Zestaw A

C#, *Base Class Library*

17 maja 2004

Streszczenie

Rozwiązanie zadań w tym zestawie polega na napisaniu programów w języku C#. Tam, gdzie jest to możliwe, należy unikać funkcji z **Win32**.

1. Aplikację z zadania 4.5 przenieść do interfejsu okienkowego.

   Okno główne aplikacji powinno być oknem typu MDI, zaś widoki danych powinny być prezentowane w oknach typu MDIChild.

   Dane (tu: dane osobowe zgromadzone w kartotece) powinny być prezentowane w kontrolce `ListView`, dwuklik elementu powinien otwierać nowy modalny formularz z edycją właściwości obiektu.

   Lista powinna mieć dodatkowo menu kontekstowe z następującymi pozycjami: *Dodaj nowy* (powinno przywołać pusty formularz właściwości obiektu), *Modyfikuj wybrany* (analogicznie do dwukliku elementu) i *Usuń wybrany* (powinno usunąć wskazany element).

   [**2p**]

2. W poprzednim zadaniu pojawia się pewna trudność: modyfikacja właściwości obiektu w osobnym formularzu oznacza konieczność odświeżenia wszystkich widoków, w których ujawniane są atrybuty tego obiektu.

   Rozwiązać ten problem według podanych niżej wskazówek.

   Rozszerzyć definicję każdego obiektu o unikatowy identyfikator, ID. Zaprojektować publiczną klasę `Powiadamianie` z publicznym zdarzeniem *ZmianaDanych*, którego sygnatura niesie ze sobą informację o identyfikatorze zmodyfikowanego obiektu. W klasie tej powinna być dodatkowo dostępna publiczna metoda, która wywoływałaby zdarzenie (zdarzenie można wywołać tylko z wnętrza klasy, która je definiuje).

   Następnie rozszerzyć definicję każdego formularza, zainteresowanego otrzymywaniem powiadomień o zmianach w danych, o rejestrowanie jego własnej funkcji na liście delegacji zdarzenia *ZmianaDanych* w chwili ładowania formularza (i oczywiście wyrejestrowywaniem funkcji z listy delegacji zdarzenia przy zamykaniu formularza). Funkcja reagująca na zaistnienie zdarzenia powinna przebudowywać widok swoich danych według informacji zawartych w parametrach zdarzenia.

   Na końcu, na wszystkich formularzach powodujących zmianę danych wywoływać zdarzenie po zaakceptowaniu zmian przez użytkownika.

   Taki mechanizm jest bardzo ogólny. Zdarzenie informujące o zmianach w danych dostępne jest publicznie, więc jego źródłem może być dowolne inne zdarzenie w aplikacji (na przykład naciśnięcie przez użytkownika przycisku OK w oknie modyfikacji właściwości obiektu). Również każdy formularz może być "subskrybentem" zdarzenia, czyli dokładać swoją własną funkcję na liście delegacji zdarzenia (jeden i ten sam formularz może być w szczególności subskrybować zdarzenie jak i wywoływać je). W momencie wywołania zdarzenia, wszystkie zarejestrowane na liście delegacji funkcje zostaną po kolei wywołane, czyli każdy formularz zareaguje na zmianę danych w sobie tylko właściwy sposób.

   [**2p**]

3. Aplikację z zadania 4.5 rozszerzyć o odczyt i zapis danych do/z jakiegoś wybranego serwera baz danych (SQL Server, MySql, Oracle, FireBird, inny) przez ADO.NET.

Należy zwrócić uwagę na to, że nie wszystkie rodzaje serwerów mają swoje dedykowane klasy w ADO.NET. Jeśli wybrany serwer nie posiada klas dedykowanych, należy spróbować skorzystać z ogólnych klas `OleDb...` lub poszukać klas dedykowanych na stronie producenta serwera (tak jest np. w przypadku MySql).

Aplikacja może zakładać, że struktura bazy danych jest przygotowana na wskazanym serwerze (program nie musi sam tworzyć nowej bazy danych). Uruchomienie aplikacji powinno być poprzedzone oknem logowania, na którym należałoby wprowadzić nazwę i hasło użytkownika bazy danych.

[**3p**]

4. Napisać w VB.NET program, który korzystając z automatyzacji i późnego wiązania otworzy nowy arkusz Excela, wpisze do niego zbiór przykładowych danych i utworzy na ich podstawie tabelę przestawną.

   [**3p**]