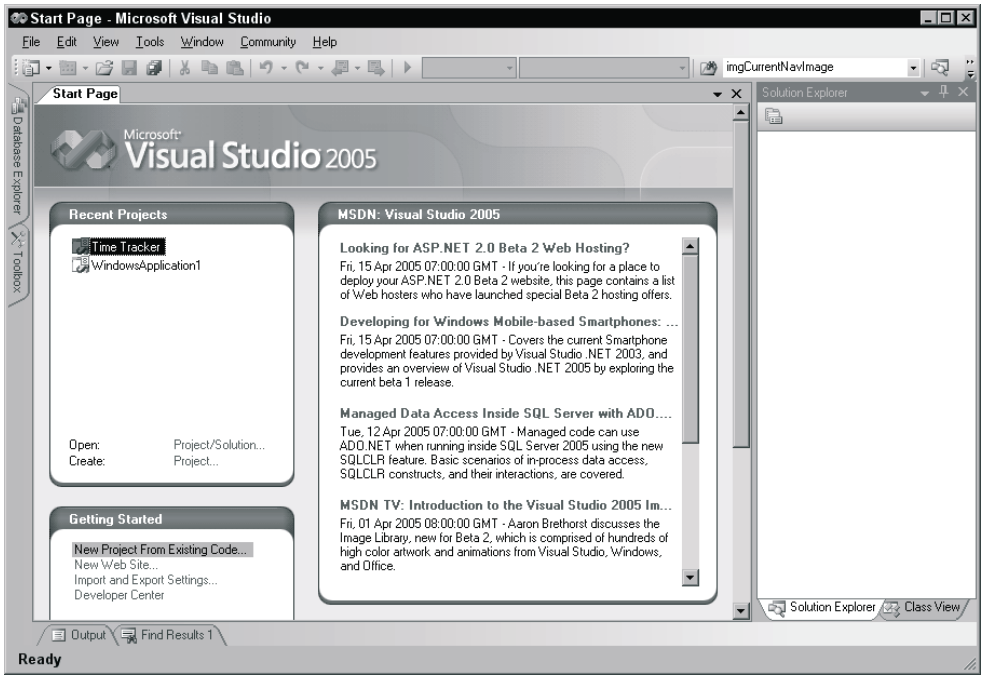
Po zakończeniu aktualizacji środowisko Visual Studio 2005 jest gotowe do użycia. Prawdziwa zabawa dopiero się rozpoczyna! Rozluźnij się więc, zrelaksuj i przygotuj na wkroczenie do świata języka Visual Basic 2005.

Środowisko programistyczne Visual Basic 2005

Pisanie aplikacji w języku Visual Basic 2005 nie wymaga korzystania ze specjalnego środowiska programistycznego tego języka. Możliwość uruchamiania kodu w języku Visual Basic 2005 jest związana z platformą .NET. W rzeczywistości można pisać cały kod w prostym edytorze tekstu, jak na przykład Notatnik. Można także wbijać gwoździe, posługując się butem zamiast młotkiem, ale pneumatyczny pistolet do gwoździ jest prawdopodobnie dużo bardziej wydajny. Programy w języku Visual Basic 2005 zdecydowanie najłatwiej jest pisać za pomocą zintegrowanego środowiska programistycznego (ang. *Integrated Development Environment* — IDE) Visual Studio 2005. Środowisko IDE jest tym, co widzisz, używając Visual Basic 2005 — to okna, pola i tak dalej. Środowisko IDE udostępnia wiele właściwości niedostępnych w zwykłych edytorach tekstu, takich jak sprawdzanie kodu, graficzna reprezentacja gotowych aplikacji czy okno wyświetlające wszystkie pliki składające się na projekt.

Ustawianie profilu

IDE to środowisko łączące ze sobą szereg narzędzi, które znacznie ułatwiają tworzenie oprogramowania. Uruchom środowisko Visual Studio 2005 i przyjrzyj się widocznym elementom. Jeśli w czasie instalacji użyłeś domyślnych ustawień, możesz otworzyć menu *Start* systemu Windows, a następnie wybrać opcję *Programy* (*Wszystkie programy* w Windows XP i Windows Server 2003)*/Microsoft Visual Studio 2005/Microsoft Visual Studio 2005*. Pojawi się wtedy ekran powitalny z widocznym oknem dialogowym *Choose Default Environment Settings*. Zaznacz opcję *Visual Basic Development Settings*, a następnie kliknij *Start Visual Studio*. Pojawi się środowisko programistyczne Visual Studio, widoczne na rysunku 1.4.



Rysunek 1.4. *Strona startowa środowiska Visual Studio 2005*

Menu

Możesz już z niecierpliwością oczekiwać, kiedy zaczniesz pisać kod. Najpierw jednak powinieneś poznać środowisko IDE i przyjrzeć się paskowi narzędzi i menu, które nie różnią się zbytnio od pasków narzędzi i menu w innych produktach Microsoftu, na przykład w programach Word, Excel czy PowerPoint.

Menu środowiska Visual Studio 2005 jest *dynamiczne*, co oznacza, że w zależności od wykonywanych operacji dostępne są różne elementy menu. W środowisku, w którym nie otwarto żadnego projektu, menu składa się jedynie z pozycji *File*, *Edit*, *View*, *Data*, *Tools*, *Window*, *Community* oraz *Help*. Po utworzeniu projektu pojawi się pełne menu środowiska Visual Studio 2005, widoczne na rysunku 1.5.

Rysunek 1.5. *Menu środowiska Visual Studio 2005*

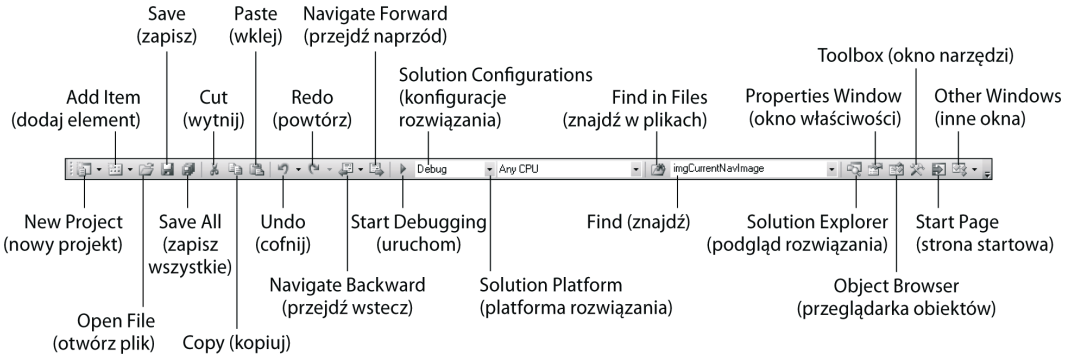
Na razie nie musisz szczegółowo znać wszystkich opcji menu. Zapoznasz się z nimi w czasie lektury tej książki. Poniżej znajduje się krótki przegląd operacji, z którymi powiązane są poszczególne elementy menu:

- ***File*** — Niemal każdy program dla systemu Windows ma menu *File* (plik). Jest to standardowy element, który zawiera przynajmniej opcję pozwalającą zakończyć działanie programu. W Visual Studio 2005 w menu tym znajdują się opcje pozwalające otwierać i zamykać poszczególne pliki, jak również całe projekty.
- ***Edit*** — Menu *Edit* (edycja) daje dostęp do standardowych elementów znanych z innych produktów: *Undo* (cofnij), *Redo* (powtórz), *Cut* (wytnij), *Copy* (kopiuj), *Paste* (wklej) i *Delete* (usuń).
- ***View*** — Menu *View* (widok) zapewnia szybki dostęp do okien udostępnianych przez środowisko, między innymi do okien *Solution Explorer*, *Properties*, *Output* i *Toolbox*.
- ***Project*** — Menu *Project* (projekt) pozwala dodawać do aplikacji nowe pliki, na przykład formularze czy klasy.
- ***Build*** — Menu *Build* (kompilacja) jest ważne, kiedy skończysz pisać aplikację i chcesz uruchamiać ją bez użycia środowiska Visual Basic 2005 (na przykład bezpośrednio z menu *Start*, tak jak inne aplikacje, choćby Word czy Access).
- ***Debug*** — Menu *Debug* (debugowanie) pozwala uruchamiać i zatrzymywać aplikację w obrębie środowiska IDE Visual Basic 2005. To menu daje także dostęp do debugera Visual Studio 2005. Debuger umożliwia stopniowe przechodzenie przez kod z jednoczesnym podglądem jego działania.
- ***Data*** — Menu *Data* (dane) pomaga używać informacji pochodzących z bazy danych. To menu jest dostępne tylko wtedy, kiedy projektujesz aplikację w trybie graficznym (w głównym oknie aktywna musi być zakładka *[Desing]*), a nie w czasie pisania kodu w edytorze kodu. Pracę z bazami danych opisują rozdziały 15. i 16.
- ***Format*** — Menu *Format* (format) także pojawia się tylko w czasie projektowania aplikacji w trybie graficznym. Elementy tego menu pozwalają manipulować wyglądem tworzonych kontrolek.
- ***Tools*** — Menu *Tools* (narzędzia) udostępnia polecenia pozwalające na konfigurację IDE Visual Studio 2005, jak również odnośniki do dostępnych zewnętrznych narzędzi.
- ***Window*** — Menu *Window* (okno) jest standardowo udostępniane w wielu aplikacjach pozwalających na jednoczesne otwieranie kilku okien, na przykład w programach Word czy Excel. Polecenia tego menu pozwalają przełączać się między oknami otwartymi w środowisku.
- ***Community*** — Menu *Community* (społeczność) daje dostęp do zasobów programistycznych i pozwala zadać pytanie, znaleźć potrzebny fragment kodu czy wysłać opinię o produkcie.
- ***Help*** — Menu *Help* (pomoc) daje dostęp do dokumentacji środowiska Visual Studio 2005. Dostęp do tych informacji jest możliwy na wiele sposobów (na przykład poprzez

odnośniki systemu pomocy, indeks lub wyszukiwanie). Menu *Help* udostępnia także opcje pozwalające połączyć się z witryną Microsoftu w celu pobrania aktualizacji lub zgłoszenia problemów.

Paski narzędzi

IDE Visual Studio 2005 udostępnia wiele pasków narzędzi, między innymi *Formatting* (formatowanie), *Image Editor* (edytor rysunków) i *Text Editor* (edytor tekstu). Te paski można dodawać i usuwać z IDE za pomocą opcji menu *View/Toolbars*. Każdy pasek daje szybki dostęp do często używanych poleceń, dzięki czemu nie trzeba przechodzić przez szereg opcji menu. Na przykład ikona *New Project* znajdująca się po lewej stronie domyślnego paska narzędzi (pasek narzędzi *Standard*) widocznego na rysunku 1.6 to odpowiednik opcji *File/New/Project*.



Rysunek 1.6. *Elementy pasków narzędzi*

Pasek narzędzi jest podzielony na grupy powiązanych ze sobą opcji, rozdzielone pionową linią. Pierwsze pięć ikon daje dostęp do często używanych opcji służących do manipulowania projektem i plikami, na przykład do otwierania i zapisywania plików. Opcje te są dostępne także w menu *File* i *Project*.

Kolejna grupa ikon służy do edycji (*Cut*, *Copy* i *Paste*). Trzecia grupa ikon pozwala cofnąć i powtórzyć edycję oraz poruszać się po kodzie.

Czwarta grupa ikon pozwala uruchomić aplikację (zielony trójkąt). Można także wybrać konfigurację rozwiązania i konkretną docelową platformę.

Kolejna grupa pozwala znaleźć tekst w kodzie dokumentu, projektu lub całego rozwiązania.

Ostatnia grupa ikon zawiera odnośniki prowadzące do okien *Solution Explorer*, *Properties*, *Toolbox*, *Object Browser*, *Start Page* i innych. Jeśli wybrane okno jest zamknięte, kliknięcie przycisku spowoduje jego automatyczne otwarcie.

Jeśli zapomnisz, do czego służy dana ikona, możesz umieścić nad nią kursor myszy. Spowoduje to pojawienie się podpowiedzi z nazwą opcji paska narzędzi.

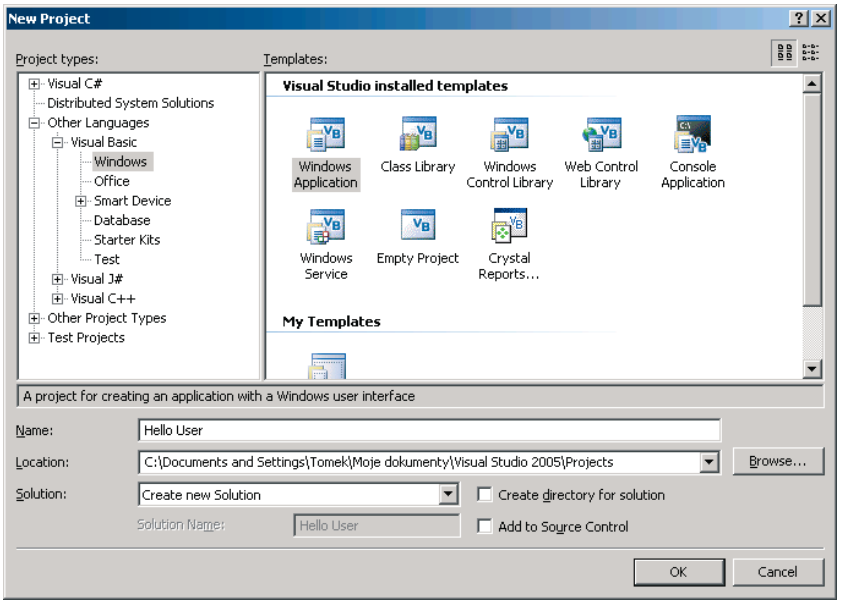
Możesz przyjrzeć się wszystkim oknom, wybierając poszczególne okna z menu *View*. Jednak przed otwarciem projektu wszystkie są puste, dlatego trudno powiedzieć, do czego służą. Najlepszy sposób na poznanie możliwości środowiska to używanie go w czasie pisania kodu.

Tworzenie prostej aplikacji

Na zakończenie przeglądu IDE musisz utworzyć projekt, dzięki czemu okna widoczne na rysunku 1.4 będą miały interesującą zawartość. Kolejne „Spróbuj sam" pokazuje, jak utworzyć bardzo prostą aplikację *HelloUser*, która pozwala użytkownikowi wpisać imię, a następnie wyświetla w oknie komunikatu pozdrowienie dla tej osoby.

Spróbuj sam Tworzenie projektu HelloUser

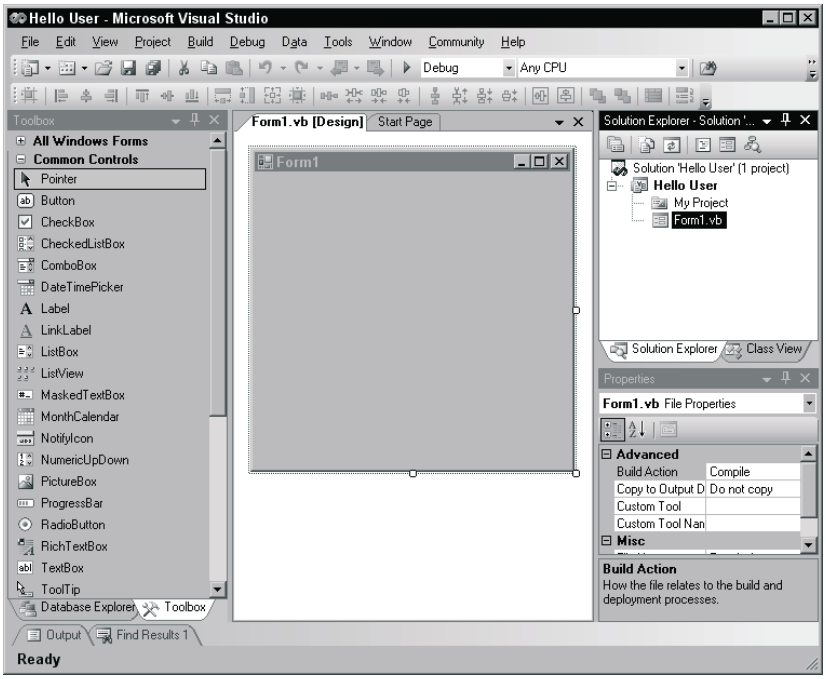
1. Kliknij przycisk *New Project* na pasku narzędzi.
2. Pojawi się okno dialogowe *New Project*. Upewnij się, że na panelu *Project types* znajdującym się po lewej stronie zaznaczona jest opcja *Visual Basic*. Następnie zaznacz opcję *Windows Application* w polu *Templates*, wpisz nazwę `Hello User` w polu tekstowym *Name* i kliknij przycisk *OK*. Okno dialogowe *New Project* powinno wyglądać teraz tak jak na rysunku 1.7.



Rysunek 1.7. *Tworzenie nowego projektu*

3. IDE utworzy pustą aplikację dla systemu Windows. W tym momencie program *Hello User* składa się z jednego pustego okna typu *Windows Form* (zwykle nazywanego formularzem) o domyślnej nazwie *Form1.vb*. Okno to widoczne jest na rysunku 1.8.

Rysunek 1.8.
Formularz



Kiedy środowisko Visual Studio 2005 tworzy nowy plik — zarówno w procesie tworzenia projektu, jak i w wyniku bezpośredniego polecenia programisty — nadaje mu nazwę odzwierciedlającą to, czym jest dany plik (w tym przypadku jest to formularz), dodając do niej numer.

Okna środowiska Visual Studio 2005

W tym momencie możesz zobaczyć podstawowe zastosowania różnych okien środowiska IDE. Przyjrzyj im się, zanim przejdziesz do dalszej części ćwiczenia. Jeśli któreś z tych okien nie jest widoczne na ekranie, możesz wybrać je w menu *View*, co spowoduje wyświetlenie potrzebnego okna. Jeśli nie odpowiada Ci położenie danego okna, zawsze możesz je przenieść, klikając pasek tytułu (niebieski pasek u góry okna) i przeciągając je w nowe miejsce. Okna środowiska IDE mogą *pływać* (zajmować niezależną pozycję) lub być *dokowane* (jak okno na rysunku 1.8). Poniższa lista krótko opisuje najważniejsze okna:

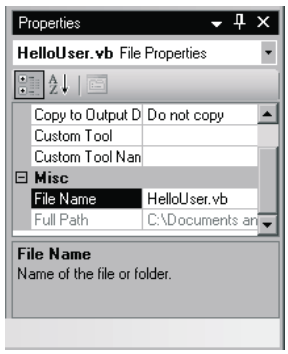
- ***Database Explorer*** — Okno *Database Explorer* daje dostęp do zdefiniowanych połączeń z bazami danych. Tutaj można tworzyć nowe połączenia z bazami danych i przeglądać istniejące. Na rysunku 1.8 okno *Database Explorer* to zakładka na dole okna *Toolbox*.
- ***Toolbox*** — Okno *Toolbox*, zwane oknem narzędzi, zawiera rozmaite kontrolki i komponenty, które można dodawać do aplikacji. Mogą to być proste przyciski, jak również niestandardowe kontrolki kupione lub napisane przez programistę.
- ***Design*** — Okno *Design* to miejsce, w którym przeprowadza się wiele operacji. W tym miejscu możesz tworzyć interfejs użytkownika. To okno zwane jest oknem projektowym.

- ***Solution Explorer*** — Okno *Solution Explorer* zawiera hierarchiczny podgląd rozwiązania. *Rozwiązanie* (ang. *solution*) może zawierać wiele projektów, podczas gdy *projekt* zawiera formularze, klasy, moduły i komponenty, które służą do rozwiązywania konkretnych problemów.
- ***Properties*** — Okno *Properties*, zwane oknem właściwości, pokazuje *właściwości* udostępniane przez wybrany obiekt. Choć właściwości można określić także w kodzie, często dużo łatwiej ustawić je w czasie projektowania aplikacji (na przykład dodając kontrolkę do formularza). Zauważ, że właściwość `File Name` formularza ma wartość `Form1.vb`. Jest to nazwa fizycznego pliku z kodem formularza i informacjami o układzie jego elementów.

Spróbuj sam Tworzenie projektu HelloUser (ciąg dalszy)

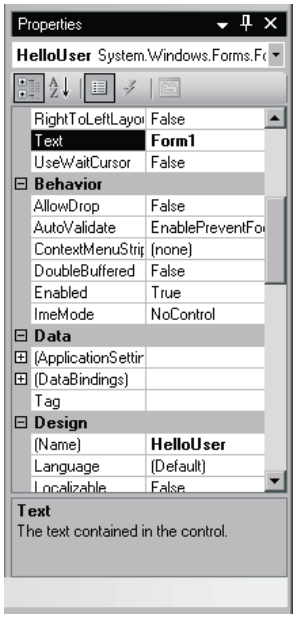
1. Zmień nazwę formularza na bardziej znaczącą. W tym celu kliknij plik *Form1.vb* w oknie *Solution Explorer*, a następnie w oknie właściwości zmień wartość właściwości `File Name` z `Form1.vb` na `HelloUser.vb`, jak przedstawia to rysunek 1.9, i wciśnij klawisz *Enter*. Aby wprowadzona zmiana została zatwierdzona, trzeba wcisnąć klawisz *Enter* lub przenieść kursor poza pole właściwości.

Rysunek 1.9.
Właściwości pliku



2. Zauważ, że zmiana właściwości jest automatycznie odzwierciedlana w oknie *Solution Explorer*, w którym plik ma teraz nazwę *HelloUser.vb*.
3. Następnie kliknij formularz widoczny w oknie projektowym. Okno właściwości wyświetli wtedy właściwości samego formularza zamiast widocznych wcześniej właściwości pliku formularza. Właściwości formularzy są zupełnie odmienne od właściwości plików. Różnica jest wynikiem dwóch odmiennych podejść do tego samego kodu. Kiedy zaznaczona jest nazwa pliku formularza w oknie *Solution Explorer*, okno właściwości przedstawia właściwości fizycznego pliku. Kiedy wybrany jest formularz w oknie projektowym, okno właściwości wyświetla wizualne i logiczne właściwości samego formularza.

   Okno właściwości umożliwia łatwe ustawianie właściwości kontrolki. Właściwości to zbiór wewnętrznych danych obiektu, które zwykle opisują jego wygląd lub działanie. Rysunek 1.10 pokazuje, że właściwości te są pogrupowane w kategorie: *Accessibility* (niewidoczna na rysunku), *Appearance* (nagłówek niewidoczny), *Behavior*, *Data*, *Design*, *Focus* (niewidoczna), *Layout* (niewidoczna), *Misc* (niewidoczna) i *Window Style* (niewidoczna).



Rysunek 1.10. *Właściwości formularza*

Patrząc na właściwości kategorii Appearance, możesz zauważyć, że mimo zmiany nazwy pliku formularza na HelloUser.vb tekst nagłówka formularza to wciąż Form1.

4. W tej chwili nagłówek (właściwość `Text`) formularza, widoczny na górnym pasku okna, to `Form1`. Nie jest to zbyt opisowy tytuł, dlatego zmień go tak, aby odzwierciedlał działanie aplikacji. Znajdź właściwość `Text` w kategorii *Appearance* okna właściwości, zmień jej wartość na `Pozdrowienie od Visual Basic 2005` i wciśnij *Enter*. Nagłówek formularza zostanie automatycznie zaktualizowany.

Jeśli masz kłopot ze znalezieniem danej właściwości, możesz kliknąć przycisk AZ na pasku narzędzi znajdującym się nad oknem właściwości. Spowoduje to uporządkowanie właściwości według nazw zamiast kategorii.

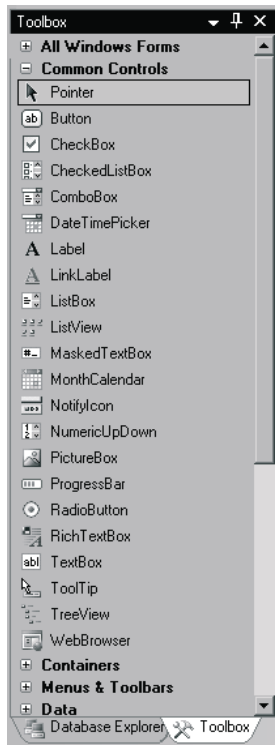
5. Program jest już gotowy. Kliknij przycisk *Start* na pasku narzędzi środowiska Visual Studio 2005 (zielony trójkąt), aby uruchomić aplikację. Jeśli czytając książkę, natrafisz na instrukcję „uruchom projekt", po prostu kliknij przycisk *Start*. W tym przypadku spowoduje to wyświetlenie pustego okna z nagłówkiem *Pozdrowienia od Visual Basic 2005*.

Utworzenie tej aplikacji było łatwe, ale na razie program nie jest zbyt użyteczny. Kolejne operacje zwiększą możliwości komunikacji z aplikacją. W tym celu należy dodać do formularza kilka kontrolek: etykietę (`Label`), pole tekstowe (`TextBox`) i dwa przyciski (`Button`). Zobaczysz, jak łatwo to zrobić za pomocą okna narzędzi. Możesz zastanawiać się, w którym momencie zaczniesz pisać kod. Już wkrótce. Wspaniałą cechą języka Visual Basic 2005 jest to, że można utworzyć dużą część aplikacji *bez* pisania jakiegokolwiek kodu. Oczywiście ten kod, choć ukryty, wciąż istnieje, a za jego generowanie odpowiada Visual Basic 2005.

Okno narzędzi

Okno narzędzi można otworzyć, wybierając opcję *View/Toolbox*, klikając ikonę *Toolbox* na standardowym pasku narzędzi lub wciskając kombinację klawiszy *Ctrl+Alt+X*. Ponadto w lewej części interfejsu widoczna jest zakładka *Toolbox*. Umieszczenie nad nią kursora myszy spowoduje wyświetlenie okna narzędzi, które częściowo przesłoni formularz.

Okno narzędzi zawiera podzielone na kategorie kontrolki i komponenty, które można umieszczać na formularzach. Kontrolki takie jak pola tekstowe, przyciski, przyciski opcji czy listy rozwijane można wybrać i *narysować* na formularzu. W aplikacji *HelloUser* potrzebne będą jedynie kontrolki z kategorii *Common Controls*. Na rysunku 1.11 widoczne są standardowe kontrolki formularzy Windows.



Rysunek 1.11. *Standardowe kontrolki w oknie narzędzi*

Kontrolki można dodawać do formularzy w dowolnej kolejności, dlatego nie jest istotne, czy dodasz najpierw etykietę, czy pole tekstowe lub przycisk. Kolejne „Spróbuj sam” opisuje dodawanie kontrolek.

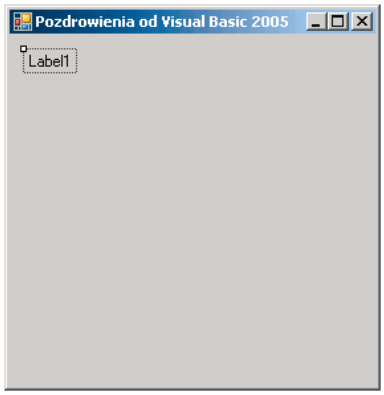
Spróbuj sam Dodawanie kontrolek do aplikacji HelloUser

1. Jeśli jeszcze tego nie zrobiłeś, zamknij aplikację. Najłatwiej można to zrobić, klikając przycisk *x* w prawym górnym rogu formularza. Możesz także kliknąć niebieski prostokąt w środowisku IDE, który wyświetla podpowiedź *Stop Debugging* po umieszczeniu nad nim kursora myszy.

2. Dodaj do formularza etykietę (kontrolka `Label`). Kliknij etykietę w oknie narzędzi, przenieś ją na formularz i umieść w wybranym miejscu. Możesz także umieszczać kontrolki na formularzu, klikając je dwukrotnie w oknie narzędzi lub klikając kontrolkę w oknie narzędzi, a następnie rysując ją na formularzu.

3. Jeśli narysowana etykieta nie znajduje się w odpowiednim miejscu, nie stanowi to problemu. Po umieszczeniu kontrolki na formularzu możesz zmieniać jej wielkość i położenie. Rysunek 1.12 przedstawia wygląd kontrolki po umieszczeniu jej na formularzu. Aby przenieść ją w inne miejsce, kliknij przerywaną krawędź kontrolki i umieść ją w wybranym położeniu. Etykieta automatycznie dopasuje swój rozmiar do tekstu wpisanego we właściwości `Text`.

Rysunek 1.12.
Etykieta Label1

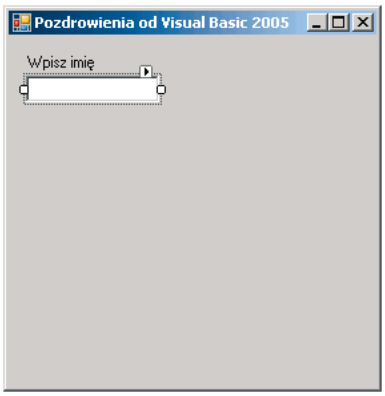


4. Po narysowaniu kontrolki na formularzu powinieneś przynajmniej określić jej nazwę oraz wyświetlany tekst. Okno właściwości, znajdujące się na prawo od okna projektowego, wyświetla teraz właściwości etykiety, o czym informuje widoczna nazwa kontrolki — `Label1`. W oknie właściwości podaj nową wartość właściwości `Text` etykiety — `Wpisz imię`. Zauważ, że po wciśnięciu klawisza *Enter* lub po umieszczeniu kursora poza właściwością etykieta automatycznie zmienia rozmiar, aby dopasować się do nowej wartości właściwości `Text`. Następnie zmień nazwę etykiety (właściwość `Name`) na `lblName`.

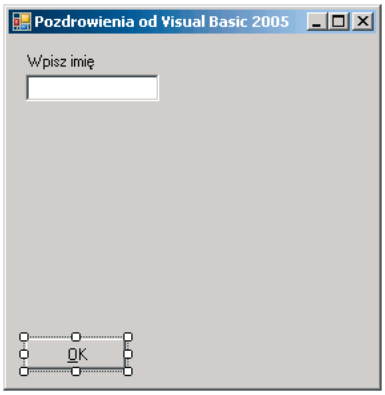
5. Teraz bezpośrednio pod etykietą dodaj pole tekstowe do wpisania imienia. W tym celu możesz powtórzyć operacje prowadzące do dodania etykiety, jednak tym razem wybierz z okna narzędzi kontrolkę `TextBox`. Po przeciągnięciu kontrolki w odpowiednie miejsce formularza (jak przedstawia to rysunek 1.13) użyj okna właściwości do zmiany nazwy kontrolki na `txtName`.

Zwróć uwagę na punkty służące do zmiany rozmiaru znajdujące się po lewej i prawej stronie kontrolki. Możesz użyć tych punktów do zmiany długości pola tekstowego.

6. W lewym dolnym rogu formularza umieść przycisk (kontrolka `Button`), wykonując te same operacje, co w przypadku etykiety i pola tekstowego. Ustaw nazwę przycisku na `btnOK`, a właściwość `Text` na `&OK`. Formularz powinien przypominać ten widoczny na rysunku 1.14.



Rysunek 1.13. *Dodawanie pola tekstowego do formularza*

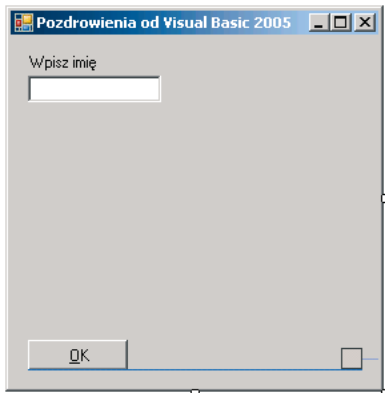


Rysunek 1.14. *Umieszczanie przycisku na formularzu*

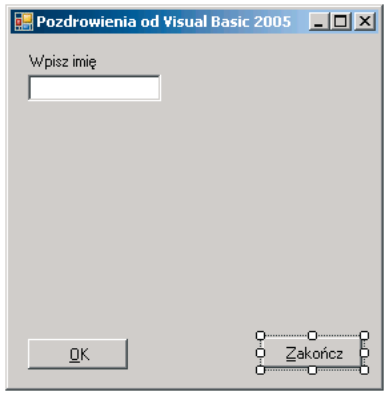
Znak & we właściwości `Text` przycisku pozwala utworzyć skrót klawiaturowy (tak zwany klawisz skrótu). Litera, przed którą znajduje się znak &, jest wyświetlana z podkreśleniem (co widać na rysunku 1.14). Informuje to użytkownika, że może wybrać ten przycisk za pomocą kombinacji klawiszy Alt+litera zamiast używać do tego myszy. Niektóre konfiguracje powodują wyświetlenie podkreślenia dopiero po wciśnięciu klawisza Alt. W tym przykładzie wciśnięcie kombinacji Alt+O daje ten sam efekt co bezpośrednie kliknięcie przycisku OK. Nie trzeba pisać specjalnego kodu do obsługi tej techniki.

7. Teraz dodaj drugi przycisk w prawym dolnym rogu formularza, przeciągając kontrolkę `Button` z okna narzędzi. Zauważ, że tym razem w trakcie zbliżania kursora myszy do prawego dolnego rogu formularza pojawi się niebieska linia, widoczna na rysunku 1.15. Ta linia pozwala wyrównać położenie nowego przycisku względem przycisku znajdującego się już na formularzu. Tego typu linie pomagają w wyrównywaniu położenia nowej kontrolki względem lewej, prawej, dolnej lub górnej krawędzi innej kontrolki, w zależności od tego, gdzie ta nowa kontrolka ma się znaleźć. Jasnoniebieska linia pozwala określić stały margines między krawędzią kontrolki a krawędzią formularza. Ustaw nazwę nowego przycisku na `btnExit`, a jego właściwość `Text` na `&Zakończ`. Formularz powinien wyglądać teraz tak jak na rysunku 1.16.

Zanim skończysz tworzyć przykładową aplikację, powinieneś zapoznać się z używaną w tej książce notacją.



Rysunek 1.15. *Wyrównywanie położenia kontrolek*



Rysunek 1.16. *Gotowy interfejs użytkownika*

Zmodyfikowana notacja węgierska

Może zauważyłeś, że nazwy nadawane kontrolkom są trochę dziwne. Każda ma przedrostek w postaci skrótowego identyfikatora typu kontrolki. Dzięki temu w kodzie dużo łatwiej zauważyć, jakiego typu jest używana kontrolka. Możesz na przykład nazwać kontrolkę `Name`, bez przedrostka `lbl` ani `txt`. Nie wiesz wtedy, czy używasz etykiety wyświetlającej imię, czy pola tekstowego do wpisywania imienia. Wyobraź sobie, że w poprzednim „Spróbuj sam" nazwałbyś etykietę `Name1`, a pole tekstowe — `Name2`. Łatwo pomylić takie kontrolki, szczególnie kiedy wraca się do pracy nad aplikacją po dłuższym okresie.

Kiedy współpracujesz z innymi programistami, bardzo istotne jest zachowanie spójności stylu kodowania. Jeden z najczęściej używanych sposobów nazywania kontrolek, przyjęty w wielu językach programowania, został zaproponowany przez dr. Charlesa Simonyiego, który przed objęciem stanowiska w Microsofcie pracował w ośrodku badawczym korporacji Xerox. Dr Simonyi wymyślił krótkie przedrostki, które pozwalają programistom łatwo identyfikować typ danych zmiennej. Ponieważ dr Simonyi pochodzi z Węgier, a przedrostki powodują, że nazwy wyglądają nieco obco, system ten przyjęło się nazywać notacją węgierską. Ponieważ oryginalna notacja tego typu służyła do pisania programów w językach C i C++, notacja używana w Visual Basic 2005 nosi nazwę zmodyfikowanej notacji węgierskiej. Tabela 1.1 przedstawia niektóre z popularnych przedrostków wykorzystywanych w tej książce.

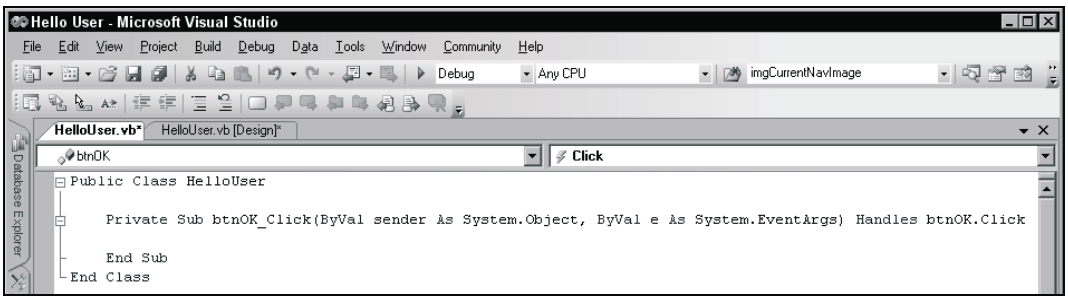
Tabela 1.1. *Wybrane przedrostki typów danych*

Kontrolka	Przedrostek
Button	btn
ComboBox	cbo
CheckBox	chk
Label	lbl
ListBox	lst
MainMenu	mnu
RadioButton	rdb
PictureBox	pic
TextBox	txt

Notacja węgierska pozwala zaoszczędzić wiele czasu przy analizie kodu napisanego przez innego programistę lub kodu, do którego wraca się po dłuższej przerwie. Jednak najważniejsza jest spójność stosowanej notacji. Kiedy zaczniesz pisać kod, wybierz określoną konwencję tworzenia nazw. Programistom języka Visual Basic 2005 zaleca się używanie zmodyfikowanej notacji węgierskiej, która stała się praktycznie standardem, jednak nie jest to konieczne. Kiedy już zdecydujesz się na określoną konwencję, stosuj ją konsekwentnie. Kiedy modyfikujesz kod napisany przez innych programistów, powinieneś dopasować się do używanej przez nich notacji. Standardowa konwencja nazywania stosowana w całym projekcie pozwala zaoszczędzić wiele godzin na etapie pielęgnacji kodu. Pora jednak wrócić do przykładowej aplikacji. Jesteś już gotowy, aby napisać pierwszy fragment kodu.

Edytor kodu

Po zdefiniowaniu formularza aplikacji *HelloUser* możesz dodać kod, dzięki któremu formularz zacznie robić coś ciekawego. Widziałeś już, jak łatwe jest dodawanie kontrolek do formularza. Obsługa działania elementów widocznych na ekranie także nie jest trudna. Aby dodać kod definiujący działanie kontrolki, należy dwukrotnie kliknąć kontrolkę. Spowoduje to otwarcie w głównym oknie edytora kodu widocznego na rysunku 1.17.



Rysunek 1.17. *Edytor kodu*

Zauważ, że w głównym oknie pojawiła się nowa zakładka. Teraz widoczne są dwie zakładki, reprezentujące okno projektowe oraz edytor kodu. Aby umieścić kontrolki na formularzu, korzystałeś z okna projektowego, podczas gdy do pisania kodu posłuży Ci edytor kodu. Środowisko Visual Studio 2005 tworzy odrębny plik z tym kodem. Graficzne elementy formularza oraz tak zwany kod ukryty znajdują się w odrębnych plikach. W tym przypadku są to pliki *HelloUser.Designer* oraz *HelloUser.vb*. Jest to jedna z przyczyn tego, że tworzenie aplikacji za pomocą języka Visual Basic 2005 jest tak wygodne i łatwe. Używając okna projektowego, możesz utworzyć graficzny interfejs aplikacji, a następnie za pomocą edytora kodu dodać kod odpowiedzialny za potrzebne operacje.

W górnej części edytora kodu widoczne są dwie listy rozwijane. Pozwalają one szybko przejść do różnych fragmentów kodu. Po umieszczeniu kursora myszy nad polem znajdującym się po lewej stronie zobaczysz podpowiedź informującą, że jest to lista *Class Name*. Po rozwinięciu tej listy zobaczysz wszystkie obiekty pojawiające się w aplikacji. Prawa lista nosi nazwę *Method Name*. Jej rozwinięcie pozwala zobaczyć listę wszystkich zdefiniowanych funkcji i procedur obiektu wybranego z listy *Class Name*. Jeśli do obsługi danego formularza potrzeba dużo kodu, listy rozwijane pozwalają szybko przejść do szukanego obszaru, przeskakując do wybranego fragmentu kodu. Jednak ponieważ w tym przypadku cały kod mieści się na ekranie, trudno jest się w nim zgubić.

Spróbuj sam Dodawanie kodu projektu HelloUser

1. Aby dodać potrzebny kod, kliknij zakładkę *Design* w celu ponownego wyświetlenia formularza. Teraz dwukrotnie kliknij przycisk *OK*. Pojawi się wtedy edytor kodu z poniższym kodem. Jest to szkielet metody obsługi zdarzenia `Click` przycisku, w którym należy wpisać kod uruchamiany w wyniku kliknięcia tej kontrolki. Ten kod to *metoda obsługi zdarzenia*.

```
Private Sub btnOK_Click(ByVal sender As System.Object, _
   ByVal e As System.EventArgs) Handles btnOK.Click

End Sub
```

Z powodu ograniczeń związanych ze składem książki nie można zmieścić deklaracji procedury w jednym wierszu. Język Visual Basic 2005 pozwala dzielić wiersze kodu. Służy do tego znak podkreślenia (_), który oznacza kontynuację wiersza. Przed tym znakiem musi znajdować się odstęp. Wszystkie odstępy na początku następnego wiersza kodu są ignorowane.

   `Sub` to przykład *słowa kluczowego*. W programowaniu słowo kluczowe to specjalne słowo, które pozwala poinformować środowisko programistyczne, że musi wykonać określoną operację. Słowo kluczowe `Sub` informuje język Visual Basic 2005, że jest to *procedura*, czyli metoda, która nie zwraca wartości. Kod wpisany między `Private Sub` a `End Sub` to procedura obsługi zdarzenia `Click` przycisku *OK*.

2. Teraz dodaj do procedury wyróżniony kod:

```
Private Sub btnOK_Click(ByVal sender As System.Object, _
   ByVal e As System.EventArgs) Handles btnOK.Click
```

```
    ' Wyświetla okno komunikatu z powitaniem użytkownika
    MessageBox.Show("Cześć " & txtName.Text & _
       "! Witaj w świecie Visual Basic 2005.", _
       "Komunikat powitalny")
End Sub
```

Książka zawiera wiele fragmentów kodu, które powinieneś wpisać, jeśli wykonujesz ćwiczenia. Zwykle będzie oczywiste, gdzie powinieneś umieścić kod, a w bardziej skomplikowanych sytuacjach przedstawione są dokładne wskazówki. Kod, który należy wpisać, znajduje się na szarym tle.

3. Po dodaniu potrzebnego kodu ponownie otwórz okno projektowe i dwukrotnie kliknij przycisk *Zakończ*. Do metody obsługi zdarzenia `btnExit_Click` dodaj wyróżniony kod.

```
Private Sub btnExit_Click(ByVal sender As System.Object, _
   ByVal e As System.EventArgs) Handles btnExit.Click

    ' Kończenie działania programu i zamykanie formularza
    Me.Close()
End Sub
```

   Możesz zastanawiać się, czym jest `Me`. `Me` to słowo kluczowe oznaczające formularz. Podobnie jak angielski zaimek *me*, jest to skrótowe określenie samego siebie.

4. Kod jest już gotowy. Nadeszła chwila prawdy — możesz sprawdzić działanie programu. Najpierw jednak zapisz wyniki swej pracy, używając opcji menu *File/Save HalloUser.vb* lub klikając przycisk *Save* na pasku narzędzi.

5. Teraz kliknij przycisk *Start* na pasku narzędzi. Okno *Output* znajdujące się na dole ekranu wyświetli wtedy wiele informacji. Jeśli nie przytrafiły Ci się żadne literówki w czasie wpisywania kodu, widoczny tekst informuje o plikach załadowanych w celu uruchomienia aplikacji.

   Na tym etapie środowisko Visual Studio 2005 *kompiluje* kod. Kompilacja to proces przekształcania kodu źródłowego w języku Visual Basic 2005 na postać zrozumiałą dla komputera. Po zakończeniu kompilacji środowisko Visual Studio 2005 uruchamia (wykonuje) program i możesz zobaczyć efekty swej pracy.

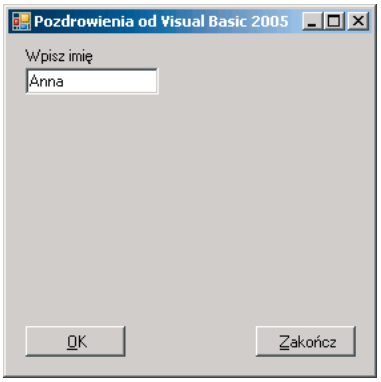
Jeśli Visual Basic 2005 natrafi na błędy, zostaną one wyświetlone jako zadania w oknie Task List. Dwukrotne kliknięcie zadania spowoduje wyświetlenie wiersza kodu powodującego problem. Więcej informacji o usuwaniu błędów znajdziesz w rozdziale 9.

6. Po uruchomieniu aplikacji zobaczysz główny formularz. Wpisz imię i kliknij przycisk *OK* (lub wciśnij kombinację klawiszy *Alt+O*). Uruchomiony formularz widoczny jest na rysunku 1.18.

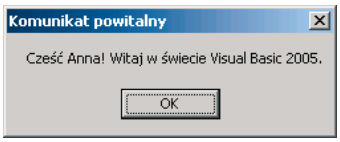
7. Pojawi się okno zwane oknem komunikatu, witające osobę, której imię zostało wpisane w polu tekstowym formularza. W tym przypadku jest to `Anna` (rysunek 1.19).

8. Po zamknięciu okna komunikatu poprzez kliknięcie przycisku *OK* kliknij przycisk *Zakończ* na formularzu. Aplikacja zakończy działanie i wrócisz do środowiska Visual Basic 2005.

Rysunek 1.18. *Działający program*



Rysunek 1.19. *Program wyświetla okno komunikatu*



Jak to działa?

Kod, który dodałeś do obsługi zdarzenia `Click` przycisku *OK*, pobiera imię podane w polu tekstowym i używa go jako części komunikatu widocznego na rysunku 1.19.

Pierwszy wiersz kodu procedury to *komentarz*, czyli tekst przeznaczony dla programistów piszących lub pielęgnujących kod, a nie dla komputera. Komentarze w języku Visual Basic 2005 rozpoczynają się od apostrofu ( ). Kompilator ignoruje tekst oznaczony jako komentarz. Więcej informacji o komentarzach znajduje się w rozdziale 3.

Metoda `MessageBox.Show` wyświetla okno komunikatu, do którego można przekazać szereg parametrów. W kodzie przykładu przekazywany jest wyświetlany ciąg znaków. Odbywa się to w wyniku *łączenia* stałych zdefiniowanych przez tekst wewnątrz cudzysłowów. Do łączenia ciągów znaków w jeden ciąg służy znak `&`.

Następny fragment kodu łączy stałą `"Cześć "` z wartością właściwości `Text` pola tekstowego `txtName` oraz z następną stałą — `"! Witaj w świecie Visual Basic 2005."`. Drugi parametr metody `MessageBox.Show` to nagłówek wyświetlany na pasku tytułu okna dialogowego.

Znak podkreślenia (_) użyty na końcu wierszy w poniższym kodzie pozwala rozbić długie wiersze na kilka krótszych. Informuje to kompilator o tym, że reszta kodu danej instrukcji znajduje się w kolejnym wierszu. Jest to technika bardzo użyteczna przy tworzeniu długich ciągów znaków, ponieważ pozwala widzieć całe fragmenty kodu w edytorze kodu bez konieczności przewijania okna w prawo.

```
    Private Sub btnOK_Click(ByVal sender As System.Object, _
ByVal e As System.EventArgs) Handles btnOK.Click

    ' Wyświetla okno komunikatu z powitaniem użytkownika
    MessageBox.Show("Cześć " & txtName.Text & _
       "! Witaj w świecie Visual Basic 2005.", _
       "Komunikat powitalny")
End Sub
```

Kolejna procedura to metoda obsługi zdarzenia `Click` przycisku *Zakończ*. Zawiera ona tylko jedną instrukcję — `Me.Close()`. Jak opisuje to punkt 3. poprzedniego „Spróbuj sam", słowo kluczowe `Me` reprezentuje formularz. Metoda `Close` formularza zamyka go i zwalnia wszystkie powiązane z nim zasoby, kończąc w ten sposób działanie programu.

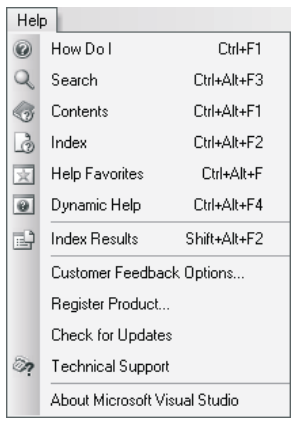
```
Private Sub btnExit_Click(ByVal sender As System.Object, _
   ByVal e As System.EventArgs) Handles btnExit.Click

   ' Kończenie działania programu i zamykanie formularza
   Me.Close()
End Sub
```

Używanie systemu pomocy

System pomocy dostępny w Visual Basic 2005 jest lepszy niż w poprzednich wersjach tego języka. Ucząc się języka Visual Basic 2005, prawdopodobnie poznasz system pomocy. Jednak warto poświęcić chwilę na jego krótki przegląd, dzięki czemu będzie Ci łatwiej znaleźć potrzebne informacje.

Menu *Help* zawiera elementy widoczne na rysunku 1.20.

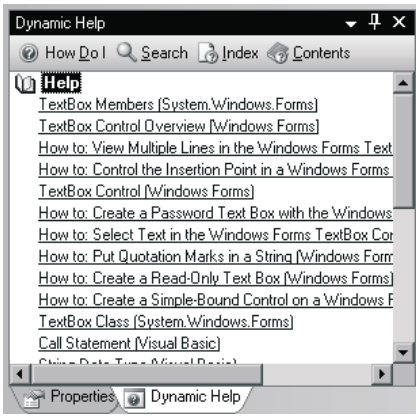
Rysunek 1.20. *Elementy menu Help*



Jak widać, to menu zawiera nieco więcej pozycji, niż ma to miejsce w zwykłych aplikacjach dla systemu Windows. Głównym tego powodem jest obszerność dokumentacji. Bardzo niewiele osób potrafi zapamiętać wszystkie informacje. Na szczęście nie stanowi to problemu, ponieważ można zawsze szybko znaleźć potrzebne informacje w systemie pomocy. Możesz o nim myśleć jako o systemie bezpieczeństwa dla pamięci.

Jedną z naprawdę fantastycznych właściwości jest pomoc dynamiczna. Kiedy wybierzesz opcję *Dynamic Help* z menu *Help*, otworzy się okno *Dynamic Help* zawierające listę tematów powiązanych z zagadnieniem, nad którym aktualnie pracujesz. Okno *Dynamic Help* można otworzyć w środowisku, wybierając opcję *Help/Dynamic Help*, po czym okno to jest dostępne jako zakładka obok okna właściwości.

Kiedy używasz pola tekstowego (na przykład pola tekstowego w aplikacji *HelloUser*) i chcesz znaleźć pewne informacje na jego temat, wystarczy zaznaczyć to pole na formularzu lub w edytorze kodu, a następnie otworzyć okno *Dynamic Help*, w którym widoczne będą wszystkie tematy pomocy dotyczące pól tekstowych, tak jak przedstawia to rysunek 1.21.



Rysunek 1.21. *Okno Dynamic Help wyświetlające teamty dotyczące wybranego zagadnienia*

Inne polecenia pomocy w menu *Help* (*Search*, *Contents* i *Index*) działają dokładnie tak samo jak w innych aplikacjach dla systemu Windows. Opcja *How Do I* wyświetla listę często używanych operacji podzielonych na kategorie. Dzięki temu można szybko i łatwo znaleźć tematy pomocy dotyczące często wykonywanych zadań.

Podsumowanie

Czy zaczynasz już widzieć, że tworzenie podstawowych aplikacji za pomocą Visual Basic 2005 nie jest zbyt trudne? Przyjrzałeś się już środowisku IDE i zobaczyłeś, że pozwala bardzo szybko utworzyć gotową aplikację. Okno narzędzi umożliwia dodawanie kontrolek do formularza i projektowanie interfejsu użytkownika w szybki i łatwy sposób. Okno właściwości ułatwia konfigurowanie tych kontrolek, a okno *Solution Explorer* umożliwia przegląd plików składających się na projekt. Ponadto napisałeś dwa fragmenty kodu.

W kolejnych rozdziałach poznasz więcej szczegółów i nauczysz się pisać kod. Zanim zagłębisz się w tajniki języka Visual Basic 2005, w następnym rozdziale dowiesz się więcej o platformie .NET. To dzięki tej platformie języki .NET są tak łatwe w użyciu, mogą ze sobą współpracować i łatwo jest się ich nauczyć.

Po lekturze tego rozdziału powinieneś opanować następujące zagadnienia:

- Pracę w zintegrowanym środowisku programistycznym (IDE).
- Dodawanie kontrolek do formularzy w oknie projektowym.
- Ustawianie właściwości kontrolek.
- Dodawanie kodu formularza w edytorze kodu.

Ćwiczenie

Ćwiczenie 1.

Utwórz aplikację dla systemu Windows z polem tekstowym i przyciskiem, która wyświetli w oknie komunikatu tekst wpisany w polu tekstowym.

Rozwiązanie tego ćwiczenia, jak również wszystkich ćwiczeń przedstawionych na końcu poszczególnych rozdziałów, znajduje się w dodatku D.