



James Foxall

Visual Basic 2015 w 24 godziny 🕒

SAMS

Helion 

Tytuł oryginału: Visual Basic® 2015 in 24 Hours, Sams Teach Yourself, Barnes & Noble Special Edition

Tłumaczenie: Andrzej Watrak

Projekt okładki: Studio Gravite / Olsztyn

Obarek, Pokoński, Pazdrijowski, Zaprucki

Materiały graficzne na okładce zostały wykorzystane za zgodą Shutterstock Images LLC.

ISBN: 978-83-283-2874-7

Authorized translation from the English language edition: Sams Teach Yourself, Visual Basic 2015 in 24 Hours, Barnes & Noble, First Edition, ISBN 0672337541; by James Foxall; published by Pearson Education, Inc, publishing as SAMS Publishing.
Copyright © 2016 by Pearson Education, Inc.

All rights reserved. No part of this book may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording or by any information storage retrieval system, without permission from Pearson Education Inc.

Polish language edition published by HELION S.A. Copyright © 2016.

Wszelkie prawa zastrzeżone. Nieautoryzowane rozpowszechnianie całości lub fragmentu niniejszej publikacji w jakiegokolwiek postaci jest zabronione. Wykonywanie kopii metodą kserograficzną, fotograficzną, a także kopiowanie książki na nośniku filmowym, magnetycznym lub innym powoduje naruszenie praw autorskich niniejszej publikacji.

Wszystkie znaki występujące w tekście są zastrzeżonymi znakami firmowymi bądź towarowymi ich właścicieli.

Autor oraz Wydawnictwo HELION dołożyli wszelkich starań, by zawarte w tej książce informacje były kompletne i rzetelne. Nie biorą jednak żadnej odpowiedzialności ani za ich wykorzystanie, ani za związane z tym ewentualne naruszenie praw patentowych lub autorskich. Autor oraz Wydawnictwo HELION nie ponoszą również żadnej odpowiedzialności za ewentualne szkody wynikłe z wykorzystania informacji zawartych w książce.

Wydawnictwo HELION

ul. Kościuszki 1c, 44-100 GLIWICE

tel. 32 231 22 19, 32 230 98 63

e-mail: helion@helion.pl

WWW: <http://helion.pl> (księgarnia internetowa, katalog książek)

Pliki z przykładami omawianymi w książce można znaleźć pod adresem:

<ftp://ftp.helion.pl/przyklady/vb1524.zip>

Drogi Czytelniku!

Jeżeli chcesz ocenić tę książkę, zajrzyj pod adres

<http://helion.pl/user/opinie/vb1524>

Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

Printed in Poland.

- [Kup książkę](#)
- [Poleć książkę](#)
- [Oceń książkę](#)

- [Księgarnia internetowa](#)
- [Lubię to! » Nasza społeczność](#)

Spis treści

Wprowadzenie	13
--------------------	----

CZĘŚĆ I ŚRODOWISKO PROGRAMISTYCZNE VISUAL BASIC 2015

Godzina 1.	Skok na głęboką wodę: programowanie w Visual Basic 2015	19
	Uruchomienie środowiska Visual Studio 2015	20
	Tworzenie nowego projektu	21
	Środowisko programistyczne Visual Studio 2015	24
	Zmianieanie cech obiektów	25
	Umieszczanie kontrolki w formularzu	29
	Projektowanie interfejsu użytkownika	31
	Tworzenie kodu obsługującego interfejs użytkownika	35
	Uruchomienie projektu	39
	Podsumowanie	41
	Pytania i odpowiedzi	42
	Warsztat	42
	Ćwiczenia	42
Godzina 2.	Obsługa środowiska Visual Studio 2015	43
	Strona startowa środowiska Visual Studio 2015	44
	Obsługa i dostosowywanie środowiska Visual Studio	46
	Dodawanie kontrolki do formularza za pomocą panelu Toolbox	51
	Ustawianie właściwości obiektów w panelu Properties	53
	Zarządzanie projektami	59
	Prowizoryczny podręcznik programowania	65
	Pomoc	67
	Podsumowanie	68
	Pytania i odpowiedzi	68
	Warsztat	69
	Ćwiczenia	69

Godzina 3.	Obiekty i kolekcje	71
	Obiekty	72
	Właściwości	72
	Metody	79
	Prosty przykład użycia obiektów	80
	Kolekcje	84
	Przeglądarka obiektów	87
	Podsumowanie	88
	Pytania i odpowiedzi	89
	Warsztat	89
	Ćwiczenia	90
Godzina 4.	Zdarzenia	91
	Kod sterowany zdarzeniami	91
	Przykładowy projekt z obsługą zdarzeń	99
	Aktualizacja nazw procedur zdarzeń	104
	Podsumowanie	105
	Pytania i odpowiedzi	105
	Warsztat	105
	Ćwiczenia	106
 CZĘŚĆ II TWORZENIE INTERFEJSU UŻYTKOWNIKA		
Godzina 5.	Podstawy tworzenia formularzy	109
	Zmianianie nazwy formularza	110
	Zmianianie wyglądu formularza	110
	Wyświetlanie i ukrywanie formularzy	121
	Podsumowanie	127
	Pytania i odpowiedzi	127
	Warsztat	128
	Ćwiczenia	128
Godzina 6.	Zaawansowane techniki projektowania formularzy	129
	Definiowanie kontroltek	129
	Tworzenie niemodalnych okien widocznych zawsze na pierwszym planie	146
	Tworzenie przezroczystych formularzy	147
	Tworzenie przewijanych formularzy	147
	Tworzenie interfejsu MDI	149
	Ustawianie formularza startowego	152

Podsumowanie	153
Pytania i odpowiedzi	153
Warsztat	154
Ćwiczenia	154
Godzina 7. Praca z tradycyjnymi kontrolkami	155
Wyświetlanie statycznego tekstu za pomocą etykiet	155
Pole do wprowadzania tekstu	157
Tworzenie przycisków	162
Tworzenie kontenerów i grup kontroltek	166
Listy elementów	170
Tworzenie list rozwijanych	176
Podsumowanie	178
Pytania i odpowiedzi	179
Warsztat	179
Ćwiczenia	180
Godzina 8. Zaawansowane kontrolki	181
Tworzenie czasomierzy	181
Tworzenie okien dialogowych z zakładkami	184
Przechowywanie obrazów w kontrolce ImageList	187
Tworzenie zaawansowanych list za pomocą kontrolki ListView	189
Tworzenie hierarchicznych list za pomocą kontrolki TreeView	194
Podsumowanie	198
Pytania i odpowiedzi	198
Warsztat	199
Ćwiczenia	199
Godzina 9. Dodawanie pasków menu i narzędzi do formularzy	201
Tworzenie menu	201
Kontrolka paska narzędzi	213
Tworzenie paska stanu	218
Podsumowanie	220
Pytania i odpowiedzi	220
Warsztat	221
Ćwiczenia	221

CZĘŚĆ III WPRAWIANIE APLIKACJI W RUCH — PROGRAMOWANIE

Godzina 10.	Tworzenie i wywoływanie procedur	225
	Tworzenie modułów kodu Visual Basic	225
	Tworzenie kodu procedur	228
	Wywoływanie procedur	233
	Wychodzenie z procedur	238
	Zapobieganie nieskończonym wywołaniom rekurencyjnym	239
	Podsumowanie	240
	Pytania i odpowiedzi	240
	Warsztat	241
	Ćwiczenia	241
Godzina 11.	Stałe, typy danych, zmienne i tablice	243
	Typy danych	244
	Definiowanie i stosowanie stałych	247
	Deklarowanie zmiennych i odwoływanie się do nich	249
	Tablice	255
	Określanie zakresu widoczności	258
	Deklarowanie zmiennych statycznych	262
	Zastosowanie zmiennych w projekcie Przeglądarka obrazów	263
	Modyfikowanie nazw zmiennych	267
	Podsumowanie	267
	Pytania i odpowiedzi	268
	Warsztat	269
	Ćwiczenia	270
Godzina 12.	Operacje na liczbach, tekstach, datach i godzinach	271
	Operacje arytmetyczne w języku Visual Basic	271
	Porównywanie wartości	275
	Operatory logiczne	276
	Operacje na ciągach znaków	278
	Operacje na datach i godzinach	283
	Podsumowanie	288
	Pytania i odpowiedzi	288
	Warsztat	289
	Ćwiczenia	289

Godzina 13.	Podejmowanie decyzji w kodzie Visual Basic	291
	Podejmowanie decyzji za pomocą instrukcji If...Then	291
	Instrukcja Select...Case wykorzystująca wyrażenie zwracające różne wartości	296
	Przechodzenie do innych miejsc kodu za pomocą instrukcji GoTo	302
	Podsumowanie	303
	Pytania i odpowiedzi	304
	Warsztat	305
	Ćwiczenia	306
Godzina 14.	Efektywne pętle	307
	Wykonywanie kodu określoną liczbę razy za pomocą instrukcji For...Next	307
	Wykonywanie kodu nieokreśloną liczbę razy za pomocą pętli Do...Loop	313
	Podsumowanie	317
	Pytania i odpowiedzi	318
	Warsztat	318
	Ćwiczenia	319
Godzina 15.	Diagnostyka kodu	321
	Umieszczanie komentarzy w kodzie	322
	Dwa podstawowe rodzaje błędów	323
	Narzędzia diagnostyczne środowiska Visual Studio	326
	Obsługa błędów za pomocą instrukcji Try...Catch...Finally	335
	Podsumowanie	341
	Pytania i odpowiedzi	341
	Warsztat	342
	Ćwiczenia	343
Godzina 16.	Tworzenie obiektów za pomocą klas	345
	Klasy	346
	Tworzenie instancji obiektów za pomocą klas	353
	Podsumowanie	359
	Pytania i odpowiedzi	359
	Warsztat	359
	Ćwiczenia	360

Godzina 17.	Interakcje z użytkownikiem	361
	Wyświetlanie komunikatów za pomocą metody <code>MessageBox.Show()</code>	361
	Tworzenie własnych okien dialogowych	367
	Uzyskiwanie informacji od użytkownika za pomocą procedury <code>InputDialog()</code>	370
	Interakcje przy użyciu klawiatury	372
	Wykorzystywanie najważniejszych zdarzeń wywołanych za pomocą myszy	375
	Podsumowanie	377
	Pytania i odpowiedzi	378
	Warsztat	378
	Ćwiczenia	379
Godzina 18.	Praca z grafiką	381
	Obiekt <code>Graphics</code>	381
	Stosowanie piór	384
	Stosowanie kolorów systemowych	385
	Przeznaczenie prostokątów	387
	Rysowanie kształtów	389
	Rysowanie tekstu	390
	Utrwalanie obrazu formularza	391
	Przykładowy projekt z obsługą grafiki	391
	Podsumowanie	397
	Pytania i odpowiedzi	397
	Warsztat	397
	Ćwiczenia	398
 CZĘŚĆ IV PRACA Z DANYMI		
Godzina 19.	Operacje na plikach	401
	Zastosowanie kontrolki <code>OpenFileDialog</code> i <code>SaveFileDialog</code>	401
	Wykonywanie operacji na plikach za pomocą obiektu <code>File</code>	407
	Wykonywanie operacji na folderach za pomocą obiektu <code>System.IO.Directory</code>	415
	Podsumowanie	416
	Pytania i odpowiedzi	416
	Warsztat	417
	Ćwiczenia	417

Godzina 20.	Operacje na rejestrze i plikach tekstowych	419
	Korzystanie z rejestru systemu Windows	419
	Odczytywanie i zapisywanie plików tekstowych	430
	Podsumowanie	438
	Pytania i odpowiedzi	439
	Warsztat	439
	Ćwiczenia	440
Godzina 21.	Praca z bazą danych	441
	Podstawy platformy ADO.NET	442
	Przetwarzanie danych	448
	Podsumowanie	458
	Pytania i odpowiedzi	458
	Warsztat	459
	Ćwiczenia	460
Godzina 22.	Drukowanie	461
	Przygotowanie projektu Przeglądarka obrazów	462
	Wydruk i podgląd wydruku dokumentu	464
	Zmiana ustawień strony	473
	Skalowanie obrazu na stronie	476
	Podsumowanie	479
	Pytania i odpowiedzi	480
	Warsztat	480
	Ćwiczenia	481
Godzina 23.	Wysyłanie wiadomości e-mail	483
	Klasy do wysyłania wiadomości e-mail	484
	Wysyłanie wiadomości za pomocą aplikacji Przeglądarka obrazów	484
	Podsumowanie	494
	Pytania i odpowiedzi	495
	Warsztat	495
	Ćwiczenia	495
CZĘŚĆ V APLIKACJE — WDRAŻANIE I NIE TYLKO		
Godzina 24.	Wdrażanie aplikacji	499
	Technologia ClickOnce	499
	Utworzenie programu instalacyjnego za pomocą kreatora Publish Wizard	500

	Sprawdzenie programu instalacyjnego aplikacji Przeglądarka obrazów	502
	Oinstalowanie udostępnionej aplikacji	504
	Zaawansowane opcje technologii ClickOnce	506
	Podsumowanie	507
	Pytania i odpowiedzi	507
	Warsztat	508
	Ćwiczenia	508
Godzina 25.	Sterowanie aplikacjami Microsoft Office 2016	509
	Sterowanie programem Microsoft Excel	510
	Sterowanie programem Microsoft Word	516
	Podsumowanie	519
	Pytania i odpowiedzi	519
	Warsztat	520
	Ćwiczenia	520
Godzina 26.	Tworzenie własnych kontroltek	521
	Tworzenie kontrolki potomnej	522
	Tworzenie kontrolki zagregowanej	527
	Podsumowanie	537
	Pytania i odpowiedzi	538
	Warsztat	538
	Ćwiczenia	539
DODATKI		
Dodatek A	Z szerokiej perspektywy	543
	Platforma .NET	543
	Język Microsoft IL	544
	Przestrzenie nazw	545
	Wspólny system typów	547
	Porządkowanie pamięci	547
	Dodatkowe materiały	547
	Podsumowanie	548
	Skorowidz	549

Godzina 6.

Zaawansowane techniki projektowania formularzy

W ciągu tej godziny poznasz następujące zagadnienia:

- ▶ dodawanie kontroltek do formularza,
- ▶ rozmieszczanie, wyrównywanie, kotwiczenie i określanie wielkości kontroltek,
- ▶ ustalanie kolejności wyróżniania kontroltek,
- ▶ ustalanie kolejności warstw kontroltek,
- ▶ tworzenie formularzy wyświetlanych zawsze na wierzchu,
- ▶ tworzenie przezroczystych formularzy,
- ▶ tworzenie interfejsów wielodokumentowych.

Formularz jest tylko ramą. Choć można go dostosowywać, zmieniając jego właściwości, jednak aby był funkcjonalny, trzeba w nim umieścić kontrolki. W poprzedniej godzinie dowiedziałeś się, jak dodawać formularze do projektu, wyświetlać je, ukrywać i ustawiać ich podstawowe właściwości. W ciągu tej godziny nauczysz się dodawać kontrolki, porządkować je i wyrównywać, aby tworzyć atrakcyjne i funkcjonalne interfejsy użytkownika. Dowiesz się również, jak utworzyć zaawansowany interfejs obsługujący wiele dokumentów (ang. *Multiple-Document Interfaces*, MDI). Po zapoznaniu się z materiałem przedstawionym w tej godzinie będziesz przygotowany do zgłębiania szczegółów dotyczących kontroltek dostępnych w języku Visual Basic.

Definiowanie kontroltek

Kontrolki są umieszczanymi w formularzu obiektami, z których korzysta użytkownik. Jeżeli wykonałeś przykłady opisane w poprzednich godzinach, potrafisz już dodawać kontrolki do formularza. Jednak w przyszłości będziesz dodawać mnóstwo kontroltek, dlatego ważne jest dokładne poznanie wszystkich aspektów tego procesu.

Dodawanie kontroltek do formularza

Wszystkie kontrolki, które można dodać do formularza, znajdują się w panelu *Toolbox*, domyślnie zadokowanym w lewej części okna środowiska programistycznego. Takie położenie panelu jest przydatne tylko wtedy, gdy kontrolki do formularza dodaje się sporadycznie.

Jednak gdy będziesz projektować zaawansowane formularze, najwygodniej będzie zadokować lub przypiąć ten panel przy prawej krawędzi okna środowiska, aby nie przesłaniał twórczonego formularza.

Kontrolki w panelu *Toolbox* podzielone są na kategorie, które można rozwijać i zwiijać. W większości projektów będziesz używać kontroltek z kategorii *Common Controls* (popularne kontrolki). Jednak w miarę nabywania umiejętności będziesz stosować bardziej zaawansowane i wyspecjalizowane kontrolki znajdujące się w innych kategoriach.

Kontrolki można dodawać do formularza na trzy sposoby, które już poznałeś. Otwórz projekt *Przeglądarka obrazów*, nad którym pracowałeś w poprzedniej godzinie (lub otwórz początkowy projekt pobrany pod adresem <ftp://ftp.helion.pl/przyklady/vb1524.zip>), i w panelu *Solution Explorer* kliknij dwukrotnie element *FormularzOpcji.vb*, aby otworzyć projekt formularza *Opcje*.

Dodanie kontrolki poprzez jej dwukrotne kliknięcie w panelu Toolbox

Najprostszym sposobem dodania kontrolki do formularza jest jej dwukrotne kliknięcie w panelu *Toolbox*. Wypróbuj teraz ten sposób: otwórz panel *Toolbox* i kliknij dwukrotnie kontrolkę *TextBox*. W lewym górnym rogu formularza pojawi się pole tekstowe (aby je zobaczyć, musisz zamknąć panel *Toolbox*, przesuując kursor myszy poza jego obszar). Nowa kontrolka, po dwukrotnym kliknięciu, jest umieszczana na wierzchu aktualnie wybranej kontrolki (nie dotyczy to niewidocznych kontroltek) i ma domyślną wielkość. Jeżeli formularz jest pusty, nowa kontrolka jest umieszczana w jego lewym górnym rogu, o czym już się przekonałeś. Po dodaniu kontrolki można ją dowolnie przesuwać i zmieniać jej wielkość.

Dodanie kontrolki poprzez jej przeciągnięcie z panelu Toolbox

Jeżeli chcesz mieć większą kontrolę nad umiejscowieniem kontrolki, przeciągnij ją do formularza. Spróbuj teraz: otwórz panel *Toolbox*, kliknij kontrolkę *Button* i przeciągnij ją do formularza. Następnie zwolnij przycisk myszy w pobliżu miejsca, w którym ma się znaleźć kontrolka.

Dodanie kontrolki poprzez jej narysowanie

Ostatnim sposobem dodania kontrolki do formularza jest jej narysowanie. Wykonaj poniższe kroki:

1. Otwórz panel *Toolbox* i kliknij kontrolkę *ListBox*, aby ją zaznaczyć.
2. Umieść kursor myszy w miejscu, w którym ma się znaleźć lewy górny róg kontrolki, a następnie naciśnij i przytrzymaj lewy przycisk myszy.
3. Przeciągnij kursor myszy do miejsca, w którym ma się znaleźć prawy dolny róg kontrolki, i zwolnij przycisk myszy.

Zostanie utworzone pole listy o wielkości odpowiadającej narysowanemu przez Ciebie prostokątowi. Jest to najdokładniejszy sposób dodawania kontroltek do formularza.

Czy wiedziałeś?

Jeżeli zamierzasz dodawać kontrolki, klikając je lub przeciągając, bardzo zalecam uwolnienie panelu *Toolbox* lub przypięcie go do okna środowiska. Panel ten, gdy jest zadokowany i ma ustawioną opcję automatycznego ukrywania, utrudnia umieszczanie kontrolek, ponieważ przesłania sporą część formularza.

Warto pamiętać, że pierwszym elementem w każdej kategorii kontrolek jest *Pointer*, który w rzeczywistości nie jest kontrolką. Po wybraniu tego elementu edytor przechodzi w tryb zaznaczania, a nie tworzenia kontrolek. W tym trybie po kliknięciu kontrolki można przeglądać jej właściwości. Jest to domyślny tryb środowiska programistycznego.

Manipulowanie kontrolkami

Dodanie kontrolek do formularza jest proste, ale rozmieszczenie ich w taki sposób, aby powstał intuicyjny i atrakcyjny interfejs użytkownika jest prawdziwym wyzwaniem. Możliwości modyfikacji formularza są niemal nieograniczone, więc nie mogę Ci doradzić, jak projektować określony interfejs (jednakże bardzo zalecam, abyś tworzył interfejsy podobne do spotykanych w komercyjnych aplikacjach). Pokażę za to techniki przemieszczania, zmieniania wielkości i rozmieszczania kontrolek tak, aby prezentowały się zgodnie z Twoimi oczekiwaniami. Gdy opanujesz te techniki, będziesz o wiele sprawniej tworzył interfejsy i miał więcej czasu na tworzenie kodu realizującego potrzebne operacje.

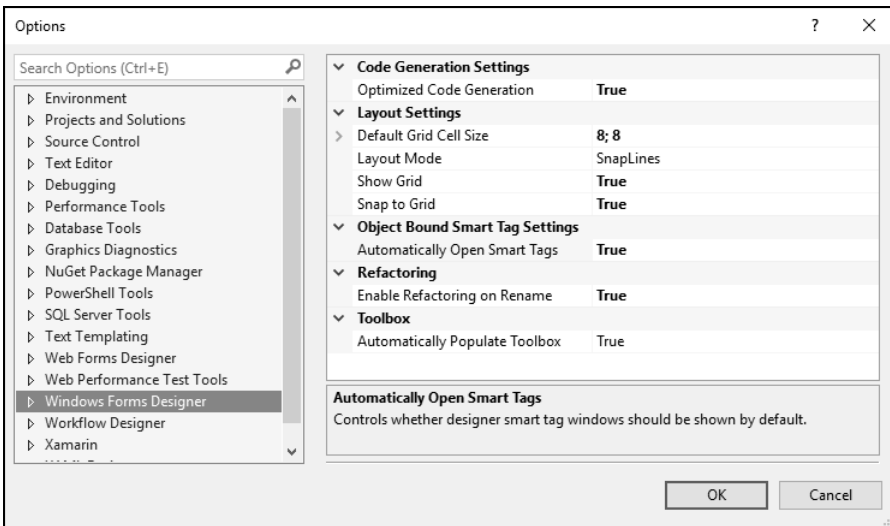
Siatka (gęstość i przyciąganie)

Zapewne podczas pracy z kontrolkami zauważyłeś, że są one „przyciągane” do niewidzialnej siatki. To nie złudzenie — rzeczywiście tak jest. Gdy siatka jest aktywna, wówczas podczas rysowania lub przesuwania kontrolki jej współrzędne są automatycznie zaokrąglane do współrzędnych najbliższego węzła siatki. Dzięki temu uzyskuje się równe wielkości i położenie kontrolek. W praktyce jednak siatka rzadko mi się przydaje, ponieważ żądane wielkości i umiejscowienie kontrolek rzadko odpowiadają położeniu węzłów siatki. Można jednak zmienić gęstość siatki, a nawet ją wyświetlić, co proponuję zrobić.

Ustawienia siatki są globalne dla całego środowiska Visual Studio — nie trzeba ich określać indywidualnie dla każdego projektu czy formularza. Aby wyświetlić siatkę, wybierz polecenie *Tools/Options*. Następnie w oknie *Options* kliknij po lewej stronie sekcję *Windows Forms Designer* (edytor formularzy Windows), jak na rysunku 6.1.

Interesują nas poniższe ustawienia:

- ▶ *Default Grid Cell Size* (domyślna wielkość oczka siatki) — ustawienie określające wielkość oczka siatki w pikselach w kierunkach poziomym i pionowym. Im mniejsza wielkość oczka, tym większa dokładność w określaniu wielkości i położenia kontrolek.
- ▶ *Layout Mode* (tryb rozmieszczania) — ustawienie określające, czy kontrolki mają być przyciągane do siatki i wyrównywane względem innych kontrolek.
- ▶ *Show Grid* (pokaż siatkę) — opcja określająca, czy w projekcie formularza ma być wyświetlana siatka.



RYSUNEK 6.1. Ustawienia siatki są globalne w skali całego środowiska Visual Studio 2015

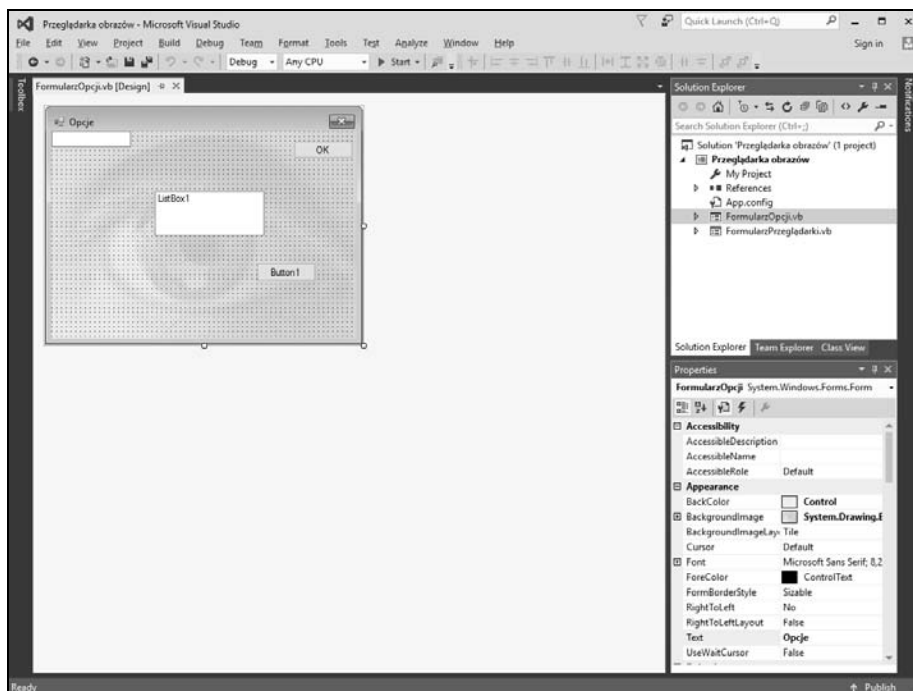
- ▶ *Snap to Grid* (przyciągaj do siatki) — opcja określająca, czy ma być stosowana siatka. Jeżeli opcja ma wartość *False*, wtedy ustawienia siatki nie są aktywne i kontrolki nie są do niej przyciągane.

W tej chwili siatka nie jest stosowana podczas rysowania kontrolki, ale w czasie przesuwania są one przyciągane do innych kontrolki, ponieważ opcja *Layout Mode* jest ustawiona na *SnapLines* (przyciągaj do linii). Ustawienie to dokładnie opiszę w dalszej części rozdziału. Teraz chcę Ci pokazać, jak funkcjonuje siatka, zmień więc powyższą opcję na *Snap to Grid*.

Dodatkowo ustaw większą gęstość siatki (mniejsze odległości między jej węzłami). Według mnie łatwiej będzie wtedy projektować formularz, ponieważ krawędzie kontrolki nie będą przyciągane do niepożądanych punktów.

Aby dopasować gęstość siatki, zmień ustawienie *Default Grid Cell Size*. Nadanie ustawieniom *Width* i *Height* mniejszych wartości pozwoli utworzyć gęstszą siatkę, umożliwiającą większą kontrolę nad wielkością i położeniem kontrolki. Duże wartości powodują powstanie rzadszej siatki, dającej mniejszą kontrolę, ponieważ krawędzie kontrolki będą przemieszczały się o większe odcinki. W takiej sytuacji dokładne określenie wielkości i położenia kontrolki nie będzie możliwe. Wykonaj poniższe kroki:

1. Zmień ustawienie *Default Grid Cell Size* na **6; 6**.
2. Zmień opcję *Show Grid* na *True*, jeżeli ustawienie jest inne.
3. Kliknij przycisk *OK*, aby zapisać zmiany i wrócić do projektu formularza. Zauważ, że pojawiły się węzły siatki, jak na rysunku 6.2. Jeżeli węzłów nie widać (jest to błąd w oprogramowaniu Visual Studio), zamknij zakładkę z projektem formularza i w panelu *Solution Explorer* kliknij dwukrotnie element *FormularzOpcji.vb*, aby odświeżyć widok. Jeżeli pojawi się pytanie, czy zapisać zmiany, kliknij przycisk *Yes*.



RYSunEK 6.2. Siatka nie musi być widoczna, aby była aktywna

Przesuń kontrolkę w formularzu lub przeciągnij jej krawędź, aby zmienić jej wielkość. Zwróć uwagę, że dzięki gęstszej siatce masz większą kontrolę nad położeniem kontrolki. Wpisz w opcji *Default Grid Cell Size* większe wartości, na przykład **25; 25**, i sprawdź efekt.

Ubočnym skutkiem ustawienia gęstszej siatki jest pogorszenie czytelności formularza. Musisz zdecydować, które ustawienie jest lepsze. Najczęściej wyłączam siatkę w swoich formularzach i wolę stosować opisaną niżej opcję przyciągania do linii.

Przy okazji

Opcja *Show Grid* określa jedynie, czy siatka jest widoczna, a nie — czy jest aktywna. Aktywność siatki określa opcja *Snap to Grid*.

Przyciąganie do linii

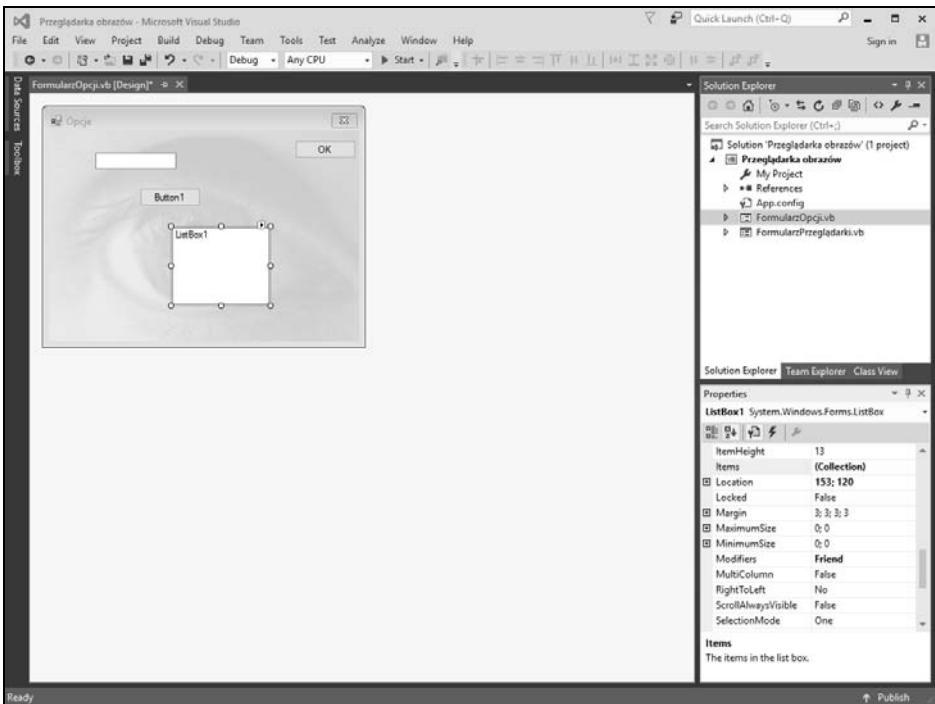
Stosunkowo nową i przydatną funkcją jest przyciąganie do linii (*SnapLines*). Włącz tę funkcję, wykonując następujące kroki:

1. Wybierz polecenie *Tools/Options*, aby otworzyć okno opcji.
2. Kliknij pozycję *Windows Forms Designer*, aby wyświetlić ustawienia siatki.
3. Zmień opcję *Layout Mode* na *SnapLines*.
4. Wyłącz wyświetlanie siatki, ustawiając opcję *ShowGrid* na *False*.
5. Kliknij przycisk *OK*, aby zapisać ustawienia.

Podobnie jak poprzednio, może być konieczne zamknięcie projektu formularza i ponowne jego otwarcie, aby zmiany zostały wprowadzone.

Przyciąganie do linii pozwala szybciej i łatwiej tworzyć interfejsy użytkownika, ponieważ kontrolki są przyciągane do niewidzialnych linii będących przedłużeniami krawędzi innych kontroltek. Najłatwiej poznać tę możliwość, wypróbując ją. Wykonaj następujące kroki:

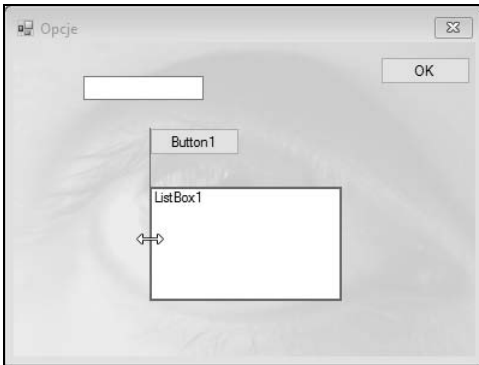
1. Rozmieść kontrolki jak na rysunku 6.3.



RYSUNEK 6.3. Rozmieść kontrolki jak na rysunku

2. Kliknij kontrolkę `ListBox`, aby ją zaznaczyć.
3. Kliknij biały kwadrat na lewej krawędzi kontrolki i przeciągnij go w lewo. W chwili gdy krawędź ta wyrówna się z krawędzią przycisku powyżej listy, pojawi się niebieska linia (widoczna na rysunku 6.4), do której zostanie przyciągnięta krawędź kontrolki.

Gdy będziesz przeciągać krawędź kontrolki w jedną lub drugą stronę, w chwili jej wyrównania z innymi kontrolkami będą pojawiały się kolejne przyciągające linie. Istnieją też poziome linie przyciągające. Oba rodzaje linii funkcjonują również podczas przesuwania kontroltek. Funkcjonalność ta może się wydawać dość błaha, ale wierz mi, jest ona bardzo przydatna i oszczędzi Ci wielu godzin mozolnej pracy.

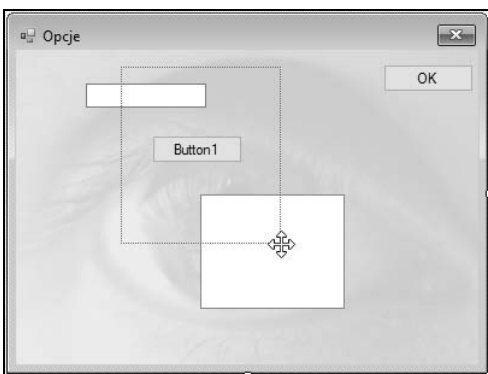


RYSunEK 6.4. Dzięki przyciągnięciu do linii wyrównywanie krawędzi kontroltek jest łatwiejsze

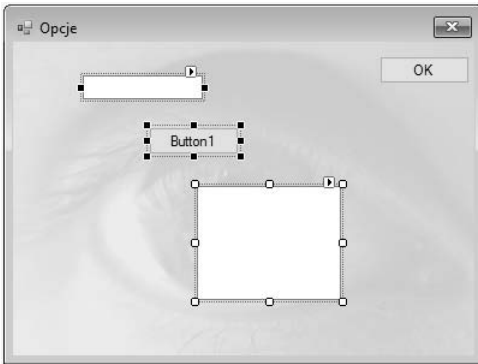
Zaznaczanie grup kontroltek

W miarę doskonalenia swoich umiejętności zauważysz, że Twoje formularze będą się stawać coraz bardziej skomplikowane. W niektórych aplikacjach formularze zawierają dziesiątki, a nawet setki kontroltek. Środowisko Visual Studio oferuje kilka sposobów ułatwiających rozmieszczanie grup kontroltek.

Domyślnie kliknięcie kontrolki powoduje jej zaznaczenie i *jednoczesne* usunięcie zaznaczenia innych kontroltek. Aby wykonać jakąś operację na kilku kontrolkach, musisz je jednocześnie zaznaczyć. Możesz to zrobić na dwa sposoby. Pierwszym jest „zarzucenie pętli”. Aby zaznaczyć w ten sposób grupę kontroltek, kliknij formularz w dowolnym miejscu, a następnie przeciągnij wskaźnik myszy w wybranym kierunku. Podczas przeciągania na formularzu będziesz rysował prostokąt. Gdy zwolnisz przycisk myszy, zostaną zaznaczone wszystkie kontrolki przecięte krawędziami prostokąta. Zwróć uwagę, że aby kontrolka została zaznaczona, nie musi się znaleźć wewnątrz prostokąta — wystarczy, że zostanie przecięta jego krawędzią. Wypróbuj ten sposób. Kliknij formularz w dowolnym miejscu w pobliżu jego lewego górnego wierzchołka i nie zwalniając przycisku myszy, przeciągnij kursor w kierunku prawego dolnego wierzchołka formularza. Przetnij lub zamknij wewnątrz prostokąta wszystkie kontrolki oprócz przycisku *OK*, jak pokazano na rysunku 6.5, a następnie zwolnij przycisk. Kontrolki zostaną zaznaczone (patrz rysunek 6.6).



RYSunEK 6.5. Kliknij formularz i przeciągnij kursor myszy, aby narysować prostokąt



RYSUNEK 6.6. Wszystkie zaznaczone kontrolki będą miały kropkowaną ramkę i uchwyty do zmiany wielkości

Gdy kontrolka zostanie zaznaczona, otrzyma kropkowaną ramkę i kilka uchwytów do zmiany wielkości (kwadratów umieszczonych na środkach krawędzi ramki i w jej narożnikach). Przyjrzyj się dokładnie tym uchwytom. Kontrolka z białymi uchwytami jest aktywną kontrolką w grupie. Narzędzia Visual Studio do modyfikowania grupy kontrolki (na przykład do ich wyrównywania lub formatowania) wykorzystują ustawienia aktywnej kontrolki. Jeżeli na przykład kontrolki na rysunku 6.6 wyrównasz do lewej krawędzi, wówczas właściwość `Left` każdej z nich zostanie ustawiona na wartość właściwości aktywnej kontrolki (tej z białymi uchwytami, w tym przypadku `Listbox`). Stosując technikę z pętlą do zaznaczania kontrolki, nie będziesz miał większego wpływu na wybór aktywnej kontrolki. Jeżeli chcesz wyrównać kontrolki do lewej krawędzi przycisku, musisz zastosować inną technikę zaznaczania kontrolki. Usuń zaznaczenie wszystkich kontrolki, klikając formularz w dowolnym miejscu (nie klikaj jednak żadnej kontrolki).

Przy okazji

Nie każdy uchwyt można przesunąć w dowolnym momencie. Jeżeli na przykład w kontrolce `TextBox` właściwość `Multiline` ustawisz na `False`, wtedy nie będziesz mógł zmienić wysokości kontrolki. Dlatego w takim przypadku kontrolka po zaznaczeniu będzie miała uchwyty tylko na bocznych krawędziach, które będziesz mógł przesunąć.

Inną techniką zaznaczania wielu kontrolki jest użycie klawiszy *Shift* i *Ctrl* podczas klikania (oba klawisze działają tak samo). Ta technika jest bardzo podobna do zaznaczania wielu plików w Eksploratorze plików. Wykonaj poniższe kroki:

1. Kliknij najniższą kontrolkę (`Listbox`), aby ją zaznaczyć. (Kontrolka tylko wtedy jest aktywna, gdy jest zaznaczona).
2. Naciśnij i przytrzymaj klawisz *Shift*, a następnie kliknij kontrolkę `TextBox` w lewej górnej części formularza. Zostaną zaznaczone dwie kontrolki. Aktywną kontrolką będzie pole listy, ponieważ zostało zaznaczone jako pierwsze. Gdy zaznaczysz kilka kontrolki, aktywna będzie ta, która otrzyma białe uchwyty. W ten sposób będziesz mógł ją łatwo zlokalizować.

3. Trzymając cały czas naciśnięty klawisz *Shift*, kliknij nową kontrolkę przycisku (ale nie przycisku *OK*). Dodasz go w ten sposób do grupy zaznaczonych kontrolek. Teraz zaznaczone są wszystkie kontrolki, przy czym aktywną jest pole listy. Możesz zwolnić klawisz *Shift*.

Przy okazji

Kliknięcie zaznaczonej kontrolki przy naciśniętym klawiszu *Shift* powoduje usunięcie jej zaznaczenia.

W razie potrzeby możesz łączyć obie opisane techniki. Na przykład możesz najpierw użyć pętli, aby zaznaczyć wszystkie kontrolki. Jeżeli wśród nich będzie kontrolka, której nie chciałeś zaznaczyć, naciśnij po prostu klawisz *Shift* i kliknij żądaną kontrolkę. W ten sposób usuniesz jej zaznaczenie.

Przy okazji

Jeżeli musisz kliknąć dwa razy tę samą kontrolkę, na przykład aby usunąć jej zaznaczenie, a potem ponownie ją zaznaczyć, zrób to po-wo-li. Jeżeli klikniesz zbyt szybko, Visual Studio potraktuje tę czynność jako podwójne kliknięcie i otworzy kod zdarzenia tej kontrolki.

Wyrównywanie kontrolek

Środowisko Visual Studio zawiera kilka formatujących kontrolek narzędzi, które możesz wykorzystać do tworzenia atrakcyjnych interfejsów użytkownika. Większość narzędzi znajduje się w pasku *Layout* pokazanym na rysunku 6.7. Wyświetl teraz ten pasek, klikając prawym przyciskiem myszy istniejący pasek narzędzi i wybierając z podręcznego menu polecenie *Layout*. Pasek ten zawiera narzędzia do wyrównywania kontrolek w poziomie i w pionie oraz do ich wyródkowywania.

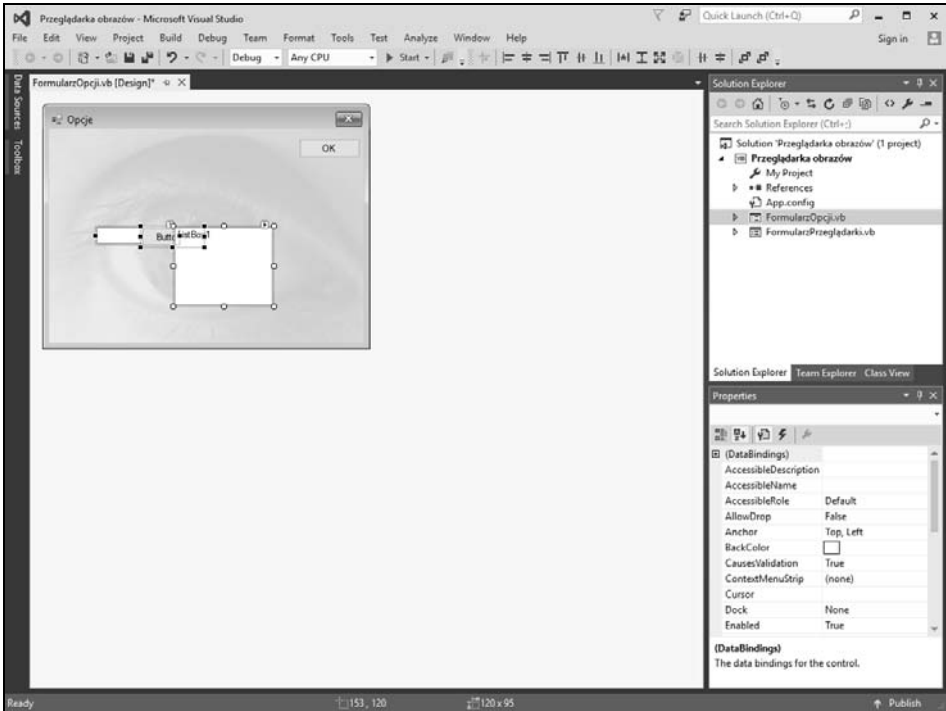


RYSUNEK 6.7. Pasek *Layout* pozwala szybko i łatwo wyrównywać kontrolki

Przesuwaj powoli kursor myszy po ikonach paska i odczytuj pojawiające się dymki. Zwróć uwagę, że za pomocą tego paska możesz wykonać następujące operacje:

- ▶ wyrównać kontrolki do siatki, jeżeli jest włączona;
- ▶ wyrównać zaznaczone kontrolki do lewej, prawej lub środkowej linii;
- ▶ wyrównać zaznaczone kontrolki do górnej, dolnej lub środkowej linii;
- ▶ ustawić taką samą wysokość, szerokość lub oba wymiary zaznaczonych kontrolek;
- ▶ ustawić jednakowe odstępy w pionie lub poziomie pomiędzy zaznaczonymi kontrolkami;
- ▶ przesunąć zaznaczone kontrolki na wierzch lub pod spód innych kontrolek.

Pierwszy przycisk wyrównuje zaznaczoną kontrolkę do siatki — to nic nadzwyczajnego. Jednak pozostałe przyciski są bardzo przydatne. Pamiętaj, że w środowisku Visual Studio aktywna kontrolka stanowi odniesienie w operacjach wyrównywania kontroltek — to jest ważne. Kliknij przycisk *Align Tops* (wyrównaj do góry) i zwróć uwagę, że zaznaczone kontrolki zostały wyrównane względem górnej krawędzi aktywnej kontrolki, co pokazuje rysunek 6.8.



RYSunEK 6.8. Aktywna kontrolka stanowi odniesienie przy wyrównywaniu grupy zaznaczonych kontroltek

Nadawanie kontrolkom tej samej wielkości

Zaznaczone kontrolki można nie tylko wyrównywać, ale również nadawać im takie same wymiary — wysokość, szerokość lub obie wartości. W tym celu trzeba użyć przycisku *Make Same Size* (ustaw tę samą wielkość) w pasku narzędzi. W naszym przykładzie będzie to wielkość (dość spora) listy. Teraz wykonaj następującą operację: w panelu *Properties* we właściwości *Size* wpisz **75; 25** i naciśnij klawisz *Tab*, aby zatwierdzić zmiany. Zwróć uwagę, że wszystkie zaznaczone kontrolki zmieniły wielkość. W ten sposób możesz łatwo i szybko zmieniać wspólne właściwości zaznaczonych kontroltek. Wkrótce opiszę dokładniej tę funkcję.

Ustawianie równych odstępów między kontrolkami

Ale to nie wszystko — jak mówią sprzedawcy. Za pomocą paska *Layout* możesz ustawiać równe odstępy między kontrolkami. Spróbuj teraz: kliknij przycisk *Make Horizontal Spacing Equal* (ustaw równe odstępy w poziomie). Odstępy między kontrolkami będą takie

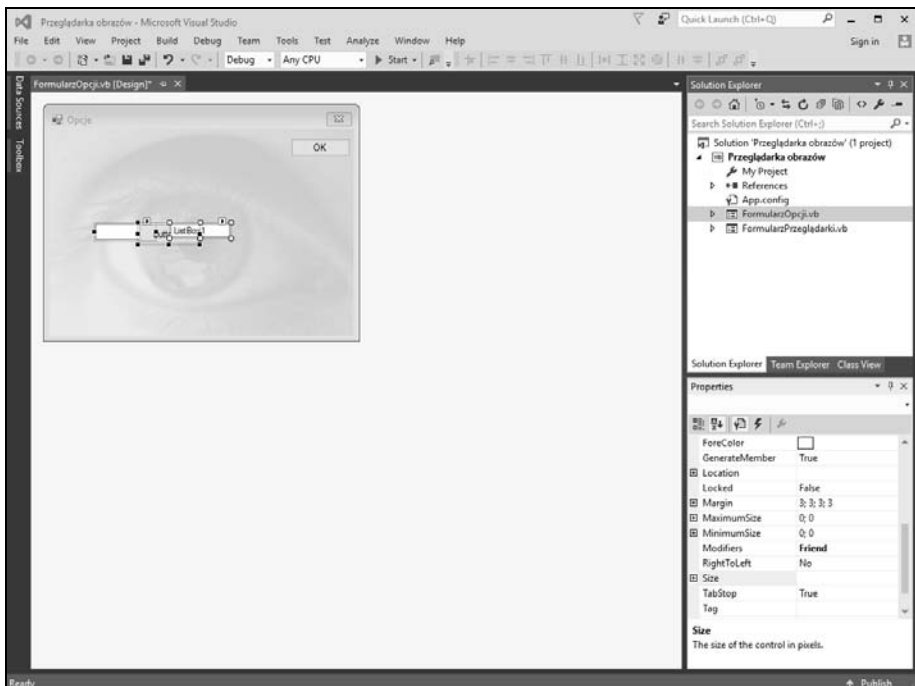
same. W poprzednich wersjach środowiska Visual Studio były dostępne dodatkowe przyciski do zwiększania i zmniejszania odstępów, ale w najnowszej wersji zostały one z jakiegoś powodu usunięte z paska narzędzi. Funkcja ta jest jednak dostępna po wybraniu polecenia *Format/Horizontal Spacing* (format/odstęp w poziomie). W menu *Format* dostępne są polecenia do zwiększania odstępów w pionie i poziomie oraz do ich usuwania. Teraz zapisz projekt, klikając w pasku narzędzi przycisk *Save All*.

Ustawianie właściwości kontroltek w grupie

Poniżej opisano technikę, o której doświadczeni programiści Visual Basic zapominają. W panelu *Properties* można zmieniać właściwości kilku zaznaczonych kontroltek.

Upewnij się, że trzy kontrolki są zaznaczone, a następnie otwórz panel *Properties* (jeżeli nie jest otwarty). Jeżeli zaznaczona jest grupa kontroltek, panel ten wygląda nieco inaczej, tzn. tak, jak widać to na rysunku 6.9:

- ▶ Nie ma właściwości *Name*, ponieważ nie można nadać kilku kontrolkom tej samej nazwy. Dlatego środowisko nie daje możliwości wykonania takiej operacji.
- ▶ Widoczne są tylko właściwości wspólne dla wszystkich kontroltek. Ponieważ zazaczyłeś kontrolki różnych typów, dostępny jest tylko niewielki podzbiór wszystkich właściwości. Jeżeli zaznaczysz kontrolki tego samego typu, widocznych będzie więcej właściwości.
- ▶ Właściwości, których wartości są różne dla zaznaczonych kontroltek (na przykład *Location*), nie mają podanych wartości.



RYСУNEK 6.9. Możesz zmienić właściwości wielu kontroltek jednocześnie, z pewnymi ograniczeniami

Wpisanie wartości powoduje zmianę właściwości każdej kontrolki. Aby sprawdzić, jak to działa, wybierz we właściwości BackColor kolor żółty. Jak się przekonasz, kolor tła wszystkich kontrollek zostanie zmieniony na żółty.

Nie będziesz używać kontrollek dodanych w ciągu tej godziny, więc kliknij pusty obszar poza panelem *Properties*, tak aby nie był bieżącym oknem, a następnie naciśnij klawisz *Delete*, aby usunąć wszystkie kontrolki.

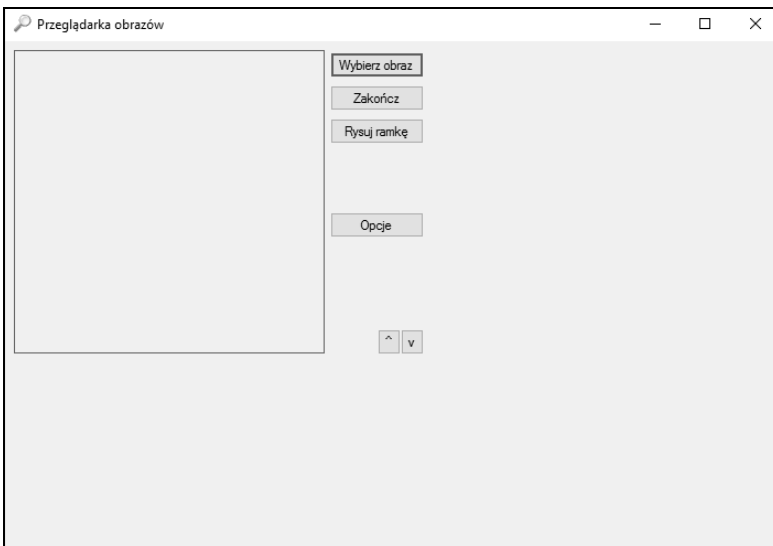
Kotwiczenie i automatyczne powiększanie kontrollek

Moje ulubione funkcje edytora formularzy w środowisku Visual Studio to kotwiczenie kontrollek do krawędzi formularza i dopasowywanie rozmiarów kontrollek podczas zmiany wielkości formularza. W przeszłości, aby osiągnąć ten efekt, trzeba było korzystać z zewnętrznych (zazwyczaj skomplikowanych) komponentów lub pisać własny kod zdarzenia *Resize*. Teraz jest to wbudowana funkcja środowiska Visual Studio 2015.

Domyślnie kontrolki są zadokowane do lewej i górnej krawędzi kontenera. Ale co trzeba zrobić, aby kontrolka zawsze znajdowała się w prawym górnym lub lewym dolnym rogu formularza? Teraz dowiesz się, jak można kotwiczyć kontrolki, aby mogły dopasowywać się do zmienianej wielkości formularza.

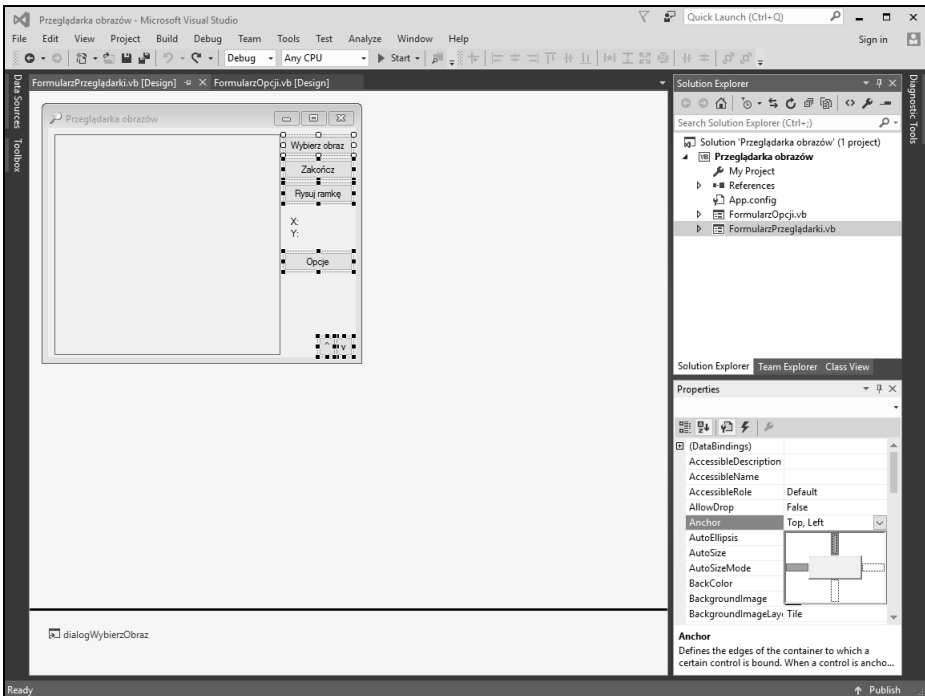
Wykonaj poniższe kroki:

1. W panelu *Solution Explorer* kliknij dwukrotnie element *FormularzPrzeglądarki.vb*. Ten formularz będziesz zmieniał.
2. Naciśnij klawisz *F5*, aby uruchomić projekt.
3. Przeciągnij prawy dolny wierzchołek formularza, aby go powiększyć. Zwróć uwagę, że kontrolki nie przemieszczają się razem z narożnikiem (patrz rysunek 6.10).



RYСУNEK 6.10. Kontrolki są domyślnie zakotwiczone przy lewym górnym wierzchołku formularza

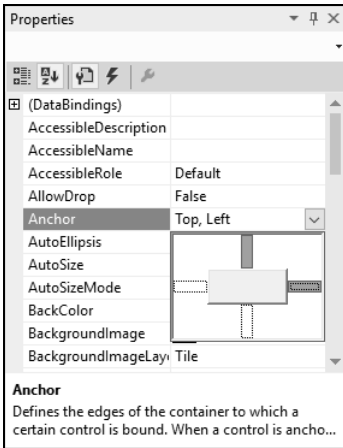
4. W pasku narzędzi kliknij przycisk *Stop Debugging*, aby zatrzymać program.
5. Kliknij przycisk *Wybierz obraz*, aby go zaznaczyć i — co najważniejsze — usunąć zaznaczenie formularza.
6. Naciśnij i przytrzymaj klawisz *Shift*, po czym kliknij kolejno przyciski *Zakończ*, *Rysuj ramkę*, *Opcje*, \wedge oraz *v*.
7. W panelu *Properties* kliknij właściwość *Anchor* (kotwica), a następnie strzałkę w dół, która się pojawi w polu wartości. Zobaczysz unikatową rozwijaną opcję, przedstawioną na rysunku 6.11.



RYSunEK 6.11. Do ustawiania właściwości *Anchor* służy unikatowa rozwijana opcja

Szary prostokąt w środku okienka reprezentuje zaznaczoną kontrolkę (kontrolki). Mniejsze prostokąty: u góry, u dołu, po lewej i po prawej stronie reprezentują możliwe zakotwiczenia do krawędzi formularza. Wypełniony prostokąt oznacza, że kontrolka jest zakotwiczona do odpowiedniej krawędzi. Aby sprawdzić, jak funkcjonuje właściwość *Anchor*, wykonaj poniższe kroki:

1. Kliknij prostokąt po lewej stronie, aby usunąć jego wypełnienie, a następnie kliknij prostokąt po prawej stronie, aby go wypełnić (patrz rysunek 6.12).
2. Kliknij dowolną inną właściwość, aby zamknąć rozwijaną opcję. Właściwość *Anchor* otrzyma wartość *Top, Right*.
3. Naciśnij klawisz *F5*, aby uruchomić projekt, a następnie przeciągnij prawy dolny wierzchołek formularza, aby go powiększyć.



RYSUNEK 6.12. Taki układ powoduje zakotwiczenie kontrolki do górnej i prawej krawędzi formularza

Ciekawe, prawda? Przyciski zostały zakotwiczone do prawej krawędzi formularza, jak na rysunku 6.13. W rzeczywistości zakotwiczenie oznacza utrzymywanie stałego odstępu krawędzi kontrolki od krawędzi formularza. Jest to *niezwykle przydatne* podczas tworzenia interfejsów użytkownika.



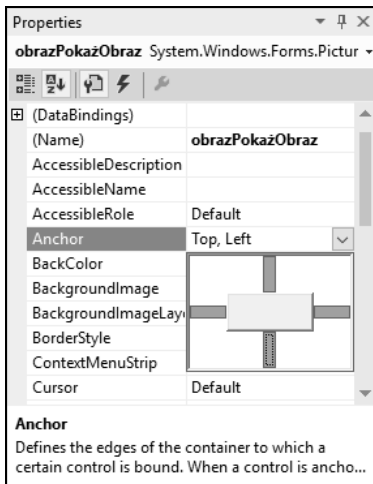
RYSUNEK 6.13. Kotwiczenie kontrolki jest niezwykle przydatną funkcjonalnością umożliwiającą tworzenie elastycznych formularzy

Zwróć uwagę, że podczas zmiany wielkości formularza ramka obrazu i etykiety ze współzrędnymi pozostają w dotychczasowym położeniu. To żaden problem, można go rozwiązać również za pomocą właściwości `Anchor`. Najpierw zmień zakotwiczenie etykiety, wykonując poniższe kroki (wcześniej zamknij uruchomiony program, jeżeli tego już nie zrobiłeś):

1. Kliknij etykietę *X*, aby ją zaznaczyć.
2. Naciśnij i przytrzymaj klawisz *Shift*, a następnie zaznacz kliknięciem etykietę *Y*.
3. Ustaw właściwość `Anchor` tak samo jak w przypadku przycisków — usuń wypełnienie lewego prostokąta i wypełnij prawy prostokąt (patrz rysunek 6.12).
4. Kliknij dowolną inną właściwość, aby zamknąć rozwijaną opcję.

Jednak ramka obrazu nieco różni się od pozostałych kontroltek. Jej górna i lewa krawędź powinny być zakotwiczone jak teraz, ale prawa i dolna powinny się przemieszczać razem z krawędziami formularza. Ten efekt jest łatwy do osiągnięcia. Wykonaj poniższe kroki:

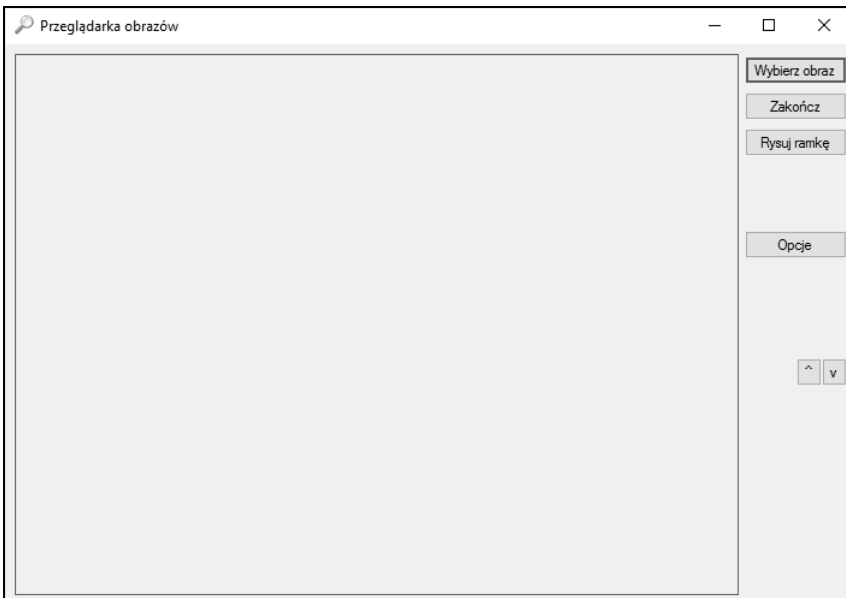
1. Kliknij ramkę obrazu, aby ją zaznaczyć.
2. Kliknij właściwość `Anchor` i wypełnij wszystkie cztery prostokąty, jak pokazano to na rysunku 6.14.



RYСУNEK 6.14. Takie ustawienie powoduje zakotwiczenie kontrolki do wszystkich czterech krawędzi formularza

Teraz naciśnij klawisz *F5*, aby uruchomić projekt, a następnie przeciągnij prawy dolny wierzchołek formularza, aby go powiększyć. Zwróć uwagę, że tym razem ramka zmienia swoje wymiary i dostosowuje się do wielkości formularza (patrz rysunek 6.15). Ta cecha przyda się podczas przeglądania większych obrazów.

Teraz, gdy wiesz, jak używać właściwości `Anchor`, możesz tworzyć elastyczne formularze bez konieczności kodowania. Jest tylko jedna pułapka: w zależności od wartości tej właściwości kontrolka może zupełnie zniknąć, jeżeli formularz zostanie bardzo zmniejszony.



RYSUNEK 6.15. Odpowiednie użycie właściwości Anchor pozwala tworzyć elastyczne formularze

Ustalanie kolejności wyróżniania kontroltek

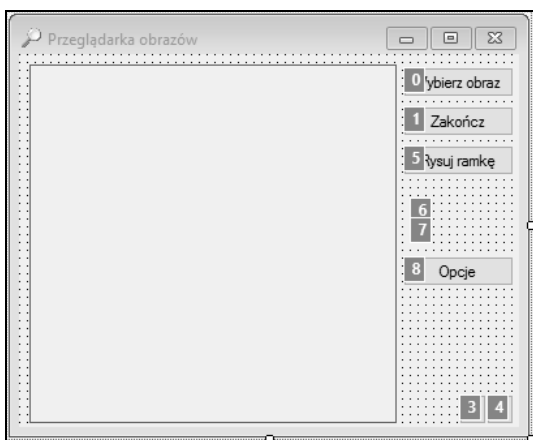
Kolejność wyróżniania kontroltek jest *często* zaniedbywana nawet przez doświadczonych programistów Visual Basic. Prawdopodobnie jako użytkownik korzystałeś z tej możliwości funkcjonalnej być może nieświadomie. Gdy podczas korzystania z formularza naciśniesz klawisz *Tab*, wyróżniana jest następna w kolejności kontrolka. W ten sposób można łatwo poruszać się po formularzu za pomocą klawiatury. Kolejność wyróżniania kontroltek jest określana za pomocą właściwości *TabIndex*. Po otwarciu formularza jest wyróżniana ta kontrolka, której powyższa właściwość ma wartość 0. Po naciśnięciu klawisza *Tab* wyróżniana jest kontrolka, której właściwość *TabIndex* ma wartość 1 itd. Gdy dodasz nową kontrolkę do formularza, środowisko Visual Studio przypisze jej właściwości *TabIndex* kolejną wartość (czyli nowa kontrolka będzie wyróżniana na końcu). Wartość właściwości *TabIndex* dla każdej kontrolki jest unikatowa, a kontrolki wyróżniane są zgodnie z rosnącymi wartościami ich właściwości.

Jeżeli kolejność wyróżniania kontroltek jest niewłaściwie ustawiona, wówczas po naciśnięciach klawisza *Tab* kontrolki wyróżniane są w przypadkowej kolejności, co może być frustrujące dla użytkownika. W przeszłości jedynym sposobem rozwiązania tego problemu było ręczne skorygowanie wartości właściwości *TabIndex* w panelu *Properties*. Przykładowo, aby umieścić kontrolkę na początku kolejki, trzeba było ustawić jej właściwość na 0, jednak wtedy środowisko Visual Studio samodzielnie zmieniało właściwości wszystkich pozostałych kontroltek. To był bardzo uciążliwy proces, wierz mi. Choć wygodna jest możliwość ręcznego ustawiania właściwości *TabIndex*, na przykład w celu umieszczenia kontrolki w środku kolejki, to o wiele lepszym sposobem jest ustawianie kolejności za pomocą myszy.

Naciśnij klawisz *F5*, aby uruchomić projekt (jeżeli go wcześniej zatrzymałeś), i zwróć uwagę, że wyróżniony jest przycisk *Wybierz obraz* (ma niebieską ramkę). Jeżeli teraz naciśniesz klawisz *Enter*, „klikniesz” przycisk, ponieważ jest on wyróżniony. Naciśnij klawisz *Tab*. Wyróżniony zostanie przycisk *Zakończ*, ponieważ ten przycisk dodałeś do formularza zaraz po przycisku *Wybierz obraz*. Naciśnij ponownie klawisz *Tab*. Na pewno spodziewałeś się, że wyróżniony zostanie przycisk *Rysuj ramkę*. Użytkownik również by tego właśnie oczekiwał. Ale zamiast tego wyróżniony został przycisk *^*, ponieważ był on kolejną kontrolką, którą dodałeś do formularza. Musisz poprawić kolejność kontroltek, więc zatrzymaj program, klikając w pasku narzędzi przycisk *Stop Debugging* lub zamykając okno programu.

Aby ustalić kolejność wyróżniania kontroltek, korzystając z edytora Visual Studio, wykonaj poniższe kroki:

1. Kliknij formularz, aby go zaznaczyć, a następnie wybierz polecenie *View/Tab Order* (widok/kolejność wyróżniania). Zwróć uwagę, że na kontrolkach pojawiły się liczby, jak na rysunku 6.16. Oznaczają one wartości właściwości *TabIndex*. Teraz widać wyraźnie, że kolejność kontroltek jest niewłaściwa.



RYSUNEK 6.16. Liczby na kontrolkach oznaczają wartości właściwości *TabIndex*

2. Kliknij przycisk *Wybierz obraz*. Tło liczby zmieni się z niebieskiego na białe, aby zasygnalizować, że zaznaczyłeś kontrolkę. Gdyby wartość właściwości *TabIndex* tej kontrolki była inna niż 0, to po kliknięciu taka wartość zostałaby jej nadana.
3. Kliknij przycisk *Zakończ*, aby wskazać, że ta kontrolka ma być wyróżniana jako następna.
4. Teraz przycisk *Rysuj ramkę* ma numer 5. Kliknij przycisk, a jego numer zmieni się na 2.
5. Kliknij pozostałe kontrolki w następującej kolejności: etykieta *X*, etykieta *Y*, przycisk *Opcje*, przycisk *^*, przycisk *v*.
6. Gdy klikniesz ostatni przycisk, tło każdej liczby stanie się z powrotem niebieskie. Zostanie w ten sposób ustalona kolejność wyróżniania kontroltek. Wybierz ponownie polecenie *View/Tab Order*, aby wyłączyć tryb ustalania kolejności.

7. Ponownie naciśnij klawisz *F5*, aby uruchomić projekt. Przekonasz się, że teraz, podczas naciskania klawisza *Tab*, kontrolki będą wyróżniane w logicznej kolejności.

Przy okazji

Możesz wyróżniać kontrolki w zadanej kolejności za pomocą kodu. W tym celu użyj metody `SelectNextControl()` kontrolki lub formularza.

Aby kontrolka nie była wyróżniana, ustaw jej właściwość `TabStop` na `False`. Użytkownik będzie mógł ją wyróżnić za pomocą myszy, ale nie za pomocą klawisza *Tab*. Jej właściwość `TabIndex` może zawierać poprawną wartość, dzięki czemu po wyróżnieniu kontrolki (na przykład poprzez jej kliknięcie), a następnie po naciśnięciu klawisza *Tab* wyróżniona zostanie kolejna właściwa kontrolka.

Zanim przejdziesz dalej, zatrzymaj program.

Warstwy kontrolek

Kolejność wyróżniania i wizualne rozmieszczenie kontrolek decydują o efektywności formularza. Jednak powyższe dwa aspekty dotyczą rozmieszczenia kontrolek na płaszczyźnie, tj. w kierunkach osi *x* i *y*. Czasami (choć zdarza się to dość rzadko) kontrolki muszą się nakładać. W takim przypadku na wierzchu znajduje się kontrolka, która została dodana do formularza jako ostatnia. Kolejność warstw kontrolek możesz zmieniać za pomocą przycisków *Bring to Front* (przenieś na wierzch) lub *Send to Back* (przenieś na spód) w pasku narzędzi *Layout*.

Czy wiedziałeś?

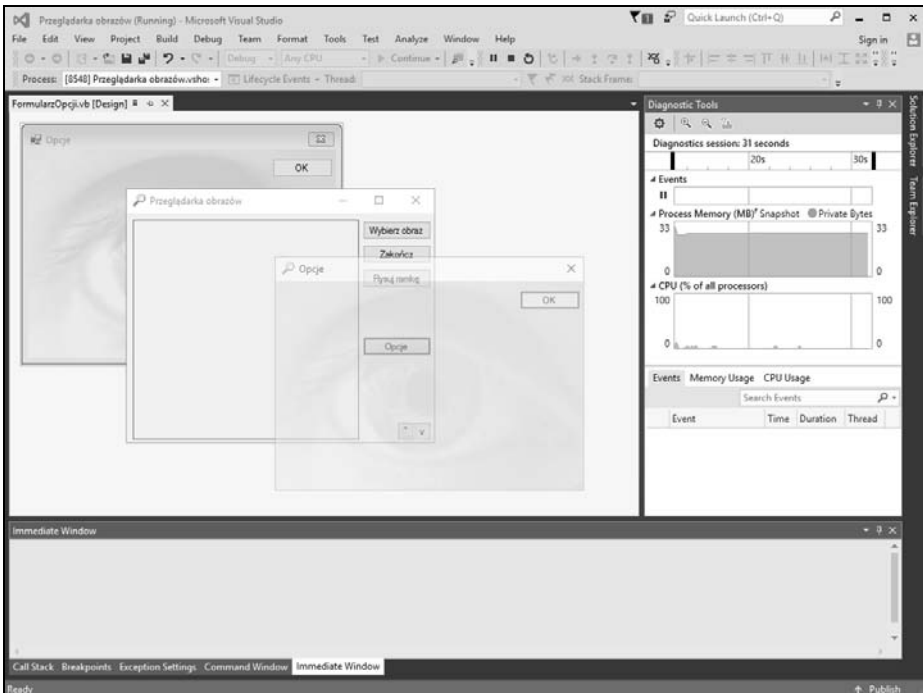
Kontrolki możesz przenosić na wierzch lub na spód za pomocą kodu. Służą do tego metody kontrolki `BringToFront()` lub `SendToBack()`.

Tworzenie niemodalnych okien widocznych zawsze na pierwszym planie

Jak już wiesz, okno po kliknięciu jest zazwyczaj przenoszone na pierwszy plan, a wszystkie pozostałe okna umieszczane są za nim (o ile nie jest to okno modalne). Czasami istnieje potrzeba utworzenia okna, które zawsze będzie widoczne na pierwszym planie, niezależnie od tego, czy będzie ono aktywne (tj. wyróżnione), czy nie. Przykładem jest formularz *Find* (znajdź) w środowisku Visual Studio i innych aplikacjach, na przykład Word. Niezależnie od tego, które inne okno jest aktywne, formularz *Find* zawsze znajduje się na wierzchu ponad innymi oknami. Takie okno możesz utworzyć, ustawiając jego właściwość `TopMost` na `True`. Nie jest to nic nadzwyczajnego, ale warto wiedzieć: zwykła zmiana właściwości lub wywołanie jednej metody często umożliwia uzyskanie efektu na pierwszy rzut oka trudnego do uzyskania.

Tworzenie przezroczystych formularzy

Bardzo ciekawa, moim zdaniem, jest stosunkowo nowa właściwość formularza `Opacity`. Służy ona do określania stopnia przezroczystości formularza i wszystkich jego kontroltek. Domyślnie właściwość ta ma wartość 100% oznaczającą, że formularz (i jego kontrolki) są całkowicie nieprzezroczyste. Wartość 0% powoduje, że formularz jest zupełnie przezroczysty (jest jednak wtedy nieprzydatny). Natomiast wartość 50% oznacza, że formularz jest półprzezroczysty, co pokazano na rysunku 6.17. Dobrym przykładem jest program Microsoft Outlook 2003 lub nowszy, w którym przezroczystość zastosowano w powiadomieniach o odebranej wiadomości. Przezroczystość powiadomień jest zmieniana od 0% do 100%, utrzymywana przez krótki czas na górnym poziomie, a następnie zmniejszana z powrotem do 0%, co powoduje zniknięcie powiadomienia. Ten efekt możesz uzyskać za pomocą prostej pętli opisanej w rozdziale 14. „Efektywne pętle”.



RYSUNEK 6.17. Formularze — duchy!

Tworzenie przewijanych formularzy

Formularz przewijany posiada paski przewijania, jeżeli zawartość nie mieści się w jego oknie. Przewijanie jest doskonałą funkcją, a do tego prostą do zaimplementowania we własnych aplikacjach.

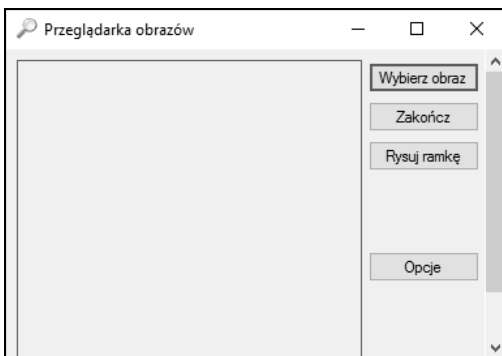
O możliwości przewijania formularza decydują trzy właściwości:

Właściwość	Opis
AutoScroll	Właściwość określająca, czy paski przewijania są widoczne w formularzu.
AutoScrollMinSize	Minimalna wielkość przewijanego obszaru. Jeżeli formularz zostanie zmniejszony tak, że jego wewnętrzne wymiary (bez ramek i paska tytułu) będą mniejsze niż ta właściwość, wówczas pojawią się paski przewijania.
AutoScrollMargin	Właściwość określająca dodatkowy margines wokół każdej kontrolki. Określa ona odległość między zewnętrzną kontrolką a krawędzią formularza, przy której pojawiają się paski przewijania formularza.

Naciśnij klawisz *F5*, aby uruchomić projekt, a następnie zmniejsz formularz, przeciągając jego prawy dolny wierzchołek w kierunku lewego górnego wierzchołka. Zwróć uwagę, że choć kontrolki będą dopasowywać się w możliwie najlepszy sposób, to gdy formularz będzie bardzo mały, wtedy niektóre kontrolki znikną. Jedynym sposobem uzyskania dostępu do nich będzie powiększenie formularza — chyba że będzie to formularz przewijany.

Wykonaj następujące kroki:

1. Jeżeli program działa, zatrzymaj go.
2. W formularzu *FormularzPrzeглядarki.vb* ustaw właściwość *AutoScroll* na *True*.
3. Naciśnij klawisz *F5*, aby uruchomić projekt.
4. Zmniejsz formularz, przeciągając jego prawy dolny wierzchołek w kierunku lewego górnego wierzchołka. Zwróć uwagę, że w trakcie tej czynności po prawej stronie formularza pojawi się pasek przewijania, jak na rysunku 6.18. Za jego pomocą możesz przewinąć zawartość formularza i uzyskać dostęp do kontroltek, które wcześniej były niedostępne.



RYСУNEK 6.18. Bez pasków przewijania niektóre kontrolki mogą być niewidoczne

Zatrzymaj program i zapisz pracę.

Tworzenie interfejsu MDI

Wszystkie utworzone przez Ciebie do tej pory programy były projektami **jednodokumentowymi** (ang. *Single-Document Interface*, SDI). W tego typu programach wszystkie formularze są sobie równoważne i nie tworzą żadnej hierarchii. Jednak w języku Visual Basic można tworzyć projekty **wielodokumentowe** (ang. *Multiple-Document Interface*, MDI). Projekt taki zawiera okno rodzicielskie (zwane kontenerem) i jedno lub kilka okien potomnych. Wewnątrz okna rodzicielskiego można otwierać dowolną liczbę dokumentów potomnych, każdy w osobnym oknie. Wszystkie okna pochodne są obsługiwane za pomocą tego samego menu i paska narzędzi, znajdujących się w oknie rodzicielskim. Okna potomne mogą znajdować się tylko wewnątrz okna rodzicielskiego.

Przy okazji

Oprócz okien potomnych aplikacje MDI mogą zawierać dowolną liczbę zwykłych okien (na przykład dialogowych).

Teraz utwórz prosty projekt MDI. W tym celu wykonaj poniższe kroki:

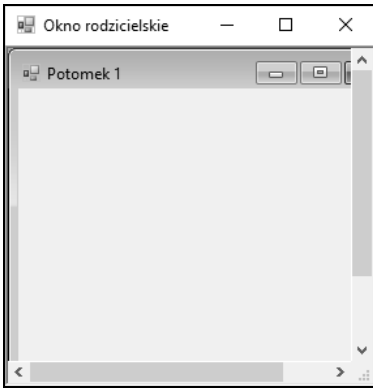
1. Jeżeli Twój program cały czas działa, zamknij go.
2. Wybierz polecenie *File/New/Project*. Wyświetli się okno *New Project* (zwróć uwagę, że jest to formularz modalny). Jeżeli pojawi się pytanie, czy zapisać zmiany w projekcie *Przeglądarka obrazów*, kliknij przycisk *Yes*.
3. Wpisz w polu *Name* nazwę **Przykład MDI** i kliknij przycisk *OK*, aby utworzyć projekt.
4. W panelu *Solution Explorer* kliknij prawym przyciskiem element *Form1.vb* i z podręcznego menu wybierz polecenie *Rename*. Zmień nazwę formularza na **FormularzRodzicielskiMDI.vb**. Następnie zmień właściwość *Text* na **Okno rodzicielskie**, a właściwość *IsMdiContainer* na **True** (jeżeli nie zmienisz tej ostatniej właściwości, nie utworzysz programu MDI). Pierwszą zmianą, jaką zauważysz, będzie wklęsłe, szare tło formularza. Jest to standardowy wygląd rodzicielskiego okna MDI. Wewnątrz niego będą znajdowały się wszystkie okna potomne.
5. Kliknij polecenie *Project/Add Windows Form*, aby dodać nowy formularz. Nadaj mu nazwę **Potomek1.vb** i ustaw jego właściwość *Text* na **Potomek 1**.
6. Dodaj w ten sam sposób jeszcze jeden formularz. Nadaj mu nazwę **Potomek2.vb** i ustaw jego właściwość *Text* na **Potomek 2**.
7. W pasku narzędzi kliknij przycisk *Save All*.
8. W panelu *Solution Explorer* kliknij dwukrotnie element *FormularzRodzicielskiMDI.vb*, aby otworzyć projekt okna rodzicielskiego.
9. Kliknij dwukrotnie formularz, aby otworzyć kod jego domyślnego zdarzenia *Load*. Wpisz poniższe instrukcje:

```
Potomek1.MdiParent = Me  
Potomek1.Show()
```

Na pewno wiesz już, co robi druga instrukcja: otwiera formularz niemodalny. Jednak nas interesuje pierwsza instrukcja, która wpisuje do właściwości *MdiParent* referencję do

bieżącego formularza, czyli formularza rodzicielskiego, ponieważ jego właściwość `IsMdiContainer` ustawiłeś na `True`. Nowy formularz po otwarciu będzie wyświetlany jako okno potomne.

Naciśnij klawisz `F5`, aby uruchomić projekt. Zwróć uwagę, że formularz potomny pojawił się wewnątrz formularza rodzicielskiego. Jeżeli zmienisz wielkość formularza rodzicielskiego tak, że okna potomne nie będą w pełni widoczne, to pojawią się paski przewijania (patrz rysunek 6.19). Jeżeli usuniesz instrukcję ustawiającą właściwość `MdiParent`, wówczas nowy formularz (który nie będzie już oknem potomnym) zostanie po prostu wyświetlony przed formularzem rodzicielskim i nie będzie go ograniczać jego ramka.



RYSUNEK 6.19. Formularz potomny wyświetlany jest wewnątrz formularza rodzicielskiego

Zatrzymaj program, klikając przycisk *Stop Debugging* w pasku narzędzi, i wykonaj poniższe kroki:

1. W panelu *Solution Explorer* kliknij dwukrotnie element *Potomek1.vb*, aby otworzyć projekt formularza.
2. Dodaj do formularza przycisk i ustaw jego właściwości jak niżej:

Właściwość	Wartość
Name	przyciskPokażPotomka2
Location	105; 100
Size	85; 23
Text	Potomek 2

3. Kliknij dwukrotnie przycisk, aby otworzyć kod jego zdarzenia `Click`, i wpisz poniższe instrukcje:

```
Potomek2.MdiParent = Me.MdiParent
Potomek2.Show()
```

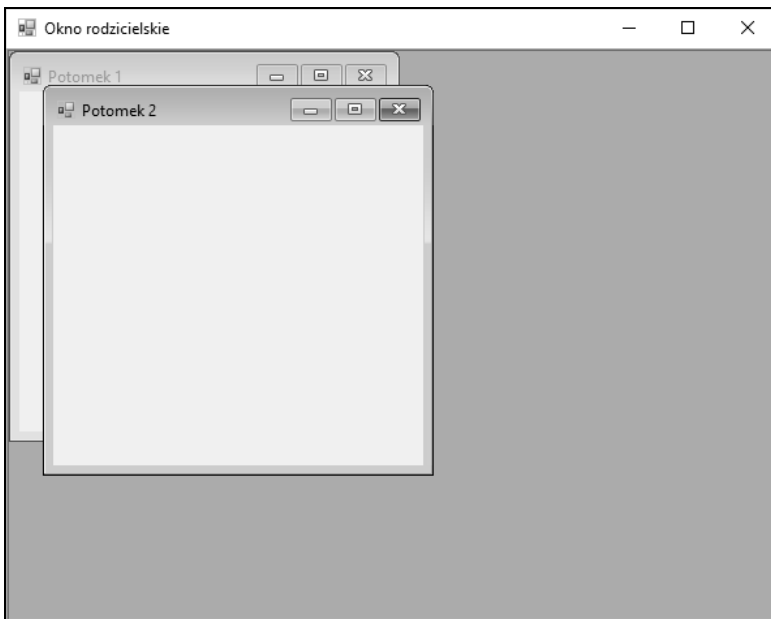
Powyższy kod powoduje wyświetlenie drugiego formularza. Zwróć uwagę, że kod ten różni się od wcześniejszego. Właściwość `MdiParent` drugiego okna potomnego nie możesz ustawić na `Me`, ponieważ słowo to jest referencją do bieżącego formularza

(o nazwie *Potomek1*, który nie jest kontenerem MDI). Jednak obiekt `Me.MdiParent` jest referencją do formularza rodzicielskiego, ponieważ tak właśnie ustawiłeś tę właściwość w tym formularzu. Możesz więc po prostu umieścić referencję do formularza rodzicielskiego we właściwości drugiego formularza potomnego. W ten sposób oba formularze będą miały tego samego rodzica.

Przy okazji

Formularzem potomnym może być każdy formularz (z wyjątkiem oczywiście formularza rodzicielskiego). Aby utworzyć formularz potomny, wpisz w jego właściwość `MdiParent` referencję do formularza zdefiniowanego jako kontener MDI.

4. Naciśnij klawisz *F5*, aby uruchomić projekt, a następnie powiększ formularz. W formularzu potomnym będzie widoczny przycisk (jeżeli go nie widać, prawdopodobnie przez pomyłkę dodałeś go do drugiego formularza potomnego). Gdy klikniesz przycisk, pojawi się drugi formularz potomny, jak na rysunku 6.20. Zwróć uwagę, że nowy formularz również jest umieszczony wewnątrz formularza rodzicielskiego.



RYSUNEK 6.20. Formularze potomne są równorzędne

Czy wiedziałeś?

Formularz rodzicielski MDI ma właściwość `ActiveMdiChild`, którą możesz wykorzystać jako referencję do bieżącego aktywnego formularza potomnego.

Przy okazji

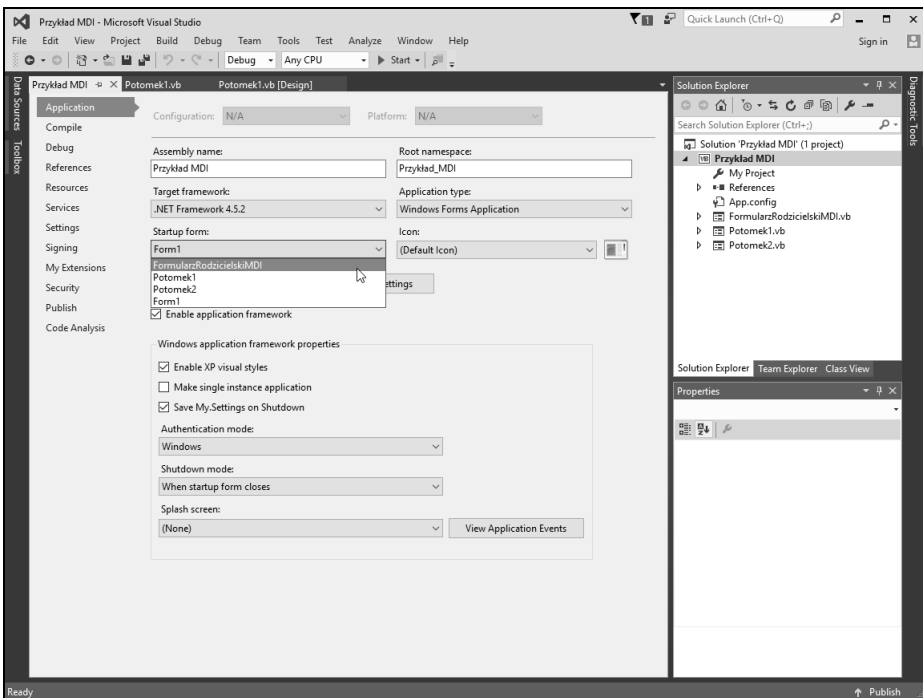
Aby po uruchomieniu projektu formularz rodzicielski był większy, ustaw jego właściwości `Size.Height` i `Size.Width` ręcznie w projekcie albo po uruchomieniu za pomocą kodu w zdarzeniu `Load`.

Musisz jeszcze wiedzieć, że można utworzyć dowolną liczbę instancji tego samego formularza. Jednak zarządzanie wieloma instrukcjami tego samego formularza jest dość trudne i wykracza poza zakres tej książki.

Ustawianie formularza startowego

W projekcie typu *Windows Forms Application* właściwość *Startup form* zawiera referencję do pierwszego formularza, automatycznie tworzonego przez środowisko Visual Studio podczas tworzenia nowego projektu.

W każdym projekcie musi być zdefiniowany formularz startowy, stanowiący punkt wejścia do programu. Możesz zmienić formularz startowy, klikając w panelu *Solution Explorer* prawym przyciskiem myszy nazwę projektu i wybierając z podręcznego menu polecenie *Properties*. Na pierwszej stronie panelu konfiguracyjnego, który się pojawi, będzie widoczna właściwość *Startup form*, jak na rysunku 6.21.



RYSUNEK 6.21. Punkt wejścia do aplikacji zdefiniowany jest za pomocą właściwości *Startup form*

Jeżeli właściwość *Startup form* będzie ustawiona na formularz potomny, wówczas program po uruchomieniu nie będzie działał zgodnie z oczekiwaniami. Wprawdzie pojawi się wskazany formularz, ale nie będzie on formularzem potomnym, ponieważ nie zostanie wykonany kod wpisujący w jego właściwości *MdiParent* referencję do formularza rodzicielskiego MDI.

Jeżeli formularze MDI wydają Ci się skomplikowane, nie przejmuj się. Większość aplikacji, które będziesz tworzył jako początkujący programista Visual Basic, będzie aplikacjami typu SDI. Gdy nabędziesz nieco doświadczenia, zaczniesz eksperymentować z projektami MDI. Pamiętaj, że nie musisz tworzyć programu MDI tylko dlatego, że potrafisz to robić. Twórz go tylko wtedy, gdy wymaga tego charakter projektu.

Podsumowanie

Gruntowna wiedza o formularzach ma niezwykle istotne znaczenie, ponieważ formularze stanowią dynamiczne podstawy, na których tworzy się interfejsy użytkownika. Jeżeli nie będziesz wiedział, jak tworzyć poprawnie funkcjonujące formularze, cała aplikacja będzie wadliwa. Wiele operacji wykonywanych na formularzach wykracza daleko poza proste ustawianie właściwości, szczególnie gdy, tworząc formularze, trzeba przyjąć punkt widzenia użytkownika. W miarę nabywania doświadczenia będziesz zagłębiał się w zawłości formularzy i operacje na nich będziesz wykonywał odruchowo.

W ciągu tej godziny nauczyłeś się wykonywać kilka interesujących czynności, na przykład tworzyć przezroczyste formularze, a oprócz tego poznałeś zaawansowane techniki, takie jak tworzenie aplikacji MDI. Dowiedziałeś się, jak definiować przewijane formularze — o tym elemencie interfejsu użytkownika trzeba zawsze pamiętać. Sporo czasu poświęciłeś na pracę z kontrolkami, które są ważnymi elementami interfejsu, ponieważ podstawową funkcją formularza jest udostępnianie kontrolki. W ciągu następnych dwóch godzin poznasz szczegóły dotyczące zaawansowanych kontrolki dostępnych w języku Visual Basic, które staną się ważnym elementem Twojego programistycznego arsenału.

Pytania i odpowiedzi

P: Czy powinienem się zajmować kotwiczeniem i przewijaniem kontrolki w każdym stworzonym formularzu?

O: Zdecydowanie nie. W większości aplikacji formularze są oknami dialogowymi, czyli formularzami modalnymi przeznaczonymi do wprowadzania danych przez użytkownika. Okno dialogowe ma zazwyczaj stałą wielkość. Styl ramki takiego okna nie pozwala na zmianę jego wielkości. W formularzu o stałej wielkości nie trzeba się zajmować kotwiczeniem kontrolki ani ich przewijaniem.

P: Jak mogę ocenić, czy projekt powinien mieć interfejs MDI?

O: Jeżeli w programie będzie otwieranych wiele instancji formularza tego samego typu, można w nim zastosować interfejs MDI. Przykładowo, jeżeli stworzysz program do edycji obrazów i użytkownik musi mieć możliwość pracy z wieloma obrazami jednocześnie, warto zastosować interfejs MDI. Ponadto możesz rozważyć utworzenie projektu MDI, jeżeli wykorzystujesz wiele formularzy obsługiwanych za pomocą wspólnego menu i paska narzędzi.

Warsztat

Quiz

1. Prawda czy fałsz? Pierwsza kontrolka w zaznaczonej grupie jest kontrolką aktywną.
2. Na ile sposobów można dodać kontrolkę do formularza?
3. W którym miejscu formularza zostanie umieszczona kontrolka, jeżeli klikniesz ją dwukrotnie w panelu *Toolbox*?
4. Jaka właściwość powoduje zakotwiczenie krawędzi kontrolki przy krawędzi formularza?
5. Co trzeba zrobić, aby wyświetlić lub ukryć siatkę formularza?
6. Który pasek narzędzi zawiera funkcje do rozmieszczania kontrolki?
7. Jaka właściwość musisz ustawić, aby zdefiniować formularz rodzicielski MDI?

Odpowiedzi

1. Prawda.
2. Są trzy podstawowe sposoby: dwukrotne kliknięcie kontrolki w panelu *Toolbox*, przeciągnięcie kontrolki do formularza oraz kliknięcie kontrolki i narysowanie jej w formularzu. Dostaniesz dodatkowe punkty, jeżeli znasz czwartą metodę (skopiowanie i wklejenie istniejącej kontrolki).
3. Kontrolka jest umieszczana na istniejącej kontrolce albo w lewym górnym rogu formularza, jeżeli nie ma jeszcze żadnej kontrolki.
4. Właściwość *Anchor*.
5. Należy zmienić opcję *Show Grid* w oknie dialogowym *Options*.
6. Pasek *Layout*.
7. Aby zdefiniować formularz rodzicielski MDI, należy jego właściwość *IsMdiContainer* ustawić na *True*.

Ćwiczenia

1. Utwórz nową aplikację typu *Windows Forms Application* i dodaj przycisk w środku formularza. Poeksperymentuj z właściwością *Anchor*, uruchamiając projekt z różnymi jej wartościami.
2. Zmień utworzony w tej godzinie projekt *Przykład MDI* tak, aby najpierw pojawiał się formularz potomny *Potomek 2*, a następnie *Potomek 1*.

Skorowidz

A

- akcelerator, 203
- aktualizacja nazw procedur zdarzeń, 104
- aktywna
 - kontrolka, 138
 - ścieżka, 40
- aplikacje typu Windows Forms Application, 22
- atrybuty, 71, 72
 - pliku, 412
- automatyczne
 - powiększanie kontroltek, 140
 - ukrywanie paneli, 50
 - uzupełnianie, 76
- automatyzacja, 509

B

- baza danych, 441
 - nawiązywanie połączenia, 443
 - zamykanie połączenia, 448
- biblioteka, 87
 - DLL, 24
 - obiektów, 510
 - typów, 510
- blok With, 403
- błąd
 - kompilacji, 323
 - wykonania, 323

C

- ciąg o zerowej długości, 328
- ciągi znaków, 278
- CLR, Common Language Runtime, 544, 562
- cyfry, 525

- czas życia obiektu, 358
- czasomierze, 181
- czyszczenie listy, 174

D

- dane, 413, 399
- data i czas, 412
- data i godzina, 283
- definiowanie
 - filtrów plików, 404
 - ikony formularza, 28
 - kolumn, 190
 - kontroltek, 129
 - menu kontekstowego, 210
 - poła do wprowadzania hasła, 162
 - stałych, 247
- deklarowanie, 66
 - procedur, 228, 232
 - tablic, 255
 - zmiennych, 249
 - jawne, 252
 - zdarzenia, 96
 - zmiennych statycznych, 262
- diagnostyka
 - koðu, 321
 - projektu, 427
- dodawanie, 272
 - dat i godzin, 284
 - elementów do listy, 171
 - elementów listy, 191
 - kontroltek, 29, 31, 51, 129, 528
 - nowego formularza, 111
 - plików projektu, 64
 - przycisków, 117, 214
 - węzłów do listy, 195
 - zakładek, 186

dokowanie paneli, 48
dostęp
do rejestru, 421
do zdarzeń obiektu, 95
dostosowanie pola tekstowego, 525
drukowanie, 461
aktualnie wyświetlanego obrazu, 468
dokumentu, 465
dwuwymiarowa tablica, 257
dymek z tekstem pomocy, 527
dynamika metod, 80
dziedziczenie, 72
dzielenie, 273

E

edytor
Items Collection Editor, 217
tekstowy, 96
elementy
kontrolki TreeView, 194
listy, 191
obiektu, 82
elipsa, 389
enkapsulacja, 72
kodu i danych, 346
etykieta, 155, 302
Excel, 510

F

filtr plików, 40, 404
flagi atrybutów pliku, 413
foldery, 415
formatowanie daty i czasu, 286
formularze, 23, 26, 109
do wysyłania wiadomości, 486
dodawanie kontroltek, 129
dodawanie narzędzi, 201
dodawanie pasków menu, 201
dodawanie przycisków, 117
modalność, 122
moduły, 225
określanie położenia, 124
określanie wielkości, 120
potomne, 151
przewijane, 147
przezroczyste, 147
przypisywanie ikony, 117

startowe, 152
techniki projektowania, 129
ukrywanie, 121
wyświetlanie, 121, 123
zamykanie, 126
zmiana wyglądu, 119
zmienianie koloru tła, 112
zmienianie nazwy, 110
zmienianie wyglądu, 110
funkcja, 228, 232
CLng(), 328, 329, 331
DateAdd(), 284, 285
InStr(), 281
IsDate(), 287
IsNumeric(), 293
Len(), 279
Microsoft.VisualBasic.Left(), 279
Microsoft.VisualBasic.Right(), 280
Mid(), 280
funkcje
konwersji typu, 247
ramki, 119
tekstowe, 279

G

GDI, Graphical Device Interface, 382
gorący klawisz, 204
grafika, 381
grupy kontroltek, 166

H

hasło, 162

I

IDE, Integrated Development Environment, 20
ikona, 117
ikony formularza, 28, 118
informacje
o zaznaczonym elemencie listy, 174
od użytkownika, 370
instancja, 81
instrukcja
Case, 297
Catch ex As Exception, 336
Dim, 257

- Do...Loop, 313
- Elseif, 295
- End Function, 238
- End If, 37
- End Sub, 230, 238
- End Try, 336
- Exit, 303
- Exit For, 309
- For, 307
- For...Next, 307, 310
- GoTo, 302
- If...Then, 37, 296
- Next, 308, 310
- print, 328
- Select...Case, 296, 298, 301
- Try...Catch...Finally, 335
- With, 403
- IntelliSense, 75, 99

- interakcje
 - przy użyciu klawiatury, 372
 - z użytkownikiem, 361
- interfejs, 346, 349
 - obiektu, 347
 - urządzenia graficznego, 382
 - użytkownika, 31, 35, 99

J

- jawne
 - deklarowanie zmiennych, 252
 - określanie typów danych, 253
- język Microsoft IL, 544, 562

K

- klasa, 26, 346
 - Attachment(), 483
 - DataAdapter, 442
 - DataSet, 442
 - DataTable, 442, 453
 - MailMessage, 483, 484
 - SMTPClient(), 483
 - SmtptClient, 484
 - SqlConnection, 442
 - Word.Application, 518
- klawiatura, 372
- klawisz
 - F5, 35
 - Shift, 137

- klient, 347, 509
- klucze główne rejestru, 420
- kod
 - do wysyłania wiadomości, 490
 - klienta, 349
 - maszynowy, 545, 563
 - obsługującego zdarzenia, 100
 - obsługujący interfejs użytkownika, 35
 - obsługujący zdarzenia, 95
 - odczytujący atrybuty pliku, 413
 - procedur, 228
 - zamykający aplikację, 38
 - zarządzalny, 544, 562
- kolejność wyróżniania kontrolki, 144
- kolekcja, 71, 84
 - Buttons, 217
 - Controls, 87
 - Items, 170, 171, 173, 214
 - Nodes, 195
 - TabPage, 185
- kolor, 55
 - tła, 112
- kolory systemowe, 385, 386
- kolumna, 190
- komentarze, 322
- kompilacja, 24
- kompilator, 244
- komponent, 510
 - dystrybucyjny, 20
 - pętli For, 308
 - projektu, 62
- komunikat, 335, 361
- konflikt zakresów widoczności, 262
- konkatenacja, 278
- kontekstowy system pomocy, 67
- kontener, 166
- kontrolka, 25, 29
 - Button, 31, 75
 - CheckBox, 165
 - ComboBo, 176
 - ContextMenuStrip, 210
 - GroupBox, 166
 - ImageList, 188, 190
 - Label, 183
 - ListBox, 134, 170
 - ListView, 189, 192, 193, 195
 - MenuStrip, 203
 - OpenFileDialog, 31, 34, 37, 40, 402
 - PageSetupDialog, 464

paska narzędzi, 213
 PictureBox, 31, 102
 PrintDocument, 464
 PrintPreviewDialog, 464, 473
 RadioButton, 168
 SaveFileDialog, 34, 406
 StatusStrip, 219
 TabControl, 186, 198
 Timer, 93
 ToolStrip, 213
 ToolTip, 527
 TreeView, 194, 197
 UserControl, 529, 530

kontrolki

niewidoczne, 33
 potomne, 522
 tradycyjne, 155
 widoczne, 31
 zaawansowane, 181
 zagregowane, 527, 531
 zakotwiczone, 140

konwersja

typu danych, 246
 w dół, 246
 w górę, 246

kopiowanie pliku, 408

kotwiczenie, 140

kontrolki, 142

kształty, 389

L

liczby ujemne, 272

linia, 389

lista, 171

elementów, 170
 hierarchiczna, 194
 kolorów, 57
 rozwijana, 176, 177
 zaawansowana, 189

literał, 279

Ł

łączenie ciągów znaków, 278

M

maksymalizowanie, 118

manipulowanie

elementami listy, 170, 171
 kontrolkami, 131
 listami, 193

mapa bitowa, 383

MDI, Multiple-Document Interface, 149

menu, 201

kontekstowe, 210
 podręczne, 64
 polecenia, 206
 Project, 64
 rozwijane, 218

metoda, 71, 79, 348, 353

Add(), 172, 193
 BaseDirectory(), 435
 Clear(), 82, 174, 198
 Close(), 126, 431
 Copy(), 409
 CreateDirectory(), 415
 CreateGraphics(), 382
 Delete(), 410
 Dispose(), 83
 DrawLine(), 389
 DrawRectangle(), 82
 Exists(), 408
 GetAttributes(), 413, 415
 GetCreationTime(), 412, 415
 GetLastAccessTime(), 412, 415
 GetLastWriteTime(), 412, 415
 Invalidate(), 396
 Me.Close(), 39
 MessageBox.Show(), 67, 266, 365–367
 Move(), 409
 ReadToEnd(), 432
 Remove(), 172, 194, 197
 SelectNextControl(), 146
 SetValue(), 424
 ShowDialog(), 37, 369, 405
 WriteLine(), 432

metody niestandardowe, 535

Microsoft Excel, 510

Microsoft Office 2016, 509

Microsoft Word, 516

miniformularz, 166

minimalizowanie, 118

mnożenie, 272
 modalność formularza, 122
 model obiektowy, 509
 modulo, 273
 moduł, 260

- kodu, 225
- klasy, 347

 moduły standardowe, 347
 modyfikowanie nazw zmiennych, 267
 mysza, 375

N

nadawanie

- nazw obiektom, 25
- wartości kluczy, 423

 narzędzia diagnostyczne, 326
 narzędzie Object Browser, 87
 nazwa

- formularza, 110
- metody, 82
- obiektu, 25, 82
- procedury zdarzeń, 104
- zmiennej, 267

 niemodalne okna, 146
 nieskończone wywołania, 239
 niestandardowe

- metody, 535
- właściwości, 534
- zdarzenia, 535

 nowy projekt, 21, 44

O

obiekt, 71, 72, 345

- Application, 513
- DataTable, 448
- Excel.Application, 513
- Exception, 338
- File, 407
- FileAttributes, 411
- formularza, 104
- Graphics, 381, 382, 383
- MouseEventArgs, 98
- My.Computer.Registry, 421
- StreamWriter, 432
- System.IO.Directory, 415
- System.IO.File, 408
- Word, 517

obiektowość, 72
 obiekty

- czas życia, 358
- przeglądanie, 54
- tworzenie, 353, 357
- tworzenie interfejsu, 347
- typu Control, 25
- usuwanie, 358
- wczesne wiązanie referencji, 356
- zapisywanie referencji, 354
- zmienianie właściwości, 54

 obraz

- formularza, 391
- w tle formularza, 114

 obsługa

- błędów, 335
- grafiki, 391
- interfejsu użytkownika, 35
- klawiatury, 373
- serwera automatyzacyjnego, 513
- spodziewanych wyjątków, 338
- środowiska Visual Studio 2015, 43
- wyjątków, 337
- zdarzenia, 36
- zdarzeń, 99, 100
- zdarzeń myszy, 377

 odbicie, 355
 odczytywanie

- atrybutów pliku, 413
- danych, 450
- daty i czasu, 412
- daty i godziny, 287
- plików tekstowych, 430, 432
- wartości kluczy, 423
- właściwości, 73

 odejmowanie, 272

- dat i godzin, 284

 odinstalowanie udostępnionej aplikacji, 504
 odstępy między kontrolkami, 138
 odwołania do tablicy, 256
 odwołanie wprzód, 25
 ograniczanie liczby znaków, 160
 okna z zakładkami, 184
 okno

- Czcionka, 56
- Define Color, 58
- do wybierania plików, 405
- Items Collection Editor, 214, 216
- New Project, 21, 22, 45

- okno
 - programu Excel, 514
 - String Collection Editor, 170
 - View Detail, 330
 - z kodem, 36
 - okrąg, 389
 - określanie
 - przedziału czasu, 285
 - typu danych, 244
 - wielkości formularza, 120
 - zakresu widoczności, 258
 - klikniętego przycisku, 365
 - OOO, object-oriented programming, 226, 346
 - opcja
 - Layout Mode, 133
 - ShowGrid, 133
 - opcje technologii ClickOnce, 506
 - operacje
 - arytmetyczne, 271
 - na ciągach znaków, 278
 - na datach i godzinach, 283
 - na plikach, 401
 - na rejestrze, 419
 - operator
 - And, 277
 - Not, 277
 - Or, 277
 - Xor, 278
 - operatory
 - logiczne, 271, 276
 - porównania, 275
 - opis właściwości, 58
 - otwieranie projektu, 46
- P**
- pamięć, 547, 565
 - panel
 - Error List, 255, 324
 - Immediate Window, 328, 330
 - Personalizacja, 385
 - Properties, 24, 53
 - Solution Explorer, 59
 - Toolbox, 30, 51, 53
 - panele
 - automatycznie ukrywane, 47, 50
 - projektowe, 47
 - swobodne, 47
 - ukrywanie, 47
 - wyświetlanie, 47
 - zadokowane, 47
 - zamknięte, 47
 - parametr, 96, 236
 - Format_pikseli, 383
 - Provider, 445
 - Przyciski, 363
 - typu MessageBoxIcon, 363
 - parametry
 - procedury, 66
 - zdarzeń, 97
 - pasek
 - Layout, 137
 - narzędzi, 51, 213, 215
 - menu rozwijane, 218
 - programowanie, 217
 - ukrywanie, 51
 - wyświetlanie, 51
 - przewijania, 159
 - stanu, 218
 - tytułu, 112
 - pętla, 307
 - For...Next, 307, 310
 - Do...Loop, 259, 313
 - Loop, 308
 - rekurencyjna, 239
 - pióro, 384
 - platforma
 - .NET, 20, 543, 561
 - ADO.NET, 442
 - plik dziennika, 433
 - pliki
 - kopiowanie, 408
 - odczytywanie atrybutów, 412
 - odczytywanie właściwości, 411
 - projektu, 64
 - przenoszenie, 409
 - tekstowe, 419
 - usuwanie, 410
 - wykonywalne, 24
 - zmiana nazwy, 410
 - pluskwa, 321
 - podgląd
 - dokumentu, 471
 - wydruku, 464
 - podmenu, 205
 - podprogram, 228, 229

- pola
 - obiektu DataRow, 451
 - opcji, 168
- pole
 - do wprowadzania hasła, 162
 - do wprowadzania tekstu, 157
 - tekstowe, 159, 525
 - tekstowe wielowierszowe, 158
 - wyboru, 165
- polecenia menu, 206
 - programowanie, 208
 - przemieszczanie, 206
 - przypisywanie skrótów klawiszowych, 212
 - usuwanie, 206
 - zaznaczane, 206
- polecenie
 - Add/New Item, 64
 - Checked, 206
 - Debug/Stop Debugging, 325
 - New Item, 64
 - Options, 96
 - Reset, 116
- polimorfizm, 72
- połączenie z bazą danych, 443
- położenie formularza, 124
- pomoc, 67
- porównywanie wartości, 275
- porządkowanie pamięci, 547, 565
- potęgowanie, 273
- późne wiązanie, 354
 - referencji obiektu, 354
- priorytety operatorów, 273
- procedura, 66, 225, 228
 - DrukujObraz(), 467
 - InputBox(), 370
 - RysujKontrolkę(), 532
 - typu Function, 232
 - typu Sub, 228
- procedury
 - niezwracające wartości, 228
 - zakres, 260
 - zwracające wartości, 232
- proces zdarzenia, 36
- program, 61
 - instalacyjny, 500
- programowanie, 233, 223
 - obiektywne, OOP, 226
 - paska narzędzi, 217
 - poleceń menu, 208
- projekt, 20
 - dodawanie kontrolek, 29
 - interfejs użytkownika, 31, 35
 - komponenty, 62
 - otwieranie, 46
 - pliki, 64
 - tworzenie, 21
 - typu Windows Forms Application, 23
 - uruchomienie, 39
 - właściwości, 63
 - z obsługą grafiki, 391
 - z obsługą zdarzeń, 99
 - zapisywanie, 27
 - zarządzanie, 59
- projektowanie
 - formularzy, 129
 - interfejsu użytkownika, 31
- projekty
 - jednodokumentowe, 149
 - wielodokumentowe, 149
- prostokąt, 387, 389
- przechowywanie
 - danych, 66
 - obrazów, 187
- przedział czasu, 285
- przeglądanie
 - plików, 36
 - rekordów danych, 453
- przeglądarka obiektów, 87
- przekazywanie parametrów, 236
- przemieszczanie poleceń menu, 206
- przenoszenie pliku, 409
- przerywanie wykonywania kodu, 335
- przestrzeń nazw, 260, 545, 563
 - System.IO, 407
- przetwarzanie danych, 448
 - w skoroszycie, 514
- przetwarzanie iteracyjne, 84
- przyciąganie do linii, 133
- przycisk, 32, 162
 - Align Tops, 138
 - anulujący, 164
 - do wysyłania wiadomości, 484
 - Drukuj, 462
 - kontrolny, 117
 - Make Same Size, 138
 - maksymalizujący, 117
 - minimalizujący, 117
 - Podgląd wydruku, 462
 - Save All, 102

przycisk
 Show All Files, 60
 Stop Debugging, 77, 84, 150
 zatwierdzający, 164

przypisywanie
 ikony, 117
 skrótów klawiszowych, 212

pułapka, 326, 328
 pułapki zaawansowane, 332
 punkt kontrolny, 335

R

ramka, 119
 referencje do biblioteki automatyzacyjnej,
 510, 517
 referencje obiektu, 354
 rejestr systemu Windows, 419
 odczytywanie ustawień, 425
 struktura, 420
 tworzenie kluczy, 422
 usuwanie kluczy, 423
 zapisywanie ustawień, 426

rekord danych, 453
 rekurencyjne wywoływanie
 procedur, 94
 zdarzeń, 93

RGB, red, green, blue, 112
 rozmiar kontrolki, 138
 rozwiązanie, 61
 rozwiązanie, solution, 20

rysowanie
 kształtów, 389
 linii, 389
 okręgów i elips, 389
 prostokąta, 82
 prostokątów, 389
 tekstu, 390

S

sekcja
 All Languages, 96
 Get, 351
 Set, 351
 Text Editor, 96

serwer, 347, 509
 automatyzacyjny, 513, 517

siatka, 131
 skalowanie obrazu, 476
 skoroszyt, 514
 słowo kluczowe
 Dim, 249
 Public, 228
 Step, 309
 Until, 314
 While, 314

sortowanie listy, 176
 sprawdzanie
 poprawności danych, 331
 kodu, 492
 programu, 84
 programu instalacyjnego, 502

stała, 243, 247
 DialogResult.OK, 37
 stałe modułowe, 250
 stan formularza, 123

sterowanie
 aplikacjami Microsoft Office 2016, 509
 programem Microsoft Excel, 510
 programem Microsoft Word, 516

stos, 239
 stosowanie
 kolorów systemowych, 385
 kontrolki OpenFileDialog, 402
 kontrolki SaveFileDialog, 406
 piór, 384
 przestrzeni nazw, 546, 564
 stałych, 247

struktura
 kolekcji, 84
 rejestru, 420

strukturalna obsługa błędów, 335
 szablon formularza, 26

Ś

środowisko
 CLR, 544, 562
 IDE, 20
 programistyczne, 17
 uruchomieniowe, 544
 Visual Studio 2015, 20

T

- tablice, 243, 255
 - wielowymiarowe, 256
- technologia ClickOnce, 499, 506
- tekst, 390
 - statyczny, 155
- test, 427
 - aplikacji klienckiej, 516
 - zintegrowanej kontrolki, 536
- testy dziennika, 437
- tworzenie
 - czasomierzy, 181
 - czytelnych komunikatów, 366
 - formularzy, 109
 - funkcji, 232
 - głównego menu, 202
 - hierarchicznych list, 194
 - instancji, 81
 - instancji obiektów, 353
 - interfejsu MDI, 149
 - interfejsu obiektu, 347
 - interfejsu użytkownika, 99, 107
 - kluczy rejestru, 422
 - końca obiektowego, 81
 - końca procedur, 228
 - kontenerów, 166
 - kontrolki potomnej, 522
 - kontrolki zintegrowanej, 527
 - list rozwijanych, 176
 - menu, 201
 - niemodalnych okien, 146
 - niestandardowego zdarzenia, 535
 - niestandardowej metody, 535
 - niestandardowej właściwości, 534
 - nowego projektu, 44
 - obiektów, 345
 - obiektu DataTable, 450
 - obiektu Graphics, 382, 383
 - obiektu SqlDataAdapter, 449
 - obiektu w deklaracji zmiennej, 357
 - okien dialogowych, 184
 - paska stanu, 218
 - pętli Do...Loop, 313
 - pliku dziennika, 433
 - poleczeń, 205
 - procedur, 225
 - programu instalacyjnego, 500
 - projektu, 21

- przewijanych formularzy, 147
- przezroczystych formularzy, 147
- przycisków, 162
- przycisku, 32
- referencji do biblioteki, 517
- rekordów danych, 455
- rozwijanego menu, 218
- skoroszytu, 514
- tablic wielowymiarowych, 256
- wielowierszowego pola tekstowego, 158
- własnych kontroltek, 521
- własnych okien dialogowych, 367
- zaawansowanych list, 189
- zaznaczanych poleceń menu, 206
- zmiennych dla opcji, 264

typ danych, 243

Date, 283

typy

- procedur, 228
- wartości kluczy rejestru, 421

U

udostępnianie

- procedur, 353
- zmiennych klasy, 349

ukrywanie

- formularzy, 121
- paneli, 47
- pasków narzędzi, 51

upakowanie bitowe, 412

uruchomienie projektu, 39

ustawianie

- formularza startowego, 152
- siatki, 132
- strony, 473
- właściwości, 73
- właściwości kontroltek, 139
- właściwości Text, 27

usuwanie

- elementów, 194
- elementów z listy, 172
- kluczy rejestru, 423
- obiektów, 358
- plików projektu, 64
- pliku, 410
- poleczeń menu, 206
- rekordów danych, 457
- spacji, 282

usuwanie

- węzłów, 197
- zakładek, 186

utrwalanie obrazu formularza, 391

użycie

- instrukcji Select Case, 301
- obiektów, 80
- pętli Do...Loop, 315
- zmiennych, 251

V

Visual Studio

- dostosowywanie środowiska, 46
- obsługa, 46
- strona startowa, 44

Visual Studio 2015, 20, 24

W

warstwy kontroltek, 146

wartości

- parametru Format_pikseli, 383
- parametru Przyciski, 363
- właściwości DashStyle, 384
- wyliczeniowe typu DialogResult, 366

wartość

- False, 293
- zmiennej, 251

wczesne wiązanie, 354

- referencji obiektu, 356

wdrażanie aplikacji, 499

węzeł, 87, 196

wiadomości e-mail, 483, 484

wielkość formularza, 29

wielowierszowe pole tekstowe, 159

Windows Forms Application, 22

wklejanie znaków, 525

własne

- kontrolki, 521
- okna dialogowe, 367

właściwości, 71, 72

- do odczytu, 351
- do zapisu, 351
- grupy, 167
- klasy, 349
- kontroltek, 139
- niestandardowe, 534
- obiektów, 25, 53, 74

obiektu MouseEventArgs, 98

obiektu Registry, 422

określające kolor, 55

pliku, 411

projektu, 63

przycisku, 33, 75

składowe, 115

tylko do odczytu, 352

tylko do zapisu, 352

właściwość, 348

AcceptButton, 164

ActiveMdiChild, 151

Anchor, 141, 143

AutoCompleteMode, 178

AutoCompleteSource, 178

AutoScroll, 148

AutoScrollMargin, 148

AutoScrollMinSize, 148

AutoSize, 183

BackColor, 55, 112, 385

BackgroundImage, 114, 116

BackgroundImageLayout, 124

BorderStyle, 55

CancelButton, 165

CheckState, 166

Columns, 190

ConnectionString, 445, 446

ContextMenuStrip, 211

Count, 193

DashStyle, 384

DialogResult, 368

DropDownStyle, 178

Enabled, 158, 160

Filter, 404

Font, 55

FormBorderStyle, 119, 220

Height, 28

Interval, 93

Items, 176

Location, 168, 202

Location.Y, 219

MaxLength, 160

Message, 338

Multiline, 158, 160

Name, 26, 31

Opacity, 311

Option strict, 254

Parent, 195

ScrollBars, 159, 160

SelectedIndex, 187

- SelectedItems, 193
- SelectionMode, 176
- ShortcutKeys, 212
- ShowInTaskbar, 126
- Size, 55
- Sorted, 176
- StartPosition, 125
- Size, 28
- TabIndex, 144, 145
- Text, 27, 156, 191
- Title, 404
- ToolTipText, 413
- TopMost, 146
- View, 190
- WindowState, 123, 124
- Width, 28
- wprowadzanie hasła, 162
- wspólny system typów, 547, 565
- wstrzymywanie kodu, 334
- wybieranie
 - ikonek, 30
 - koloru, 114
 - obiektów, 54
 - plików, 405
- wychodzenie
 - z pętli, 309, 313
 - z procedur, 238
- wyciek pamięci, 83
- wydruk, 464
- wygląd kontrolki, 530
- wyjątek, 324, 337
 - InvalidCastException, 325, 328, 339
 - OutOfMemoryException, 340
 - StackOverflow, 94
- wyjątki
 - nieobsłużone, 340
 - spodziewane, 338
- wyodrębnianie części daty, 286
- wyrażenia, 251
- wyrównywanie
 - kontrolki, 137
 - tekstu, 158
- wysyłanie wiadomości, 484
- wyświetlanie
 - formularza, 121, 123
 - ikony, 125, 363
 - komunikatów, 335, 361
 - obrazów, 41, 116
 - okna programu Excel, 514
 - paneli, 47

- pasków narzędzi, 51
- pliku dziennika, 436
- przycisków, 363
- tekstu, 112
- wywołania rekurencyjne, 239
- wywoływanie
 - metod, 79
 - procedur, 225, 233
 - zdarzeń, 92

Z

- zaawansowane cechy pułapek, 333
- zagnieżdżanie
 - instrukcji, 295
 - konstrukcji If...Then, 296
- zakładka, 185
 - Toolbox, 24
 - Web, 58
- zakończenie pętli, 309
- zakres
 - bloku, 259
 - modułu, 260
 - procedury, 260
 - przestrzeni nazw, 260
 - wartości, 245
 - widoczności, 243, 258
- zamykanie
 - formularza, 118, 126
 - programu, 38
- zapisywanie
 - plików tekstowych, 430
 - projektu, 27
 - referencji obiektu, 354
 - ustawień w rejestrze, 426
- zarządzanie
 - plikami projektu, 59
 - projektami, 59
- zastępowanie ciągu znaków, 282
- zastosowanie zmiennych, 263
- zaznaczane polecenia menu, 206
- zaznaczanie grup kontrolki, 135
- zdarzenia, 35, 71, 91
 - klawiatury, 373
 - myszy, 375
 - niestandardowe, 535
 - pola tekstowego, 162
 - wywoływane przez system operacyjny, 93
 - wywoływane przez użytkownika, 92

zdarzenie

- Click, 36, 77, 163, , 233 375
- FormClosed, 376
- FormClosing, 266
- KeyDown, 373
- KeyPress, 373, 526
- KeyUp, 373
- Load, 265
- MouseDown, 163, 375
- MouseEnter, 375
- MouseLeave, 102, 375
- MouseMove, 103, 163, 375
- MouseUp, 163, 375
- Paint, 395
- SelectedIndexChanged, 187
- TextChanged, 93, 163, 526
- Tick, 183

- zgłoszenie wyjątku, 324

zmienianie

- cech obiektów, 25
- koloru tła, 112
- nazwy pliku, 410
- nazwy formularza, 110
- rekordów danych, 455
- ustawień strony, 473
- wielkości formularza, 28
- wyglądu formularza, 110

- zmienna, 66, 249
 - typu Object, 354
- zmiennie, 243
 - klasy, 349
 - publiczne, 264
 - statyczne, 262
 - z opcjami, 265
- znak kontynuacji, 83

PROGRAM PARTNERSKI

GRUPY WYDAWNICZEJ HELION



1. ZAREJESTRUJ SIĘ
2. PREZENTUJ KSIĄŻKI
3. ZBIERAJ PROWIZJĘ

Zmień swoją stronę WWW
w działający bankomat!

Dowiedz się więcej i dołącz już dzisiaj!

<http://program-partnerski.helion.pl>

Visual Basic jest bardzo popularnym językiem programowania wysokiego poziomu rozwijanym przez firmę Microsoft. Jego składnię oparto na języku Basic i unowocześniono, zapewniając przy tym wykorzystanie technologii ActiveX. Visual Basic w wersji 2015 zasadniczo różni się od swoich poprzedników: jest zdecydowanie lepszy, ma większe możliwości, a funkcjonalnością dorównuje takim językom jak C++. Jednak konsekwencją tego rozwoju jest większa złożoność języka.

Niniejsza książka ma jeden cel: jak najszybciej nauczyć Cię poprawnego programowania w języku Visual Basic. W ciągu 26 lekcji, z których każda powinna zająć najwyżej godzinę, przyswoisz sobie praktyczne umiejętności pozwalające na samodzielne zbudowanie kompletnej aplikacji. Gruntownie poznasz środowisko programistyczne Visual Studio 2015 i takie elementy interfejsu użytkownika jak formularze i kontrolki, a także dowiesz się, jak je wykorzystać, aby przygotować atrakcyjną i funkcjonalną aplikację. W książce znajdziesz też liczne przykłady starannie objaśnionego kodu.

W książce omówiono:

- środowisko Visual Basic 2015, jego interfejs, funkcje i narzędzia
- podstawowe pojęcia programistyczne i szczegóły programowania w Visual Basic 2015
- rodzaje kontrolki i ich zastosowanie
- pracę z danymi tekstowymi, plikami graficznymi i bazami danych
- pracę z systemem plików użytkownika i interakcje z zewnętrznymi aplikacjami Windows
- zagadnienia dotyczące platformy Microsoft .NET

James Foxall jest prezesem Tigerpaw Software. Przedsiębiorstwo to rozwija oprogramowanie do automatyzacji procesów biznesowych dla licznych firm informatycznych, telekomunikacyjnych, integratorów systemowych i wielu innych. Foxall jest autorytetem w dziedzinie usprawniania procesów biznesowych, tworzenia interfejsów i standardów aplikacji Windows. Napisał 15 książek poświęconych wykorzystaniu zdobyczy technicznych do lepszego prowadzenia biznesu.



Poznaj Visual Basic — i programuj dla Windows!

sięgnij po **WIĘCEJ**



KOD KORZYŚCI

Helion

SAMS

księgarnia internetowa



<http://helion.pl>

Helion SA
ul. Kościuszki 1c, 44-100 Gliwice
tel.: 32 230 98 63
e-mail: helion@helion.pl
<http://helion.pl>

zamówienia telefoniczne



0 801 339900

Sprawdź najnowsze promocje:
● <http://helion.pl/promocje>
Książki najchętniej czytane:
● <http://helion.pl/bestsellery>
Zamów informacje o nowościach:
● <http://helion.pl/nowosci>



0 601 339900

ISBN 978-83-283-2874-7



Informatyka w najlepszym wydaniu

cena: 89,00 zł