Microsoft®

# Microsoft® Visual Basic® 2010 Krok po kroku

*Michael Halvorson*

*przekład: Leszek Biolik, Janusz Machowski*

Microsoft® Visual Basic® 2010 Krok po kroku

© 2010 APN PROMISE Sp. z o. o.

Authorized translation of English edition of
**Microsoft® Visual Basic® 2010 Step by Step**
© 2010 Michael Halvorson

This translation is published and sold by permission of O'Reilly Media, Inc., which owns or controls of all rights to publish and sell the same.

APN PROMISE Sp. z o. o., biuro: ul. Kryniczna 2, 03-934 Warszawa
tel. +48 22 35 51 642, fax +48 22 35 51 699
e-mail: mspress@promise.pl

Wszystkie prawa zastrzeżone. Żadna część niniejszej książki nie może być powielana ani rozpowszechniana w jakiejkolwiek formie i w jakikolwiek sposób (elektroniczny, mechaniczny), włącznie z fotokopiowaniem, nagrywaniem na taśmy lub przy użyciu innych systemów bez pisemnej zgody wydawcy.

Microsoft, Microsoft Press, Access, ActiveX, Arc, Azure, DataTips, Excel, Expression, Halo, IntelliSense, Internet Explorer, MS, MSDN, MS-DOS, PowerPoint, SharePoint, Silverlight, SQL Server, Visual Basic, Visual C#, Visual C++, Visual InterDev, Visual Studio, Windows, Windows Azure, Windows Server, Windows Vista oraz Zoo Tycoon są zarejestrowanymi znakami towarowymi Microsoft Corporation.

Wszystkie inne nazwy handlowe i towarowe występujące w niniejszej publikacji mogą być znakami towarowymi zastrzeżonymi lub nazwami zastrzeżonymi odpowiednich firm odnośnych właścicieli.

Przykłady firm, produktów, osób i wydarzeń opisane w niniejszej książce są fikcyjne i nie odnoszą się do żadnych konkretnych firm, produktów, osób i wydarzeń. Ewentualne podobieństwo do jakiejkolwiek rzeczywistej firmy, organizacji, produktu, nazwy domeny, adresu poczty elektronicznej, logo, osoby, miejsca lub zdarzenia jest przypadkowe i niezamierzone.

APN PROMISE Sp. z o. o. dołożyła wszelkich starań, aby zapewnić najwyższą jakość tej publikacji. Jednakże nikomu nie udziela się rękojmi ani gwarancji.
APN PROMISE Sp. z o. o. nie jest w żadnym wypadku odpowiedzialna za jakiekolwiek szkody będące następstwem korzystania z informacji zawartych w niniejszej publikacji, nawet jeśli APN PROMISE została powiadomiona o możliwości wystąpienia szkód.

ISBN: 978-83-7541-065-5

Przekład: Leszek Biolik, Janusz Machowski
Redakcja: Marek Włodarz
Korekta: Ewa Swędrowska
Skład i łamanie: MAWart Marek Włodarz

*Dla Henry'ego*

# Spis treści

**Podziękowania** . . . . . xi

**Wstęp** . . . . . xiii

Wersje języka Visual Basic . . . . . xiv
Od czego najlepiej zacząć? . . . . . xvi
Wymagania dotyczące sprzętu i oprogramowania . . . . . xvii
Oprogramowanie w wersji testowej . . . . . xvii
Korzystanie z obrazu dysku CD . . . . . xviii
Odinstalowanie plików pomocniczych . . . . . xxiv
Konwencje przyjęte w książce . . . . . xxv
Przydatne łącza i adresy . . . . . xxvi
Wsparcie dla tej książki . . . . . xxvi

**Część I Wprowadzenie do Microsoft Visual Basic 2010**

**1 Poznawanie środowiska Visual Studio Integrated Development Environment** . . . . . 3

Środowisko programowania Visual Studio . . . . . 4
Narzędzia Visual Studio . . . . . 8
Designer . . . . . 10
Uruchamianie programu Visual Basic . . . . . 11
Okno Properties . . . . . 13
Przenoszenie i zmienianie rozmiarów narzędzi programistycznych . . . . . 17
Przenoszenie i zmienianie rozmiarów okien narzędziowych . . . . . 18
Dokowanie okien narzędziowych . . . . . 19
Ukrywanie okien narzędziowych . . . . . 21
Przełączanie się między otwartymi plikami i narzędziami przy użyciu funkcji IDE Navigator . . . . . 22
Otwieranie przeglądarki sieci Web w Visual Studio . . . . . 23
Uzyskiwanie pomocy . . . . . 24
Zarządzanie ustawieniami systemu pomocy . . . . . 25
Korzystanie z klawisza F1 . . . . . 26
Dopasowywanie ustawień IDE do opisywanych ćwiczeń . . . . . 29
Ustawianie IDE do programowania w języku Visual Basic . . . . . 29
Sprawdzanie ustawień projektu i kompilatora . . . . . 31
O krok dalej: wyjście z programu Visual Studio . . . . . 34
Krótkie podsumowanie rozdziału 1 . . . . . 35

**2 Pisanie pierwszego programu** . . . . . 37

Lucky Seven: nasz pierwszy program Visual Basic . . . . . 38
Etapy programowania . . . . . 38

Tworzenie interfejsu użytkownika . . . . . 39
Ustawianie właściwości . . . . . 45
Właściwości pola obrazu . . . . . 49
Pisanie kodu . . . . . 52
Rzut oka na procedurę *Button1_Click* . . . . . 57
Uruchamianie aplikacji Visual Basic . . . . . 59
Przykładowe projekty na dysku . . . . . 60
Budowanie pliku wykonywalnego . . . . . 61
Wdrażanie aplikacji . . . . . 63
O krok dalej: Rozbudowa programu . . . . . 64
Krótkie podsumowanie rozdziału 2 . . . . . 66

**3 Korzystanie z kontrolek przybornika Toolbox** . . . . . 67

Podstawowe wykorzystanie kontrolek: program „Hello World" . . . . . 68
Używanie kontrolki *DateTimePicker* . . . . . 73
Program „Birthday" . . . . . 73
Kontrolki do pobierania danych wejściowych . . . . . 78
Używanie pól grupy i przycisków opcji . . . . . 81
Przetwarzanie danych wejściowych przy użyciu pól listy . . . . . 84
Kilka słów na temat terminologii . . . . . 88
O krok dalej: Używanie kontrolki *LinkLabel* . . . . . 91
Krótkie podsumowanie rozdziału 3 . . . . . 95

**4 Używanie menu, pasków narzędzi i okien dialogowych** . . . . . 97

Dodawanie menu przy użyciu kontrolki *MenuStrip* . . . . . 98
Dodawanie klawiszy dostępu do poleceń menu . . . . . 100
Przetwarzanie wyborów dokonanych w menu . . . . . 102
Dodawanie pasków narzędzi za pomocą kontrolki *ToolStrip* . . . . . 108
Używanie kontrolek okien dialogowych . . . . . 111
Procedury zdarzeń zarządzające popularnymi oknami dialogowymi . . . 112
O krok dalej: Przypisywanie klawiszy skrótu do menu . . . . . 118
Krótkie podsumowanie rozdziału 4 . . . . . 121

**Część II Podstawy programowania**

**5 Zmienne i formuły Visual Basic oraz .NET Framework** . . . . . 125

Anatomia instrukcji programu Visual Basic . . . . . 125
Używanie zmiennych do zapamiętywania informacji . . . . . 126
Rezerwowanie miejsca na zmienne: instrukcja ***Dim*** . . . . . 126
Jawne deklarowanie zmiennych . . . . . 128
Używanie zmiennych w programie . . . . . 129
Używanie zmiennych do przechowywania danych wejściowych . . . . . 133
Używanie zmiennych do przekazywania danych wyjściowych . . . . . 135
Praca z różnymi typami danych . . . . . 137
Stałe: zmienne, które się nie zmieniają . . . . . 144
Używanie operatorów w programach Visual Basic . . . . . 146
Podstawowe operatory matematyczne: **+**, **–**, ***** i **/** . . . . . 146

Używanie operatorów zaawansowanych: ****, ***Mod*, ^** i **&** . . . . . 150
Korzystanie z metod klasy Math w .NET Framework . . . . . 155
O krok dalej: ustalanie kolejności działań . . . . . 158
Używanie nawiasów w formułach . . . . . 159
Krótkie podsumowanie rozdziału 5 . . . . . 160

**6 Używanie struktur decyzyjnych** . . . . . 161

Programowanie sterowane zdarzeniami . . . . . 162
Używanie wyrażeń warunkowych . . . . . 163
Struktura decyzyjna *If ... Then* . . . . . 164
Sprawdzanie kilku warunków w strukturze decyzyjnej ***If ... Then*** . . . . 165
Używanie operatorów logicznych w wyrażeniach warunkowych . . . . . 170
Operatory „skrótowe" ***AndAlso*** i ***OrElse*** . . . . . 172
Struktura decyzyjna *Select Case* . . . . . 174
Używanie operatorów porównania w strukturze Select Case . . . . . 176
O krok dalej: Wykrywanie zdarzeń myszy . . . . . 180
Krótkie podsumowanie rozdziału 6 . . . . . 182

**7 Używanie pętli i zegarów** . . . . . 183

Pisanie pętli *For ... Next* . . . . . 184
Używanie zmiennej – licznika w wielowierszowej kontrolce *TextBox* . . . . 185
Tworzenie złożonych pętli *For ... Next* . . . . . 188
Używanie licznika o większym zasięgu . . . . . 191
Pisanie pętli *Do* . . . . . 194
Unikanie pętli nieskończonych . . . . . 195
Kontrolka *Timer* . . . . . 198
Tworzenie zegara cyfrowego przy użyciu kontrolki *Timer* . . . . . 199
Używanie obiektu *Timer* do ustawiania limitu czasu . . . . . 202
O krok dalej: Wstawianie fragmentów kodu . . . . . 206
Krótkie podsumowanie rozdziału 7 . . . . . 209

**8 Debugowanie programów Visual Basic** . . . . . 211

Znajdywanie i poprawianie błędów . . . . . 212
Trzy rodzaje błędów . . . . . 212
Rozpoznawanie błędów logicznych . . . . . 213
ABC debugowania: Korzystanie z trybu debugowania . . . . . 214
Śledzenie zmiennych za pomocą okna Watch . . . . . 220
Wizualizatory: Narzędzia do debugowania wyświetlające dane . . . . . 222
Używanie okien Immediate i Command . . . . . 224
Przełączanie się do okna Command . . . . . 226
O krok dalej: Usuwanie punktów przerwania . . . . . 227
Krótkie podsumowanie rozdziału 8 . . . . . 227

**9 Wychwytywanie błędów przy użyciu strukturalnej obsługi błędów** . . . . . 229

Obsługa błędów przy użyciu instrukcji *Try ... Catch* . . . . . 230
Kiedy używać obsługi błędów . . . . . 230
Definiowanie pułapki: blok kodu ***Try ... Catch*** . . . . . 232

Błędy związane ze ścieżką i napędem dysku . . . . . 232
Pisanie obsługi błędu dotyczącego napędu dysku . . . . . 236
Używanie klauzuli *Finally* do przeprowadzania zadań porządkowania . . 237
Bardziej złożona obsługa błędu *Try ... Catch* . . . . . 239
Obiekt ***Exception*** . . . . . 239
Określanie liczby powtórzeń . . . . . 243
Używanie zagnieżdżonych bloków ***Try ... Catch*** . . . . . 245
Porównanie obsługi błędów z metodami programowania defensywnego . . . . . 246
O krok dalej: instrukcja *Exit Try* . . . . . 247
Krótkie podsumowanie rozdziału 9 . . . . . 249

**10 Tworzenie modułów i procedur** . . . . . 251

Korzystanie z modułów . . . . . 251
Tworzenie modułu . . . . . 252
Używanie zmiennych publicznych . . . . . 255
Tworzenie procedur . . . . . 260
Pisanie procedur Function . . . . . 261
Składnia procedury Function . . . . . 262
Wywoływanie procedury Function . . . . . 263
Używanie funkcji do przeprowadzania obliczeń . . . . . 263
Pisanie procedur Sub . . . . . 267
Składnia procedury Sub . . . . . 267
Wywołanie procedury Sub . . . . . 268
Używanie procedury Sub do wprowadzania danych . . . . . 269
O krok dalej: przekazywanie argumentów poprzez wartość i poprzez odniesienie . . . . . 274
Krótkie podsumowanie rozdziału 10 . . . . . 275

**11 Używanie tablic do zarządzania danymi liczbowymi i tekstowymi** . . . . . 277

Korzystanie z tablic zmiennych . . . . . 278
Tworzenie tablicy . . . . . 278
Deklarowanie tablicy o stałym rozmiarze . . . . . 279
Rezerwowanie pamięci . . . . . 280
Korzystanie z elementów tablicy . . . . . 281
Deklarowanie tablicy i przypisywanie jej wartości początkowych . . . . 282
Tworzenie tablicy o stałym rozmiarze do przechowywania temperatur . . . . . 284
Tworzenie tablicy dynamicznej . . . . . 288
Rezerwowanie zawartości tablicy za pomocą instrukcji *ReDim Preserve* . 291
Użycie instrukcji ***ReDim*** w przypadku tablic trójwymiarowych . . . . . 292
O krok dalej: przetwarzanie dużych tablic przy użyciu metod klasy *Array* . . . . . 293
Klasa ***Array*** . . . . . 293
Krótkie podsumowanie rozdziału 11 . . . . . 299

**12 Korzystanie z kolekcji** . . . . . 301
Posługiwanie się kolekcjami obiektów . . . . . 301
Odwoływanie się do obiektów kolekcji . . . . . 302
Pisanie pętli ***For Each ... Next*** . . . . . 302
Eksperymenty z obiektami kolekcji ***Controls*** . . . . . 303
Używanie właściwości ***Name*** w pętli ***For Each ... Next*** . . . . . 306
Tworzenie własnych kolekcji . . . . . 308
Deklarowanie nowych kolekcji . . . . . 309
O krok dalej: kolekcje VBA . . . . . 313
Wprowadzenie makra przeznaczonego dla programu Word . . . . . 315
Krótkie podsumowanie rozdziału 12 . . . . . 316
**13 Metody przetwarzania plików tekstowych i ciągów** . . . . . 317
Odczytywanie plików tekstowych . . . . . 317
Obszar nazw ***My*** . . . . . 318
Klasa ***StreamReader*** . . . . . 320
Używanie metody ***ReadAllText*** . . . . . 322
Zapisywanie plików tekstowych . . . . . 325
Metoda ***WriteAllText*** . . . . . 326
Klasa ***StreamWriter*** . . . . . 326
Stosowanie metody ***WriteAllText*** . . . . . 327
Przetwarzanie ciągów znaków za pomocą klasy *String* . . . . . 331
Sortowanie tekstu . . . . . 334
Korzystanie z kodów ASCII . . . . . 335
Sortowanie ciągów w polu tekstowym . . . . . 336
Analiza kodu programu Sort Text . . . . . 339
Ochrona tekstu za pomocą podstawowego szyfrowania . . . . . 341
O krok dalej: użycie operatora *Xor* . . . . . 345
Analiza kodu programu szyfrowania . . . . . 347
Krótkie podsumowanie rozdziału 13 . . . . . 349

**Część III Projektowanie interfejsu użytkownika**

**14 Zarządzanie formularzami i kontrolkami podczas wykonywania kodu** . . . . . 355
Dodawanie nowych formularzy do programu . . . . . 355
Jak korzystać z formularzy . . . . . 356
Praca z wieloma formularzami . . . . . 356
Używanie właściwości ***DialogResult*** w formularzu wywołującym . . . . . 363
Rozmieszczanie formularzy na pulpicie . . . . . 364
Minimalizowanie, maksymalizowanie i przywracanie wielkości okien . 368
Dodawanie kontrolek do formularza w trakcie działania programu . . . . . 369
Rozmieszczanie kontrolek na formularzu . . . . . 372
O krok dalej: określenie obiektu startowego . . . . . 376
Krótkie podsumowanie rozdziału 14 . . . . . 378

**15 Dodawanie grafiki i animacji** . . . . . 381
Dodawanie grafiki przy użyciu obszaru nazw *System.Drawing* . . . . . 382
Korzystanie z systemu współrzędnych formularza . . . . . 382
Klasa ***System.Drawing.Graphics*** . . . . . 383
Stosowanie w formularzu procedury zdarzenia Paint . . . . . 384
Tworzenie animacji w programach . . . . . 386
Przesuwanie obiektów na formularzu . . . . . 386
Właściwość ***Location*** . . . . . 387
Tworzenie animacji przy użyciu obiektu ***Timer*** . . . . . 388
Powiększanie i zmniejszanie obiektów w trakcie działania programu . . . 393
O krok dalej: zmiana przezroczystości formularza . . . . . 395
Krótkie podsumowanie rozdziału 15 . . . . . 397

**16 Dziedziczenie formularzy i tworzenie klas bazowych** . . . . . 399
Dziedziczenie formularza przy użyciu narzędzia Inheritance Picker . . . . . 400
Tworzenie własnych klas bazowych . . . . . 406
Dodawanie do projektu nowej klasy . . . . . 407
O krok dalej: dziedziczenie klasy bazowej . . . . . 415
Krótkie podsumowanie rozdziału 16 . . . . . 419

**17 Korzystanie z drukarek** . . . . . 421
Używanie klasy *PrintDocument* . . . . . 421
Drukowanie tekstu z obiektu pola tekstowego . . . . . 426
Drukowanie wielostronicowych plików tekstowych . . . . . 430
O krok dalej: dodawanie okien dialogowych Print Preview i Page Setup 437
Krótkie podsumowanie rozdziału 17 . . . . . 443

**Część IV Programowanie dotyczące baz danych i sieci Web**

**18 Wprowadzenie do ADO.NET** . . . . . 447
Programowanie baz danych z użyciem ADO.NET . . . . . 447
Terminologia baz danych . . . . . 448
Praca z bazą danych Access . . . . . 450
Okno Data Sources . . . . . 459
Używanie kontrolek powiązanych do wyświetlania informacji z bazy danych . . . . . 464
O krok dalej: instrukcje SQL, LINQ i filtrowanie danych . . . . . 468
Krótkie podsumowanie rozdziału 18 . . . . . 473

**19 Udostępnianie danych przy użyciu kontrolki *DataGridView*** . . . . 475
Stosowanie kontrolki *DataGridView* do wyświetlania rekordów bazy danych . . . . . 475
Formatowanie komórek kontrolki *DataGridView* . . . . . 486
Dodawanie drugiego obiektu widoku siatki danych . . . . . 489
O krok dalej: aktualizowanie źródłowej bazy danych . . . . . 493
Krótkie podsumowanie rozdziału 19 . . . . . 497

**20 Tworzenie witryn i stron sieci Web za pomocą narzędzi Visual Web Developer i ASP.NET** . . . . . 499

Omówienie ASP.NET . . . . . 500
Strony sieci Web a formularze systemu Windows . . . . . 501
Kontrolki serwera . . . . . 502
Kontrolki HTML . . . . . 502
Tworzenie witryn sieci Web za pomocą narzędzia Visual Web Developer . . . . . 503
Wymagania oprogramowania ASP.NET . . . . . 503
Korzystanie z narzędzia Web Page Designer . . . . . 507
Dodawanie kontrolek serwera do witryny sieci Web . . . . . 510
Tworzenie procedur zdarzeń dla kontrolek stron sieci Web . . . . . 512
Dostosowywanie szablonu witryny sieci Web . . . . . 518
Wyświetlanie rekordów bazy danych na stronie sieci Web . . . . . 520
O krok dalej: definiowanie tytułów witryn sieci Web w przeglądarce Internet Explorer . . . . . 528
Krótkie podsumowanie rozdziału 20 . . . . . 531

**Dodatek Gdzie można znaleźć dodatkowe informacje** . . . . . 533

Witryny sieci Web dotyczące języka Visual Basic . . . . . 533
Witryny wideo . . . . . 535
Książki poświęcone programowaniu przy użyciu języka Visual Basic i programu Visual Studio . . . . . 536
Programowanie w języku Visual Basic . . . . . 536
Microsoft .NET Framework . . . . . 536
Tworzenie aplikacji bazodanowych z użyciem ADO.NET . . . . . 537
Programowanie witryn sieci Web za pomocą ASP.NET . . . . . 538
Programowanie dla pakietu Office . . . . . 538
Książki poświęcone ogólnym zagadnieniom programowania i informatyki . . . . . 538

**Indeks** . . . . . 541

**O autorze** . . . . . 569

# Podziękowania

Pisanie książki poświęconej zagadnieniom programowania jest fascynujące, ponieważ cały proces rozpoczyna się dużo wcześniej, zanim rzeczywiście ukończone zostaną prace nad oprogramowaniem. Autorzy spotykają się z projektantami oprogramowania i wydawcami książek, analizują specyfikacje produktu i wczesne jego wydania, przeglądają komentarze i opinie Czytelników dotyczące poprzednio oferowanych wersji, opracowują harmonogram pisania książki i rozpoczynają sprawdzanie swoich pomysłów w oparciu o wersje beta produktu. Ten interaktywny proces pozwala skupiać się na istotnych kwestiach i trwa bez przerwy (przy narastającym zapale), aż do ukończenia prac nad oprogramowaniem i przesłania ostatecznej wersji książki do drukarni.

Wydawnictwo Microsoft Press to idealne miejsce do tworzenia książek dotyczących programowania komputerów. Na każdym etapie procesu wydawniczego utalentowani członkowie zespołu współpracują razem, by doskonalić wartościowe zasoby i pogłębiać kontakty techniczne, tworzyć nowatorskie strategie wdrożeniowe produktu, zgłębiać niewidoczne możliwości pojawiających się technologii i dobierać najwłaściwsze środki ich opisu. W ciągu kilku lat, to dynamiczne i innowacyjne środowisko wydawnicze w sposób znaczący korzystnie przyczyniło się do rozwoju książki *Microsoft Visual Basic 2010 Krok po kroku*, będącej już ósmym wydaniem tej pozycji.

Pragnę gorąco podziękować za wsparcie i wskazówki osobom, które pomogły zaplanować, napisać, sprawdzić i wydać tę książkę (w kolejności, w jakiej współpracowałem): Ben Ryan, Devon Musgrave, Valerie Woolley, Susan McClung i Christian Holdener. W szczególności dziękuję Valerie Woolley za zapał, dzięki któremu mogłem napisać książkę zgodnie z harmonogramem i podtrzymywaniu w przekonaniu, że nasza książką dobrze wkomponuje się w serię wydawniczą *Krok po kroku,* z której tak dobrze jest znane wydawnictwo Microsoft Press. Jestem także niezwykle wdzięczny zespołowi projektowemu Microsoft Visual Studio 2010 za udostępnienie wersji beta i wersji RC produktu.

Jak zawsze, chciałbym okazać głęboką wdzięczność i uczucie mojej rodzinie za stałe wsparcie podczas pisania projektów i prac naukowych.

# Wstęp

Microsoft Visual Basic 2010 to istotna aktualizacja i ulepszenie popularnego języka programowania i kompilatora – Visual Basic, czyli technologii, którą z przyjemnością wykorzystują miliony programistów na całym świecie. Visual Basic 2010 nie jest produktem autonomicznym, lecz kluczowym składnikiem produktu Microsoft Visual Studio 2010 – kompletnego systemu projektowego, który pozwala tworzyć doskonałe aplikacje dla systemu Windows, sieci Web, urządzeń przenośnych, a także hostów w innych środowiskach. Niezależnie od tego, czy zakupimy jedną z wersji komercyjnych Visual Studio 2010 (opisanych w dalszej części Wstępu), czy pobierzemy darmową wersję Visual Basic 2010 Express w celach testowych, praca będzie ekscytująca. Najnowsze funkcje języka Visual Basic zwiększają naszą produktywność i sprawność programistyczną w szczególności, jeśli zajmujemy się integracją informacji pochodzących z baz danych, multimediami oraz stronami i witrynami sieci Web. Ponadto istotna korzyść z poznania języka Visual Basic i środowiska Visual Studio Integrated Development Environment (IDE) polega na tym, że możemy wykorzystywać wiele z tych samych narzędzi, pisząc programy przy użyciu Microsoft Visual C++ 2010, Microsoft Visual C# 2010, Microsoft Visual Web Developer 2010 czy innych popularnych produktów.

Książka *Microsoft Visual Basic 2010 Krok po kroku* to kompletne wprowadzenie do programowania w języku Visual Basic przy użyciu jego najnowszej wersji – Visual Basic 2010. Ten praktyczny podręcznik opracowany został z uwzględnieniem różnych poziomów zaawansowania Czytelników i przy założeniu, że najskuteczniejszym sposobem nauczania jest wykonywanie konkretnych zadań. Najlepszą metodą doskonalenia umiejętności w zakresie tak złożonej technologii, jak Visual Basic, jest poznanie najważniejszych technik programowania poprzez starannie przygotowane kursy, które mogą być realizowane przez studenta zgodnie z jego własnym harmonogramem i tempem. Książka ta nie jest poradnikiem „od A do Z", zawierającym suche opisy i odniesienia. Jest to praktyczny podręcznik programowania, który daje Czytelnikowi kontrolę nad sposobem uczenia się, projektowania i realizacji zadań. W rezultacie początkujący mogą poznać podstawy programowania w języku Visual Basic osadzone w kontekście przydatnych, realnych aplikacji, a doświadczeni programiści języka Visual Basic szybko opanują najważniejsze narzędzia i metody udostępnione w najnowszej wersji Visual Basic 2010.

Uzupełnieniem tego podejścia jest struktura książki – 4 części tematyczne, 20 rozdziałów, 56 ćwiczeń typu procedury krok po kroku oraz przykładowe programy. Książka pozwala szybko i w zabawny sposób nauczyć się tworzenia w języku Visual Basic 2010 profesjonalnie wyglądających aplikacji, które działają pod kontrolą systemu Windows i w środowisku przeglądarek internetowych!

## Wersje języka Visual Basic

Na początek kilka słów o „naszym pochodzeniu". W latach od 1991 do 1998 firma Microsoft opracowała sześć wersji języka Visual Basic (wersje od 1.0 do 6.0), które zrewolucjonizowały projektowanie aplikacji dla systemu Windows poprzez spopularyzowanie programowania, którego motorem były zdarzenia i które oparte było na języku QuickBasic i środowisku IDE. Po okresie intensywnego rozwoju i wprowadzania istotnych innowacji, firma Microsoft wydała program Visual Basic .NET 2002, czyli obiektowy język programowania ściśle związany z Visual Basic, ale zaimplementowany na platformie Microsoft .NET Framework, kompletnego zbioru bibliotek kodu rozwiązań opracowanych tak, by mogły być używane przez większość nowych aplikacji uruchamianych w systemach Windows. Ulepszone wersje języka Visual Basic pojawiały się w latach 2003, 2005 i 2008, a język Visual Basic stał się komponentem środowiska Visual Studio. W tym czasie w produkcie istotnie rozwinięta została obsługa baz danych, sieci Internet i narzędzi projektowania zespołowego, a także nadal usprawniana była platforma .NET Framework. Obecnie język Visual Basic 2010 jest tak ściśle zintegrowany z oprogramowaniem Visual Studio, że jest dostępny jedynie jako komponent pakietu Visual Studio 2010, w skład którego wchodzi Visual C#, Visual C++, Visual Web Developer i inne narzędzia projektowe Microsoft .NET. Od roku 2005 zarówno język Visual Basic, jak i program Visual Studio nie są oznaczane przydomkiem „.NET", chociaż produkty te nadal bazują na technologii .NET Framework.

Produkt Visual Studio 2010 jest sprzedawany w różnych konfiguracjach: Professional, Premium, Ultimate i Express. W książce starano się zachować zgodność ze wszystkimi wersjami produktów Visual Basic 2010 i Visual Studio 2010, ale w szczególności z narzędziami i metodami dostępnymi w wersji Visual Studio 2010 Professional i Visual Basic 2010 Express. Pomimo że program Visual Basic 2010 jest podobny do wersji Visual Basic 2008, to jednak ze względu na istniejące różnice i wprowadzone ulepszenia lepiej jest wykonywać zamieszczone w książce ćwiczenia korzystając z nowej wersji oprogramowania – Visual Basic 2010.

**Uwaga** Oprogramowanie Visual Studio 2010 nie zostało zamieszczone na dołączonym do książki dysku CD. Dysk CD zawiera pliki używane w ćwiczeniach, przykładowe bazy danych i inne przydatne informacje wymagane do działania programu Visual Studio 2010 (sprzedawanego oddzielnie). Jeśli nie mamy oprogramowania Visual Studio, możemy pobrać darmową wersję Visual Basic 2010 Express, które zawiera zaskakująco dużą liczbę funkcji produktów Visual Studio Professional, Premium lub Ultimate (rzecz jasna nie wszystkie funkcje). W trakcie omawiania ćwiczeń pojawiają się czasem uwagi informujące, że dana funkcja nie jest dostępna w wersji Visual Basic 2010 Express. Jeśli korzystamy z wersji Visual Basic 2010 Express, podczas realizacji rozdziału 20 „Tworzenie witryn i stron sieci Web za pomocą narzędzi Visual Web Developer i ASP.NET", do przeprowadzenia ćwiczeń trzeba będzie pobrać oprogramowanie Visual Web Developer 2010 Express. Narzędzie Visual Web Developer dołączone jest tylko do wersji Visual Studio Professional, Premium i Ultimate.

## Pobieranie wersji programu Visual Basic 2010 Express

Jak już nadmieniono wcześniej, jeśli nie mamy wersji oprogramowania Visual Studio 2010 Professional, Visual Studio 2010 Premium lub Visual Studio 2010 Ultimate, możemy przeprowadzić większość ćwiczeń zamieszczonych w tej książce w oparciu o pobraną z sieci Internet darmową wersję Visual Basic 2010 Express. Dzięki temu mamy okazję nauczyć się programowania w języku Visual Basic i przekonać się samemu, czy warto zaktualizować oprogramowanie do pełnej wersji Visual Studio.

W celu pobrania programu Visual Basic 2010 Express należy wykonać następujące instrukcje:

1. Otwórz przeglądarkę sieci Web (na przykład Internet Explorer) i przejdź do strony to *http://www.microsoft.com/express*.
2. Pobierz program Visual Basic 2010 Express zgodnie z wyświetlanymi instrukcjami.

   W witrynie Express Web możemy porównać cechy produktu w wersji Express z pełnymi wersjami Visual Studio. Chociaż pomiędzy pełnymi wersjami a wersją Visual Basic 2010 Express istnieją istotne różnice, jednak różnice te nie mają znaczenia w kontekście metod poznawania podstawowych technik i funkcji programowania w języku Visual Basic. Ćwiczenia przeprowadzone w oparciu o wersję Express pozwolą zdecydować, czy potrzebna jest nam aktualizacją do jednej z pełnych wersji programu Visual Studio. Rozpocznijmy więc naukę programowania w języku Visual Basic i przekonajmy się, jak działa ten kurs!

# Od czego najlepiej zacząć?

Książka ma pomóc w opanowaniu kilku najważniejszych dziedzin i może być używana przez początkującego programistę, przez programistę, który chce zmienić język programowania lub chce zaktualizować wersję Visual Basic 2008. Poniższa tabela powinna pomóc w znalezieniu miejsca, od którego najlepiej zacząć czytanie książki.

| Jeśli... | Wykonaj następujące instrukcje |
|---|---|
| Jesteś początkującym programistą | **1.** Zainstaluj pliki do ćwiczeń zgodnie z opisem w części „Instalowanie i korzystanie z plików pomocniczych" zamieszczonej w dalszym fragmencie wstępu.<br>**2.** Opanuj podstawy języka Visual Basic 2010 przerabiając kolejno rozdziały od 1 do 17.<br>**3.** Zapoznaj się z częścią IV „Programowanie dotyczące baz danych i sieci Web" w stopniu odpowiadającym zainteresowaniom lub doświadczeniu. |
| Aktualizujesz wersję Visual Basic 2005 lub 2008 | **1.** Zainstaluj pliki do ćwiczeń zgodnie z opisem w części „Instalowanie i korzystanie z plików pomocniczych".<br>**2.** Opanuj tematykę rozdziałów od 1 do 4, przejrzyj rozdziały 5 do 17 i opanuj rozdziały od 18 do 20. |
| Przeprowadzasz aktualizację z wersji Visual Basic 6.0 | **1.** Zainstaluj pliki do ćwiczeń zgodnie z opisem w części „Instalowanie i korzystanie z plików pomocniczych".<br>**2.** Przeczytaj uważnie rozdziały od 1 do 4, by zapoznać się z nowymi funkcjami środowiska projektowego Visual Studio 2010.<br>**3.** Przejrzyj rozdziały od 5 do 13, by przypomnieć sobie podstawy programowania opartego na zdarzeniach, używanie zmiennych i tworzenie struktur decyzyjnych. Specjalną uwagę zwróć na rozdziały 5, 6, 9 i 12.<br>**4.** Opanuj kolejno tematykę rozdziałów 14 – 20, by poznać nowe funkcje języka Visual Basic 2010, dotyczące projektowania interfejsu użytkownika, tworzenia aplikacji korzystających z baz danych i sieci Web. |
| Korzystasz z książki po przerobieniu wszystkich rozdziałów | **1.** Stosuj indeks do wyszukiwania konkretnych informacji, a spis treści do wyszukania tematów ogólnych.<br>**2.** Przeglądaj tabele „Krótkie podsumowanie rozdziału", by szybko przypomnieć sobie sposoby realizacji najważniejszych zadań. Tematy w tabeli „Krótkie podsumowanie rozdziału" są wymienione w tej samej kolejności, w jakiej były omawiane w danym rozdziale. |

## Wymagania dotyczące sprzętu i oprogramowania

Spełnienie poniższych wymagań, dotyczących sprzętu i oprogramowania, konieczne jest do wykonania ćwiczeń prezentowanych w książce:

- System operacyjny Windows 7, Windows Vista, Windows XP, Windows Server 2003 lub Windows Server 2008
- Visual Studio 2010 (Professional, Premium, Ultimate) lub Visual Basic 2010 Express
- Procesor 1,6 GHz
- 1 GB pamięci RAM
- 3 GB wolnego miejsca na dysku twardym
- Napęd dysku twardego o obrotach min. 5400 RPM
- Karta grafiki zgodna z DirectX 9 funkcjonująca przy rozdzielczości 1024 × 768 lub wyższej
- Napęd DVD

Potrzebny będzie także dostęp do komputera na poziomie administracyjnym.

**Uwaga** Ćwiczenia w książce i pliki do ćwiczeń były testowane przy użyciu programów Visual Studio 2010 Professional i Visual Basic 2010 Express pod kontrolą systemu Windows 7. Jeśli używane są inne wersje Visual Studio 2010, mogą pojawić się pewne różnice. W szczególności, jeśli używana jest wersja Visual Basic 2010 Express, kilka funkcji nie będzie dostępnych w programie. Ponadto wszystkie zrzuty ekranu zamieszczone w książce zostały wykonane w systemie Windows 7. Jeśli stosowana jest inna wersja systemu Windows lub Windows Server, na niektórych ekranach mogą pojawić się różnice.

## Oprogramowanie w wersji testowej

Materiał tej książki był przeglądany i testowany w oparciu o wersję RC (Release Candidate) programu Visual Studio 2010. Wersja RC to ostatnia wersja przed pojawieniem się końcowej wersji Visual Studio 2010. Mamy nadzieję, że materiały prezentowane w książce są w pełni zgodne z końcowym wydaniem programu Visual Studio 2010 i Visual Basic 2010. Jeśli zauważone zostaną różnice lub błędy, wszystkie będą opublikowane w łatwo dostępnym artykule Microsoft Knowledge Base w Internecie (zob. informacje w części „Wsparcie techniczne” na końcu tego rozdziału).

# Korzystanie z obrazu dysku CD

Obraz dysku CD towarzyszącego książce jest dostępny na stronie wydawcy przy opisie książki w zakładce Dodatkowe informacje, pod adresem:

*http://www.ksiazki.promise.pl/aspx/produkt.aspx?pid=55288*

Na podstawie tego obrazu można wykonać fizyczny dysk CD lub zainstalować go jako napęd wirtualny. Dysk CD zawiera pliki pomocnicze potrzebne do przeprowadzenia ćwiczeń omawianych w książce. Na przykład podczas omawiania sposobów wyświetlania tabel bazy danych na formularzu za pomocą kontrolki *DataGridView* otwierany jest jeden z plików pomocniczych – baza danych uczelni nazwana Faculty2010.accdb – a następnie używane są narzędzia programu Visual Studio do uzyskania dostępu do tej bazy danych. Korzystając z plików pomocniczych nie marnujemy czasu na tworzenie plików, które nie są istotne dla tematyki omawianej w ćwiczeniu. W ten sposób możemy skoncentrować się na opanowywaniu metod programowania w języku Visual Basic 2010. Dzięki plikom i instrukcjom krok po kroku, poznawanie zagadnień odbywa się poprzez wykonywanie konkretnych zadań, co jest najprostszą i najbardziej efektywną metodą zdobywania umiejętności.

## Instalowanie plików pomocniczych

Zainstalowanie plików pomocniczych wymaga około 10 MB wolnego miejsca na dysku. Poniżej prezentowana jest procedura instalacji plików pomocniczych tak, by mogły być używane w trakcie ćwiczeń.

1. Umieść w napędzie dysk CD dołączony do tej książki.

**Uwaga** Automatycznie powinny zostać wyświetlone informacje dotyczące licencji EULA (End-User License Agreement). Jeśli informacje te nie są wyświetlane, możemy dwukrotnie kliknąć plik StartCD.exe (na dysku CD). Jeśli korzystamy z systemu Windows 7 lub Windows Vista, możemy kliknąć menu Komputer (w menu Start), dwukrotnie kliknąć ikonę napędu dysku CD, a następnie dwukrotnie kliknąć plik StartCD.exe.

2. Przejrzyj informacje umowy licencyjnej EULA. Aby zaakceptować warunki umowy, zaznacz opcję Accept, a następnie kliknij przycisk Next.

   Wyświetlone zostaje menu zawierające opcje dotyczące książki.
3. Kliknij opcję Install Practice Files.
4. Postępuj zgodnie z instrukcjami wyświetlanymi na ekranie.

**Uwaga** Najłatwiej przeprowadzać ćwiczenia omawiane w tej książce, jeśli zaakceptujemy domyślną lokalizację instalacji plików (folder C:\Vb10sbs). Jeśli lokalizacja ta zostanie zmieniona, będziemy musieli ręcznie modyfikować ścieżki w kilku plikach pomocniczych tak, by mogły być znalezione ważne składniki (na przykład plik z grafiką czy plik bazy danych).

5. Po zainstalowaniu plików usuń dysk CD z napędu i schowaj do opakowania.

   Jeśli zaakceptowane zostały ustawienia domyślne, na dysku twardym utworzony został folder C:\Vb10sbs, w którym zapisane zostały pliki pomocnicze. Dla każdego rozdziału utworzony został jeden folder, wewnątrz folderu C:\Vb10sbs (niektóre pliki są gotowymi projektami, a w przypadku innych plików wymagane będzie wprowadzenie pewnych instrukcji). Jeśli pojawiają się problemy przy uruchomieniu jakiegokolwiek pliku pomocniczego, przydatne informacje można znaleźć w książce, w tekście opisującym te pliki.

## Korzystanie z plików pomocniczych

W każdym rozdziale wyjaśniono, jak i kiedy korzystać z plików pomocniczych. Ilekroć zachodzi potrzeba użycia pliku pomocniczego, w książce umieszczone są instrukcje dotyczące otwierania tego pliku. Rozdziały zbudowane zostały w oparciu o scenariusze przypominające rzeczywiste projekty, dzięki czemu zdobyte umiejętności łatwo będziemy mogli wykorzystywać w pracy.

**Uwaga** W programie Visual Basic 2010 wykorzystywany jest nowy format plików projektów i rozwiązań i z tego względu nie będziemy mogli otwierać plików pomocniczych przeznaczonych dla tej książki przy użyciu starszych wersji oprogramowania Visual Basic czy Visual Studio. Aby sprawdzić, jaka wersja programu Visual Basic lub Visual Studio jest aktualnie używana, można w menu Help kliknąć polecenie About.

Program Visual Studio możemy bardzo łatwo dostosowywać, a jego konfiguracja pozwala na otwieranie i zapisywanie projektów różnymi metodami. Instrukcje w książce zasadniczo korzystają z ustawień domyślnych programu Visual Studio. Dodatkowe informacje na temat ustawień środowiska projektowego oraz ich wpływu na sposób pisania programów i używania plików pomocniczych znaleźć można w rozdziale 1 „Poznawanie środowiska Visual Studio Integrated Development Environment", w części opisującej dostosowywanie ustawień środowiska IDE do ćwiczeń.

Dla osób, które lubią znać wszystkie szczegóły, poniżej zamieszczono listę projektów Visual Basic dołączonych do dysku CD. Każdy projekt znajduje się w swoim własnym folderze i ma kilka plików pomocniczych. Przyjrzyjmy się, czym będziemy się zajmować!

| Projekt | Opis |
|---|---|
| **Rozdział 1** | |
| MusicTrivia | Prosty program zapraszający do kursu programowania i wyświetlający zdjęcie cyfrowe. |
| **Rozdział 2** | |
| Lucky7 | Nasz pierwszy program – gra będąca symulacją maszyny losującej Las Vegas Lucky Seven. |
| **Rozdział 3** | |
| Birthday | Program używa kontrolki *DateTimePicker* do pobierania danych. |
| CheckBox | Zaprezentowanie działania kontrolki *CheckBox* i jej właściwości. |
| Hello | Program Hello Word, który prezentuje kontrolki *Label* i *TextBox*. |
| List Box | Zaprezentowanie działania kontrolki *ListBox* gromadzącej dane wejściowe. |
| Radio Button | Zaprezentowanie działania kontrolki *RadioButton* gromadzącej dane wejściowe. |
| WebLink | Zaprezentowanie działania kontrolki *LinkLabel*, która otwiera przeglądarkę sieci Web w aplikacji Visual Basic. |
| **Rozdział 4** | |
| Menu | Prezentacja sposobów używania kontrolek okien dialogowych, pasków narzędzi i menu w Visual Studio. |
| **Rozdział 5** | |
| Advanced Math | Zaawansowane wykorzystanie operatorów dzielenia całkowitego, reszty z dzielenia, potęgowania i konkatenacji ciągów znakowych. |
| Basic Math | Podstawowe użycie operatorów dodawania, odejmowania, mnożenia i dzielenia. |
| Constant Tester | Użycie stałej do przechowywania ustalonej wielkości matematycznej. |
| Data Types | Zaprezentowanie podstawowych typów danych języka Visual Basic i ich użycia w pracy ze zmiennymi. |
| Framework Math | Zaprezentowanie klas i metod matematycznych platformy .NET Framework. |
| Input Box | Pobieranie danych za pomocą funkcji *InputBox*. |
| Variable Test | Deklarowanie i używanie zmiennych do przechowywania informacji. |
| **Rozdział 6** | |
| Select Case | Użycie struktury decyzyjnej *Select … Case* i kontrolki *ListBox* do wyświetlania komunikatu powitania w kilku językach. |
| User Validation | Użycie struktury decyzyjnej *If … Then … Else* i kontrolki *MaskedTextBox* do zarządzania procesem logowania. |

| Projekt | Opis |
|---|---|
| **Rozdział 7** | |
| Celsius Conversion | Konwersja temperatury ze skali Fahrenheita na Celsjusza za pomocą pętli *Do*. |
| Digital Clock | Prosty zegar cyfrowy prezentujący działanie kontrolki *Timer*. |
| For Loop | Prezentacja działania pętli *For … Next* do wyświetlenia tekstu w kontrolce *TextBox* i użycie funkcji *Chr* do zawijania tekstu. |
| For Loop Icons | Użycie w procedurze zdarzenia globalnej zmiennej typu licznik jako alternatywne działanie pętli. Wyświetlenie grafiki za pomocą kontrolki *PictureBox*. |
| Timed Password | Zaprezentowanie sposobu użycia kontrolki *Timer* do tworzenia programu logowania z ograniczonym czasem wprowadzania hasła. |
| Windows Version Snippet | Zaprezentowanie sposobu użycia polecenia Insert Snippet do wyświetlenia aktualnej wersji systemu Windows uruchomionego na komputerze klienckim. |
| **Rozdział 8** | |
| Debug Test | Symulowanie problemu w celu zaprezentowania możliwości debugowania narzędzi Visual Studio. |
| **Rozdział 9** | |
| Disc Drive Error | Przerywanie działania, jeśli dysk CD lub DVD zostaje nieprawidłowo użyty. Projekt ten służy jako baza do omówienia obsługi błędów w języku Visual Basic. |
| Disc Drive Handler | Pełna obsługa błędu ładowania pliku zaprezentowana w oparciu o składnię *Try … Catch*. |
| **Rozdział 10** | |
| Text Box Sub | Procedura Sub ogólnego przeznaczenia, która dodaje elementy do listy. |
| Track Wins | Ulepszona wersja programu losującego Lucky7 używanego w rozdziale 2, w której używane są zmienne publiczne i funkcje obliczania procentu wygranych gier. |
| **Rozdział 11** | |
| Array Class Sorts | Prezentacja sposobu tworzenia dużych tablic liczb całkowitych i operowania na nich. Ilustracja działania metod *Array.Sort* i *Array.Reverse* oraz użycia kontrolki *ProgressBar* do informowania użytkownika o postępie realizacji zadania sortowania. |
| Dynamic Array | Obliczenie średniej temperatury dla dowolnej liczby dni przy użyciu tablicy dynamicznej. |
| Fixed Array | Obliczenie średniej temperatury tygodniowej przy użyciu tablicy o stałym rozmiarze. |

| Projekt | Opis |
|---|---|
| **Rozdział 12** | |
| Controls Collection | Użycie pętli *For Each … Next* i kolekcji *Controls* w programie Visual Studio do przesuwania obiektów formularza. |
| URL Collection | Zaprezentowanie kolekcji zdefiniowanej przez użytkownika, która zawiera listę adresów URL (Uniform Resource Locator) stron ostatnio odwiedzanych przez użytkownika. |
| **Rozdział 13** | |
| Encrypt Text | Zaprezentowanie działania metod *Chr*, *Asc*, *Length*, *Substring* i *FileExists*, jak również schematu prostego szyfrowania tekstu w pliku. Nauka przydatnych metod przetwarzania tekstu. |
| Quick Note | Prosta aplikacja typu notatnik, na podstawie której zaprezentowane jest działanie właściwości *Clock.LocalTime*, metody *WriteAllText* oraz kontrolek *TextBox*, *MenuStrip* i *SaveFileDialog*. |
| Sort Text | Edytor plików tekstowych z paskiem menu, na podstawie którego prezentowane jest działanie poleceń Open, Close, Save As, Insert Date, Sort Text i Exit. Program zawiera moduł *ShellSort* sortowania tablic, który może być wykorzystywany w innych projektach. |
| Text Browser | Wyświetlenie zawartości pliku tekstowego w programie napisanym w języku Visual Basic. Zaprezentowanie działania poleceń menu i okna dialogowego, obsługi błędów *Try … Catch* i metody *ReadAllText*. Program służy jako podstawa dla innych programów w tym rozdziale. |
| Xor Encryption | Analiza działania klasy *StreamWriter* oraz metod *OpenTextFileWriter* i *ReadAllText* do zarządzania plikami. Użycie operatora *Xor* do szyfrowania plików za pomocą klucza wprowadzonego przez użytkownika. |
| **Rozdział 14** | |
| Add Controls | Zaprezentowanie sposobu dodawania kontrolek do formularza systemu Windows w trakcie działania programu przy użyciu kodu programu (a nie za pomocą narzędzia Designer). |
| Anchor and Dock | Użycie właściwości *Anchor* i *Dock* formularza do dostosowania położenia obiektów po uruchomieniu programu. |
| Desktop Bounds | Użycie właściwości *StartPosition* i *DesktopBounds* do określania położenia formularza po uruchomieniu programu. Zaprezentowanie działania właściwości *FormBorderStyle*, struktury *Rectangle* i metody *ShowDialog*. |
| Lucky Seven Help | Ulepszony program Lucky7 (Track Wins) używany w rozdziale 10, wzbogacony o drugi formularz, w którym wyświetlane są informacje pomocy. |

| Projekt | Opis |
|---|---|
| **Rozdział 15** | |
| Draw Shapes | Zaprezentowanie kilku przydatnych metod graficznych obszaru nazw *System.Drawing*, a w tym *DrawEllipse*, *FillRectangle* i *DrawCurve*. |
| Moving Icon | Animacja ikony na formularzu, przesuwanie ikony z góry na dół formularza, ilekroć kliknięty zostaje przycisk Move Down. |
| Transparent Form | Zaprezentowanie sposobów zmiany przezroczystości formularza za pomocą obiektu *Me* i właściwości *Opacity*. |
| Zoom In | Symulacja powiększania lub zbliżania się do obiektu (w ćwiczeniu używany jest obraz Ziemi o dużej rozdzielczości). |
| **Rozdział 16** | |
| Form Inheritance | Użycie narzędzia Visual Studio Inheritance Picker do tworzenia formularza, który dziedziczy cechy i funkcje innego formularza. |
| Person Class | Zaprezentowanie sposobu działania nowych klas, właściwości i metod w projektach Visual Basic. Nowa klasa *Person* opisuje pracownika, przechowuje jego imię, nazwisko i datę urodzenia i zawiera metodę obliczania wieku pracownika. |
| **Rozdział 17** | |
| Print Dialogs | Prezentacja sposobów tworzenia okien dialogowych Print Preview i Page Setup. |
| Print File | Obsługa bardziej skomplikowanych zadań związanych z drukowaniem, jak drukowanie wielu stron tekstu z zawijaniem wierszy. Program zawiera fragmenty kodu, które mogą być używane w innych projektach. |
| Print Graphics | Drukowanie grafiki z programu napisanym w języku Visual Basic przy użyciu obsługi błędu, metody *Print* i *DrawImage*. |
| Print Text | Prezentacja sposobu drukowania prostego tekstu w programie napisanym w języku Visual Basic. |
| **Rozdział 18** | |
| ADO Faculty Form | Zaprezentowanie sposobu wykorzystywania ADO.NET do nawiązywania połączeń z bazą danych Microsoft Access 2007 i wyświetlania informacji pobranych z bazy. |
| **Rozdział 19** | |
| DataGridView Sample | Prezentacja sposobu używania kontrolki *DataGridView* do wyświetlania na formularzu wielu tabel. Ilustracja powiązań pasków nawigacji, zestawów danych i adapterów tabel z obiektami na formularzu. |

| Projekt | Opis |
|---|---|
| **Rozdział 20** | |
| Chap20 | Zaprezentowanie wykorzystania narzędzi Visual Web Developer i ASP.NET 4 do utworzenia w przeglądarce internetowej kalkulatora wysokości rat kredytu samochodowego, wyświetlenie informacji pomocy i rekordów z bazy danych. |

# Odinstalowanie plików pomocniczych

Poniższa procedura umożliwia usunięcie plików pomocniczych zapisanych na dysku przez program instalacyjny Visual Basic 2010 Step by Step. Po odinstalowaniu plików pomocniczych możemy ręcznie usuwać wszystkie pliki utworzonych samodzielnie projektów.

### Jeśli komputer działa pod kontrolą systemu Windows 7 lub Windows Vista:

1. W Panelu sterowania, w kategorii Programy kliknij polecenie Odinstaluj program.
2. Na liście programów zaznacz program Microsoft Visual Basic 2010 Step by Step, a następnie kliknij przycisk Odinstaluj.
3. Usuń pliki pomocnicze zgodnie z wyświetlanymi instrukcjami.

### Jeśli komputer działa pod kontrolą systemu Windows XP:

1. W Panelu sterowania otwórz aplet Dodaj lub usuń programy.
2. Na liście aktualnie zainstalowanych programów zaznacz pozycję Microsoft Visual Basic 2010 Step by Step. Kliknij przycisk Usuń.
3. Usuń pliki pomocnicze zgodnie z wyświetlanymi instrukcjami.

# Konwencje przyjęte w książce

Możemy zaoszczędzić trochę czasu, jeśli przed przystąpieniem do przeprowadzania ćwiczeń w tej książce zapoznamy się z przyjętym sposobem prezentowania informacji i stosowanymi elementami, które ułatwiają zrozumienie omawianych tematów. Poniżej wymienione zostały konwencje stylistyczne stosowane w książce oraz opis najważniejszych cech książki.

## Konwencje

- Nazwy wszystkich elementów programu – kontrolek, obiektów, funkcji, właściwości itp. – wyróżniono *kursywą*.
- Ćwiczenia praktyczne przedstawiono w postaci ponumerowanych list instrukcji (1, 2 itd.).
- Tekst, który musi być wpisany, wyróżniono **pogrubieniem**.
- Wykonując ćwiczenia, czasami możemy napotkać listy właściwości, które należy zdefiniować w programie Visual Studio. Wartość właściwości została ujęta cudzysłowami, ale nie należy ich wpisywać.
- Znak plus (+) pomiędzy dwoma nazwami klawiszy oznacza, że naciskać je należy jednocześnie. Na przykład, sformułowanie „naciśnij Alt+Tab" oznacza, że należy nacisnąć i przytrzymać klawisz Alt, a następnie nacisnąć klawisz Tab.
- Akapity wyróżnione etykietami Uwaga, Wskazówka i Ważne zawierają dodatkowe informacje lub alternatywne sposoby realizacji danej czynności. Przed kontynuowaniem ćwiczenia należy zapoznać się z tymi informacjami.

## Inne cechy

- Fragmenty tekstu wyróżniane są specjalnymi ramkami, a ich zawartość pozwala zdobyć wiedzę na temat specjalnych technik programowania, informacji uzupełniających lub funkcji powiązanych z omawianą tematyką, a często są też wyjaśnieniem trudnej terminologii czy sugestią dotyczącą dalszych obszarów analizy.
- Na końcu każdego rozdziału znajduje się fragment zatytułowany „O krok dalej", w którym w oparciu o przedstawione w rozdziale informacje pokazano uzupełnienia i rozszerzenia danej tematyki.
- Szybkie i zwięzłe przypomnienie sposobów wykonywania zadań omawianych w rozdziale znaleźć można w tabeli „Krótkie podsumowanie rozdziału". Po przeczytaniu całej książki tabele te mogą służyć także jako tematyczne referencje i szybkie przypomnienie metod realizacji konkretnych zadań.

# Przydatne łącza i adresy

Poniżej wymieniono witryny, które warto odwiedzić szukając pomocy dotyczącej oprogramowania Visual Studio 2010 czy innych tematów omawianych w książce.

## Wsparcie techniczne dla programu Visual Studio 2010

Poszukiwania odpowiedzi na pytania dotyczące programu Visual Studio 2010 warto rozpocząć od dwóch witryn firmy Microsoft:

- *http://msdn.microsoft.com/vbasic/* (strona główna Microsoft Visual Basic Developer Center)
- *http://www.microsoft.com/communities/* (społeczności techniczne związane z produktami i technologiami firmy Microsoft)

Obie witryny umożliwiają nawiązanie kontaktu z programistami zawodowo korzystającymi z języka Visual Basic, pracownikami firmy Microsoft, a także umożliwiają dostęp do blogów, grup dyskusyjnych, prezentacji, konsultacji technicznych i grup zainteresowań tematyką dotyczącą języka Visual Basic. Dodatkowe informacje na temat tych i innych zasobów znaleźć można w dodatku „Gdzie można znaleźć dodatkowe informacje".

# Wsparcie dla tej książki

Dołożono wszelkich starań dla zapewnienia dokładności książki i materiałów pomocniczych zamieszczonych na dysku CD. Zauważone błędy i ewentualne zmiany będą publikowane w artykule biblioteki Microsoft Knowledge Base. Firma Microsoft Press gromadzi informacje na temat książki i dołączonego dysku CD za pośrednictwem następującej witryny:

*http://www.microsoft.com/learning/support/books/*

Wszelkie komentarze, pytania lub pomysły dotyczące książki bądź materiałów pomocniczych lub kwestie, co do których nie można znaleźć rozwiązania w wymienionych powyżej witrynach, prosimy kierować do wydawnictwa Microsoft Press za pośrednictwem adresu poczty elektronicznej *mspinput@microsoft.com*.

Prosimy zwrócić uwagę, że pod tym adresem nie można uzyskać pomocy technicznej dla produktów firmy Microsoft, czym również nie zajmuje się autor książki.

CZĘŚĆ I

# Wprowadzenie do Microsoft Visual Basic 2010


**1** Poznawanie środowiska Visual Studio Integrated Development Environment *3*

**2** Pisanie pierwszego programu *37*

**3** Korzystanie z kontrolek przybornika Toolbox *67*

**4** Używanie menu, pasków narzędzi i okien dialogowych *97*


Część I zawiera przegląd podstawowych technik programowania w Microsoft Visual Basic 2010 oraz wprowadzenie do najczęściej wykorzystywanych narzędzi i funkcji. Dowiemy się, jak używać środowiska Visual Studio 2010 Integrated Development Environment (IDE), czyli zintegrowanego środowiska programowania obfitującego w różne narzędzia, okna i polecenia menu, a także zobaczymy krok po kroku, jak zbudować od podstaw i uruchomić kilka interesujących programów. Jest to punkt wyjścia dla wszystkich tych, którzy dopiero zaczynają programować w języku Visual Basic lub korzystali dotąd z jego wcześniejszych wersji.

W rozdziale 2 pokażemy, jak za pomocą kontrolek, formularzy, właściwości i odpowiedniego kodu zaprogramować automat do gier Lucky Seven (szczęśliwa siódemka). Rozdział 3 zawiera przegląd najbardziej przydatnych kontrolek z przybornika Toolbox, przydatnych do wyświetlania informacji lub udostępniania użytkownikowi możliwości wyboru, pobierania danych wejściowych, operowania datami i godzinami oraz łączenia się z siecią Web. Głównym tematem rozdziału 4 jest dodawanie menu, pasków narzędzi i okien dialogowych do programów Visual Basic, umożliwiających im uzyskiwanie charakteru komercyjnych aplikacji Windows.

ROZDZIAŁ 1

# Poznawanie środowiska Visual Studio Integrated Development Environment

**Po ukończeniu tego rozdziału Czytelnik będzie potrafił:**

- Używać środowiska Visual Studio Integrated Development Environment.
- Otwierać i uruchamiać programy Visual Basic.
- Zmieniać ustawienia właściwości.
- Przenosić, skalować, dokować i automatycznie ukrywać okna narzędzi.
- Używać funkcji IDE Navigator.
- Otwierać przeglądarkę sieci Web w Visual Studio.
- Korzystać z systemu pomocy i zarządzać jego ustawieniami.
- Dostosowywać ustawienia IDE do procedur „krok po kroku" opisanych w tej książce.

Możemy przystąpić do pracy z Microsoft Visual Studio 2010? W tym rozdziale zdobędziemy umiejętności potrzebne do uruchamiania i wykorzystywania zintegrowanego środowiska programowania, czyli Visual Studio 2010 Integrated Development Environment (IDE) – środowiska, w którym odbywa się pisanie programów Microsoft Visual Basic. Ten rozdział powinni przeczytać zarówno ci, którzy dopiero zaczynają programować w języku Visual Basic, jak i ci, którzy używali już wcześniejszych wersji Visual Basic lub Visual Studio.

W tym rozdziale poznamy podstawowe polecenia menu oraz procedury programowania w Visual Studio. Otworzymy i uruchomimy prosty program Visual Basic o nazwie Music Trivia; zmienimy ustawienie programowe zwane *właściwością*; ponadto poćwiczymy przenoszenie, skalowanie, dokowanie i ukrywanie okien narzędziowych. Dowiemy się również, jak przełączać się między plikami i narzędziami za pomocą funkcji IDE Navigator, jak otwierać przeglądarkę sieci Web z poziomu Visual Studio, jak uzyskiwać więcej informacji przy użyciu dokumentacji systemu pomocy w trybie online i jak dostosowywać IDE do zamieszczonych w tej książce instrukcji „krok po kroku". Są to typowe zadania wykonywane podczas większości sesji programowania z użyciem Visual Studio i wkrótce powinny stać się odruchowe dla każdego Czytelnika (o ile jeszcze to nie nastąpiło).

# Środowisko programowania Visual Studio

Na początek krótka uwaga do Czytelników znających Visual Studio 2008: chociaż wersja Visual Studio 2010 ma wiele wewnętrznych ulepszeń, jej środowisko IDE jest w dużym stopniu takie samo, jak w wersji Visual Studio 2008. Ponieważ jednak niektórzy mogą nie znać Visual Studio, w tym rozdziale omówimy jego podstawy. Osoby te powinny również wiedzieć, że chociaż w książce omawiany jest Visual Basic, większość funkcji dostępnych w środowisku Visual Studio IDE dotyczy w równym stopniu wersji Visual Basic, Microsoft Visual C++, jak i Microsoft Visual C#. Wszystkie te programy (a właściwie narzędzia kompilacji) oferują takie samo środowisko IDE, które można teraz sprawdzić uruchamiając Visual Studio i przyglądając się temu produktowi.

**Ważne** Krótka uwaga. Jeśli nie zainstalowałeś jeszcze plików ćwiczeniowych dołączonych do tej książki, zrób to teraz, ponieważ wkrótce będziemy z nich korzystać. Zajrzyj na chwilę do podrozdziałów „Od którego miejsca najlepiej zacząć" oraz „Dysk CD i pliki ćwiczeniowe" we Wstępie do książki i zgodnie z instrukcją zainstaluj pliki (dobrze jest umieścić pliki z projektami i związane z nimi podfoldery w folderze C:\Vb10sbs). Trzeba mieć również zainstalowany w swoim komputerze program Visual Studio 2010, na przykład wydanie Visual Studio 2010 Professional.(większość podawanych ćwiczeń da się również wykonać w wersji Visual Studio 2010 Express). Po skończeniu instalacji można wrócić do lektury rozdziału.

## Uruchomienie Visual Studio 2010

1. Na pasku zadań systemu Windows kliknij przycisk Start, potem Wszystkie programy, a następnie folder Microsoft Visual Studio 2010.

   Na ekranie pojawi się lista folderów i ikon w folderze Microsoft Visual Studio 2010.

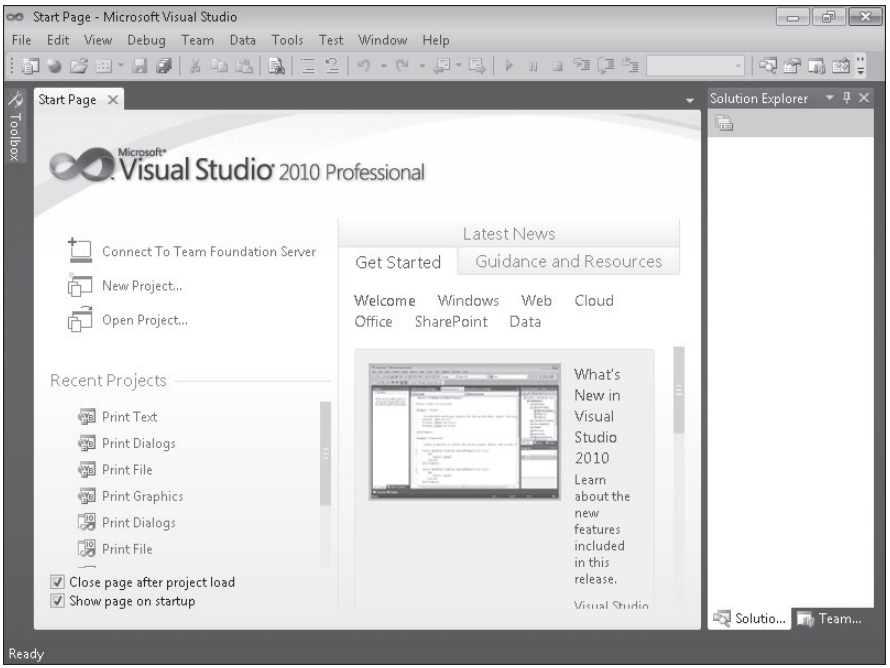
2. Kliknij ikonę Microsoft Visual Studio 2010.

**Wskazówka** Jeśli jest używana wersja Visual Basic 2010 Express, należy kliknąć ikonę Microsoft Visual Basic 2010 Express.

Jeśli jest to pierwsze uruchomienie Visual Studio, program musi przez chwilę skonfigurować środowisko. Gdy pojawi się monit dotyczący preferencji programowania, należy wybrać ustawienia dla języka Visual Basic.

Po uruchomieniu programu Visual Studio na ekranie pojawia się środowisko programowania z wieloma menu, narzędziami i oknami składowymi, jak to jest pokazane na rysunku (okna te są czasem nazywane *oknami narzędziowymi*). Powinna być również widoczna strona startowa (Start Page) ze zbiorem kart zawierających łącza, wskazówki i zasoby do nauki, wiadomości oraz opcje projektu. Strona startowa jest wyczerpującym źródłem informacji o projekcie, jak również o zasobach

oferowanych przez społeczność programistów Visual Basic. Jest to jedna ze ścieżek otrzymywania nowych informacji o Visual Studio po zakupie tego oprogramowania (zamieszczony tu zrzut ekranu jest prawdopodobnie mniej szczegółowy niż u Czytelnika, ale wynika to z zastosowania rozdzielczości 800 x 600 w celu zapewnienia większej czytelności tekstu).

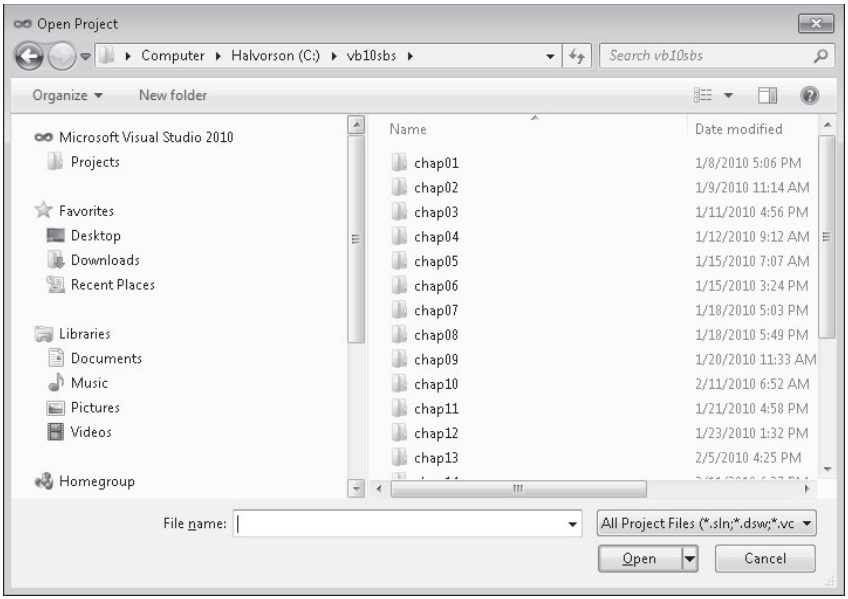



Pierwszą rzeczą, jaką robi większość programistów po uruchomieniu Visual Studio, jest otwarcie jakiegoś istniejącego projektu – bądź to gotowego już rozwiązania, nad którym chcą jeszcze popracować, bądź to aktualnie tworzonego projektu. Zadaniem Czytelnika będzie otwarcie projektu specjalnie opracowanego przez autora – programu Music Trivia.

## Otwarcie projektu Visual Basic

1. Na stronie Start Page kliknij łącze Open Project.

   Na ekranie pojawi się okno dialogowe Open Project, pokazane na kolejnym rysunku (okno to można również wyświetlić wybierając polecenie Open Project z menu File lub naciskając klawisze CTRL+O). Nawet dla kogoś, kto nigdy dotąd nie używał programu Visual Studio, okno dialogowe Open Project powinno być zrozumiałe, ponieważ przypomina okno dialogowe Otwórz z programu Microsoft Office Word lub Microsoft Office Excel.




**Wskazówka** W oknie dialogowym Open Project, z lewej strony, widnieje kilka lokalizacji. Szczególnie przydatny jest folder Projects w ramach Microsoft Visual Studio 2010. Standardowo program Visual Studio zapisuje projekty właśnie w tym folderze, zakładając dla każdego z nich oddzielny podfolder. Do organizacji naszych projektów ćwiczeniowych będziemy jednak używać innego folderu, jak to zostanie opisane poniżej. W zależności od konfiguracji komputera i systemu operacyjnego mogą być również dostępne inne lokalizacje, takie jak Favorites (Ulubione) i Libraries (Biblioteki) (zrzuty ekranu zamieszczone w tej książce pochodzą z systemu Windows 7).

**2.** Przejdź do folderu C:\Vb10sbs na dysku twardym.

Folder C:\Vb10sbs stanowi domyślną lokalizację dla obszernej kolekcji przykładowych plików do tej książki, o ile Czytelnik zastosował się do instrukcji zamieszczonej w części Wstępu zatytułowanej „Instalowanie i używanie plików ćwiczeniowych". Jeśli pliki te nie zostały jeszcze zainstalowane, należy zamknąć okno dialogowe Open Project i dokonać instalacji przy użyciu dysku CD dołączonego do książki.

**3.** Otwórz folder Chap01\Musictrivia i kliknij dwukrotnie plik rozwiązania MusicTrivia (jeśli system pokazuje rozszerzenia nazw plików, plik ten powinien mieć zakończenie .sln).

W efekcie Visual Studio pobierze formularz, właściwości oraz kod programu rozwiązania MusicTrivia. Strona startowa może być nadal widoczna na środku ekranu. W prawym górnym rogu ekranu, w oknie Solution Explorer, pojawi się lista plików należących do wybranego rozwiązania.

**Rozwiązywanie problemów** Jeśli pojawi się komunikat o błędzie informujący, że projekt wybrany do otwarcia ma nowszy format pliku, może to oznaczać próbę załadowania plików Visual Basic 2010 do starszej wersji oprogramowania Visual Basic (wcześniejsze wersje Visual Basic nie potrafią otworzyć projektów Visual Basic 2010 znajdujących się na dysku CD dołączonym do książki). Aby sprawdzić aktualnie używaną wersję Visual Basic, należy wybrać polecenie About z menu Help.

Visual Studio udostępnia specjalne pole wyboru o nazwie Always Show Solution (zawsze pokazuj rozwiązanie), umożliwiające kontrolę kilku opcji IDE dotyczących rozwiązania. Pole to znajduje się na karcie Projects and Solutions/General okna dialogowego Options, wyświetlanego po wybraniu polecenia Options z menu Tools. Jeśli to pole jest zaznaczone, to dla każdego nowego rozwiązania tworzony jest oddzielny podfolder, a sam projekt i jego pliki są umieszczane w oddzielnym folderze poniżej rozwiązania. Ponadto zaznaczenie pola wyboru Always Show Solution powoduje wyświetlanie w IDE kilku opcji związanych z rozwiązaniami, takich jak polecenia w menu File i wpis rozwiązania w Solution Explorer. Jeśli więc ktoś lubi tworzyć oddzielne foldery dla swoich rozwiązań i widzieć związane z nimi polecenia i ustawienia, powinien zaznaczyć to pole wyboru. Więcej informacji o tych opcjach zostanie podanych pod koniec tego rozdziału.

## Projekty i rozwiązania

W Visual Studio, programy w trakcie powstawania są nazywane zwykle *projektami* lub *rozwiązaniami*, ponieważ obejmują wiele indywidualnych składników, a nie tylko jeden plik. Programy Visual Basic 2010 składają się z pliku projektu (.vbproj) i pliku rozwiązania (.sln). Oglądając te pliki w przeglądarce, takiej jak Windows Explorer, można zauważyć w ikonach plików rozwiązań małą liczbę 10, wskazującą numer wersji. (Visual Basic 2010 ma wewnętrzne oznaczenie VB 10).

Plik projektu zawiera informacje właściwe dla jednego programowanego zadania. Plik rozwiązania zawiera informacje o jednym lub wielu projektach. Pliki rozwiązań są przydatne do zarządzania wieloma pokrewnymi projektami. Przykłady dołączone do tej książki mają zwykle po jednym projekcie w każdym rozwiązaniu, co oznacza, że otwarcie pliku projektu (.vbproj) ma ten sam skutek, co otwarcie pliku rozwiązania (.sln). Jednak w przypadku rozwiązań obejmujących wiele projektów należy otwierać pliki rozwiązań. Visual Basic 2010 oferuje nowy format pliku projektu i rozwiązania, ale podstawowa terminologia używana w wersji Visual Basic 2005 lub 2008 nadal ma zastosowanie.

## Narzędzia Visual Studio

W tym momencie należy poświęcić chwilę na zbadanie środowiska Visual Studio IDE i zapoznanie się z kilkoma narzędziami programistycznymi i oknami używanymi w trakcie tego kursu. Ci, którzy pisali już wcześniej programy w języku Visual Basic, rozpoznają wiele z tych narzędzi (choć zapewne nie wszystkie). Wszystkie te elementy stanowią razem składniki używane do budowania, organizowania i testowania swoich programów. Niektóre narzędzia pomagają również uzyskać informacje o zasobach dostępnych w systemie, takich jak bazy danych i połączenia z witrynami sieci Web. Jest także kilka narzędzi systemu pomocy.

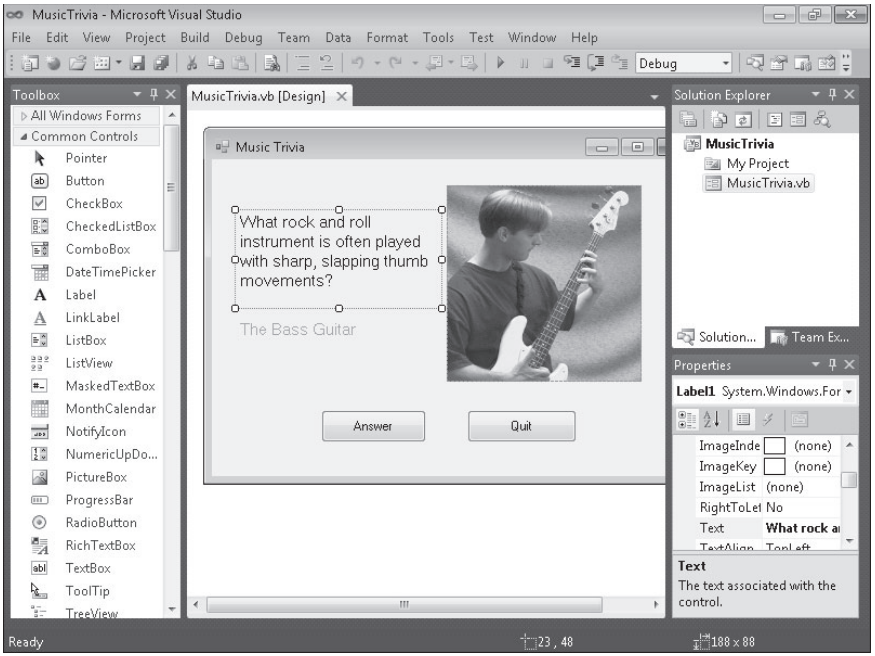
*Pasek menu* zapewnia dostęp do większości poleceń sterujących środowiskiem programowania. Menu i polecenia działają podobnie, jak we wszystkich programach opartych na systemie Windows. Można się do nich dostawać za pomocą klawiatury lub myszy. Poniżej paska menu znajduje się pasek narzędzi *Standard* (standardowy). Jest to zbiór przycisków służących za skróty do poleceń sterujących Visual Studio IDE. Zakłada się, że Czytelnik używał już programu Word, Excel lub innej aplikacji opartej na systemie Windows i jest już dostatecznie obeznany z paskami narzędzi oraz z popularnymi poleceniami, takimi jak Open, Save, Cut czy Paste. Będzie jednak zapewne pod wrażeniem liczby i różnorodności pasków narzędzi oferowanych przez Visual Studio do zadań programistycznych. W książce tej omówimy tylko niektóre z tych pasków; pełną ich listę można uzyskać w każdej chwili klikając dowolny pasek narzędzi prawym przyciskiem myszy.

Wzdłuż dolnej krawędzi ekranu może być widoczny pasek zadań systemu Windows. Używając tego paska można przełączać się między poszczególnymi składnikami Visual Studio i aktywować inne programy uruchomione w systemie. Mogą być także widoczne ikony Windows Internet Explorer, narzędzi antywirusowych i innych programów zainstalowanych w komputerze. Na większości rysunków w tej książce pasek zadań jest ukryty, aby zwiększyć ilość miejsca na IDE.

Na kolejnym zrzucie ekranu widać kilka narzędzi i okien Visual Studio IDE. Nie należy się martwić różnicami między tym rysunkiem a aktualnym wyglądem własnego ekranu. Więcej o tych elementach (i sposobach dostosowywania widoku) dowiemy się w dalszej części rozdziału.

Jak widać na rysunku, główne narzędzia dostępne w Visual Studio IDE, to okna Designer, Solution Explorer i Properties oraz przybornik Toolbox. Na ekranie Czytelnika mogą występować również bardziej wyspecjalizowane narzędzia, takie jak Server Explorer czy Object Browser albo mogą mieć one postać kart w IDE. Ponieważ każdy programista ma inne upodobania, trudno przewidzieć, co się pojawi na ekranie Czytelnika, jeśli korzystał już wcześniej z Visual Studio (w książce pokazywany jest zwykle wygląd ekranu „zaraz po pobraniu" lub inaczej „po wyjęciu z pudełka").

Jeśli jakieś narzędzie nie jest widoczne, a jest potrzebne, wystarczy kliknąć menu View i wybrać to narzędzie. Ponieważ menu View przez lata się rozrosło, Microsoft przeniósł niektóre rzadziej używane narzędzia z tego menu do podmenu o nazwie Other Windows. I tam trzeba szukać narzędzi, których nie widać w menu View.




Faktyczny rozmiar i kształt narzędzi oraz okien zależy od konfiguracji środowiska programowania. Visual Studio pozwala wyrównywać i doczepiać, inaczej *dokować*, okna, aby były widoczne tylko te elementy, które są potrzebne. Można również częściowo ukrywać narzędzia w formie *zakładek dokumentów* wzdłuż krawędzi środowiska programowania, a następnie przełączać się szybko między nimi. Próba ustalenia, które narzędzia są ważne w pierwszej kolejności, a które można odsunąć na dalszy plan, jest trudnym zadaniem na początku nauki bogatego interfejsu Visual Studio. Prawdopodobnie najlepiej jest tak zorganizować ekran monitora i pulpit Windows, aby uzyskać jak najwięcej miejsca. Ale nawet wtedy elementy mogą wydawać się stłoczone.

**Wskazówka** Chociaż większość zrzutów ekranu w tej książce była robiona przy rozdzielczości 800 × 600 – aby zwiększyć czytelność IDE – autor stosuje do pisania kodu rozdzielczość 1280 x 1024. Aby zmienić rozdzielczość ekranu w systemie Windows 7, wystarczy kliknąć pulpit Windows prawym przyciskiem myszy i wybrać polecenie Screen Resolution (rozdzielczość ekranu). W systemie Windows Vista należy kliknąć pulpit prawym przyciskiem myszy i wybrać polecenie Personalize (personalizuj).

Celem tych wszystkich skomplikowanych narzędzi jest dodanie do IDE wielu nowych i przydatnych funkcji, a jednocześnie zapewnienie sprytnych mechanizmów zarządzania nimi. Do mechanizmów tych należą takie funkcje, jak dokowanie, automatyczne ukrywanie i pływanie i kilka innych stanów okien, które zostaną opisane w dalszej części tego rozdziału. Dla osób poznających dopiero Visual Studio najlepszym sposobem poradzenia sobie z tym natłokiem jest ukrycie narzędzi, które nie będą używane zbyt

często, aby pozostawić jak najwięcej miejsca na rzeczy ważne. Do kluczowych narzędzi dla początkujących programistów w Visual Basic – narzędzi, które będą wkrótce wykorzystywane w tej książce – należą: Designer, okno Properties, Solution Explorer oraz przybornik Toolbox. Dopiero w dalszej części książki będą używane okna Server Explorer, Class View, Object Browser i Debug.

W kolejnych ćwiczeniach przystąpimy do eksperymentowania z podstawowymi narzędziami Visual Studio IDE. Dowiemy się również, jak wyświetlać przeglądarkę sieci Web w Visual Studio i jak ukrywać chwilowo niepotrzebne narzędzia.

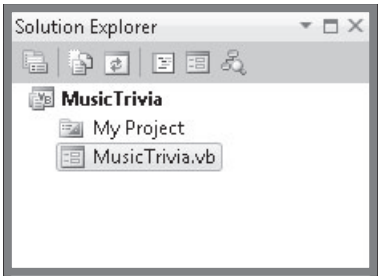
## Designer

Jeśli zostało wykonane ostatnie ćwiczenie („Otwórz projekt Visual Basic"), to projekt MusicTrivia powinien być już załadowany do środowiska Visual Studio. Mimo to interfejs użytkownika, inaczej *formularz*, dla tego projektu może nie być jeszcze widoczny na ekranie (bardziej skomplikowane projekty mogą obejmować wiele formularzy, ale ten prosty program zawiera tylko jeden). Aby wyświetlić formularz projektu MusicTrivia w środowisku IDE, można użyć narzędzia Solution Explorer.

### Wyświetlenie okna Designer

1. Odszukaj okno Solution Explorer w prawym górnym rogu środowiska programowania Visual Studio. Jeśli nie widać tego okna (bo jest ukryte w niewidocznym miejscu w postaci karty albo nie jest aktualnie wyświetlone), kliknij polecenie Solution Explorer w menu View.

   Po załadowaniu projektu MusicTrivia okno Solution Explorer powinno wyglądać następująco:




2. Kliknij formularz MusicTrivia.vb w oknie Solution Explorer.

   Wszystkie pliki formularzy, łącznie z używanym w ćwiczeniu, mają widoczną niewielką ikonę, która pozwala je łatwo zidentyfikować. Po kliknięciu pliku formularza Visual Studio zaznacza go w oknie Solution Explorer i wyświetla pewne informacje o tym pliku w oknie Properties (jeśli jest ono widoczne).

3. U góry okna Solution Explorer kliknij przycisk View Designer, aby wyświetlić interfejs użytkownika programu.

Formularz MusicTrivia w oknie Designer powinien wyglądać następująco:

U góry okna Designer można zauważyć kartę zatytułowaną MusicTrivia.vb [Design]. Po jej kliknięciu zostanie wyświetlony kod programu skojarzony z formularzem MusicTrivia. Podobnie klikając można przełączać się między pozostałymi kartami widocznymi u góry okna Designer. Więcej o kodzie programu i o karcie Code Editor będziemy mówić w rozdziale 2., „Pisanie pierwszego programu".

Teraz spróbujemy uruchomić nasz program Visual Basic w Visual Studio.

## Uruchamianie programu Visual Basic

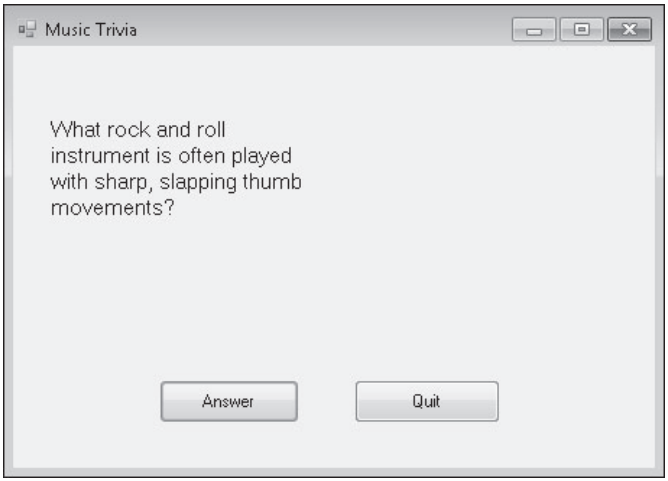
Music Trivia jest prostym programem Visual Basic zaprojektowanym w celu zademonstrowania narzędzi programistycznych w Visual Studio. Widoczny formularz obejmuje pięć obiektów (dwie etykiety, rysunek i dwa przyciski). Dodane trzy linie kodu powodują wyświetlanie przez program prostego pytania i wyświetlanie odpowiedniej odpowiedzi (program „zdradza" w tym momencie odpowiedź, ponieważ jest w trybie projektowania, ale podczas wykonywania odpowiedź jest ukryta). Więcej informacji o tworzeniu obiektów i dodawaniu kodu programu zostanie podanych w rozdziale 2. Teraz spróbujemy od razu uruchomić program w Visual Studio IDE.

### Uruchomienie programu Music Trivia

1. Kliknij przycisk Start Debugging (zielona strzałka w prawo) na pasku narzędzi Standard, aby uruchomić program Music Trivia w Visual Studio.

**Wskazówka** Aby uruchomić program w środowisku programowania Visual Studio można również nacisnąć klawisz F5 lub wybrać polecenie Start Debugging z menu Debug.

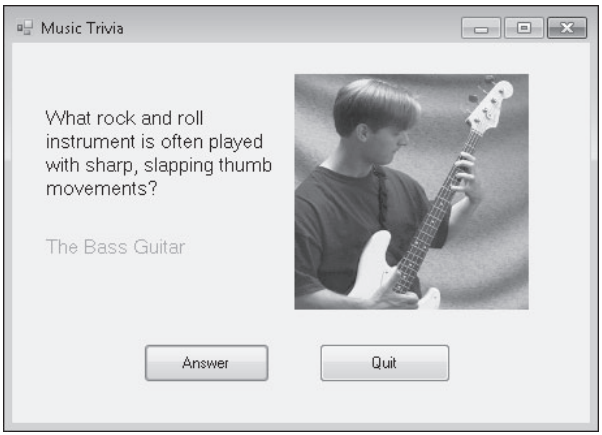
W efekcie Visual Studio pobierze projekt i skompiluje go do postaci *podzespołu* (strukturalnej kolekcji modułów, danych i informacji manifestu programu), przygotuje program do testowania lub *debugowania*, a następnie (jeśli kompilacja zakończy się pomyślnie) uruchomi program w środowisku programowania. Podczas wykonywania programu jego ikona będzie widoczna na pasku zadań. Po chwili na ekranie ponownie pojawi się formularz MusicTrivia, tym razem z ukrytym zdjęciem i etykietą odpowiedzi, jak to jest pokazane na rysunku:




Program Music Trivia wyświetli pytanie: „What rock and roll instrument is often played with sharp, slapping thumb movements?" (Jak się nazywa instrument rockandrollowy, na którym gra się często wykonując ostre uderzenia kciukiem).

**2.** Kliknij przycisk Answer, aby wyświetlić odpowiedź na pytanie.

Poniżej pytania program wyświetli odpowiedź (The Bass Guitar – gitara basowa), a następnie zdjęcie nieznanego gitarzysty basowego z Seattle demonstrującego wspomnianą technikę. Oznacza to, że program działa.



3. Kliknij Quit, aby zamknąć program.

   Formularz zostanie zamknięty i znowu stanie się aktywne środowisko Visual Studio IDE.

## Uwagi o właściwościach

W języku Visual Basic każdy element interfejsu użytkownika (user interface - UI) w programie (łącznie z samym formularzem) ma zdefiniowany zbiór właściwości. Właściwości te można ustawić w dowolnym momencie za pomocą okna Properties (właściwości). Do właściwości można się również odwoływać z poziomu kodu, co umożliwia manipulowanie nimi podczas wykonywania programu (elementy UI odbierające dane wejściowe często używają właściwości do przekazywania informacji do wnętrza programu). Na początku opanowanie koncepcji właściwości może sprawiać trudności. Aby w tym pomóc, posłużymy się przykładem zaczerpniętym z życia codziennego.

Jako analogię weźmy rower: rower jest obiektem używanym do jazdy z jednego miejsca w inne. Ponieważ obiekt jest obiektem fizycznym, ma kilka nieodłącznych cech, takich jak nazwa producenta, kolor, przerzutki, hamulce, koła oraz określony typ (może to być rower szosowy, górski albo tandem). W terminologii języka Visual Basic cechy te są nazywane *właściwościami*, a rower *obiektem*. Większość właściwości roweru jest definiowanych w momencie jego powstawania. Ale są i takie właściwości (opony, prędkość podróżna oraz dodatki, takie jak światła i lusterka), które mogą się zmieniać w trakcie użytkowania roweru. Rower może mieć nawet właściwości „niematerialne" (czyli niewidoczne), na przykład datę produkcji, nazwisko właściciela, wartość albo stan wypożyczenia. Aby jeszcze bardziej skomplikować sytuację, jakaś firma lub sklep może mieć więcej niż jeden rower lub (co bardziej prawdopodobne) całą flotę rowerów i to o różnych właściwościach. W trakcie korzystania z języka Visual Basic będziemy ustawiali właściwości wielu obiektów i organizowali je na różne wygodne sposoby.

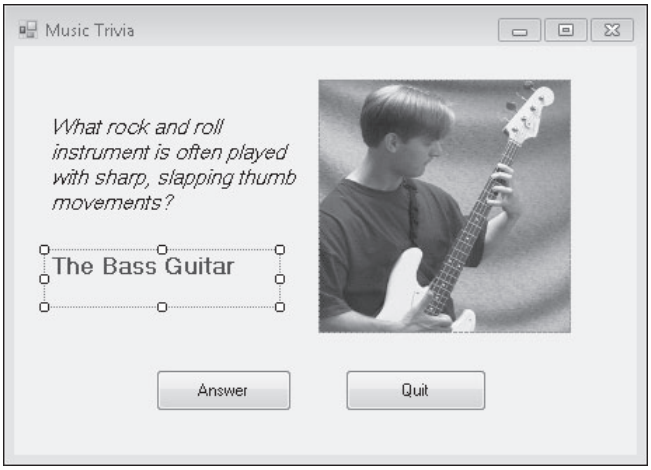
# Okno Properties

W IDE do zmiany cech, czyli *ustawień właściwości* elementów UI formularza można używać okna Properties. Ustawienie właściwości jest *własnością* jednego z obiektów w programie. Ustawienia właściwości można zmieniać podczas tworzenia interfejsu użytkownika wykorzystując do tego okno Properties. Można też za pomocą edytora kodu dodać kod, który będzie zmieniał jedno lub wiele ustawień właściwości podczas wykonywania programu. Na przykład można zmienić pytanie zadawane przez program Music Trivia, tak aby było ono wyświetlane z zastosowaniem innej czcionki, innego rozmiaru albo innego sposobu wyrównania. (Visual Studio pozwala wyświetlać tekst z użyciem dowolnej czcionki zainstalowanej w systemie, podobnie jak w przypadku programu Excel czy Word).

Obiekt *Label2* ma własny zbiór ustawień właściwości. Chociaż nazwy tych właściwości są takie same jak w przypadku obiektu *Label1*, wartości ich ustawień są inne, dzięki czemu obiekt *Label2* jest całkowicie niezależny.

9. Kliknij przycisk wielokropka dla właściwości Font, zmień styl czcionki na Bold (pogrubiona), a rozmiar na 12 punktów i na koniec kliknij OK.
10. Przewiń okno Properties do właściwości *ForeColor* i następnie kliknij jej nazwę w lewej kolumnie.
11. Kliknij strzałkę ForeColor w prawej kolumnie, potem kliknij kartę Custom i następnie kliknij kolor ciemnopurpurowy.

    Tekst w obiekcie *Label2* na formularzu zostanie w efekcie pogrubiony i zmieni kolor na purpurowy, jak to jest pokazane na rysunku*.




* Ze względu na fakt, że książka nie jest drukowana w kolorze, barwy Czytlenik będzie mógł zobaczyć jedynie na ekranie.

Gratulacje! Właśnie nauczyłeś się ustawiać właściwości obiektów programu Visual Basic przy użyciu okna Properties w Visual Studio. To jedna z ważnych umiejętności potrzebnych do zostania programistą języka Visual Basic.

## Przenoszenie i zmienianie rozmiarów narzędzi programistycznych

Z powodu licznych narzędzi programistycznych zapełniających ekran środowisko Visual Studio IDE może stać się dość zatłoczonym miejscem. Aby zapewnić użytkownikowi pełną kontrolę nad kształtem i rozmiarami elementów tego środowiska, Visual Studio umożliwia przenoszenie, skalowanie, dokowanie oraz automatyczne ukrywanie większości elementów interfejsu używanych do budowania programów.

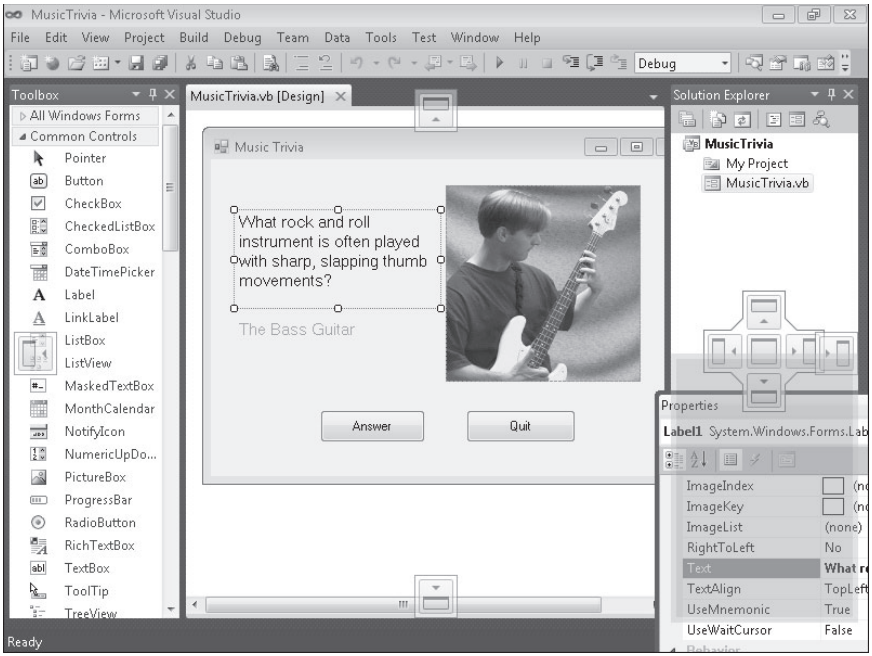
Aby przenieść jakieś okno narzędziowe w Visual Studio, wystarczy kliknąć jego pasek tytułu i przeciągnąć cały obiekt w nowe miejsce. Po zrównaniu okna z krawędzią innego okna zostaje ono doczepione do tego drugiego okna, czyli inaczej *zadokowane*. Okna zadokowane mają przewagę nad innymi, ponieważ pozostają zawsze widoczne (nie są przesłaniane przez inne okna). Aby zobaczyć większy fragment zadokowanego okna, wystarczy przeciągnąć jedną z jego krawędzi, aby powiększyć wnętrze.

Aby całkowicie zamknąć jakieś okno, wystarczy kliknąć przycisk Zamknij w prawym górnym rogu tego okna. Takie okno można zawsze otworzyć później wybierając odpowiednie polecenie z menu View.

Opcją pośrednią między zadokowaniem a zamknięciem okna może być dla niektórych włączenie automatycznego ukrywania okna narzędziowego z boku środowiska Visual Studio IDE. W tym celu należy kliknąć niewielki przycisk pinezki Auto Hide z prawej strony paska tytułu narzędzia. Spowoduje to usunięcie okna z pozycji dokowania i umieszczenie tytułu narzędzia na brzegu środowiska programowania na niezachodzącej karcie. Z chwilą włączenia automatycznego ukrywania okna narzędziowego jest ono nadal widoczne, dopóki wskaźnik myszy pozostaje w obszarze tego okna. Po przesunięciu wskaźnika nad inny obszar IDE, okno zostanie usunięte z widoku.

Aby przywrócić okno, które zostało automatycznie ukryte, należy kliknąć jego kartę na brzegu środowiska programowania lub przytrzymać mysz nad tą kartą (automatycznie ukryte okno można rozpoznać po skierowanej w bok pinezce na jego pasku tytułu). Metoda przytrzymania wskaźnika myszy nad tytułem przypomina zabawę w „akuku" – innymi słowy, aby szybko skorzystać z automatycznie ukrytego okna, lepiej jest kliknąć jego kartę, sprawdzić lub ustawić potrzebne informacje, a następnie przesunąć wskaźnik, aby ponownie ukryć okno. Jeśli zajdzie potrzeba trwałego przywrócenia okna na ekranie, wystarczy ponownie kliknąć przycisk pinezki Auto Hide. Okno stanie się znowu widoczne, a pinezka zmieni kierunek w dół.

Inną przydatną funkcją Visual Studio jest możliwość wyświetlania okien w postaci pliku kart dokumentów (okien z uchwytami kart ukrytych częściowo jedno za drugim) oraz dokowanie okien przy użyciu wskazówek dokowania wyświetlanych w postaci niewielkich kwadratów na obrzeżach IDE, jak również centralnie ulokowanego „diamentu wskazówek", pokazanego na następnej stronie rysunku.




Wskazówki dokowania są to ruchome ikony wyświetlane w obszarze roboczym IDE podczas przesuwania okna lub narzędzia z pozycji zadokowanej w nowe miejsce. Ponieważ ze wskazówkami dokowania są skojarzone zacieniowane prostokątne obszary IDE, można podejrzeć wynik operacji dokowania przed jej ostatecznym zatwierdzeniem. W Visual Studio 2010 funkcja ta została znacząco ulepszona i można podglądać wiele różnych konfiguracji rozmieszczenia, z których żadna nie zostanie zrealizowana zanim nie nastąpi zwolnienie przycisku myszy.

Ponieważ opanowanie technik dokowania i automatycznego ukrywania wymaga trochę prób, dobrze jest wykorzystać następne ćwiczenia do poeksperymentowania z funkcjami zarządzania oknami oferowanymi przez IDE. Po ukończeniu tych ćwiczeń każdy będzie już potrafił swobodnie konfigurować narzędzia Visual Studio w najwygodniejszy dla siebie sposób.

## Przenoszenie i zmienianie rozmiarów okien narzędziowych

Aby przenieść okno narzędziowe w Visual Studio lub zmienić jego rozmiar, należy wykonać podane niżej działania. Ćwiczenie to pokazuje, jak można manipulować oknem Properties, ale można zamiast niego wybrać inne okno narzędziowe.

### Przeniesienie okna Properties i zmiana jego rozmiaru

1. Jeśli okno Properties nie jest widoczne w środowisku programowania, kliknij przycisk Properties Window na pasku narzędzi Standard.

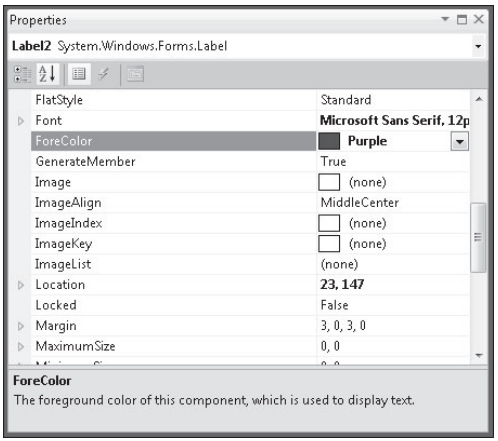
   Spowoduje to aktywację okna Properties w IDE i podświetlenie jego paska tytułu.

2. Kliknij dwukrotnie pasek tytułu okna Properties, aby wyświetlić je w postaci okna pływającego (niezadokowanego).

3. Używając paska tytułu okna Properties przeciągnij je w inne miejsce środowiska programowania, ale nie dokuj go (na razie).

   Zmiana położenia okien w środowisku Visual Studio IDE daje trochę swobody w dostępie do narzędzi i w kształtowaniu wyglądu środowiska programowania. Za chwilę zmienimy rozmiar okna Properties, aby móc widzieć więcej ustawień właściwości jednocześnie.

4. Ustaw wskaźnik myszy na prawym dolnym narożniku okna Properties, tak aby wskaźnik zmienił kształt na dwustronnie skierowaną strzałkę (wskaźnik zmiany rozmiaru). Następnie przeciągnij prawy dolny narożnik okna w dół i w prawo, aby je powiększyć, jak to jest pokazane na rysunku.




   Większe okno daje możliwość szybszej pracy i poprawia czytelność. Gdy trzeba wyświetlić większy fragment zawartości jakiegoś okna, można zawsze swobodnie je przenieść lub zmienić jego rozmiar.

## Dokowanie okien narzędziowych

Jeśli jakieś okno narzędziowe w środowisku programowania jest oknem pływającym, można ustawić je z powrotem w oryginalnym miejscu dokowania wciskając klawisz CTRL i klikając dwukrotnie pasek tytułu tego okna (jak było powiedziane w poprzednim ćwiczeniu, dwukrotne kliknięcie paska tytułu okna zadokowanego powoduje jego „oddokowanie”). Można również zaczepić lub zadokować pływające okno narzędziowe

w innym miejscu. Taka potrzeba może zaistnieć na przykład wtedy, gdy trzeba zrobić więcej miejsca w Visual Studio dla konkretnego zadania programistycznego, takiego jak tworzenie interfejsu użytkownika w oknie Designer. Spróbujemy teraz zadokować okno Properties w innym miejscu.

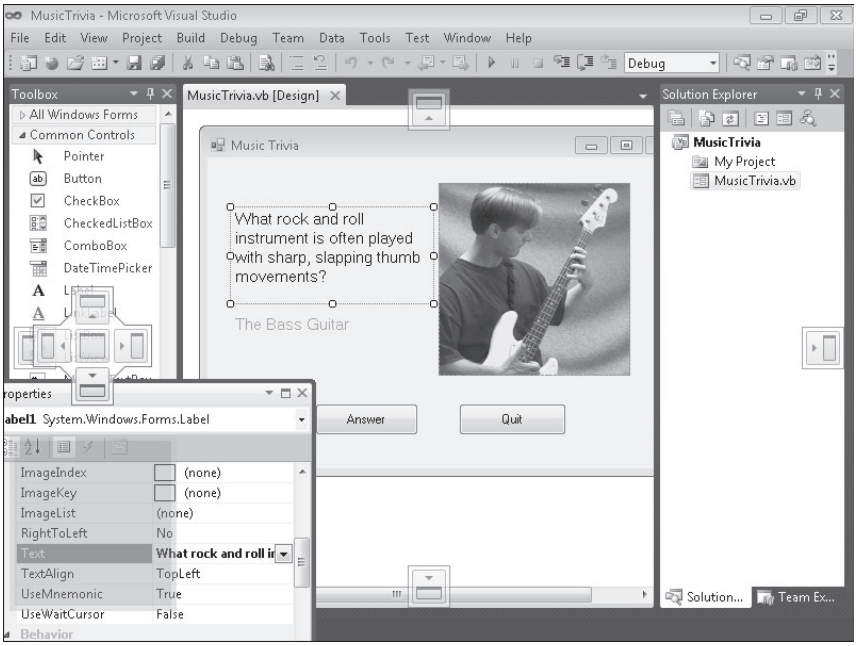
## Dokowanie okna Properties

1. Sprawdź, czy okno Properties (lub inne okno narzędziowe, które chcesz zadokować) jest oknem pływającym.

   Jeśli wykonałeś poprzednie ćwiczenie, to okno Properties jest aktualnie niezadokowane.

2. Przeciągnij pasek tytułu okna Properties w kierunku górnej, dolnej, prawej lub lewej krawędzi środowiska programowania (do wyboru!), pilnując, aby wskaźnik myszy znalazł się nad jedną ze wskazówek dokowania na obrzeżach Visual Studio IDE albo nad kolekcją czterech lub więcej wskazówek dokowania, zwaną wspólnie *diamentem wskazówek*.

   Po ustawieniu wskaźnika myszy nad wskazówką dokowania okno Properties zostanie przyciągnięte w to miejsce i pojawi się niebieski, cieniowany prostokąt wskazujący, jak będzie wyglądać okno po zwolnieniu przycisku myszy (jest to pokazane na kolejnym rysunku). Warto pamiętać, że jest kilka dozwolonych miejsc dokowania okien narzędziowych w Visual Studio, tak więc można wypróbować dwa lub trzy z nich, zanim wybierze się to najwłaściwsze (powinno to być miejsce wygodne i nie kolidujące z innymi narzędziami).



3. Zwolnij przycisk myszy, aby zadokować okno Properties.

   Okno wskoczy na swoje nowe miejsce.

**Wskazówka** Aby zmienić okno na zadokowane, ukryte, pływające lub wyświetlić je jako kartę, można kliknąć prawym przyciskiem myszy pasek tytułu okna (lub kartę, jeśli taka jest postać narzędzia), a następnie wybrać żądany wariant. O ile okno Properties sprawdza się bardzo dobrze jako okno zadokowane, to większe okna (na przykład strona testowa Visual Studio) z reguły najlepiej jest zmienić w karty.

4. Spróbuj zadokować okno Properties w kilku innych miejscach, aby oswoić się z działaniem tej funkcji.

   Chociaż wiele z tych procedur może na początku sprawiać problemy, to już po krótkim czasie staną się rutyną. Zasadniczo wielkość okien jest tak dobierana, aby było w nich dość miejsca na informacje używane podczas wykonywania istotniejszych zadań w oknach Designer i Code Editor.

## Ukrywanie okien narzędziowych

Aby ukryć okno narzędziowe, wystarczy kliknąć przycisk z pinezką Auto Hide po prawej stronie paska tytułu. Okno zostanie wówczas schowane pod kartą narzędzi na brzegu IDE. Ponowne kliknięcie spowoduje przywrócenie okna w miejscu dokowania. W celu automatycznego ukrycia okna narzędziowego można również użyć polecenia Auto Hide z menu Window (albo kliknąć prawym przyciskiem myszy pasek tytułu i z menu podręcznego wybrać polecenie Auto Hide). Wypróbujemy to teraz.

### Użycie funkcji Auto Hide

1. Odszukaj przycisk pinezki Auto Hide na pasku tytułu okna Properties.

   Aktualnie pinezka powinna być skierowana w dół, czyli „wciśnięta”, co oznacza, że okno Properties jest „przypięte” na stałe i automatyczne ukrywanie jest wyłączone.

2. Kliknij przycisk Auto Hide na pasku tytułu okna Properties.

   Okno Properties zostanie usunięte z ekranu i zastąpione małą kartą o nazwie Properties. Korzyścią płynącą z włączenia automatycznego ukrywania jest oczywiście uzyskanie dodatkowego miejsca w Visual Studio. Takie ukryte okno jest jednak nadal szybko dostępne.

3. Przytrzymaj wskaźnik myszy nad kartą Properties (możesz również kliknąć kartę Properties, jeśli chcesz).

   Okno Properties natychmiast pojawi się z powrotem.

4. Przesuń wskaźnik myszy w inne miejsce IDE, a okno znowu zniknie.

5. Na koniec ponownie wyświetl okno Properties, a następnie kliknij przycisk pinezki na pasku tytułu okna Properties.

   Okno Properties powróci w swoje stare miejsce dokowania i będzie można go używać bez obawy, że zniknie.

   Poświęć teraz chwilę na przenoszenie, zmienianie rozmiarów, dokowanie i automatyczne ukrywanie okien narzędziowych w Visual Studio, aby uzyskać swoją wersję idealnego środowiska pracy. W dalszej części tej książki co jakiś czas będzie można zmienić ustawienia okien, aby dostosować obszar roboczy do nowych narzędzi, których będziemy używać.

**Wskazówka** Visual Studio pozwala zapisać ustawienia okien i środowiska programowania, a następnie skopiować je do innego komputera lub udostępnić członkom tego samego zespołu programistów. Aby użyć tej funkcji, należy wybrać polecenie Import And Export Settings z menu Tools i zastosować się do instrukcji kreatora eksportu danych (zapisania w pliku) lub ich importu (pobrania z pliku).

# Przełączanie się między otwartymi plikami i narzędziami przy użyciu funkcji IDE Navigator

Visual Studio udostępnia funkcję, która jeszcze bardziej ułatwia przełączanie się między otwartymi plikami i narzędziami w środowisku programowania. Funkcja ta nosi nazwę IDE Navigator i pozwala na przełączanie się za pomocą kombinacji klawiszy, podobnie jak w przypadku przełączania się między otwartymi programami na pasku zadań Windows. Wypróbujemy ją teraz.

## Korzystanie z funkcji IDE Navigator

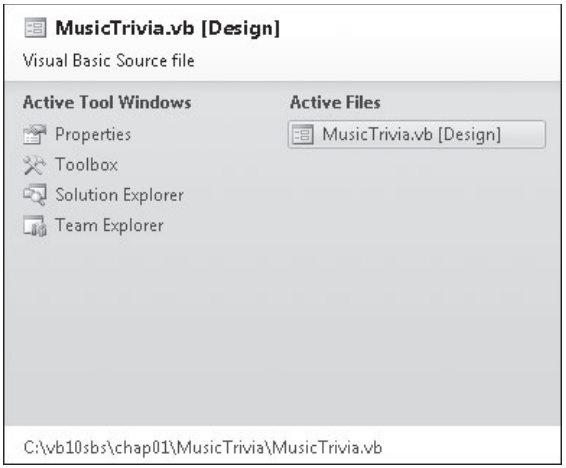
1. Wciśnij klawisz CTRL i naciśnij klawisz TAB, aby otworzyć okno IDE Navigator.

   Na ekranie pojawi się okienko IDE Navigator z listą aktywnych (otwartych) plików i narzędzi w IDE, podobne do pokazanego na rysunku.
2. Trzymając wciśnięty klawisz CTRL naciśnij parę razy klawisz TAB, aby pokrążyć po plikach i przenieść zaznaczenie do tego właściwego.

   W celu krążenia w odwrotnym kierunku należy wcisnąć klawisze CTRL+SHIFT i wtedy naciskać TAB (dla uzyskania lepszego efektu można otworzyć jeszcze jedno okno lub nawet dwa, przez co cykliczność tej operacji będzie bardziej widoczna).
3. Trzymając wciśnięty klawisz CTRL naciśnij klawisze strzałek, aby pokrążyć po aktywnych plikach i aktywnych narzędziach.

   Można również zaznaczyć aktywny plik (lub narzędzie) klikając jego nazwę.
4. Na koniec zwolnij klawisz CTRL.

   Element, który był ostatnio zaznaczony w okienku IDE Navigator, stanie się aktywny.




**WSKAZÓWKA** Aby wykonać krążenie po aktywnych narzędziach bez otwierania okienka IDE Navigator, można nacisnąć klawisze ALT+F7. Kombinacja SHIFT+ALT+F7 pozwala zrobić to samo, tylko w odwrotnym kierunku.

# Otwieranie przeglądarki sieci Web w Visual Studio

Zaletą Visual Studio jest możliwość otwarcia prostej przeglądarki sieci Web wewnątrz środowiska programowania. Przeglądarka ta ma postać karty dokumentu w IDE, przez co zajmuje niewiele miejsca, a w razie potrzeby może zostać błyskawicznie otwarta. Można oczywiście otworzyć autonomiczną przeglądarkę sieci Web (taką jak Internet Explorer) i trzymać ją obok na pasku zadań systemu Windows, ale przeglądarka działająca *wewnątrz* Visual Studio znacznie ułatwia sprawdzanie witryn sieci Web i kopiowanie danych do środowiska programowania. Spróbujemy teraz użyć tej wewnętrznej przeglądarki.

## Otwieranie przeglądarki sieci Web w Visual Studio

1. Rozwiń menu View, wybierz podmenu Other Windows i kliknij polecenie Web Browser.

   Na ekranie pojawi się okno Web Browser pokazane na następnym rysunku.

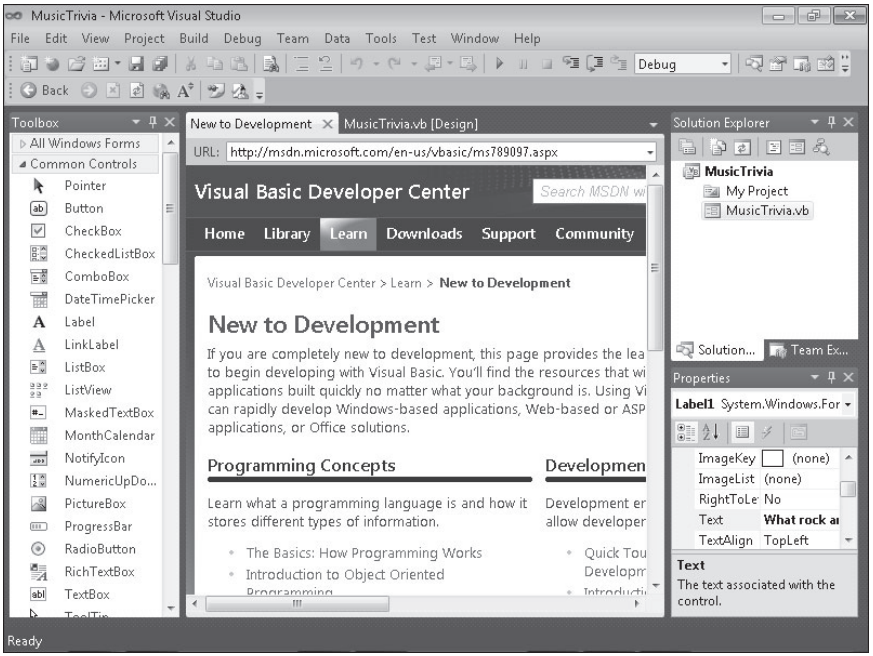
   Przeglądarka ma standardowo postać karty dokumentu, ale można ją zmienić w pływające lub zadokowane okno. W tym celu należy kliknąć pasek tytułu prawym przyciskiem myszy i następnie wybrać polecenie Float lub Dock.

**Wskazówka** Standardową stronę wyświetlaną w oknie Web Browser można zmienić za pomocą ustawień dostępnych w oknie dialogowym Options. Aby otworzyć to okno, należy rozwinąć menu Tools i wybrać polecenie Options. Następnie trzeba zaznaczyć pole wyboru Show All Settings, rozwinąć Environment i kliknąć Web Browser. Na koniec pozostaje tylko zmienić ustawienie Home Page na adres URL (Uniform Resource Locator) wybranej przez siebie strony.

2. Wypróbuj działanie przeglądarki i sposób jej funkcjonowania w IDE.

   Chociaż przeglądarka ta jest prostsza niż Internet Explorer czy inna w pełni funkcjonalna przeglądarka, może być wygodnym dodatkiem do zestawu narzędzi Visual Studio. Zawsze też można otworzyć Internet Explorer (lub inną przeglądarkę) bezpośrednio z paska zadań systemu Windows.

3. Na zakończenie kliknij przycisk Close z prawej strony karty lub paska tytułu przeglądarki sieci Web, aby ją zamknąć.




# Uzyskiwanie pomocy

Visual Studio udostępnia system pomocy (Help), którego można używać do lepszego poznania środowiska IDE, języka programowania Visual Basic oraz środowiska Microsoft .NET Framework. Przyjrzymy się teraz przez chwilę tym zasobom, zanim przejdziemy do rozdziału 2, w którym utworzymy nasz pierwszy program.

Ci, którzy używali już wersji Visual Studio 2008, zauważą pewne różnice w systemie pomocy Visual Studio 2010. Najbardziej znacząca jest taka, że system działa

obecnie w przeglądarce sieci Web. Podstawowe różnice, o których trzeba pamiętać, są wymienione w tabeli 1-1.

**Tabela 1-1** Porównywanie systemu pomocy w różnych wersjach Visual Studio

| Visual Studio 2008 | Visual Studio 2010 |
|---|---|
| Lokalna pomoc była wyświetlana w autonomicznej aplikacji o nazwie Microsoft Document Explorer. | Lokalna pomoc jest wyświetlana w przeglądarce sieci Web. |
| Aplikacja Document Explorer była połączona z Visual Studio i mogła być aktualizowana tylko razem z Visual Studio. | Ponieważ system pomocy jest oparty na przeglądarce i niezależny od Visual Studio, można go częściej aktualizować. |
| Lokalna pomoc była rzadziej aktualizowana. | System pomocy może być aktualizowany na żądanie przy użyciu narzędzia Help Library Manager. |
| Działanie F1 było czasem bardzo spowolnione. | Działanie F1 jest szybsze, a wyniki wyszukiwania lepsze. |
| System pomocy udostępniał kompletne drzewo spisu treści dla wszystkich tematów. | System pomocy udostępnia uproszone drzewo spisu treści, które obejmuje tylko temat nadrzędny, bieżący i podrzędne. |
| Lokalna pomoc obejmowała indeks. | Nie ma już dostępu do indeksu. |
| Lokalna pomoc i pomoc online działały zupełnie inaczej. | Lokalna pomoc i pomoc online działają bardzo podobnie. |
| System pomocy obejmował zwykle wiele języków, takich jak Visual Basic, C#, C++ i JScript, co utrudniało czytanie dokumentacji. | Dokumentacja systemu pomocy obejmuje różne języki i ma postać kart, dzięki czemu wyświetlana jest informacja tylko o jednym języku. |

**Uwaga** Ponieważ system pomocy został wydzielony z Visual Studio 2010 i może być regularnie aktualizowany, sytuacja u Czytelnika może się różnić od opisanej w następnym podrozdziale.

## Zarządzanie ustawieniami systemu pomocy

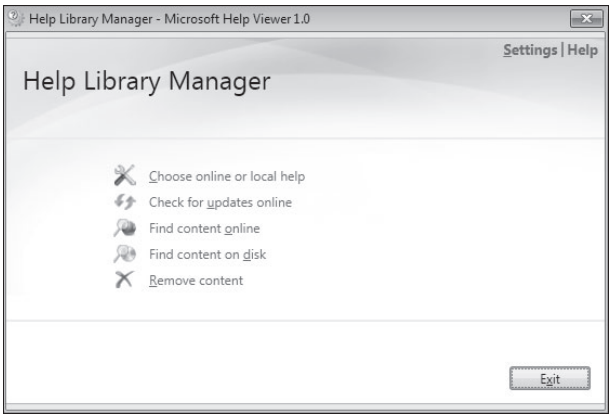
Visual Studio udostępnia narzędzie Help Library Manager do zarządzania dokumentacją i ustawieniami systemu pomocy. Używając tego narzędzia można wybrać pomoc online lub lokalną, sprawdzić aktualizacje w trybie online oraz wyszukać lub usunąć jakąś zawartość.

Dokumentacja systemu pomocy dla Visual Studio 2010 jest dostarczana na dwa sposoby: lokalnie lub w trybie online. Lokalna pomoc jest zwykle dodawana podczas instalacji Visual Studio 2010 (można również dodać ją później za pomocą narzędzia Help Library Manager). Lokalna pomoc jest aktualizowana okresowo, ale trzeba sprawdzać dostępność aktualizacji w Help Library Manager. Pomoc online jest dostępna pod

adresem *http://msdn.Microsoft.com/library/*. Jeśli komputer jest połączony z Internetem, zazwyczaj lepiej jest używać pomocy online, ponieważ zapewnia to zawsze dostęp do najnowszej wersji dokumentacji.

### Zarządzanie ustawieniami systemu pomocy

1. Rozwiń menu Help i kliknij Manage Help Settings. Gdy pojawi się okno dialogowe Set Local Content Location, kliknij OK, aby zaakceptować standardową lokalizację. Na ekranie pojawi się pokazane niżej okno Help Library Manager.




2. Kliknij Choose Online Or Local Help (wybierz pomoc online lub lokalną).

   Spowoduje to wyświetlenie okienka Settings umożliwiającego wybór typu używanej pomocy.
3. Jeśli masz połączenie z Internetem, upewnij się, że jest zaznaczona opcja I Want To Use Online Help (chcę używać pomocy online) i kliknij OK.
4. W oknie Help Library Manager kliknij Install Content From Online (zainstaluj treść z sieci).
5. Sprawdź zawartość systemu pomocy. Jeśli się zdecydujesz, możesz ją zainstalować lokalnie.
6. Kliknij Cancel.
7. Zbadaj pozostałe opcje w oknie Help Library Manager.
8. Na zakończenie zamknij okno Help Library Manager.

## Korzystanie z klawisza F1

Jaki jest najszybszy sposób uzyskania pomocy podczas pracy w Visual Studio? Zwykle najlepiej jest nacisnąć klawisz F1. Visual Studio oferuje „pomoc kontekstową", czyli związaną z aktualnie używanym słowem kluczowym lub wykonywanym zadaniem. Chociaż naciśnięcie klawisza F1 nie zawsze powoduje wyświetlenie dokładnie pasującej informacji, to zazwyczaj przenosi użytkownika do tej części dokumentacji systemu

pomocy, która stanowi dobry punkt wyjścia. Jeśli więc potrzebna jest pomoc, zawsze warto pomyśleć o użyciu klawisza F1.

## Używanie klawisza F1

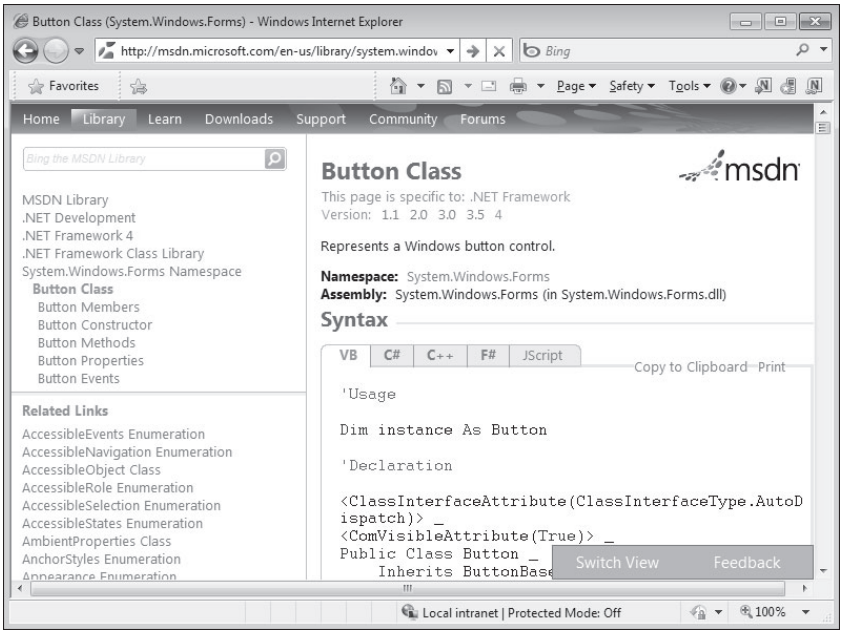
1. Kliknij obiekt *Label1* na formularzu.
2. Naciśnij klawisz F1. Jeśli pojawi się okno dialogowe z pytaniem, czy chcesz wyświetlić pomoc z Internetu, kliknij Yes.

   Na ekranie powinien pojawić się temat Label w MSDN.

> **Wskazówka** Jeśli komputer nie jest połączony z Internetem, a jest zainstalowana lokalna dokumentacja, można zmienić ustawienie na korzystanie z lokalnej pomocy.

3. Przejdź z powrotem do Visual Studio.
4. Kliknij przycisk Answer na formularzu.
5. Naciśnij klawisz F1.

   Na ekranie powinien pojawić się temat Button w MSDN. W zależności od widoku, może to wyglądać podobnie jak na rysunku:




MSDN ma aktualnie kilka widoków. Zamieszczony tu widok nosi nazwę *lightweight view* (widok wagi lekkiej). Aby wybrać ten widok u siebie, należy kliknąć łącze Lightweight View lub łącze Switch View. Łącze Switch View jest widoczne w prawym dolnym rogu ekranu.

## Wewnątrz pomocy MSDN

Trzeba zapamiętać kilka rzeczy, które pozwolą na najlepsze wykorzystywanie dokumentacji systemu pomocy. Po pierwsze, u góry zawartości okna jest podana informacja o wersji. MSDN obsługuje wiele wersji Visual Studio i .NET Framework. Jak zobaczymy w dalszej części tej książki, aktualną wersją .NET Framework jest wersja 4.

W sekcji Syntax zawartości systemu pomocy należy koniecznie wybrać kartę VB. Zapewni to wyświetlanie jedynie składni i fragmentów kodu języka Visual Basic. Pozostałe języki będą ukryte, co ułatwi czytanie dokumentacji. Dokonany wybór zostanie zapamiętany i uwzględniony przy następnym otwarciu dokumentacji.

Z lewej strony okna pomocy widnieje uproszczony spis treści. Tytuł aktualnie wyświetlonego tematu jest pogrubiony i ma inny kolor. Powyżej bieżącego tematu znajdują się tematy nadrzędne, a poniżej - podrzędne. Poniżej spisu treści widnieje sekcja Related Links. W tej sekcji znajdują się tematy pokrewne. Klikając łącza w obszarze spisu treści można poruszać się po całej dokumentacji. Powyżej spisu treści znajduje się pole wyszukiwania, oferujące jeszcze jeden sposób docierania do właściwych miejsc dokumentacji.

Tabela 1-2 zawiera kilka wskazówek dotyczących systemu pomocy, które mogą być przydatne podczas nauki Visual Studio IDE, Visual Basic i .NET Framework.

**Tabela 1-2** Lokalizacja tematów pomocy w Visual Studio 2010

| Aby uzyskać pomoc na temat | Należy |
|---|---|
| Dokumentacja | Wybrać polecenie View Help z menu Help.<br>*lub*<br>Otworzyć stronę *http://msdn.Microsoft.com/library/* w przeglądarce, aby wyświetlić pomoc online. |
| Visual Studio IDE | Zaznaczyć element w Visual Studio i nacisnąć klawisz F1.<br>*lub*<br>Przeszukać dokumentację systemu pomocy dla „Visual Studio 2010”. |
| Okno dialogowe Visual Studio | Kliknąć przycisk systemu pomocy (ze znakiem zapytania) na pasku tytułu okna dialogowego. |
| Visual Basic | Przeszukać dokumentację systemu pomocy dla „Getting Started with Visual Basic”. |
| .NET Framework | Przeszukać dokumentację systemu pomocy dla „.NET Framework 4”. |
| Windows Forms | Przeszukać dokumentację systemu pomocy dla „Getting Started with Windows Forms”. |
| Słowo kluczowe lub instrukcja w oknie Code Editor | Zaznaczyć słowo kluczowe lub instrukcję programu i nacisnąć klawisz F1. |

# Dopasowywanie ustawień IDE do opisywanych ćwiczeń

Podobnie jak w przypadku okien narzędziowych i innych elementów środowiska IDE również ustawienia kompilatora w Visual Studio można gruntownie dostosowywać. Należy teraz koniecznie sprawdzić kilka z tych ustawień u siebie, aby konfiguracja Visual Studio była kompatybilna, opisywanymi dalej ćwiczeniami „krok po kroku". Zobaczymy również, jak ogólnie dostosowywać Visual Studio, tak aby z chwilą nabrania wprawy w programowaniu każdy mógł skonfigurować ten program w sposób najwygodniejszy dla siebie.

## Ustawianie IDE do programowania w języku Visual Basic

Pierwsze ustawienie, które należy sprawdzić, zostało dokonane podczas instalacji Visual Studio w komputerze. Pojawiło się wówczas pytanie, jak ma być skonfigurowane ogólne środowisko programowania w Visual Studio. Ponieważ Visual Studio jest narzędziem programistycznym o wielu zastosowaniach, do wyboru jest wiele opcji – programowanie w Visual Basic, programowanie w Visual C++, programowanie w Visual C#, programowanie dla sieci Web i tak dalej. Dokonany wybór ma wpływ nie tylko na edytor kodu i dostępne narzędzia programistyczne, ale także na polecenia w menu i na pasku narzędzi oraz na zawartość kilku okien narzędziowych. Z tego powodu, jeśli ktoś planuje używać tej książki do nauki programowania w języku Visual Basic, a pierwotnie skonfigurował Visual Studio dla innego języka, niektóre polecenia z menu i procedury opisane w tej książce nie będą pasowały dokładnie do istniejącej konfiguracji.

Na szczęście można usunąć tę niespójność i jednocześnie przećwiczyć zmianę ustawień środowiska za pomocą polecenia Import And Export Settings z menu Tools. Zobaczymy za chwilę, jak zmienić ustawienia środowiska na potrzeby programowania w języku Visual Basic i dopasować je do tej książki.

### Ustawianie IDE na potrzeby programowania w języku Visual Basic

1. W menu Tools kliknij polecenie Import And Export Settings.

**Wskazówka** W wersji Visual Basic 2010 Express należy rozwinąć menu Tools, kliknąć Settings i następnie kliknąć Import And Export Settings.

Używając kreatora, który się pojawi na ekranie, można zapisać ustawienia środowiska w celu wykorzystania ich w innym komputerze, pobrać ustawienia z innego komputera albo zresetować lokalne ustawienia.

2. Kliknij Reset All Settings, a następnie kliknij Next.

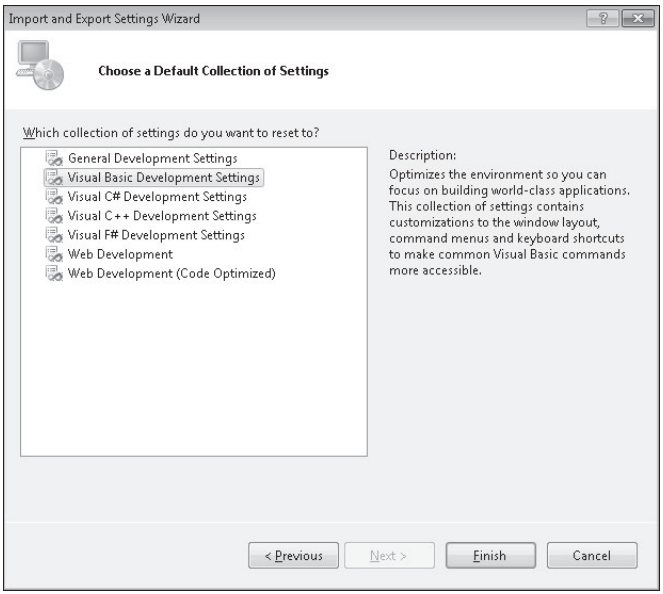
Program Visual Studio zapyta, czy przed skonfigurowaniem IDE dla innego środowiska programowania ma zapisać bieżące ustawienia w pliku. Dobrze jest zawsze utworzyć taką kopię zapasową, aby móc przywrócić ostatnie ustawienia, gdyby nowe okazały się błędne.

3. Sprawdź, czy jest zaznaczona opcja Yes, Save My Current Settings (tak, zapisz moje bieżące ustawienia) i zanotuj nazwę pliku i lokalizację folderu, w którym Visual Studio planuje zapisać ustawienia.

   Aby przywrócić później stare ustawienia, należy użyć tego samego kreatora i zaznaczyć opcję Import Selected Environmental Settings (importuj wybrane ustawienia środowiska).

4. Kliknij Next, aby wyświetlić standardową listę ustawień dostępnych dla Visual Studio.

   W zależności od zbioru zainstalowanych składników Visual Studio pojawi się lista ustawień podobna do pokazanej na kolejnym rysunku:




5. Kliknij opcję Visual Basic Development Settings (jeśli nie jest zaznaczona), a następnie przycisk Finish.

**Wskazówka** W wersji Visual Basic 2010 Express należy kliknąć Expert Settings, a następnie Finish.

Kreator zmieni dotychczasowe ustawienia IDE, w tym polecenia menu, paski narzędzi oraz ustawienia w kilku oknach dialogowych, oknach narzędziowych i w edytorze kodu. W każdej chwili można bez obaw powtórzyć ten proces dostosowania

i ponownie zresetować ustawienia, na przykład, jeśli zmiana nastąpiła omyłkowo albo trzeba dostosować Visual Studio do innego środowiska programowania.

6. Kliknij Close, aby zakończyć pracę kreatora.

## Sprawdzanie ustawień projektu i kompilatora

Jeśli zostały wybrane prawidłowe ustawienia dla środowiska Visual Basic, będzie można przystąpić do ćwiczeń z programowania. Jeśli jednak nie ma takiej pewności – bo na przykład po dokonaniu konfiguracji korzystaliśmy przez jakiś czas z programu Visual Studio 2010 albo sam komputer jest współużytkowany przez wielu programistów, którzy mogli zmodyfikować standardowe ustawienia (dotyczy to na przykład komputerów w sali do ćwiczeń) – należy wykonać opisane niżej działania, aby sprawdzić, czy ustawienia związane z projektami, rozwiązaniami i kompilatorem Visual Basic pasują do obowiązujących w książce.

### Sprawdzanie ustawień projektów i kompilatora

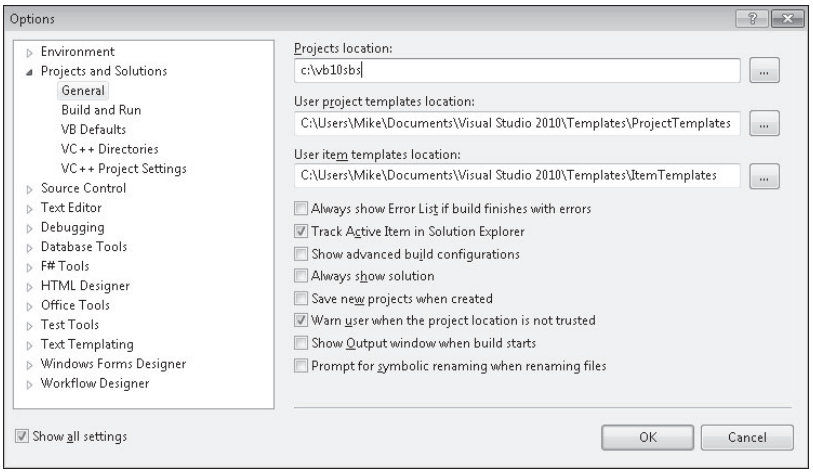
1. Wybierz polecenie Options z menu Tools, aby wyświetlić okno dialogowe Options.

   Okno dialogowe Options jest miejscem dostosowywania wielu ustawień Visual Studio. Aby wyświetlić wszystkie dostępne ustawienia, należy zaznaczyć pole wyboru Show All Settings w lewym dolnym rogu tego okna.

2. Rozwiń kategorię Projects And Solutions i kliknij pozycję General.

   Ta grupa pól wyboru i opcji odpowiada za konfigurację ustawień Visual Studio dotyczących projektów i rozwiązań.

3. Aby Twoje programy były zgodne z przykładami w tej książce, zastosuj ustawienia pokazane na poniższym rysunku.



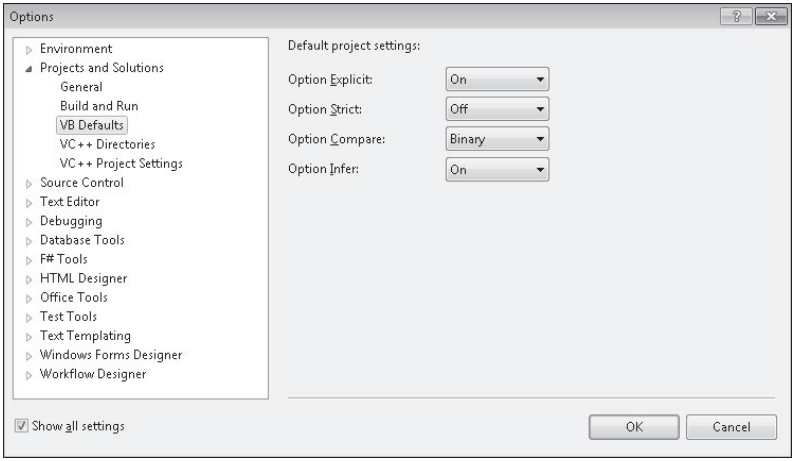
W szczególności dobrze jest usunąć zaznaczenie pól wyboru Always Show Solution i Save New Projects When Created (jeśli są zaznaczone). Pierwsza z tych opcji powoduje wyświetlanie w IDE dodatkowych poleceń dotyczących rozwiązań. Nie jest to konieczne w przypadku rozwiązań zawierających tylko jeden projekt (taka sytuacja ma miejsce w przypadku większości programów w tej książce). Druga opcja powoduje wstrzymywanie się przez Visual Studio z zapisaniem projektu do momentu kliknięcia polecenia Save All w menu File i podania miejsca dla pliku. Taki „opóźniony zapis" pozwala utworzyć program testowy, skompilować go i sprawdzić, a nawet uruchomić bez fizycznego zapisania projektu na dysku – wygodna możliwość, gdy trzeba utworzyć na próbę krótki program, a następnie usunąć go całkowicie (odpowiednikiem tej sytuacji w dziedzinie przetwarzania tekstu jest otwarcie nowego dokumentu programu Word, wprowadzenie adresu dla etykiety pocztowej, wydrukowanie jej, a następnie wyjście z programu Word bez zapisywania pliku). Przy standardowych ustawieniach podczas wykonywania ćwiczeń z tej książki będzie pojawiać się monit o zapisanie każdego projektu zaraz po jego utworzeniu. Można również z góry zażądać zapisywania tych projektów, zaznaczając pole wyboru Save New Projects When Created.

Jak można również zauważyć na rysunku w polu tekstowym Projects Location wpisana jest lokalizacja „C:\Vb10sbs". Jest to standardowe miejsce plików z przykładami do tej książki. Większość tworzonych dalej projektów będzie zapisywana w tym samym folderze i będzie miała prefiks „My", aby odróżnić je od gotowych projektów dostarczonych razem z książką w celu analizy (trzeba koniecznie pamiętać o zmianie tego ustawienia w swoim komputerze).

Po dostosowaniu wymienionych ustawień można przejść do sprawdzania ustawień kompilatora Visual Basic.

**4.** Kliknij pozycję VB Defaults w oknie dialogowym Options.

Visual Studio wyświetli listę czterech ustawień kompilatora: Option Explicit, Option Strict, Option Compare i Option Infer. Ekran powinien wyglądać tak, jak na rysunku.



Szczegółowy opis tych ustawień wykracza poza zakres tego rozdziału. Niemniej należy upewnić się, czy opcja Option Explicit jest ustawiona na On, a opcja Option Strict na Off (są to domyślne ustawienia dla środowiska Visual Basic w Visual Studio). Ustawienie Explicit On wymusza deklarowanie zmiennych przed ich pierwszym użyciem w programie – bardzo dobra praktyka, do której zachęcam. Ustawienie Strict Off pozwala na konwersję w pewnych okolicznościach zmiennych i obiektów różnego typu bez generowania błędu przez kompilator (na przykład, może to dotyczyć przypisania liczby do pola tekstowego). Chociaż jest to potencjalnie ryzykowna praktyka programowania, ustawienie Option Strict Off jest wygodne dla pewnego typu programów demonstracyjnych. Bez tego ustawienia kilka projektów z tej książki może wygenerować komunikaty błędów podczas próby uruchomienia.

Opcja Compare określa metodę porównywania podczas sortowania i porównywania różnych napisów. Więcej informacji o porównywaniu napisów i sortowaniu tekstu można znaleźć w rozdziale 13.

Opcja Infer pojawiła się w wersji Visual Basic 2008. Po ustawieniu Option Strict na Off i Option Infer na On można deklarować zmienne bez jawnego określania typu danych; w przypadku takiej deklaracji kompilator Visual Basic wywnioskuje typ danych (lub postawi hipotezę na jego temat) na podstawie pierwszego przypisania wartości do zmiennej. Projektanci kompilatora Visual Basic dodali to ustawienie, aby ułatwić pisanie kodu przy jednoczesnym zachowaniu korzyści z deklaracji typu. Więcej informacji na ten temat można znaleźć w rozdziale 5.

Zasadniczo dobrze jest ustawiać opcję Option Infer na Off, aby uniknąć nieoczekiwanych efektów podczas używania zmiennych w programach. Takie ustawienie obowiązuje w większości przykładowych projektów zamieszczonych na dołączonym dysku CD.

5. Posprawdzaj w oknie dialogowym Options dodatkowe ustawienia związane ze środowiskiem programowania i z Visual Studio. Na koniec kliknij przycisk OK, aby zamknąć to okno.

Teraz można już wyjść z programu Visual Studio i przystąpić do programowania.

# O krok dalej: wyjście z programu Visual Studio

Każdy rozdział tej książki kończy się podrozdziałem zatytułowanym „Jeden krok dalej", który pozwala przećwiczyć dodatkową czynność związaną z aktualnym tematem. Za „Jeden krok dalej" występuje zawsze tabela „Krótkie podsumowanie", która przypomina najważniejsze zagadnienia omówione w rozdziale. Jeśli więc trzeba szybko coś sprawdzić, można skorzystać z tej tabeli.

Po skończeniu korzystania z Visual Studio danego dnia można zapisać wszystkie otwarte projekty i zamknąć środowisko programowania. Spróbujemy teraz to zrobić.

## Wychodzenie z programu Visual Studio

1. Zapisz wszystkie zmiany dokonane w programie klikając przycisk Save All na pasku narzędzi Standard.

   Jak zostało to opisane w poprzednim podrozdziale, standardowo w Visual Studio 2010 każdy program musi mieć nadaną nazwę w momencie rozpoczęcia projektu lub rozwiązania, natomiast określenie lokalizacji pliku oraz zapisanie projektu jest konieczne dopiero po kliknięciu przycisku Save All lub wybraniu polecenia Save All z menu File. Ponieważ w naszym projekcie nastąpiło kilka zmian, trzeba je teraz zapisać.

2. W menu File kliknij polecenie Exit.

   Program Visual Studio zostanie zamknięty. Czas przejść do rozdziału 2. i naszego pierwszego programu!