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.

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.

Spis treści

CZĘŚĆ III WPRAWIANIE APLIKACJI W RUCH — PROGRAMOWANIE

Godzina 6.

Zaawansowane techniki projektowania formularzy

W ciągu tej godziny poznasz następujące zagadnienia:

- **dodawanie kontrolek do formularza,**
- **rozmieszczanie, wyrównywanie, kotwiczenie i określanie wielkości kontrolek,**
- **ustalanie kolejności wyróżniania kontrolek,**
- **ustalanie kolejności warstw kontrolek,**
- **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 kontrolek dostępnych w języku Visual Basic.

Definiowanie kontrolek

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 kontrolek, dlatego ważne jest dokładne poznanie wszystkich aspektów tego procesu.

Dodawanie kontrolek 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 tworzonego formularza.

Kontrolki w panelu *Toolbox* podzielone są na kategorie, które można rozwijać i zwijać. W większości projektów będziesz używać kontrolek 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*, przesuwając kursor myszy poza jego obszar). Nowa kontrolka, po dwukrotnym kliknięciu, jest umieszczana na wierzchu aktualnie wybranej kontrolki (nie dotyczy to niewidocznych kontrolek) 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 kontrolek 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.

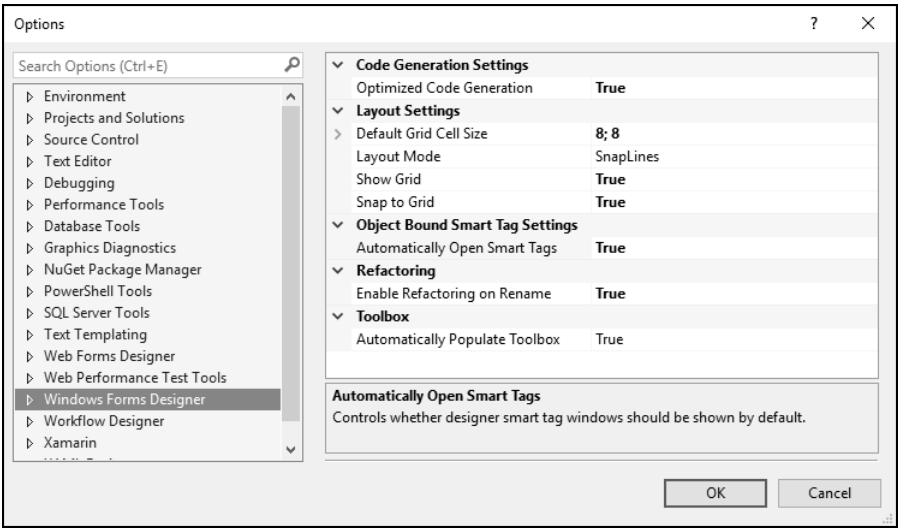
Siatka (gęstość i przyciąganie)

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

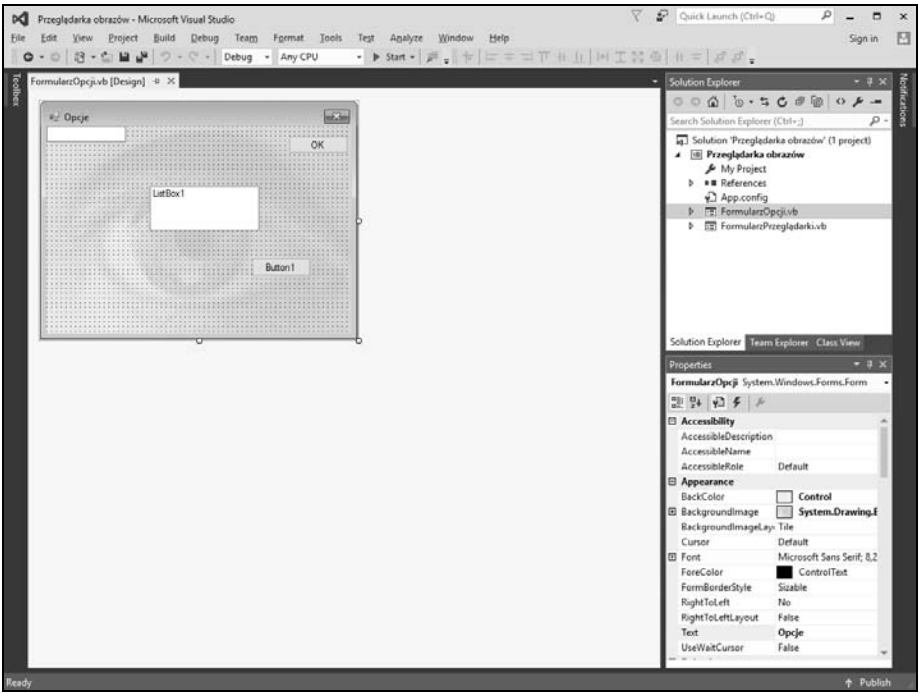
- *Snap to Grid* (przyciągaj do siatki) — opcja określająca, czy ma być stosowana siatka. Jeżeli opcja ma wartość *False*, wtedy ustawienia siatki nie są aktywne i kontrolki nie są do niej przyciągane.

W tej chwili siatka nie jest stosowana podczas rysowania kontrolek, ale w czasie przesuwania są one przyciągane do innych kontrolek, 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 kontrolek 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 kontrolek. Duże wartości powodują powstanie rzadszej siatki, dającej mniejszą kontrolę, ponieważ krawędzie kontrolek 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.

Ubocznym skutkiem ustawienia gęstszej siatki jest pogorszenie czytelności formularza. Musisz zadecydować, 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

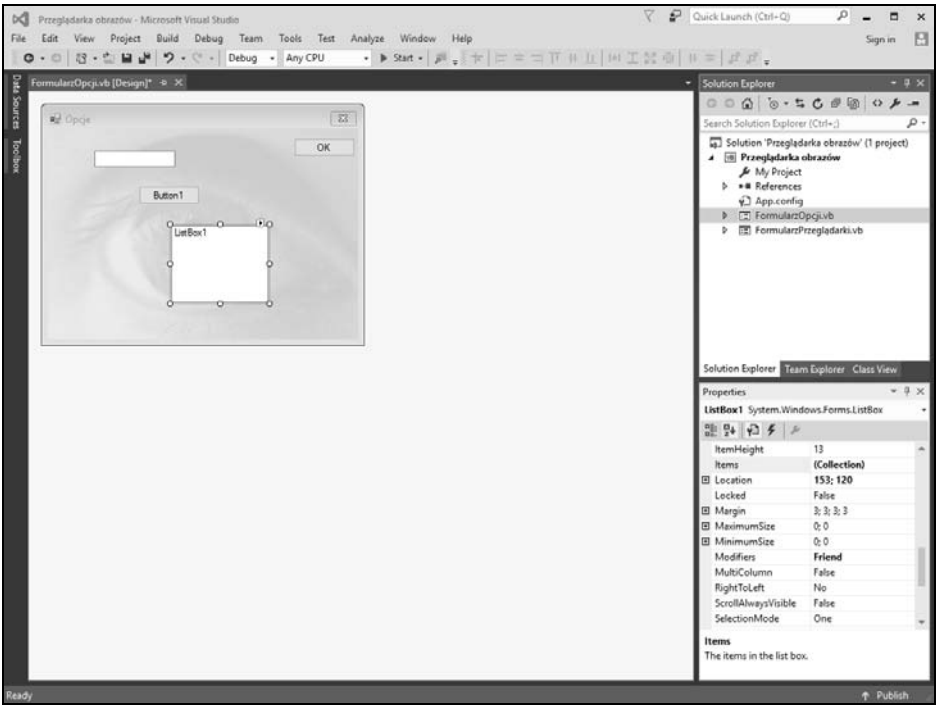
Stosunkowo nową i przydatną funkcją jest przyciąganie do linii (`SnapLines`). Włącz tę funkcję, wykonując następujące kroki:

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 kontrolek. Najłatwiej poznać tę możliwość, wypróbowując ją. Wykonaj następujące kroki:

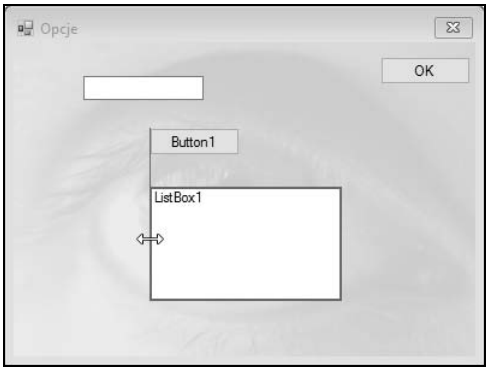
1. Rozmieść kontrolki jak na rysunku 6.3.



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 kontrolek. Funkcjonalność ta może się wydawać dość błaha, ale wierz mi, jest ona bardzo przydatna i oszczędzi Ci wielu godzin mozolnej pracy.

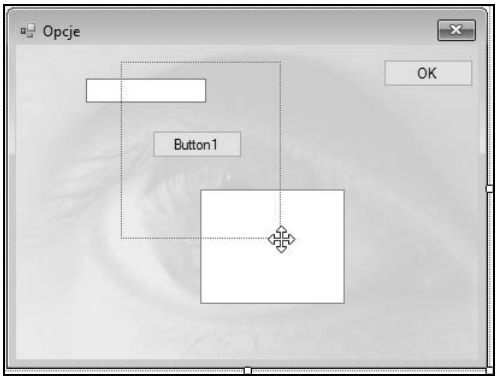


RYSUNEK 6.4. Dzięki przyciąganiu do linii wyrównywanie krawędzi kontrolek jest łatwiejsze

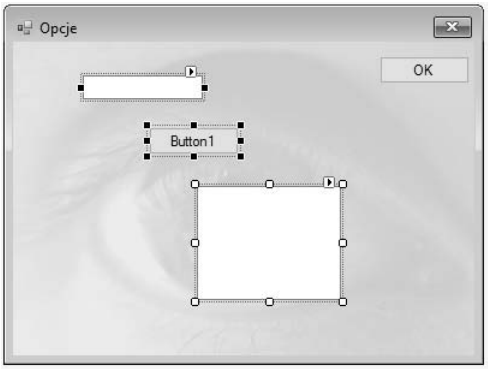
Zaznaczanie grup kontrolek

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

Domyślnie kliknięcie kontrolki powoduje jej zaznaczenie i *jednoczesne* usunięcie zaznaczenia innych kontrolek. 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 kontrolek, 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 kropkowane ramki 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 kontrolek (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 kontrolek, nie będziesz miał większego wpływu na wybór aktywnej kontrolki. Jeżeli chcesz wyrównać kontrolki do lewej krawędzi przycisku, musisz zastosować inną technikę zaznaczania kontrolek. Usuń zaznaczenie wszystkich kontrolek, klikając formularz w dowolnym miejscu (nie klikaj jednak żadnej kontrolki).

Przy okazji

Nie każdy uchwyt można przesuwać 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 przesuwać.

Inną techniką zaznaczania wielu kontrolek 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 kontrolek, 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

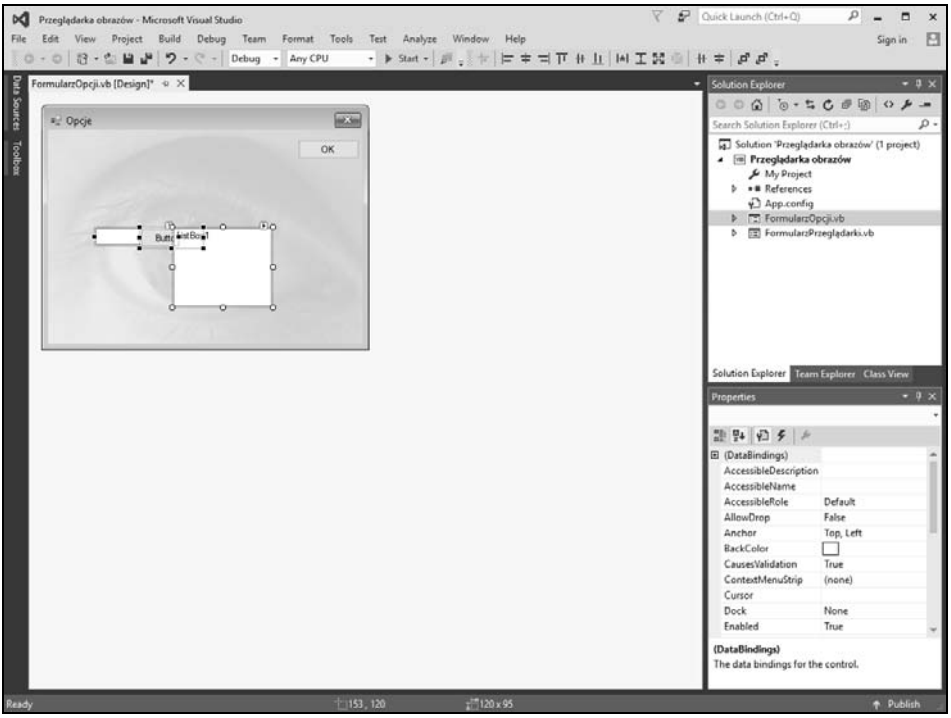
Środowisko Visual Studio zawiera kilka formatujących kontrolki narzędzi, które możesz wykorzystać do tworzenia atrakcyjnych interfejsów użytkownika. Większość narzędzi znajduje się w pasku *Layout* pokazanym na rysunku 6.7. Wyświetl teraz ten pasek, klikając prawym przyciskiem myszy istniejący pasek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

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 kontrolek — 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 kontrolek

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 kontrolek. 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

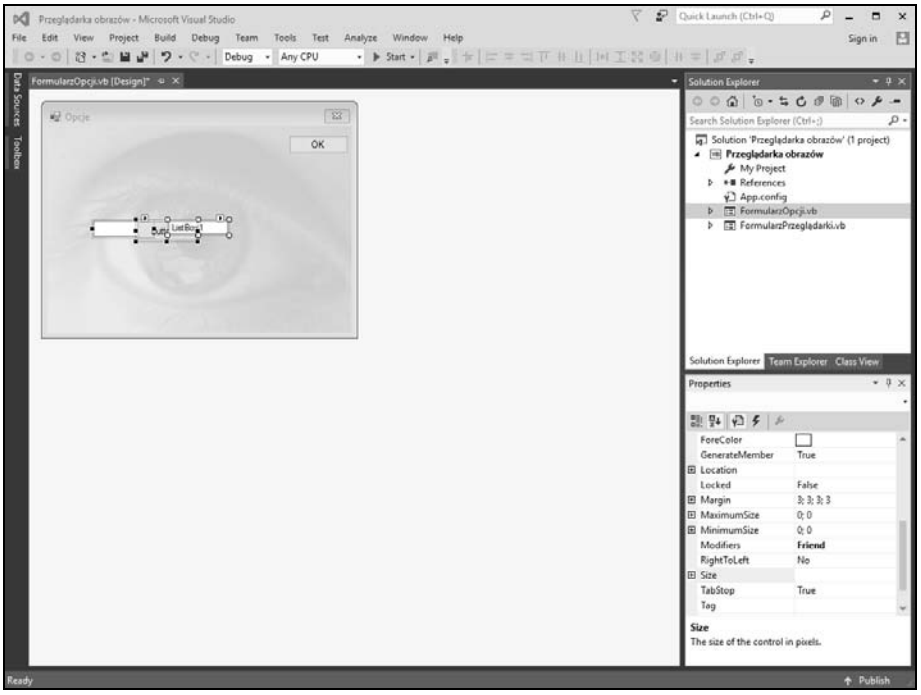
same. W poprzednich wersjach środowiska Visual Studio były dostępne dodatkowe przyciski do zwiększania i zmniejszania odstępów, ale w najnowszej wersji zostały one z jakiegoś powodu usunięte z paska narzędzi. Funkcja ta jest jednak dostępna po wybraniu polecenia *Format/Horizontal Spacing* (format/odstępy 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 kontrolek w grupie

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

Upewnij się, że trzy kontrolki są zaznaczone, a następnie otwórz panel *Properties* (jeżeli nie jest otwarty). Jeżeli zaznaczona jest grupa kontrolek, 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 kontrolek. Ponieważ zaznaczyłeś 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 kontrolek (na przykład `Location`), nie mają podanych wartości.



RYSUNEK 6.9. Możesz zmienić właściwości wielu kontrolek 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 kontrolek zostanie zmieniony na żółty.

Nie będziesz używać kontrolek 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.

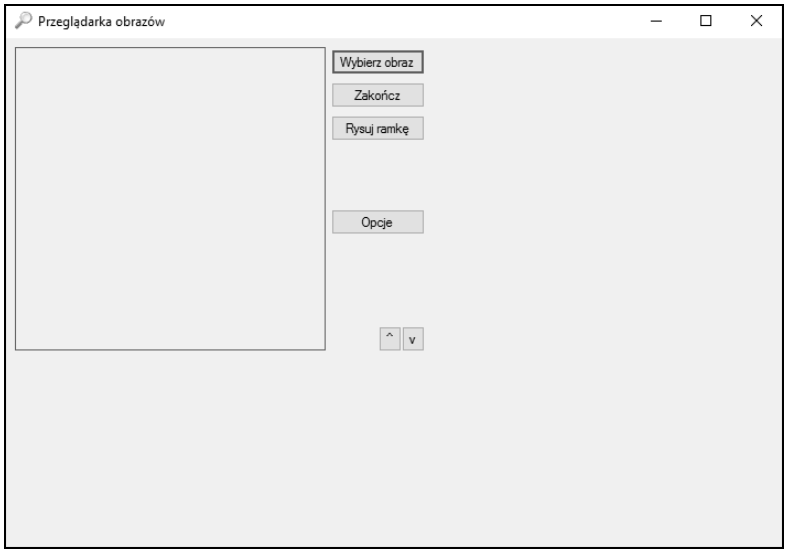
Kotwiczenie i automatyczne powiększanie kontrolek

Moje ulubione funkcje edytora formularzy w środowisku Visual Studio to kotwiczenie kontrolek do krawędzi formularza i dopasowywanie rozmiarów kontrolek podczas zmieniania 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.

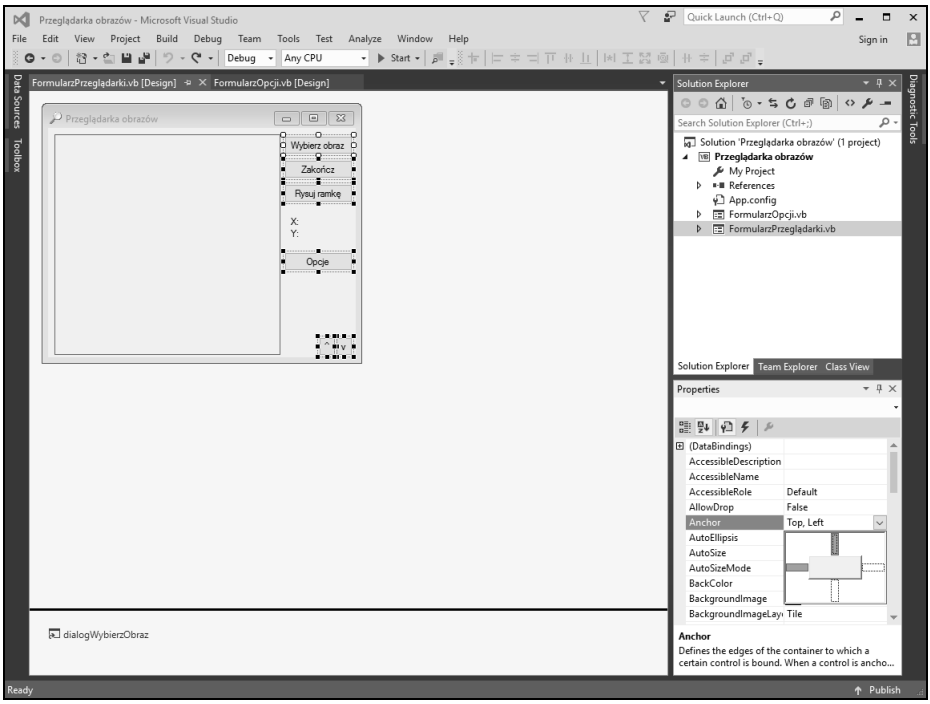
Wykonaj poniższe kroki:

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).



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

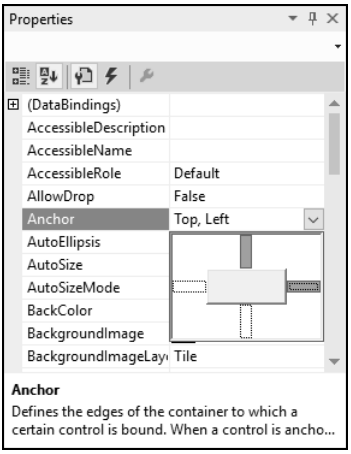
4. W pasku narzędzi kliknij przycisk *Stop Debugging*, aby zatrzymać program.
5. Kliknij przycisk *Wybierz obraz*, aby go zaznaczyć i — co najważniejsze — usunąć zaznaczenie formularza.
6. Naciśnij i przytrzymaj klawisz *Shift*, po czym kliknij kolejno przyciski *Zakończ*, *Rysuj ramkę*, *Opcje*, ^ oraz *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

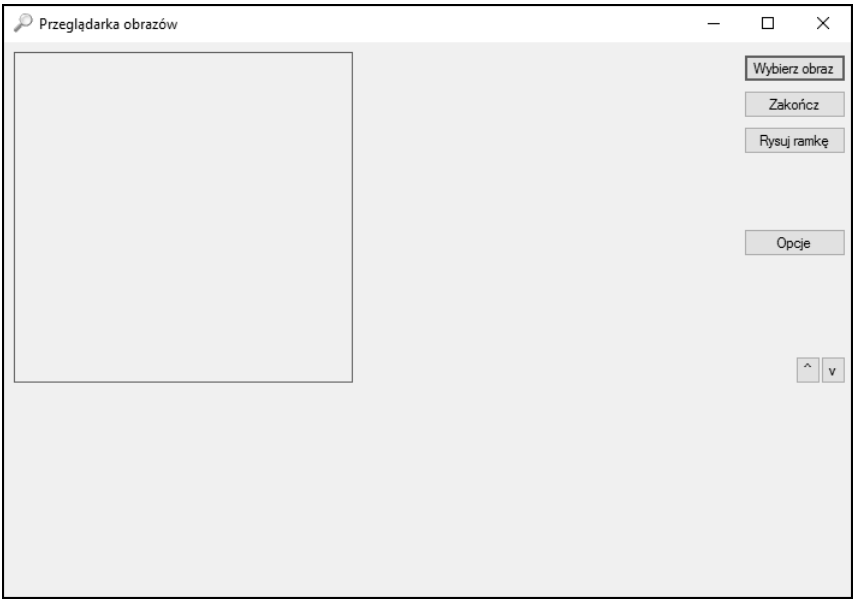
Szary prostokąt w środku okienka reprezentuje zaznaczoną kontrolkę (kontrolki). Mniejsze prostokąty: u góry, u dołu, po lewej i po prawej stronie reprezentują możliwe zakotwiczenia do krawędzi formularza. Wypełniony prostokąt oznacza, że kontrolka jest zakotwiczona do odpowiedniej krawędzi. Aby sprawdzić,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



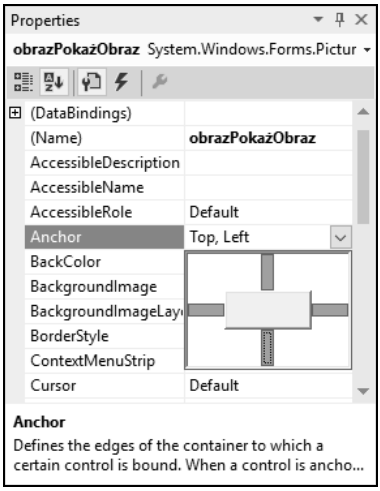
RYSUNEK 6.13. Kotwiczenie kontrolek jest niezwykle przydatną funkcjonalnością umożliwiającą tworzenie elastycznych formularzy

Zwróć uwagę, że podczas zmiany wielkości formularza ramka obrazu i etykiety ze współrzędnymi pozostają w dotychczasowym położeniu. To żaden problem, można go rozwiązać również za pomocą właściwości Anchor. Najpierw zmień zakotwiczenie etykiet, 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 kontrolek. 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:

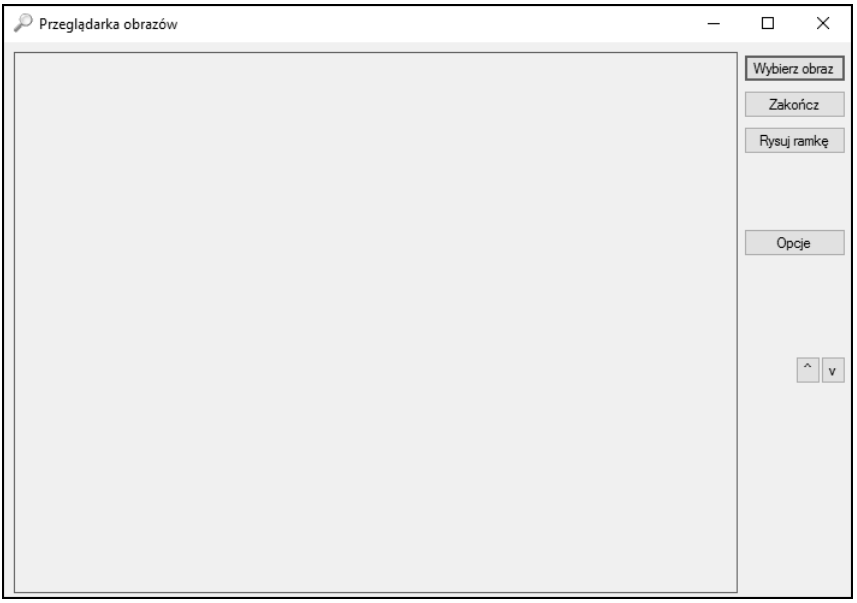
1. Kliknij ramkę obrazu, aby ją zaznaczyć.
2. Kliknij właściwość Anchor i wypełnij wszystkie cztery prostokąty, jak pokazano to na rysunku 6.14.



RYSUNEK 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 kontrolek

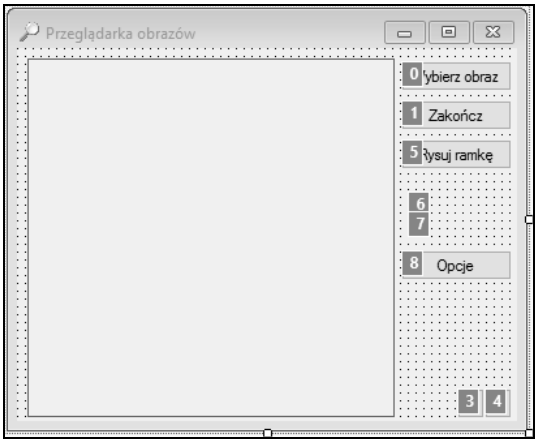
Kolejność wyróżniania kontrolek jest *często* zaniedbywana nawet przez doświadczonych programistów Visual Basic. Prawdopodobnie jako użytkownik korzystałeś z tej możliwości funkcjonalnej być może nieświadomie. Gdy podczas korzystania z formularza naciśniesz klawisz *Tab*, wyróżniana jest następna w kolejności kontrolka. W ten sposób można łatwo poruszać się po formularzu za pomocą klawiatury. Kolejność wyróżniania kontrolek 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 kontrolek 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 kontrolek. 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 kontrolek, więc zatrzymaj program, klikając w pasku narzędzi przycisk *Stop Debugging* lub zamykając okno programu.

Aby ustalić kolejność wyróżniania kontrolek, 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 kontrolek 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 kontrolek. 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?

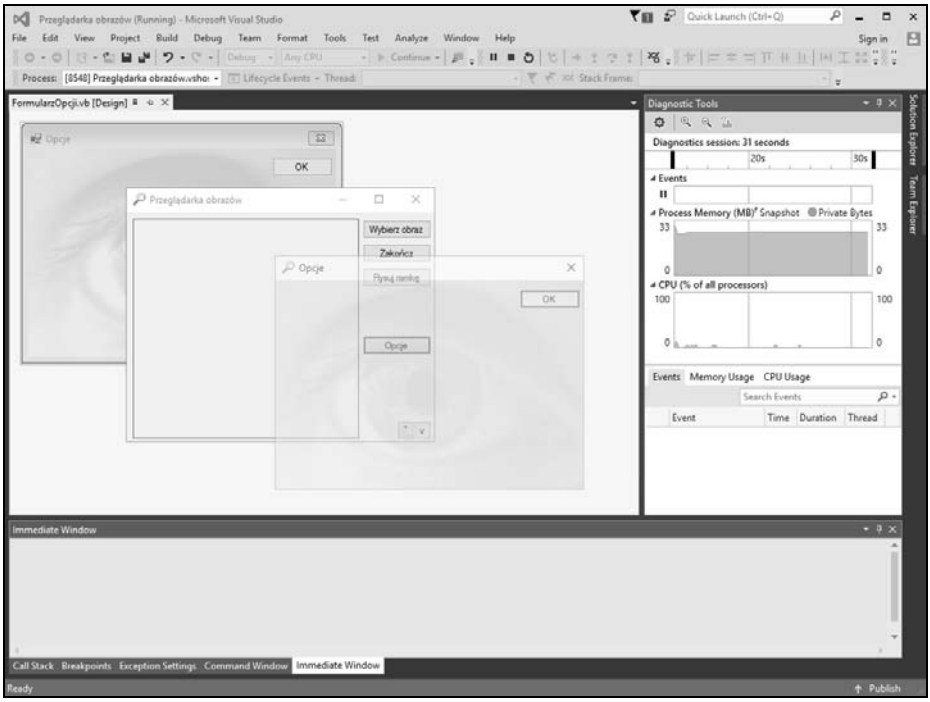
> Kontrolki możesz przenosić na wierzch lub na spód za pomocą kodu. Służą do tego metody kontrolki `BringToFront()` lub `SendToBack()`.

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 kontrolek. 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.

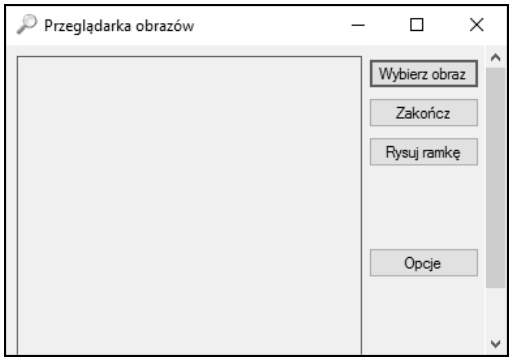
O możliwości przewijania formularza decydują trzy właściwości:

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 *FormularzPrzeglądarki.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 kontrolek, które wcześniej były niedostępne.



RYSUNEK 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:

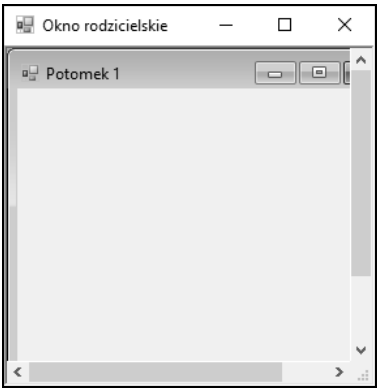
1. Jeżeli Twój program cały czas działa, zamknij go.
2. Wybierz polecenie *File/New/Project*. Wyświetli się okno *New Project* (zwróć uwagę, że jest to formularz modalny). Jeżeli pojawi się pytanie, czy zapisać zmiany w projekcie *Przeglądarka obrazów*, kliknij przycisk *Yes*.
3. Wpisz w polu *Name* 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()
```

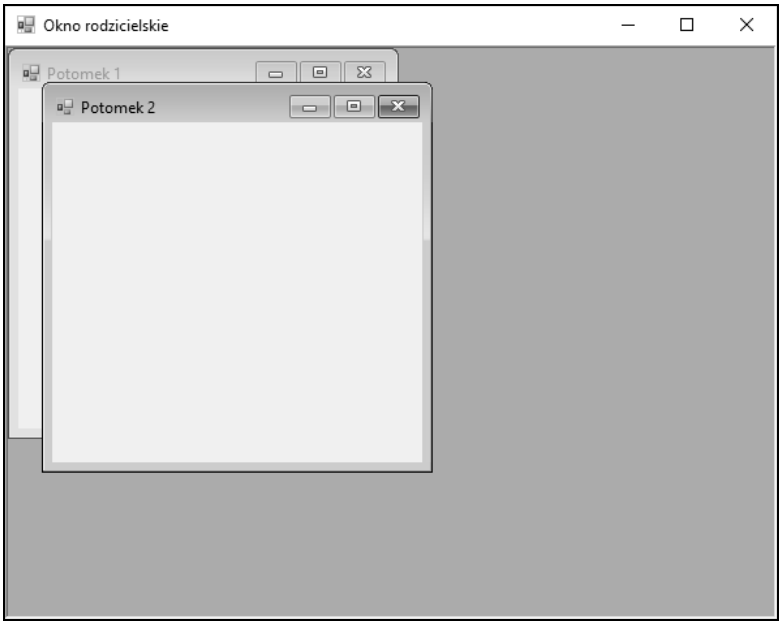
Powyższy kod powoduje wyświetlenie drugiego formularza. Zwróć uwagę, że kod ten różni się od wcześniejszego. Właściwości `MdiParent` drugiego okna potomnego nie możesz ustawić na `Me`, ponieważ słowo to jest referencją do bieżącego formularza

(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 właściwości `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

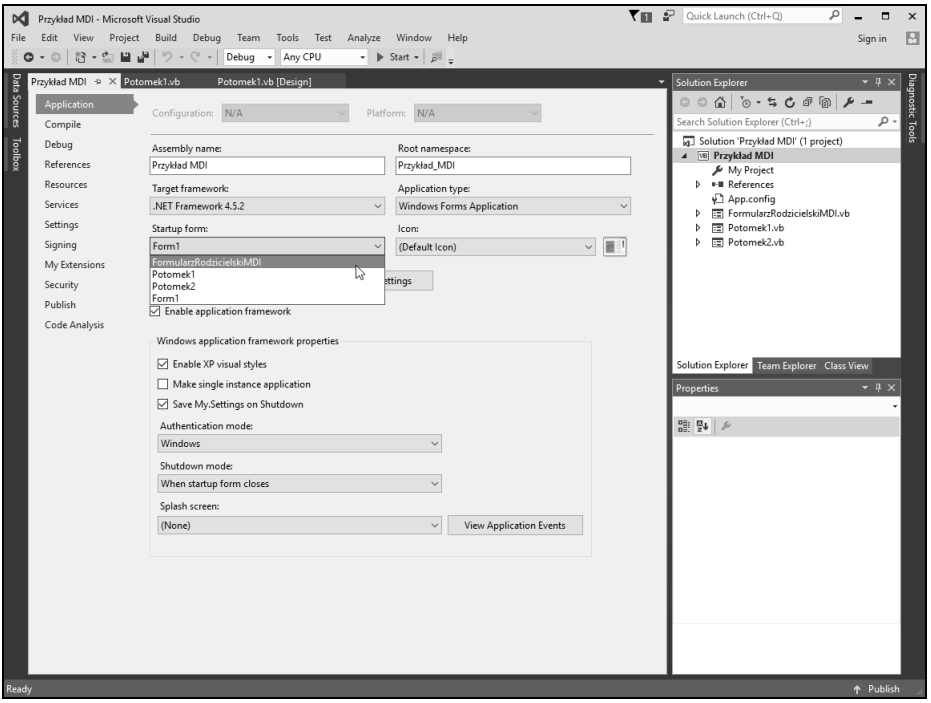
> Aby po uruchomieniu projektu formularz rodzicielski był większy, ustaw jego właściwości `Size.Height` i `Size.Width` ręcznie w projekcie albo po uruchomieniu za pomocą kodu w zdarzeniu `Load`.

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 zawiłości 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 kontrolek. W ciągu następnych dwóch godzin poznasz szczegóły dotyczące zaawansowanych kontrolek dostępnych w języku Visual Basic, które staną się ważnym elementem Twojego programistycznego arsenału.

Pytania i odpowiedzi

P: Czy powinienem się zajmować kotwiczeniem i przewijaniem kontrolek w każdym tworzonym 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 kontrolek 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 tworzysz 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 kontrolek?
7. Jaką 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

E

F

G

H

I

J

K

L

Ł

M

N

O

P

R

S

Ś

T

U

V

W

Z

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 kontrolek 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 procesorów 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!

Helion

SAMS

księgarnia internetowa
http://helion.pl
zamówienia telefoniczne
0 801 339900
0 601 339900

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

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

ISBN 978-83-283-2874-7

Informatyka w najlepszym wydaniu

cena: 89,00 zł