helion.pl
KSIĘGARNIA INTERNETOWA · WYDAWNICTWO

IDŹ DO
PRZYKŁADOWY ROZDZIAŁ
SPIS TREŚCI

KATALOG KSIĄŻEK
KATALOG ONLINE
ZAMÓW DRUKOWANY KATALOG

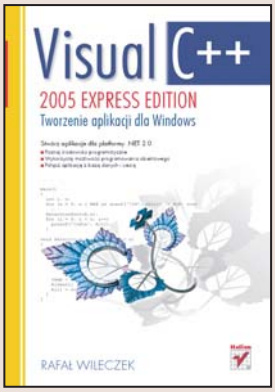
TWÓJ KOSZYK
DODAJ DO KOSZYKA

CENNIK I INFORMACJE
ZAMÓW INFORMACJE O NOWOŚCIACH
ZAMÓW CENNIK

CZYTELNIA
FRAGMENTY KSIĄŻEK ONLINE

Visual C++ 2005 Express Edition. Tworzenie aplikacji dla Windows

Autor: Rafał Wileczek
ISBN: 83-246-0488-X
Format: B5, stron: 168
Przykłady na ftp: 195 kB



Stwórz aplikacje dla platformy .NET 2.0

- Poznaj środowisko programistyczne
- Wykorzystaj możliwości programowania obiektowego
- Połącz aplikację z bazą danych i siecią

środowisko programistyczne Visual C++ 2005 Express Edition to uproszczona i okrojona wersja Visual C++ 2005. Mimo tego jest bardzo przydatnym i wydajnym narzędziem dla programisty chcącego tworzyć aplikacje dla platformy .NET. Umożliwia korzystanie z bibliotek .NET Framework ułatwiających żmudny proces tworzenia aplikacji. Jest świetnym narzędziem nie tylko dla początkującego twórcy aplikacji, ale także dla tych programistów, którzy nie potrzebują zaawansowanych środowisk w wersjach Enterprise. Dodatkowym atutem środowiska Visual C++ 2005 Express Edition jest możliwość pobrania go i używania za darmo do listopada 2007 roku.

Książka „Visual C++ 2005 Express Edition. Tworzenie aplikacji dla Windows" to podręcznik, dzięki któremu poznasz możliwości tego środowiska programistycznego tworząc własne aplikacje. Dowiesz się, jak zaprojektować interfejs użytkownika korzystając z komponentów zawartych w .NET Framework. Nauczysz się łączyć aplikacje z bazami danych oraz z siecią Internet. Przeczytasz także o obsłudze plików graficznych i dźwiękowych oraz drukowaniu. Znajdziesz tu również dodatek, w którym opisano podstawy programowania w języku C++.

- Określanie parametrów nowego projektu w Visual C++ 2005 Express Edition
- Kompilacja i uruchamianie aplikacji
- Podstawy programowania obiektowego
- Tworzenie graficznych interfejsów użytkownika
- Połączenia z bazami danych i internetem
- Programowanie wielowątkowe
- Drukowanie grafiki i dokumentów

Rozwiń swoje umiejętności programowania oraz tworzenia profesjonalnych aplikacji dla platformy .NET 2.0

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

Spis treści

Rozdział 2.

Aplikacja z graficznym interfejsem użytkownika (Windows Forms Application) — zaczynamy

W rozdziale 1. zapoznałeś się ze środowiskiem Visual C++ 2005 Express Edition — utworzyliśmy projekt aplikacji okienkowej (*Windows Forms Application*), ale go nie rozwijaliśmy. W tym rozdziale zajmiemy się podstawami tworzenia graficznego interfejsu użytkownika z wykorzystaniem komponentów dostarczanych przez platformę .NET w przestrzeni nazw `Windows.Forms`. Komponenty te pozwalają na utworzenie aplikacji o dowolnym zastosowaniu, w pełni funkcjonalnej i — co ważniejsze — o przyjaznym i znajomym dla współczesnego użytkownika komputerów wyglądzie.

Wbrew temu, co by się mogło wydawać, tworzenie graficznego interfejsu użytkownika jest sztuką i rządzi się ściśle określonymi zasadami. Istnieje wiele opracowań prezentujących teoretyczne i praktyczne aspekty tworzenia interfejsów użytkownika, poparte badaniami psychologów i socjologów, analizujących zachowania, reakcje i przyzwyczajenia użytkowników. Właśnie przyzwyczajenia użytkowników mają spore znaczenie podczas projektowania wyglądu aplikacji — nieraz spotkaliśmy się z pozytywną reakcją (naszą lub naszych znajomych) na wygląd i obsługę programu, kiedy użytkownik odkrył, że dany program obsługuje się podobnie jak inny, np. popularny edytor tekstów. Istotą współczesnych programów jest łatwość i intuicyjność obsługi. Tworząc program, nie chcemy, aby użytkownik musiał uczyć się jego obsługi od zera. Bardzo dobrym i pożądanym nawykiem jest podpatrywanie rozwiązań stosowanych przez „wielkich" — chodzi przede wszystkim o układ menu i pasków narzędziowych, stosowanie odpowiednich elementów do informowania użytkownika o stanie aplikacji itp.

W tym rozdziale nie będziemy zajmować się teorią tworzenia programów z graficznym interfejsem użytkownika, natomiast spróbujemy napisać program, za pomocą którego nauczysz się stosować podstawowe elementy kontrolne (przyciski, pola tekstowe, etykiety itp.) i programować zdarzenia generowane przez elementy kontrolne.

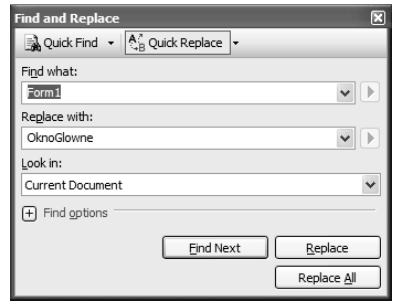
Projekt i okno główne

Zaczniemy od otwarcia projektu, który wykorzystaliśmy w rozdziale 1. Najpierw zajmiemy się oknem głównym. Kod okna głównego został wygenerowany przez kreatora aplikacji jako klasa o niefortunnej nazwie `Form1` i umieszczony w pliku *Form1.h*. My zwykle chcemy, żeby nazwy wszelkich klas, zmiennych, stałych i metod były znaczące, dlatego też przemianujemy nazwę klasy naszego okna na `OknoGlowne`. W tym celu:

- przejdź do widoku kodu okna głównego i odszukaj w nim nagłówek deklaracji klasy `Form1` (`public ref class Form1 : public System::Windows::Forms::Form`); kliknij w ten nagłówek (umieścisz w nim kursor);
- w oknie *Properties*, służącym do zmiany właściwości poszczególnych elementów projektu (po pierwszym użyciu będzie już zawsze widoczne i jego zawartość będzie się zmieniała w zależności od tego, który w danym momencie element czy plik będzie wskazany), znajdź właściwość o nazwie `Name` umieszczoną w nawiasach (właściwości domyślnie podzielone są na kategorie tematyczne, więc czasem łatwiej daną właściwość znaleźć, domyślając się, do jakiej kategorii należy) i zmień jej wartość na `OknoGlowne` — po zatwierdzeniu zmiany (np. przez naciśnięcie klawisza *Enter*) nazwa klasy zostanie zmieniona;
- w następnym kroku musisz zmienić nazwę pliku — przejdź zatem do okienka *Solution Explorer* i kliknij prawym przyciskiem myszy nazwę pliku *Form1.h*; z menu kontekstowego wybierz opcję *Rename* i wprowadź nowa nazwę pliku, w naszym przypadku *OknoGlowne.h*;

W języku C++/CLI nazwa pliku nie musi odpowiadać nazwie klasy publicznej umieszczonej w tym pliku (pojęcie klasy wyjaśniamy w rozdziale 3.), ale trzymanie się tej zasady może znacznie poprawić przejrzystość projektu.

- pozostaje wskazać plikowi zawierającemu punkt startowy programu (w naszym przypadku jest to plik o nazwie *Projekt.cpp* znajdujący się w folderze *Source Files*), w którym pliku znajduje się definicja okna głównego programu i jaka jest nazwa klasy okna głównego; w tym celu otwórz plik *Projekt.cpp*, klikając na niego dwukrotnie, i zamień wystąpienie napisu `Form1` na `OknoGlowne` — można to zrobić automatycznie, używając mechanizmu „znajdź i zamień” dostępnego poprzez menu *Edit* (rysunek 2.1).



Rysunek 2.1. *Zmiana nazwy pliku nagłówkowego i klasy w pliku źródłowym projektu*

Po dokonaniu tych zmian warto projekt przebudować — nie powinno być żadnych problemów.

Wprowadziliśmy porządek do naszego projektu, możemy zatem zająć się właściwościami okna głównego. Jeżeli kiedykolwiek odwoływaliśmy się do okna *Properties*, to jest ono widoczne po prawej stronie w oknie środowiska. Kliknijmy w projekt okna głównego naszej aplikacji — w oknie *Properties* wyświetlone zostaną właściwości okna głównego. Właściwości te służą przede wszystkim do konfiguracji wyglądu i zachowania naszego okna. Ustalmy, że nasze okno będzie miało rozmiar początkowy 400 na 300 pikseli i będzie umieszczane każdorazowo po uruchomieniu w centrum ekranu. W tym celu musimy w oknie *Properties* zmienić wartości właściwości `Size` oraz `StartPosition`. We właściwości `Size` zmieniamy domyślne rozmiary okna na założone przez nas, natomiast w przypadku właściwości `StartPosition` wybieramy z listy rozwijalnej wartość `CenterScreen`. Teraz, po przebudowaniu i uruchomieniu, okno naszego programu zawsze będzie się pojawiało na środku ekranu.

Na pewno zauważyłeś, że na belce programu nadal znajduje się napis `Form1`. Za napis, który jest wyświetlany na belce okna, odpowiada właściwość `Text` — zmieńmy jej wartość (poprzez okno *Properties*), np. na `Mój pierwszy program`.

Warto dokładniej przyjrzeć się właściwościom dostępnym poprzez okno *Properties* i poeksperymentować z różnymi ustawieniami — efekty mogą być zadziwiające (o czym można się przekonać, czytając dalej tę książkę).

Elementy kontrolne

Elementy kontrolne to wszystkie składniki interfejsu użytkownika, które mogą mieć za pośrednictwem użytkownika wpływ na działanie programu. Podstawowe elementy kontrolne, z których będziemy korzystać podczas tworzenia interfejsu użytkownika w przykładowym programie, to:

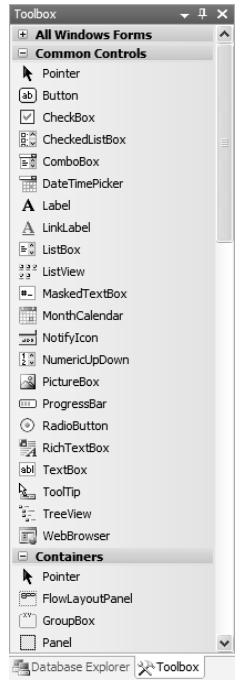
- przyciski (`Button`);
- pola tekstowe (`TextBox`);
- pola wyboru (`CheckBox`);
- listy rozwijane (`ComboBox`);

- etykiety (`Label`);
- menu główne programu (`MenuStrip`);
- pasek narzędziowy (`ToolStrip`);
- pasek stanu (`StatusStrip`).

Platforma .NET poprzez przestrzeń nazw `Windows.Forms` udostępnia oczywiście o wiele więcej elementów kontrolnych — będziemy je wprowadzać w miarę potrzeby w kolejnych rozdziałach. W tym rozdziale będziemy po kolei tłumaczyć, co trzeba zrobić, aby umieścić dany element kontrolny w obszarze okna, i jak zmienić jego właściwości. W następnych rozdziałach wprowadzimy dla ułatwienia tabelkę, która będzie zawierała typ elementu kontrolnego, jego nazwę oraz zmienione właściwości — wpłynie to niewątpliwie na przejrzystość rozdziałów.

Zacznijmy więc od wyposażenia naszego programu w menu główne. Po prawej stronie okna głównego środowiska schowane jest okienko *Toolbox* (rysunek 2.2) zawierające elementy kontrolne, za pomocą których będziemy budować naszą aplikację. Dla ułatwienia pracy można to okienko „przypiąć" — po prawej stronie pojawi się dwuczęściowe okienko składające się z okna *Toolbox* u góry i okna *Properties* u dołu.

Rysunek 2.2.
Okno Toolbox

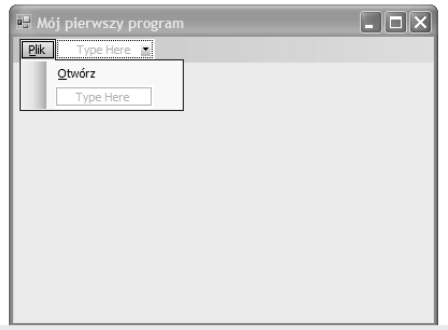


Elementy kontrolne w oknie *Toolbox* podzielone zostały na kategorie, aby ułatwić wyszukiwanie konkretnego elementu. Znajdźmy element o nazwie `MenuStrip`, wskażmy go lewym przyciskiem myszy, a następnie przeciągnijmy na projekt okna głównego naszej aplikacji — okno to zostanie wyposażone w menu główne, którego pozycję musimy teraz wprowadzić, wykorzystując bardzo prosty edytor. Najpierw jednak po-

winniśmy zmienić nazwę obiektu naszego menu — w tym celu klikamy w menu i udajemy się do okna *Properties*, gdzie zmieniamy właściwość `Name` na `menuGlowne`.

Rysunek 2.3 przedstawia sposób edycji pozycji menu głównego.

Rysunek 2.3. *Edycja menu głównego*

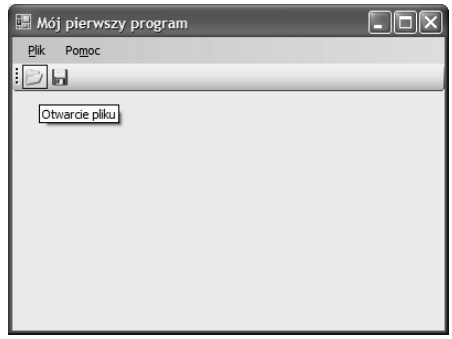


Warto pamiętać, że wpisanie symbolu `&` przed literą w nazwie pozycji menu (oraz w przypadku innych elementów kontrolnych) spowoduje podkreślenie tej litery i wprowadzenie klawisza skrótu dla danej opcji (na przykład, aby rozwinąć menu *Plik*, wystarczy nacisnąć kombinację klawiszy *Alt+P*).

Nasze menu będzie się składać z pozycji *Plik* i *Pomoc*. Menu *Plik* będzie zawierało pozycje *Otwórz*, *Zapisz* i *Zakończ*, natomiast menu *Pomoc* będzie zawierało pozycję *O programie...* Oprócz nazw poszczególnych pozycji, na etapie edycji menu można zadecydować (przez wybranie opcji z listy rozwijalnej), jakiego typu będzie dana pozycja menu (np. separator, lista rozwijalna itp.) lub czy dana pozycja będzie podmenu.

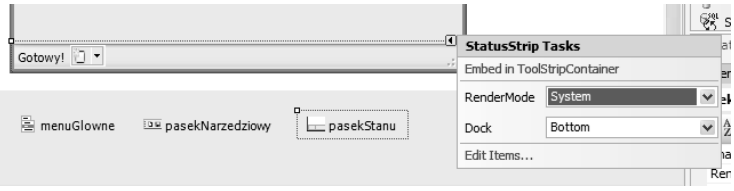
Teraz wyposażymy nasz program w pasek narzędziowy, który będzie zawierał dwa przyciski dublujące działanie pierwszych dwóch pozycji menu. Odnajdźmy w oknie *Toolbox* komponent o nazwie `ToolStrip` i umieśćmy go w oknie głównym programu. Następnie, w podobny sposób jak podczas dodawania nowych pozycji menu, dodajmy do paska narzędziowego dwa podstawowe elementy (klikając ikonkę dodawania elementu). Podstawowym elementem paska narzędziowego jest przycisk, ale można dodać też inne elementy, np. etykietę, listę rozwijalną, pasek postępu itp. Musimy również pamiętać o zmianie nazwy obiektu paska narzędziowego (właściwości `Name` przypisać wartość, np. `pasekNarzedziowy`). Warto przygotować sobie kilka ikonek, które będzie można umieścić na przyciskach paska narzędziowego — ikonki te wprowadzamy za pomocą właściwości `Image` w oknie *Properties* dla danego przycisku (wybieramy plik graficzny, najlepiej z obrazkiem o rozmiarach 16 na 16 pikseli). Dodatkowo właściwość `ToolTipText` przycisku zawiera tekst wyświetlany w „dymku" po umieszczeniu kursora myszy nad elementem kontrolnym — warto zmienić ten tekst na rzeczywistą podpowiedź. Rysunek 2.4 przedstawia projekt okna głównego z paskiem narzędziowym i dwoma dodanymi przyciskami.

Bardzo przydatnym elementem każdego programu z graficznym interfejsem użytkownika jest pasek stanu, umieszczany u dołu okna. Zawiera on najczęściej informacje, którymi użytkownik niekoniecznie chciałby być bombardowany (pod postacią różnych okien dialogowych) — chodzi tu głównie o postęp pobierania pliku, wyświetlanie podpowiedzi lub statusu operacji itp.



Rysunek 2.4. *Okno główne z paskiem narzędziowym*

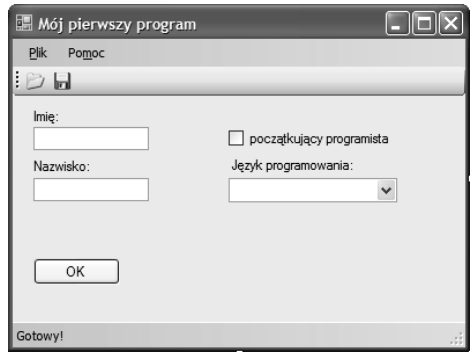
Pasek stanu dodajemy do okna w podobny sposób jak w przypadku poprzednich komponentów — przez przeciągnięcie i upuszczenie na projekcie okna komponentu `StatusStrip`. Po umieszczeniu paska stanu w oknie aplikacji powinniśmy zmienić jego nazwę (właściwość `Name` — np. `pasekStanu`). Podstawowym elementem, który może być umieszczony na pasku stanu, jest etykieta — dodaje się ją w analogiczny sposób jak przyciski w pasku narzędziowym, przez kliknięcie ikony dodawania etykiety. Dodajmy zatem jedną etykietę i przypiszmy jej właściwości `Text` napis `Gotowy!`. Twoją uwagę na pewno przykuł wygląd paska stanu, znacznie różniący się od wyglądu pozostałych elementów (paska narzędziowego i menu). Aby dostosować wygląd paska stanu do pozostałych komponentów, kliknijmy piktogram przedstawiający strzałkę, umieszczony w prawym górnym narożniku paska stanu, i w okienku zadań ustawmy *RenderMode* na *Professional* (rysunek 2.5).



Rysunek 2.5. *Zmiana wyglądu paska stanu*

Pasek stanu mamy przygotowany. Możemy zatem wyposażyć okno naszego programu w pozostałe elementy kontrolne. Dodajmy do naszego okna trzy etykiety (`Label`) i ustawmy ich właściwości `Text` jako `Imię:`, `Nazwisko:` i `Język programowania:`. Następnie dodajmy do okna głównego dwa pola tekstowe (`TextBox`) i nazwijmy je (`Name`) `txtImie` i `txtNazwisko`. Teraz dodajmy jedno pole wyboru (`CheckBox`) i nazwijmy je `cbPoczatkujacy` (właściwości `Text` przypiszmy `początkujący programista`). Na koniec dodajmy jedną listę rozwijalną (`ComboBox`) i nazwijmy ją `cmbJezyk` oraz jeden przycisk (`Button`) — nazwiemy go `btnOK` i właściwość `Text` ustalimy na `OK`. Jak te wszystkie operacje wykonać — już wiemy. Rysunek 2.6 przedstawia poprawne rozmieszczenie elementów kontrolnych w oknie programu.

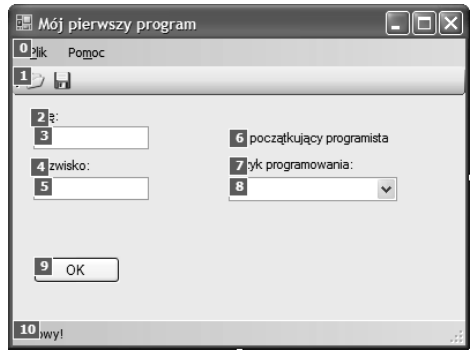
Lista rozwijalna zawierać będzie klika nazw języków programowania. Ponadto będziemy chcieli, żeby użytkownik mógł wpisać inny język programowania, niewymieniony na liście. Najpierw wprowadzimy języki programowania do wyboru. Kliknijmy w widoku projektu naszą listę rozwijalną prawym przyciskiem myszy i wybierzmy pozycję *Edit Items*. Zostaniemy przeniesieni do okna dialogowego umożliwiającego wprowadzenie elementów listy rozwijalnej, tak jak pokazano na rysunku 2.7.



Rysunek 2.6. *Rozmieszczenie elementów kontrolnych w oknie programu*

Rysunek 2.7. *Wprowadzanie elementów listy rozwijalnej*

Jak pewnie zauważyłeś, wykorzystując klawisz *Tab*, możemy się przemieszczać w pewnej logicznej kolejności po elementach kontrolnych okna. Spróbujmy w takim razie ustalić kolejność przechodzenia klawiszem *Tab* przez elementy kontrolne w naszym programie. W tym celu w widoku projektu okna z menu *View* wybierzmy pozycję *Tab Order*. Nasze okno zmieni wygląd tak jak na rysunku 2.8 — obok każdego elementu kontrolnego pojawi się numerek, oznaczający kolejność przechodzenia za pomocą klawisza *Tab*.

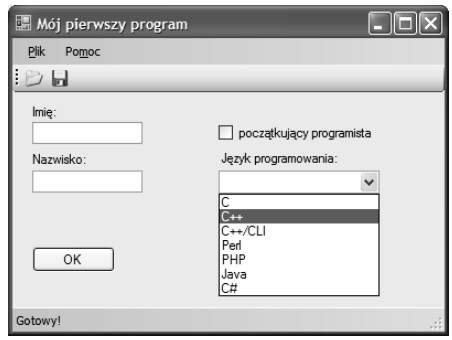


Rysunek 2.8. *Ustalanie kolejności przechodzenia między elementami kontrolnymi z wykorzystaniem klawisza Tab*

Kolejność przejść ustalamy, klikając numerki od pierwszego elementu do ostatniego zgodnie z przyjętym założeniem. Edycję przejść wyłączamy, wybierając ponownie *View/Tab Order*.

Przykładowy program jest już gotowy. Możemy go przebudować i uruchomić; powinniśmy otrzymać efekt jak na rysunku 2.9.

Rysunek 2.9. *Działający program*



Brakuje tylko jednej rzeczy: obsługi zdarzeń generowanych przez elementy kontrolne.

Zdarzenia

Elementy kontrolne generują zdarzenia — obsługa tych zdarzeń to podstawa działania programów z graficznym interfejsem użytkownika. Zacznijmy od oprogramowania zdarzenia kliknięcia przycisku *OK* naszego okna głównego. Możemy to zrobić na dwa sposoby. Można kliknąć przycisk, przenieść się do okna *Properties*, przełączyć się na widok zdarzeń (ang. *Events*) za pomocą przycisku z błyskawicą, odnaleźć zdarzenie `Click` i kliknąć dwukrotnie puste pole po prawej stronie tego zdarzenia. Można też po prostu kliknąć dwukrotnie przycisk. Obie te akcje spowodują przeniesienie nas do okna edycji kodu, gdzie będziemy mogli oprogramować to zdarzenie. Wygenerujmy więc w jeden z tych sposobów funkcję obsługi zdarzenia `Click` przycisku *OK* i wprowadźmy do niej kod z listingu 2.1.

Listing 2.1. *Funkcja obsługi zdarzenia kliknięcia w przycisk OK*

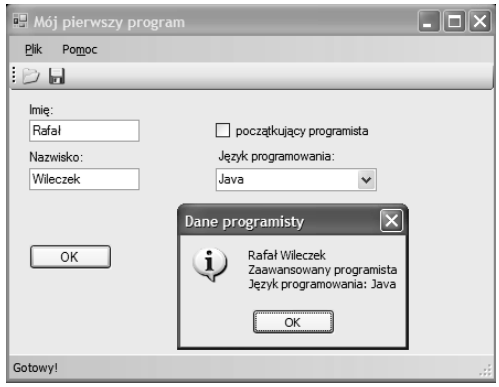
```
private: System::Void btnPokaz_Click(System::Object^  sender, System::EventArgs^  e)
{
   String^ komunikat = txtImie->Text + " " + txtNazwisko->Text + "\n";
   if (cbPoczatkujacy->Checked) {
    komunikat += "Początkujący programista\n";
   } else {
    komunikat += "Zaawansowany programista\n";
   }
   if (cmbJezyk->Text->Length != 0) {
    komunikat += "Język programowania: " + cmbJezyk->Text;
   }
   MessageBox::Show(komunikat, "Dane programisty", MessageBoxButtons::OK,
   MessageBoxIcon::Information);
}
```

W pierwszym wierszu funkcji zmiennej `komunikat` przypisujemy wartości właściwości pól tekstowych połączone spacją i zakończone znakiem nowego wiersza.

Środowiska programistyczne Microsoft wyposażone są w technologię IntelliSense, dzięki której w sposób szybki i efektywny możemy uzyskać podpowiedź dotyczącą wyboru metody czy właściwości dla danego obiektu w oknie edycji kodu. Podpowiedź można tez uruchomić ręcznie, za pomocą kombinacji klawiszy *Ctrl+*Spacja.

Dalej za pomocą instrukcji warunkowej `if` sprawdzamy, czy użytkownik zaznaczył pole wyboru `cbPoczatkujacy` (właściwość `Checked` przyjmuje wartości `True` lub `False` w zależności od tego, czy pole zostało zaznaczone, czy nie). Jeśli tak, do zmiennej komunikat dołączamy informację `Początkujący programista`, w przeciwnym razie do zmiennej komunikat dołączamy informację `Zaawansowany programista`. Następnie sprawdzamy, czy z listy rozwijalnej wybrany został język programowania (lub też użytkownik wpisał nazwę języka). Jeśli tak (długość ciągu znaków we właściwości `Text` jest większa od zera), do zmiennej `komunikat` dołączamy zawartość pola edycyjnego listy rozwijalnej. Na koniec wyświetlamy komunikat, używając wygodnego i łatwego w zastosowaniu predefiniowanego okna informacyjnego `MessageBox`. Efekt działania programu widoczny jest na rysunku 2.10.

Rysunek 2.10. *Efekt obsługi zdarzenia Click przycisku OK.*



Obsługę zdarzeń pozostałych przycisków i pozycji menu pozostawiamy Czytelnikowi jako ćwiczenie własne.

Podsumujmy: w rozdziale tym nauczyliśmy się tworzyć aplikacje z graficznym interfejsem użytkownika i pisać funkcje obsługi zdarzeń. Będziemy teraz mogli — eksperymentując z oknem *Properties* — tworzyć różne aplikacje oraz rozwijać przykładowe programy przedstawione w następnych rozdziałach. Ponieważ pewne umiejętności już mamy, począwszy od rozdziału 5., lista komponentów użytych do utworzenia interfejsu użytkownika umieszczana będzie we wspomnianej już tabeli.