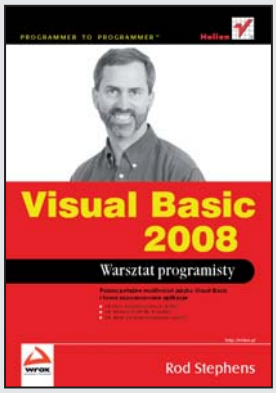
Visual Basic 2008. Warsztat programisty



Autor: Rod Stephens
Tłumaczenie: Łukasz Piwko
ISBN: 978-83-246-1842-2
Tytuł oryginału: Visual Basic 2008 Programmer's Reference
Format: 172x245, stron: 1224

Poznaj potężne możliwości języka Visual Basic i twórz zaawansowane aplikacje

- Jak pisać zaawansowane programy?
- Jak dobierać kontrolki do zadań?
- Jak dzielić program na mniejsze części?

Visual Basic jest niezwykle wydajnym językiem, pozwalającym na tworzenie zaawansowanych aplikacji, m.in. dzięki wykorzystywaniu jednego z najpotężniejszych środowisk programistycznych, jakie kiedykolwiek powstały – Visual Studio. Środowisko to udostępnia narzędzia takie, jak edytory graficzne czy funkcja automatycznego uzupełniania, które sprawiają, że budowa aplikacji staje się intuicyjna i znacznie sprawniejsza. Najnowsza wersja tego języka – Visual Basic 2008 – oferuje także opcje formularzy XAML i nowe kontrolki, a ponadto z aplikacją zintegrowano SQL Server 2005 Compact Edition, dzięki czemu pojawiło się wbudowane wsparcie dla języka LINQ w komunikacji z bazą danych.

Książka „Visual Basic 2008. Warsztat programisty" jest doskonałym podręcznikiem zarówno dla początkujących, jak i zaawansowanych programistów. Zawiera ona opis technologii i środowiska programistycznego, a także wiele przykładów i porad z dokładnie przedstawionym oraz przetestowanym kodem, zgodnym z wersją 2008 tego języka. Przykłady te pozwalają dogłębnie zrozumieć wszelkie jego zawiłości. Z książki dowiesz się, jak korzystać z formularzy i kontrolek oraz zwiększać funkcjonalność istniejących klas i obsługiwać błędy. Nauczysz się rysować obrazy przy użyciu interfejsu urządzenia graficznego, a także stosować wiele różnych obiektów, aby budować zaawansowane i nowoczesne aplikacje.

- Edytor kodu Visual Basic
- Struktura programu i modułu
- Typy danych, zmienne i stałe
- Operatory
- Procedury i funkcje
- Instrukcje sterujące
- Obsługa błędów
- Składnia zapytań i funkcje LINQ
- Tworzenie niestandardowych kontrolek
- Tworzenie nazw i klasy kolekcyjne
- Grafika i tekst
- Przetwarzanie obrazów
- Drukowanie i raportowanie
- Obiekty systemu plików
- Windows Communication Foundation

Bądź profesjonalistą – szlifuj swój warsztat programisty!

Helion SA
ul. Kościuszki 1c
44-100 Gliwice
tel. 032 230 98 63
e-mail: helion@helion.pl

Spis treści

8

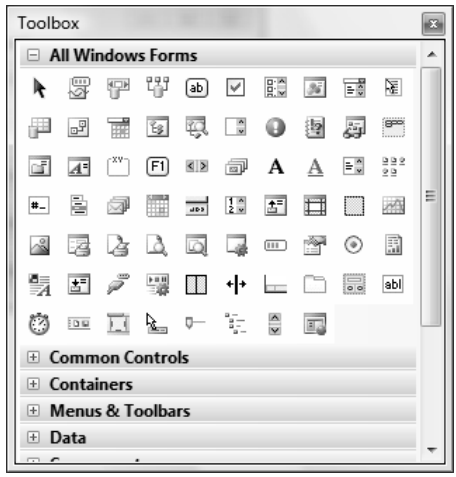
Wybieranie kontrolek Windows Forms

Kontrolki są niezwykle ważną częścią każdej interaktywnej aplikacji. Dostarczają użytkownikom informacje (`Label`, `ToolTip`, `TreeView`, `PictureBox`) oraz organizują je, dzięki czemu łatwiej jest je zrozumieć (`GroupBox`, `Panel`, `TabControl`). Umożliwiają wprowadzanie danych (`TextBox`, `RichTextBox`, `ComboBox`, `MonthCalendar`), wybieranie opcji (`RadioButton`, `CheckBox`, `ListBox`), kontrolowanie aplikacji (`Button`, `MenuStrip`, `ContextMenuStrip`) oraz uzyskiwanie dostępu do obiektów znajdujących się poza programem (`OpenFileDialog`, `SaveFileDialog`, `PrintDocument`, `PrintPreviewDialog`). Niektóre kontrolki oferują wsparcie dla innych kontrolek (`ImageList`, `ToolTip`, `ContextMenuStrip`, `ErrorProvider`).

W tym rozdziale znajduje się tylko bardzo krótki opis standardowych kontrolek Windows Forms oraz kilka wskazówek pomagających w podjęciu decyzji, której z nich użyć w określonym celu. Znacznie więcej informacji na ten temat (najważniejsze własności, metody i zdarzenia) zawiera Dodatek G.

Przeglądanie kontrolek

Na rysunku 8.1 przedstawiono okno Visual Basica *Toolbox* ze standardowymi kontrolkami Windows Forms. Ponieważ pozwala ono zarówno na dodawanie, jak i usuwanie kontrolek, na Twoim komputerze może zawierać nieco inny ich zestaw.



Rysunek 8.1. *Visual Basic udostępnia dużą liczbę standardowych kontrolek Windows Forms*

W poniższej tabeli znajdują się krótkie opisy kontrolek widocznych na rysunku 8.1. Zachowano kolejność kontrolek na rysunku (zaczynając od pierwszego rzędu na górze i patrząc od lewej do prawej).

Kontrolka	Przeznaczenie
Rząd 1	
`Pointer`	Jest to narzędzie wskaźnika, a nie kontrolka. Kliknięcie go spowoduje odznaczenie wszystkich kontrolek zaznaczonych na formularzu. Później będzie można zaznaczyć nowe.
`BackgroundWorker`	Wykonuje zadanie asynchronicznie, a kiedy skończy, powiadamia program główny.
`BindingNavigator`	Tworzy interfejs użytkownika do nawigacji przez źródło danych. Udostępnia na przykład przyciski pozwalające użytkownikowi poruszanie się do przodu i wstecz przez dane, a także dodawanie i usuwanie rekordów itd.
`BindingSource`	Hermetyzuje źródło danych formularza i udostępnia metody nawigacji po nich.
`Button`	Prosty przycisk. Kiedy użytkownik go kliknie, program wykona jakąś akcję.
`CheckBox`	Pole do zaznaczania i czyszczenia przez użytkownika.
`CheckedListBox`	Lista elementów z polami wyboru, które użytkownik może zaznaczać i czyścić.
`ColorDialog`	Umożliwia użytkownikowi wybranie standardowego lub niestandardowego koloru.
`ComboBox`	Pole tekstowe z listą rozwijaną, w którym użytkownik może wprowadzić lub wybrać wartość tekstową.
`ContextMenuStrip`	Menu pojawiające się po kliknięciu kontrolki prawym przyciskiem myszy. Należy ustawić własność `ContextMenuStrip` dla dowolnej kontrolki; reszta odbędzie się automatycznie.

Kontrolka	Przeznaczenie
Rząd 2	
`DataGridView`	Kontrolka siatki, która pozwala względnie łatwo wyświetlać duże ilości skomplikowanych danych z powiązaniami hierarchicznymi lub typu sieciowego.
`DataSet`	Znajdujący się w pamięci schowek danych z własnościami podobnymi do relacyjnej bazy danych. Przechowuje obiekty reprezentujące tabele, które zawierają wiersze i kolumny. Może też przedstawiać wiele koncepcji baz danych, na przykład indeksy i powiązania kluczy obcych.
`DateTimePicker`	Pozwala użytkownikowi wybrać datę i godzinę w jednym z kilku stylów.
`DirectoryEntry`	Reprezentuje węzeł w hierarchii Active Directory.
`DirectorySearcher`	Przeszukuje hierarchię Active Directory.
`DomainUpDown`	Pozwala użytkownikowi przewijać listę opcji do wyboru za pomocą klikania strzałek skierowanych w górę i w dół.
`ErrorProvider`	Wyświetla obok kontrolki wskaźnik błędu.
`EventLog`	Daje dostęp do dzienników zdarzeń systemu Windows.
`FileSystemWatcher`	Powiadamia aplikację o zmianach w katalogu lub pliku.
`FlowLayoutPanel`	Wyświetla swoje kontrolki w rzędach lub kolumnach. Na przykład ustawia je jedną obok drugiej, aż skończy się miejsce w poziomie. Wtedy przechodzi do nowego rzędu.
Rząd 3	
`FolderBrowserDialog`	Pozwala na wybór folderu.
`FontDialog`	Pozwala ustawić własności pisma (nazwę czcionki, rozmiar, pogrubienie itd.).
`GroupBox`	Dla przejrzystości grupuje powiązane ze sobą kontrolki. Dodatkowo definiuje domyślną grupę `RadioGroup` dla przycisków `RadioButton`, które zawiera.
`HelpProvider`	Wyświetla pomoc dla kontrolek, które ją posiadają, jeśli użytkownik aktywuje kontrolkę i naciśnie klawisz *F1*.
`HScrollBar`	Poziomy pasek przewijania.
`ImageList`	Zawiera zbiór obrazów, których mogą używać inne kontrolki. Na przykład te wyświetlane przez kontrolkę `TabControl` na jej kartach są przechowywane w kontrolce `ImageList`. W swoim kodzie również można pobierać obrazy na własny użytek z listy `ImageList`.
`Label`	Wyświetla tekst, którego użytkownik nie może zmodyfikować, zaznaczyć kliknięciem ani przeciągnąć.
`LinkLabel`	Wyświetla etykietę, której niektóre części mogą być hiperłączami. Kiedy użytkownik kliknie hiperłącze, program wykona jakieś działania.
`ListBox`	Wyświetla listę elementów do wyboru dla użytkownika. W zależności od tego, jakie ustawienia własności tej kontrolki wprowadzono, będzie można zaznaczyć jeden lub kilka elementów.
`ListView`	Wyświetla listę elementów w jednym z czterech możliwych widoków — `LargeIcon`, `SmallIcon`, `List` oraz `Details`.

Kontrolka	Przeznaczenie
Rząd 4	
`MaskedTextBox`	Pole tekstowe, które wymaga, by wprowadzane w nim dane miały określony format (na przykład numer telefonu lub kod pocztowy).
`MenuStrip`	Reprezentuje główne menu, podmenu i elementy menu formularza.
`MessageQueue`	Umożliwia komunikację pomiędzy różnymi aplikacjami.
`MonthCalendar`	Wyświetla kalendarz, który pozwala użytkownikowi wybrać daty.
`NotifyIcon`	Wyświetla ikonę w zasobniku systemowym lub polu stanu.
`NumericUpDown`	Pozwala użytkownikowi zmienić liczbę za pomocą strzałek.
`OpenFileDialog`	Pozwala użytkownikowi wybrać plik do otwarcia.
`PageSetupDialog`	Pozwala użytkownikowi ustawić własności drukowanych stron. Na przykład umożliwia określenie podajnika papieru drukarki, rozmiaru strony, marginesów i orientacji (pionowa lub pozioma).
`Panel`	Kontener kontrolek. Za pomocą własności `Anchor` i `Dock` można sprawić, że kontrolka ta będzie zmieniać rozmiar — zawarte w niej kontrolki również ulegną modyfikacji. Kontener ten może automatycznie wyświetlać paski przewijania i definiować grupę `RadioButton` dla wszystkich przycisków `RadioButton` w niej się znajdujących.
`PerformanceCounter`	Daje dostęp do liczników wydajności systemu Windows.
Rząd 5	
`PictureBox`	Wyświetla obraz. Dodatkowo udostępnia przydatną powierzchnię do rysowania.
`PrintDialog`	Wyświetla standardowe okno dialogowe drukowania. Użytkownik może wybrać w nim drukarkę, strony do wydrukowania, a także ustawić opcje drukarki.
`PrintDocument`	Reprezentuje dane, które mają zostać wysłane do drukarki. Program może wykorzystywać ten obiekt do drukowania i wyświetlania widoków druku.
`PrintPreviewControl`	Wyświetla podgląd wydruku w obrębie jednego z formularzy aplikacji.
`PrintPreviewDialog`	Wyświetla podgląd wydruku w standardowym oknie dialogowym.
`Process`	Pozwala programowi na interakcję z procesami oraz na uruchamianie i zatrzymywanie ich.
`ProgressBar`	Wyświetla szereg kolorowych pasków, które wskazują postęp długiej operacji.
`PropertyGrid`	Wyświetla informacje o obiekcie w formacie podobnym do używanego w oknie *Properties* w czasie projektowania.
`RadioButton`	Reprezentuje jedną z wzajemnie wykluczających się opcji. Kiedy użytkownik kliknie taki przycisk, Visual Basic odznaczy wszystkie pozostałe kontrolki `RadioButton` w tej samej grupie. Grupy są definiowane przez kontrolki `GroupBox` i `Panel` oraz klasę `Form`.
`ReportViewer`	Wyświetla raport wygenerowany lokalnie lub zdalnie na serwerze raportowym. Zdalne przetwarzanie wymaga serwera raportującego Microsoft SQL Server 2005.

Kontrolka	Przeznaczenie
Rząd 6	
`RichTextBox`	Pole tekstowe, które obsługuje rozszerzenia tekstu wzbogaconego.
`SaveFileDialog`	Pozwala użytkownikowi nazwać plik, w którym program ma zapisać dane.
`SerialPort`	Reprezentuje port szeregowy oraz udostępnia metody do jego kontroli, odczytu z niego i zapisu do niego.
`ServiceController`	Reprezentuje usługę Windows i pozwala zarządzać usługami.
`SplitContainer`	Pozwala użytkownikowi przeciągać pionową lub poziomą linię, dzięki czemu można podzielić dostępną przestrzeń na dwie części.
`Splitter`	Udostępnia linię, którą użytkownik może przeciągać, aby podzielić dostępną przestrzeń pomiędzy dwie kontrolki. O aranżacji i rozmiarze tych ostatnich decydują własności `Dock`, kolejność na stosie i kontrolka `Splitter`. Kontrolka `SplitterContainer` automatycznie oddziela dwa kontenery, dzięki czemu zazwyczaj jest łatwiejsza w użyciu.
`StatusStrip`	Tworzy obszar (zazwyczaj w dolnej części formularza), w którym aplikacja może wyświetlać komunikaty o stanie, małe obrazki i inne wskaźniki stanu aplikacji.
`TabControl`	Wyświetla szereg kart związanych ze stroną, zawierających własne kontrolki. Użytkownik klika jedną z nich, aby wyświetlić odpowiednią stronę.
`TableLayoutPanel`	Wyświetla kontrolki na siatce.
`TextBox`	Wyświetla tekst, który użytkownik może zmienić.
Rząd 7	
`Timer`	Okresowo wyzwala zdarzenie. Program może w odpowiedzi na to zdarzenie podjąć jakieś działania.
`ToolStrip`	Wyświetla zbiór przycisków, menu rozwijanych i innych narzędzi, które pozwalają użytkownikowi kontrolować aplikację.
`ToolStripContainer`	Kontener, który pozwala kontrolce `ToolStrip` zadokować się przy niektórych lub wszystkich jego krawędziach. Kontrolkę `ToolStripContainer` można zadokować przy formularzu, aby umożliwić użytkownikowi zadokowanie kontrolki `ToolStrip` przy każdej krawędzi tego formularza.
`ToolTip`	Wyświetli chmurkę, jeśli użytkownik najedzie na odpowiednią kontrolkę.
`TrackBar`	Pozwala użytkownikowi przeciągać wskaźnik wzdłuż paska w celu wybrania wartości liczbowej.
`TreeView`	Wyświetla dane hierarchiczne w graficznej formie, która przypomina drzewo.
`VScrollBar`	Pionowy pasek przewijania.
`WebBrowser`	Przeglądarka internetowa w kontrolce. Tę ostatnią można umieścić na formularzu i za pomocą jej metod przejść do strony internetowej. Kontrolka ta wyświetla wyniki dokładnie tak samo, jak gdyby użytkownik używał samodzielnej przeglądarki. Jednym z pożytecznych jej zastosowań jest wyświetlanie pomocy internetowej.

Przykładowy program o nazwie *UseToolStripContainer*, który można pobrać z serwera FTP wydawnictwa Helion (*ftp://ftp.helion.pl/przyklady/vb28wp.zip*), zawiera kontrolkę `ToolStripContainer` z dwiema kontrolkami `ToolStrip`. Można je przeciągać i dokować przy krawędziach kontrolki `ToolStripContainer`.

Szczegółowe opisy kontrolek znajdują się w Dodatku G — „Komponenty i kontrolki Windows Forms".

Wybieranie kontrolek

Zapamiętanie wszystkich szczegółów dotyczących wymienionych kontrolek jest bardzo trudne. Przy takiej liczbie narzędzi do wyboru czasami ciężko zdecydować, które z nich najlepiej się nada do wykonania określonego zadania.

Aby uprościć kod obsługujący błędy, powinno się wybierać kontrolki o jak najwęższym zakresie funkcjonalności, które mogą wykonać dane zadanie. Bardziej restrykcyjne zmniejszają prawdopodobieństwo podania przez użytkownika nieprawidłowych danych.

Restrykcyjne kontrolki podnoszą też poziom bezpieczeństwa aplikacji. Wyświetlając listę opcji do wyboru, zamiast pozwalać użytkownikom wpisać, co im się podoba, program może chronić sam siebie przed problemami. Na przykład dwoma najczęściej spotykanymi rodzajami ataków na strony internetowe jest przepełnienie bufora, które polega na wpisaniu w polu tekstowym znacznie więcej tekstu niż potrzeba, a także SQL injection — wpisanie w polu tekstowym specjalnie spreparowanego kodu, mającego na celu zmianę działania bazy danych. Podanie zestawu opcji zamiast zezwolenia użytkownikowi na wpisanie, co chce, oddala ryzyko tych dwóch problemów.

Załóżmy na przykład, że użytkownik musi wybrać jeden z rozmiarów — Small, Medium lub Large. Można pozwolić mu wprowadzić wybraną opcję do pola tekstowego, ale w takim przypadku mógłby napisać na przykład `Nutria`. Program musiałby sprawdzić, czy użytkownik wpisał złe słowo, a jeśli tak — wyświetlić komunikat o błędzie. Możliwe też, że trzeba by zająć cenne miejsce na ekranie w celu pokazania listy opcji do wyboru.

Lepszym pomysłem byłoby użycie grupy kontrolek `RadioButton` lub kontrolki `ComboBox` z własnością `DropDownStyle` ustawioną na `DropDownList`. Dzięki temu użytkownik będzie widział listę pozycji do wyboru i mógł wybrać tylko jedną z prawidłowych opcji. Jeśli program inicjuje kontrolki jakimiś wartościami, zamiast pozostawiać je niezdefiniowane, wie, że zawsze zaznaczona jest jakaś poprawna opcja.

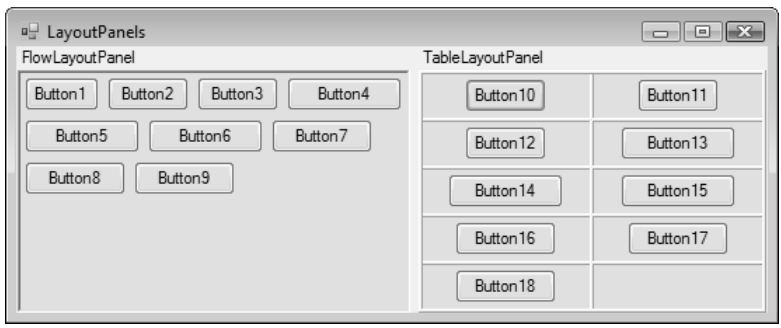
W kolejnych podrozdziałach umieszczono opisy różnych kategorii kontrolek. Znajdują się tam też wskazówki, kiedy ich używać.

Kontrolki kontenery

Te kontrolki zawierają, grupują i pomagają ustawiać inne kontrolki. Zaliczają się do nich `FlowLayoutPanel`, `TableLayoutPanel`, `GroupBox`, `Panel`, `TabControl` oraz `SplitContainer`.

Kontrolka `FlowLayoutPanel` ustawia znajdujące się w niej kontrolki w rzędach i kolumnach. Jeśli na przykład jej własność `FlowDirection` jest określona jako `LeftToRight`, ustawia ona swoje kontrolki w rzędach od lewej do prawej. Kiedy miejsce w jednym rzędzie się skończy, dana kontrolka przejdzie do kolejnego. Kontrolka `FlowLayoutPanel` jest szczególnie przydatna do tworzenia zestawów narzędzi. Warto jej używać również w sytuacjach, gdy trzeba wyświetlić jak najwięcej kontrolek jednocześnie, a ich dokładne ustawienie nie jest zbyt ważne.

Kontrolka `TableLayoutPanel` wyświetla swoją zawartość na siatce. Wszystkie komórki w jednym wierszu mają tę samą wysokość, a wszystkie komórki w jednej kolumnie — jednakową szerokość. Natomiast kontrolka `FlowLayoutPanel` umieszcza kontrolki jedną obok drugiej, aż zapełni wiersz — wtedy przechodzi do nowego. Na rysunku 8.2 przedstawiono przykładowy program o nazwie *LayoutPanels* — można go pobrać z serwera FTP wydawnictwa Helion.



Rysunek 8.2. *Kontrolka FlowLayoutPanel umieszcza kontrolki blisko siebie, zaś TableLayoutPanel ustawia je na siatce*

Kontrolka `GroupBox` znajduje zastosowanie w grupowaniu powiązanych ze sobą kontrolek lub kontrolek `RadioButton` w grupie (kontrolka `RadioButton` została opisana w podrozdziale „Wybór opcji”, który znajduje się w dalszej części tego rozdziału). Tworzy widoczne obramowanie i podpis, dzięki czemu użytkownik łatwiej domyśli się, o co chodzi w skomplikowanym formularzu.

Podstawową zasadą przy projektowaniu interfejsów użytkownika jest to, że można w jednym czasie ewaluować od pięciu do dziewięciu elementów. Lista pięciu czy sześciu opcji jest do zaakceptowania, ale spis, który zawiera kilkadziesiąt pozycji, może wprowadzać zamieszanie. Jeśli umieści się opcje w kategoriach oddzielonych od siebie wizualnie za pomocą kontrolki `GroupBox`, można sprawić, że interfejs będzie znacznie łatwiejszy do zrozumienia. Zamiast próbować zająć się kilkudziesięcioma opcjami do wyboru naraz, użytkownik może rozbić problem na mniejsze części i zająć się każdą grupą osobno.

Kontrolka Panel również może zawierać kontrolki RadioButton w grupie. Nie wyświetla jednak obramowania, przez co trzeba w jakiś inny sposób zaznaczyć, że dane przyciski należą do jednej grupy. Można na przykład użyć kilku kontrolek Panel w jednym rzędzie, z których każdy zawierałby kolumnę kontrolek RadioButton. Wtedy użytkownik mógłby wybrać po jednej opcji z każdej kolumny.

Jedną z najważniejszych własności kontrolki Panel jest jej zdolność do automatycznego wyświetlania pasków przewijania. Jeśli własność AutoScroll kontrolki Panel jest ustawiona na wartość True, a w kontrolce tej znajdzie się więcej treści, niż może ona pomieścić, automatycznie pojawią się paski przewijania, aby użytkownik mógł zobaczyć całą jej zawartość. Jednak przewijanie w tę i z powrotem może być uciążliwe, dlatego nie jest to najlepszy wybór dla danych, które trzeba często przeglądać. Jeśli użytkownik musi ciągle przechodzić pomiędzy różnymi kontrolkami w przewijanym panelu, lepszym rozwiązaniem byłoby dla niego użycie kontrolki TabControl.

Kontrolka TabControl grupuje dane na kartach. Użytkownik może szybko przechodzić pomiędzy nimi. Kontrolka ta w razie potrzeby wyświetli paski przewijania, chociaż nie jest to zbyt zgrabne. TabControl dobrze sprawdza się w przypadku danych dających podzielić się w naturalny sposób na grupy, które można umieścić na poszczególnych kartach. Nie jest jednak dobrym rozwiązaniem, gdy użytkownik musi porównywać dane znajdujące się na jednej karcie z zawartymi w innej.

Kontrolka SplitContainer pozwala użytkownikowi podzielić obszar ekranu na dwa przystające do siebie regiony. Zawiera dwie kontrolki Panel, w których można umieszczać swoje własne kontrolki. Kiedy użytkownik przeciąga linię dzielącą oba panele, kontrolka odpowiednio zmienia rozmiary obu. Można ustawić własność AutoScroll tych paneli na True, aby w razie potrzeby automatycznie wyświetlały paski przewijania.

Kontrolka SplitContainer okazuje się przydatna, gdy formularz jest za mały, by pomieścić wszystkie dane, które program musi wyświetlić. Dzięki niej użytkownik może powiększyć jedną część ekranu kosztem drugiej. Jest to szczególnie przydatne, gdy w celu porównania wartości z tych dwóch regionów trzeba je jednocześnie widzieć.

Mimo że kontenery SplitContainer można zagnieżdżać w innych kontenerach tego samego typu, najłatwiej ich używać, kiedy oddzielają tylko dwa regiony. Duże grupy kontrolek SplitContainer, które oddzielają wiele obszarów, są zazwyczaj zatłoczone i mało czytelne.

Przykładowy program o nazwie *UseSplitter*, który można pobrać z serwera FTP wydawnictwa Helion, dzieli swój formularz na dwa regiony za pomocą kontrolki SplitContainer. Ta ostatnia zawiera dwie kontrolki: Panel i Splitter, co w sumie daje formularz podzielony na dwa regiony. Dzięki temu, że za pomocą kontrolki Splitter nie trzeba ustawiać własności Dock ani kolejności na stosie, jak w kontrolce Splitter, kontrolka SplitContainer jest łatwiejsza w użyciu.

Opisane tu kontenery pomagają odpowiednio ustawiać zawarte w nich kontrolki. Własności Anchor i Dock wszystkich kontrolek znajdujących się wewnątrz tych kontenerów działają względem nich. Wyobraźmy sobie na przykład, że umieszczono serię przycisków z własnością Anchor = Top, Left, Right wewnątrz kontrolki SplitContainer, dzięki czemu mają one taką samą szerokość, jaką posiada zawierający je panel. Kiedy użytkownik przeciągnie linię dzielącą, przyciski automatycznie zmienią rozmiary, aby pasować do otaczającego je panelu.

Wybór opcji

Kontrolki opcji do wyboru pozwalają użytkownikowi wybrać odpowiednie wartości. Właściwie stosowane, pozwalają zminimalizować ryzyko wprowadzenia niepoprawnych danych. Dzięki temu potrzebna jest mniejsza ilość kodu obsługującego błędy.

Do kontrolek tych zaliczają się: `CheckBox`, `CheckedListBox`, `ComboBox`, `ListBox`, `RadioButton`, `DateTimePicker`, `MonthCalendar`, `DomainUpDown`, `NumericUpDown`, `TrackBar`, `HScrollBar` oraz `VScrollBar`.

Kontrolka `CheckBox` pozwala na zaznaczenie wybranej opcji bez względu na inne. Aby pozwolić na wybranie tylko jednej pozycji z kilku, należy użyć kontrolki `RadioButton`. Jeśli formularz wymaga więcej niż pięciu do siedmiu kontrolek `CheckBox`, warto rozważyć możliwość użycia w zamian kontrolki `CheckedListBox`.

Kontrolka `CheckedListBox` pozwala użytkownikowi wybrać spośród kilku niezależnych od siebie opcji. Zazwyczaj ma postać listy kontrolek `CheckBox` z paskami przewijania, które pokazują się w razie potrzeby.

Kontrolka `ComboBox` pozwala użytkownikowi na zdecydowanie się na jedną opcję. Jest ona szczególnie przydatna, gdy jej własność `DropDownStyle` jest ustawiona na `DropDownList`, ponieważ dzięki niej można wybrać daną pozycję z listy rozwijanej w dół. Aby umożliwić użytkownikowi zdecydowanie się na jakąś wartość lub wprowadzenie takiej, której nie ma na liście, należy ustawić własność `DropDownStyle` na `Simple` lub `DropDown`. Kontrolka ta spełnia podobną rolę do prostej kontrolki `ListBox`, ale zajmuje mniej miejsca.

Kontrolka `ListBox` wyświetla listę opcji do wyboru. Można ją tak skonfigurować, aby użytkownik mógł zdecydować się na jedną lub więcej pozycji. Kontrolka `ListBox` zajmuje więcej miejsca od `ComboBox`, ale może być łatwiejsza w użyciu, jeśli jej lista jest bardzo długa. Oczywiście gdy użytkownik ma do wyboru zbyt wiele opcji, łatwo może przez przypadek odznaczyć wszystkie zaznaczone wcześniej pozycje i zaznaczyć kolejny element. Aby ułatwić życie osobie korzystającej z komputera, warto zastanowić się nad użyciem w zamian kontrolki `CheckedListBox`, która nie jest obciążona tym problemem.

Kontrolka `RadioButton` pozwala wybrać jedną opcję ze zbioru. Na przykład trzy kontrolki tego typu mogą reprezentować opcje `Small`, `Medium` i `Large`. Jeśli użytkownik zaznaczy jedną z nich, pozostałe zostaną automatycznie odznaczone. Ten typ kontrolki jest przydatny, gdy lista opcji nie jest duża. Przynosi też korzyść pokazania użytkownikowi wszystkich opcji do wyboru naraz. Jeśli lista opcji jest długa, należy rozważyć wybór kontrolki `ListBox` lub `ComboBox`.

Kontrolki `DateTimePicker` i `MonthCalendar` pozwalają użytkownikowi wybierać daty i godziny. Zapewniają poprawność wprowadzonych danych, dzięki czemu są z reguły lepsze od innych kontrolek, które służą do wyboru daty i godziny. Jeśli na przykład pozwoli się użytkownikowi wprowadzić datę w kontrolce `TextBox` w postaci liczbowej, trzeba będzie dodatkowo napisać kod, który sprawdzi, czy nie wpisał on czegoś w rodzaju „29 luty 2008”.

Kontrolki `DomainUpDown` i `NumericUpDown` pozwalają przewijać listę opcji. Jeśli spis ten jest krótki, lepszym rozwiązaniem będzie wybranie kontrolki `ListBox` lub `ComboBox`. Jednak `DomainUpDown` i `NumericUpDown` zajmują bardzo mało miejsca, a więc mogą być przydatne na mocno zatło-

czonych formularzach. Gdy użytkownik kliknie jedną ze strzałek kontrolki, będzie mógł bardzo szybko przewinąć wartości. Zatem kontrolki te mogą być przydatne w przypadku bardzo długich list opcji.

Kontrolka `TrackBar` pozwala na wybranie wartości liczbowej za pomocą przeciąganego wskaźnika. Zajmuje ona znacznie więcej miejsca od kontrolki `NumericUpDown`, ale jest też bardziej intuicyjna. Ponadto wymaga sporej zręczności przy dużym przedziale dozwolonych wartości.

`HScrollBar` i `VScrollBar` pozwalają użytkownikowi na wybór opcji liczbowej za pomocą przeciągania „kciuka", podobnie jak kontrolka `TrackBar`. Wszystkie trzy mają nawet podobne własności. Główna różnica przejawia się w ich wyglądzie. Kontrolki `ScrollBar` są bardziej elastyczne, jeśli chodzi o rozmiary (`TrackBar` ma ustaloną wysokość w stosunku do szerokości), więc niektórym użytkownikom mogą wydawać się bardziej eleganckie. Jednak osoby korzystające z komputera są przyzwyczajone do zwykłego przewijania obszarów formularza, a więc użycie ich jako pasków do wybierania liczb może wywoływać zdziwienie.

Wprowadzanie danych

Czasami kontrolki opisane w poprzednim podrozdziale okazują się niepraktyczne. Na przykład nie da się wprowadzić rozsądnie długiej historii pracy lub komentarza za pomocą `ComboBox` lub `RadioButton`.

Kontrolki `RichTextBox`, `TextBox` i `MaskedTextBox` pozwalają na wpisywanie tekstu z niewieloma ograniczeniami. Są one najbardziej przydatne, gdy użytkownik musi wprowadzić duże ilości tekstu, który nie wymaga walidacji.

Kontrolka `TextBox` jest mniej skomplikowana i łatwiejsza w użyciu niż `RichTextBox`. Dlatego jeśli nie są konieczne dodatkowe funkcje tej drugiej, można używać pierwszej z wymienionych kontrolek. Jeżeli dodatkowe funkcje (takie jak możliwość wyboru kroju pisma, wcięcia, wyrównanie akapitów, indeks górny i dolny, różne kolory, cofanie i ponawianie więcej niż jednego kroku itd.) są niezbędne, należy użyć `RichTextBox`.

`MaskedTextBox` jest kontrolką `TextBox`, która wymaga, aby wprowadzane dane były w określonym formacie. Może na przykład wymuszać podawanie numeru telefonu następująco: 234-567-8901. Jest przydatna tylko w przypadku krótkich pól ze ściśle ustalonym formatem danych. Redukuje natomiast ryzyko wprowadzenia przez użytkownika niepoprawnych informacji.

Wyświetlanie danych

Kontrolki opisane w tym podrozdziale wyświetlają dane użytkownikowi. Należą do nich kontrolki `Label`, `DataGridView`, `ListView`, `TreeView` oraz `PropertyGrid`.

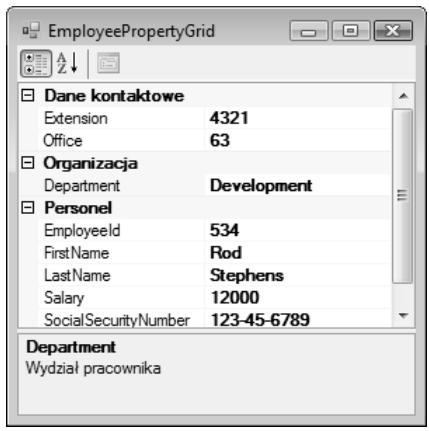
Pierwsza z nich wyświetla tekst, który użytkownik może tylko przeczytać — nie da się go ani zaznaczyć, ani zmodyfikować. Ponieważ nie można zaznaczyć tekstu, nie da się go również skopiować do schowka. Jeśli tekst ten zawiera wartość, którą użytkownik mógłby chcieć skopiować i wkleić gdzieś indziej (na przykład numer seryjny, numer telefonu, adres e-mail, adres URL itd.), można użyć kontrolki `TextBox` z własnością `ReadOnly` ustawioną na `True`.

Kontrolka `DataGridView` służy do wyświetlania danych tabelarycznych. Potrafi także wyświetlać po kilka tabel połączonych powiązaniami typu element główny-szczegół, które użytkownik może łatwo i szybko przeszukiwać. Istnieje również możliwość takiego skonfigurowania tej kontrolki, by można było aktualizować znajdujące się w niej dane.

Kontrolka `ListView` wyświetla dane, które naturalnie wyglądają jak zbiór ikon lub lista wartości z kolumnami zawierającymi dodatkowe szczegóły. Można posortować te dane według kolumny z elementami lub szczegółami, ale wymaga to trochę pracy.

Kontrolka `TreeView` wyświetla dane hierarchiczne w formie drzewa — podobnego do tego, które zawiera katalogi w Explorerze Windows. Można pozwolić użytkownikowi na edytowanie etykiet jego węzłów.

Kontrolka `PropertyGrid` wyświetla informacje o obiekcie w formacie podobnym do używanego przez okno `Properties` w trybie projektowania. Pozwala ona użytkownikowi zorganizować własności alfabetycznie lub według kategorii. Umożliwia też edytowanie ich wartości. Na rysunku 8.3 przedstawiono przykładowy program o nazwie *EmployeePropertyGrid* (można go pobrać z serwera FTP wydawnictwa Helion), który wyświetla informacje o obiekcie `Employee` w kontrolce `PropertyGrid`.



Rysunek 8.3. *Kontrolka PropertyGrid wyświetla własności obiektu*

Informowanie użytkownika

W tym podrozdziale opiszę kontrolki dostarczające informacje użytkownikowi. Zaliczają się do nich `ToolTip`, `HelpProvider`, `ErrorProvider`, `NotifyIcon`, `StatusStrip` oraz `ProgressBar`. Ich głównym celem jest poinformowanie użytkownika, co się dzieje, bez zabraniania mu kontynuowania wykonywania innych czynności. Na przykład kontrolka `ErrorProvider` oznacza pole jako niepoprawne, ale nie blokuje wprowadzania danych w innych miejscach.

`ToolTip` wyświetla krótką informację na temat przeznaczenia kontrolki, na którą użytkownik najechał kursorem. `HelpProvider` dostarcza bardziej szczegółowe informacje o przeznaczeniu danej kontrolki, kiedy użytkownik aktywuje ją i naciśnie klawisz *F1*. Wysokiej jakości aplikacje dostarczają zarówno chmurki (ang. *tooltip*), jak i pomoc w przycisku *F1* dla każdej kontrolki, której funkcja może nie być oczywista. Opcje te nie są zbyt natrętne

— i pojawiają się tylko wtedy, gdy użytkownik ich potrzebuje. Dlatego lepiej jest przesadzić ze zbyt dużą ilością pomocy, niż dostarczyć jej za mało.

`ErrorProvider` oznacza kontrolki zawierające niepoprawne dane. Lepiej jest używać kontrolek wyboru opcji, które nie pozwalają na wprowadzanie błędnych informacji, ta jednak jest przydatna, kiedy okazuje się to niemożliwe.

Kontrolka `NotifyIcon` wyświetla niewielką ikonę w obszarze powiadomień paska zadań, która informuje użytkownika o stanie aplikacji. Jest szczególnie przydatna w przypadku programów działających w tle, którym nie trzeba ciągle poświęcać uwagi. Jeśli aplikacja wymaga natychmiastowej reakcji ze strony użytkownika, powinna wyświetlać okno dialogowe lub ramkę z komunikatem, zamiast zdawać się na kontrolkę `NotifyIcon`.

> *Obszar powiadomień paska zadań, zwany również zasobnikiem systemu Windows, to niewielkie miejsce na pasku zadań. Po prawej stronie wyświetla godzinę i datę oraz ikony, które oznaczają status różnych działających aplikacji.*

Kontrolka `StatusStrip` wyświetla obszar (zazwyczaj w dolnej części formularza), w którym program może podać użytkownikowi pewne informacje na temat swojego stanu. Może on mieć postać niewielkiej ikony lub krótkiego komunikatu tekstowego. Przekazuje znacznie więcej informacji niż kontrolka `NotifyIcon`, ale jest widoczny tylko wtedy, gdy widać formularz.

Kontrolka `ProgressBar` pokazuje, jaka część długiego zadania została ukończona. Zazwyczaj zadanie to jest wykonywane synchronicznie, przez co użytkownik nie ma nic więcej do roboty, jak tylko wpatrywać się w pasek postępu. Kontrolka `ProgressBar` pozwala zorientować się, że operacja nie utknęła w miejscu.

Inicjowanie akcji

Kontrolki każdego rodzaju reagują na zdarzenia, a więc każda z nich może inicjować akcję. Niemniej jednak użytkownicy oczekują wykonywania różnych czynności tylko od niektórych kontrolek. Na przykład naciśnięcie przycisku ma włączać jakąś akcję, ale kliknięcie etykiety lub pola wyboru nie powinno uruchamiać długich procesów.

Aby uniknąć zamieszania, należy do uruchamiania akcji korzystać tylko z tych kontrolek, które są w tym celu używane najczęściej. Należą do nich: `Button`, `MenuStrip`, `ContextMenuStrip`, `ToolStrip`, `LinkLabel`, `TrackBar`, `HScrollBar`, `VScrollBar` oraz `Timer`. Wymienione kontrolki — z wyjątkiem `Timer` — pozwalają użytkownikowi zainicjować akcję.

Wszystkie te kontrolki komunikują się z programem za pomocą procedur obsługi zdarzeń. Na przykład procedura obsługi zdarzenia `Click` kontrolki `Button` w normalnej sytuacji zmusza aplikację do wykonania jakichś czynności, kiedy użytkownik kliknie przycisk.

Inne kontrolki również dysponują zdarzeniami, które mogą inicjować akcje. Na przykład `CheckBox` udostępnia zdarzenia `CheckChanged` i `Click`, których można użyć do wykonania jakichś działań. Jeśli przechwyci się odpowiednie zdarzenia, będzie można użyć prawie każdej kontrolki do zainicjowania określonej akcji. Ponieważ głównym przeznaczeniem tych narzędzi nie jest wykonywanie kodu, nie zostały one wymienione w tym podrozdziale.

Kontrolka `Button` pozwala użytkownikowi zmusić program do wykonania wybranej funkcji. Przycisk jest zazwyczaj cały czas widoczny na formularzu, a więc najczęściej przydaje się, gdy trzeba często wykonywać jakąś akcję lub jest ona częścią tego, co stanowi główne zadanie aplikacji. Dla rzadziej wykonywanych akcji należy używać kontrolek `MenuStrip` i `ContextMenuStrip`.

Elementy w kontrolce `MenuStrip` również pozwalają użytkownikowi zmusić program do wykonania wybranych akcji. W jej przypadku konieczne jest wykonanie większej liczby czynności, niż gdy używa się przycisku, ponieważ trzeba otworzyć menu, zaleźć interesującą opcję i kliknąć ją. Należy jednak pamiętać o tym, że menu zajmują mniej miejsca na formularzu niż przyciski. Do często używanych opcji znajdujących się w menu można także przypisać skróty klawiszowe (typu *F5* lub *Ctrl+S*) — ich uruchamianie jest nawet łatwiejsze niż używanie przycisków.

`ContextMenuStrip` posiada takie same wady i zalety, jakie ma kontrolka `MenuStrip`. Jest ona jednak dostępna tylko z danych kontrolek na formularzu, a więc przydaje się dla poleceń mających zastosowanie jedynie w określonych warunkach. Na przykład *Zapisz* ma zastosowanie do wszystkich danych załadowanych przez program, a więc powinno znajdować się w kontrolce `MenuStrip`. Z polecenia usuwającego określony obiekt z rysunku da się skorzystać tylko w stosunku do tego obiektu. Dzięki umieszczeniu tej opcji w kontrolce `ContextMenuStrip`, która jest związana z obiektem, pozostaje ona ukryta, gdy użytkownik pracuje nad czymś innym. Dodatkowo sprawia, że związek pomiędzy akcją (usunięcie) a obiektem jest oczywisty — zarówno dla programu, jak i użytkownika.

Kontrolka `ToolStrip` łączy niektóre najlepsze cechy menu i przycisków. Wyświetla szereg przycisków, dzięki czemu można ich użyć bez wchodzenia do menu. Są one małe, a znajdują się w górnej części formularza, więc nie zajmują tyle miejsca co duże przyciski.

Zwykle przyciski lub przyciski kontrolki `ToolStrip`, które znajdują się na formularzu, reprezentują często używane polecenia z menu. Te ostatnie można uruchamiać za pomocą skrótów klawiszowych lub przycisków, jeśli nie pamięta się skrótów.

`LinkLabel` wyświetla tekst — podobnie jak kontrolka `Label`. Dodatkowo pokazuje niebieski tekst z podkreśleniem, specjalny kursor, kiedy użytkownik najeżdża na niego, a także uruchamia zdarzenie, gdy kliknie się ten tekst. Dzięki temu z kontrolki tej warto korzystać, gdy kliknięcie tekstu ma sprawiać, że wykonana zostanie jakaś akcja. Na przykład na stronie internetowej kliknięcie odnośnika spowoduje przejście do wskazywanej przez niego strony.

`TrackBar`, `HScrollBar` i `VScrollBar` pozwalają użytkownikowi wybrać wartość liczbową za pomocą przeciągania rączki. Jak napisałem wcześniej w podrozdziale „Wybór opcji”, dzięki tym kontrolkom można wybierać wartości liczbowe. Istnieje też możliwość używania ich do interaktywnego wykonywania pewnych akcji. Na przykład z pasków przewijania często korzysta się do przewijania części formularza. Bardziej ogólnie: służą do zmuszania programu do podejmowania działań na podstawie jakiejś nowej wartości. Na przykład za pomocą paska przewijania można pozwolić użytkownikowi wybrać komponenty czerwieni, zieleni i niebieskiego dla obrazu. W chwili zmiany przez niego wartości paska przewijania program aktualizuje kolory grafiki.

Kontrolka `Timer` wyzwala określoną akcję w równych odstępach czasu. Kiedy uruchamia ona swoje zdarzenie, program wykonuje akcję.

Wyświetlanie grafiki

Kontrolki opisane w tym podrozdziale służą do wyświetlania obrazów graficznych na ekranie lub prezentacji ich na wydruku. Są to `Form`, `PictureBox`, `PrintPreviewControl`, `PrintDocument` oraz `PrintPreviewDialog`.

Kontrolka `Form` (która także może wyświetlać obrazy graficzne) umożliwia rysowanie, ale z reguły lepiej jest rysować w `PictureBox` niż w samym formularzu. Dzięki temu łatwiej jest przenieść grafikę przy zmianie projektu formularza. Jeśli na przykład dojdziesz do wniosku, że obraz może być za duży, możesz z łatwością przenieść kontrolkę `PictureBox` do przewijanej kontrolki `Panel`. Znacznie trudniej byłoby przepisać kod, który miałby przesunąć rysunek z kontrolki `Form` do `PictureBox`.

Kontrolka `PrintPreviewControl` wyświetla podgląd wydruku obiektu `PrintDocument`. Program reaguje na zdarzenia wyzwalane przez obiekt `PrintDocument`. `PrintPreviewDocument` wyświetla wyniki w kontrolce znajdującej się na jednym z formularzy programu.

`PrintPreviewDialog` wyświetla obrazy z obiektu `PrintDocument` — podobnie jak kontrolka `PrintPreviewControl` — ale posiada własne okno dialogowe. Jeśli nie ma konieczności zaprojektowania w specjalny sposób podglądu wydruku, łatwiej jest użyć `PrintPreviewDialog`, niż budować własne okno dialogowe za pomocą kontrolki `PrintPreviewControl`. `PrintPreviewDialog` udostępnia wiele funkcji, które pozwalają użytkownikowi na powiększenie, przewijanie i zmienianie stron podglądanego dokumentu. Samodzielne zaimplementowanie ich wszystkich wymagałoby bardzo dużo pracy.

Wyświetlanie okien dialogowych

Visual Basic oferuje szeroki wybór okien dialogowych, które pozwalają użytkownikowi na dokonywanie standardowych wyborów. Użycie właściwego z nich jest zazwyczaj łatwe, ponieważ każde ma bardzo konkretne przeznaczenie. Poniższa tabela zawiera listę tych okien dialogowych i krótki opis ich funkcji.

Okno dialogowe	Przeznaczenie
`ColorDialog`	Wybór koloru.
`FolderBrowserDialog`	Wybór folderu (katalogu).
`FontDialog`	Wybór czcionki.
`OpenFileDialog`	Wybór pliku do otwarcia.
`PageSetupDialog`	Ustawienia strony do druku.
`PrintDialog`	Wydruk dokumentu.
`PrintPreviewDialog`	Wyświetlenie podglądu wydruku.
`SaveFileDialog`	Wybór pliku do zapisania.

Okna te demonstruje program *UseDialogs*, który można pobrać z serwera FTP wydawnictwa Helion.

Wspieranie innych kontrolek

Wiele kontrolek Visual Basica wymaga wsparcia ze strony innych tego typu narzędzi. Dwie kontrolki najczęściej używane przez inne to `ImageList` i `PrintDocument`. Zaliczają się do nich również `DataConnector` i `DataNavigator`.

`ImageList` przechowuje obrazy, które wyświetlają inne kontrolki. Kod programu może także pobierać z niej obrazy i używać ich w dowolny sposób.

`PrintDocument` wspomaga drukowanie i podglądanie wydruku. Generuje grafikę wysyłaną do drukarki albo kontrolki `PrintPreviewDialog` lub `PrintPreviewControl`.

`DataConnector` tworzy łącze pomiędzy źródłem danych a kontrolkami z nią związanymi. Za pomocą jej metod program może nawigować po danych, sortować je, filtrować oraz aktualizować. Ponadto kontrolka ta odpowiednio aktualizuje związane z nią kontrolki.

Kontrolka `DataNavigator` udostępnia metody do nawigowania po źródłach danych, takich jak `DataConnector`.

Kontrolki niestandardowe

W Visual Basicu jest dostępnych mnóstwo różnych kontrolek, które są gotowe do użycia. Nie są to jednak wszystkie te, z których można korzystać. Poprzez kliknięcie prawym przyciskiem myszy w oknie *Toolbox* i wybranie opcji *Choose Items* można uzyskać dostęp do ogromnej listy dostępnych w systemie komponentów .NET i COM.

Można także używać kontrolek utworzonych przez inne firmy, które oferują je w sprzedaży lub udostępniają bezpłatnie w sieci. Wiele z nich wykonuje specjalistyczne zadania, na przykład generuje kody kreskowe, tworzy kształty formularzy, zniekształca obrazy czy tworzy specjalne efekty graficzne.

Są też kontrolki rozszerzające funkcjonalność standardowych tego typu narzędzi. Istnieje kilka kontrolek rysujących dwu- i trójwymiarowe wykresy i grafy. Inne oferują bardziej rozbudowane usługi raportowania od narzędzi dostępnych w Visual Studio.

Gdy wpisze się frazę `windows forms controls` w dowolnej większej wyszukiwarce internetowej, będzie można znaleźć mnóstwo witryn internetowych, które oferują kontrolki do pobrania bezpłatnie lub za pewną kwotę. Oto kilka adresów, które warto odwiedzić:

- MVPs.org (*www.mvps.org*) — witryna prowadząca do zasobów udostępnianych przez ludzi związanych z programem Microsoftu Most Valuable Professional (MVP). Projekt Common Controls Replacement Project (*ccrp.mvps.org*) oferuje kontrolki duplikujące i rozszerzające standardowe kontrolki programuVisual Basic 6. Strona ta nie jest już rozwijana, ale niektóre ze starych kontrolek szóstej wersji Visual Basica mogą stać się inspiracją do budowy własnych nowych narzędzi. Witryna *MVPs.org* jest ponadto dobrym ogólnym źródłem danych.

- Windows Forms .NET (*windowsclient.net*) — oficjalna strona Microsoftu, założona dla społeczności zainteresowanej Windows Forms .NET.
- ASP.NET (*www.asp.net*) — oficjalna społeczność ASP.NET Microsoftu.
- Download.com (*www.download.com*).
- Shareware.com (*www.shareware.com*).
- Shareware Connection (*www.sharewareconnection.com*).

Wymienione wyżej witryny są dobre do rozpoczęcia poszukiwań, ale nie zamykają listy. Kontrolki do pobrania można znaleźć w setkach (o ile nie tysiącach) stron internetowych.

Gdy pobiera się niestandardowe kontrolki i produkty, zawsze należy zachowywać pewną powściągliwość. Dodanie każdego tego typu narzędzia do projektu powoduje uzależnienie tego ostatniego od niego. Gdy przenosi się efekt pracy do nowszej wersji Visual Basica, należy się upewnić, że tam również dana kontrolka będzie działała. Jeśli powstanie nowa wersja tego narzędzia, trzeba będzie sprawdzić, czy będzie działała w używanej edycji programu. Jeśli nie, można utknąć ze starą — już niewspieraną wersją kontrolki.

Problem staje się jeszcze poważniejszy, gdy kontrolki i narzędzia współpracują ze sobą. Jeśli zmieni się cokolwiek, konieczne będzie znalezienie zestawu wersji wszystkich narzędzi, które będą ze sobą zgodne.

Osobiście staram się używać niestandardowych kontrolek jak najrzadziej, ponieważ przy pisaniu tej książki nie mogę wychodzić z założenia, że czytelnik posiada określone narzędzia innych firm. Korzystam z WinZipa (www.WinZip.com) i Internet Download Managera (www.InternetDownloadManager.com) poza projektami, ale nie wewnątrz nich.

Kontrolek niestandardowych należy używać wtedy, gdy pozwalają zaoszczędzić dużo czasu. Jednak przed ich zastosowaniem warto przemyśleć, ile czasu zajęłoby poradzenie sobie bez nich, a ile zastąpienie ich innymi w razie potrzeby migracji do nowszej wersji Visual Basica.

Podsumowanie

Kontrolki stanowią główne połączenie pomiędzy użytkownikiem a aplikacją. Umożliwiają programowi wysyłanie danych do użytkownika, a użytkownikowi kontrolowanie programu. Kontrolki są wszędzie, w praktycznie każdej aplikacji działającej w systemie Windows. Tylko niewielka liczba programów uruchomionych w tle może się bez nich obyć.

W tym rozdziale krótko opisałem przeznaczenie standardowych kontrolek Visual Basica. Zamieściłem tu też wskazówki dotyczące odpowiednich z nich w przypadku wykonywania różnych zadań. Bardziej szczegółowy opis tych kontrolek znajduje się w Dodatku G.

Nawet gdy wie się wszystko o kontrolkach, nie ma się gwarancji, że stworzony interfejs użytkownika będzie odpowiedni. Projektowanie intuicyjnych i łatwych w użyciu interfejsów jest sztuką. Dobry projekt umożliwia użytkownikowi wykonywanie pracy w naturalny sposób, przy czym zminimalizowana jest ilość marnowanego czasu. Źle zaprojektowany interfejs może przeszkadzać w wykonywaniu czynności, a nawet zamienić najprostsze z nich w walkę z programem.

Informacje na temat budowy użytecznych aplikacji można znaleźć w książkach poświęconych projektowaniu interfejsów. Objaśniono w nich typowe problemy i rozwiązania związane z tym ostatnimi. Wiele też można nauczyć się poprzez badanie istniejących już aplikacji, które są uznane za dobre. Przyjrzyj się układowi ich formularzy i okien dialogowych. Nie należy podkradać zastosowanych tam rozwiązań, ale spróbować zrozumieć, dlaczego kontrolki zostały w nich poukładane właśnie w taki sposób. Przyjrzyj się aplikacjom, które szczególnie lubisz, a do tego są wyjątkowo łatwe w użyciu. Porównaj je z programami, które uważasz za niezgrabne i mało przejrzyste.

Ten rozdział stanowi wprowadzenie do kontrolek Windows Forms. W rozdziale 9. — „Używanie kontrolek Windows Forms” — znajdziesz bardziej szczegółowe objaśnienie sposobów ich wykorzystywania. Zamieszczone tam zostaną opisy metod dodawania kontrolek do formularza w trakcie projektowania i działania programu, a także wskazówki, jak używać ich własności, metod i zdarzeń.