MS Access

Zaawansowany

Materiały szkoleniowe

Wersja demonstracyjna

KONTAKT

Adres

Expose sp. z o. o.
ul. Skierniewicka 10a
01-230 Warszawa

Telefon

+ 48 22 465 88 88
+ 48 22 240 19 99

Online

biuro@expose.pl
www.expose.pl
www.chcesieuczyc.pl

Konspekt kursu MS Access – Zaawansowany

1. Zarządzanie tabelami
   - Zaawansowane właściwości pól tabeli
   - Zmienianie klucza podstawowego
   - Relacje – opcje zaawansowane
   - Sprzężenia między tabelami
   - Kompaktowanie i naprawa bazy danych
   - Optymalizacja projektu tabeli
   - Analiza wydajności
   - Dokumentator bazy danych
2. Projektowanie kwerend przy użyciu języka SQL
   - Składnia języka SQL
   - Wybrane instrukcje SQL
   - Funkcje agregujące
   - Sortowanie i grupowanie danych
   - Kwerendy funkcjonalne: tworzące tabele, aktualizujące, dołączające dane, usuwające
3. Zaawansowane projektowanie formularzy
   - Wykorzystanie zdarzeń w formularzach
   - Formularze oparte o kwerendy
   - Praca z podformularzami
   - Formanty ActiveX
   - Obsługa zdarzeń przy użyciu języka VBA
   - Formularze wielostronicowe
4. Tworzenie zaawansowanych raportów
   - Wykorzystywanie szablonów raportów
   - Raporty wielokolumnowe
   - Grupowanie danych
   - Kontrolki ActiveX
   - Generowanie raportów w oparciu o kwerendy
5. Migracja danych do programu Microsoft SQL Server
   - Przygotowanie serwera bazy danych Microsoft SQL Server
   - Migracja niektórych lub wszystkich części bazy danych
   - Microsoft SQL Server Migration Assistant for Access
   - Wstęp do SQL Server Management Studio Express

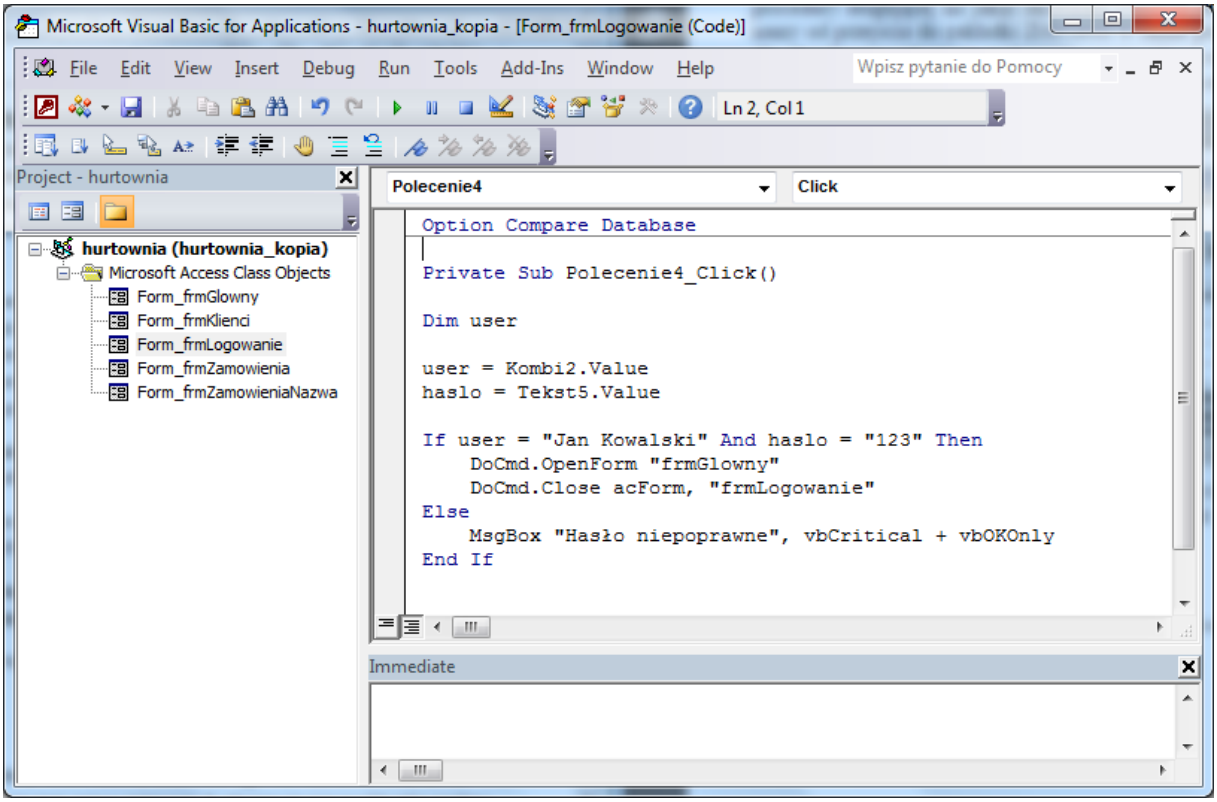
6. Podstawy języka VBA
   - Budowa edytora kodu
   - Zmienne i typy danych
   - Zasięg zmiennych
   - Komentarze
   - Instrukcje warunkowe (If, ElseIf)
   - Instrukcja Case
   - Pętle (For, While)
7. Dostęp do danych przy użyciu języka VBA
   - Obiekty DAO oraz ADO
   - Wykorzystanie obiektu Recordset
   - Wykorzystanie języka SQL
   - Instrukcje transakcyjne
   - Tworzenie własnych obiektów
   - Tworzenie rozbudowanych formularzy dla użytkownika

1 Podstawy języka VBA

Visual Basic for Applications (VBA) jest standardowym językiem makropoleceń dodawanym przez firmę Microsoft do praktycznie wszystkich składników pakietu biurowego MS Office. Przy jego pomocy można w stosunkowo prosty sposób znakomicie zwiększyć funkcjonalność aplikacji tworzących ten pakiet biurowy. W swojej zasadniczej części język VBA jest wspólny we wszystkich aplikacjach.

W trakcie projektowania dowolnego formularza może zajść potrzeba przygotowania procedury reagującej na jakąś określoną sytuację. Przygotowanie takiej procedury zaczynamy od przejścia do okna właściwości formularza – zakładka Zdarzenie. W kolejnym kroku dla określonego zdarzenia wybieramy opcję [Procedura zdarzenia] i poprzez klik przycisku z trzema kropkami przechodzimy do okna edytora VBA.

Okno edytora VBA możemy także otworzyć poprzez wywołanie odpowiedniego polecenia z karty Narzędzia baz danych lub korzystając ze skrótu alt+F11.



Kod programu i podprogramy pisze się i edytuje w oknie modułu. Każdy podprogram to ciąg instrukcji języka wykonujących poszczególne operacje.

Podprogram to pewien kod zapisany w języku programowania, który składa się z ciągu logicznych etapów umożliwiających wykonanie określonego zadania.

Rozróżnia się dwa typy podprogramów:

- Procedury,
- Funkcje.

Procedura to podprogram, który nie zwraca wartości i którego nie można wywołać ani w wyrażeniu, ani przez przypisanie jego nazwy zmiennej. Funkcjonuje ona zwykle jako osobny kod wywoływany przez zdarzenie związane z formularzem lub raportem.

Procedura funkcyjna zawsze zwraca wartość. Procedury funkcyjne są zapisywane i przechowywane w modułach (moduły stanowią pojemniki na kod procedur, można je traktować jako bibliotekę procedur).

Widok, jaki zobaczymy po przejściu do edytora VBA zależy w dużym stopniu od naszych ustawień, w pokazanej wyżej ilustracji pokazane są 4 okna:

Project Explorer – pokazuje organizację kodu w moduły w danym projekcie,

Properties – pokazuje właściwości aktywnego obiektu

Okno procedur – pokazuje procedury w danym oknie

Okno Immediate – okno instrukcji bezpośrednich

Budowa makr

```
sub nazwa()
    'komentarze pomijane przez VBA
    'dokumentują sens fragmentów programu
    instrukcje - zestaw poleceń
end sub
```

Moduł może zawierać wiele programów i funkcji

Znaczenie kolorów w edytorze

kolor niebieski - słowa kluczowe VBA

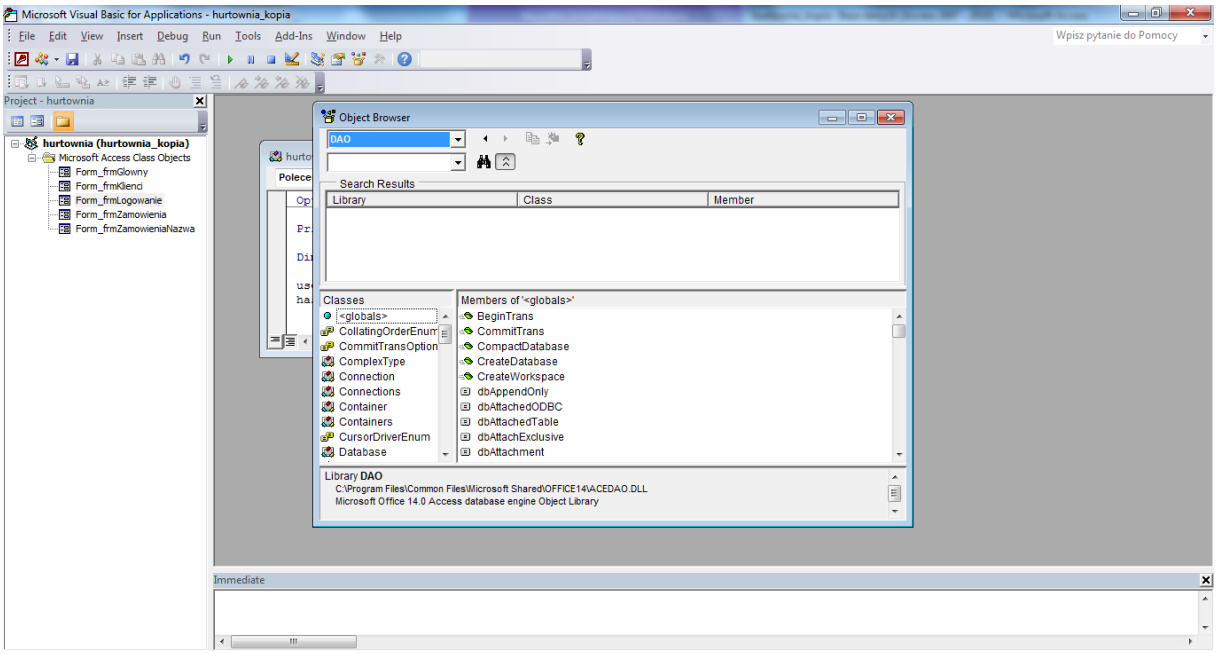
kolor zielony - wszystkie dodatkowe informacje, opisy - jednym słowem tekst własny programisty, w którym możemy wpisywać wszystkie uwagi. Zaczynają się one od górnego apostrofu ' , kończą naciśnięciem Enter czyli przejściem do następnej linii.

Przeglądarka obiektów

Przeglądarka Obiektów czyli Object Browser jest bardzo przydatnym narzędziem wyświetlającym wszystkie właściwości i metody dostępne dla danego obiektu. Po uaktywnieniu edytora Visual Basic przeglądarkę można uruchomić na jeden z trzech sposobów:

- Klawiszem F2
- Z menu View wybrać polecenie Object Browser
- Klikając przycisk Object Browser znajdujący się na pasku narzędzi Standard

Okno przeglądarki Object Browser zostało przedstawione poniżej.



Elementy programu VBA

- Słowa kluczowe – polecenia sterujące wykonywaniem programu – słowa (if), skróty (mid) i zbitki skrótów (rmid) z języka angielskiego.
- Identyfikatory – nazwy zmiennych, stałych, obiektów, programów/podprogramów/funkcji
- Komentarze

Nazwy makr

- ciągi liter i cyfr
- nie mogą zawierać znaków specjalnych: <spacja>, (), :, ;, itp. (ale mogą znak podkreślenia _)
- mogą mieć dowolną długość
- nazwa MUSI rozpoczynać się literą

Nazwy makr można zmieniać w dowolnym momencie, gdy nie są uruchomione, należy jednak pamiętać o tym, że nazwa może być użyta gdzieś w naszych innych programach korzystających z tego makra jako podprogramu;

Zmienne i stały

Cały kod programu grupowany jest w tzw. moduły odpowiadające utworzonym formularzom czy raportom. Poza nimi mogą być tworzone moduły ogólne oraz moduły klas. W ramach moduły kod grupowany jest w procedury lub funkcje, może być także umieszczany w sekcji deklaracji, gdzie deklaruje się zmienne i stały wykorzystywane w kodzie. Z pojęciem modułów ściśle związane są takie pojęcia jak zasięg zmiennych i procedur. Procedury i funkcje zdefiniowane na poziomie modułu formularza dostępne są tylko wewnątrz tego modułu. Z kolei procedury i funkcji zdefiniowane w modułach dostępne są dla wszystkich innych procedur w danym projekcie. Podobne regulacje dotyczącą deklaracji zmiennych. Zmienne zadeklarowane w sekcji deklaracji modułu ogólnego są dostępne dla wszystkich procedur i funkcji projektu, zmienne deklarowane w sekcji deklaracji modułu formularza są dostępne dla wszystkich procedur i funkcji tego modułu.

Do zadeklarowania zmiennych używa się jednego z trzech słów kluczowych: Dim, Private lub Public. Przykładowa instrukcja

```
Dim ile As Integer
```

Deklaruje zmienną o nazwie ile, która będzie przechowywać dnae typu liczba całkowita. W przypadku rozpoczęcia deklaracji zmiennej za pomocą słowa kluczowego Dim jej zasięg regulowany jest wyłącznie miejscem deklaracji.

Zmienna może być zadeklarowana słowem kluczowym Private lub Public, ale wyłącznie w sekcji deklaracji modułu (a nie w procedurze lub funkcji). Zmienne deklarowane przy pomocy słowa Private są dostępne jedynie dla procedur czy funkcji w tym module. Deklaracje zmiennych przy pomocy słowa Public oznacza, że zmienna taka jest dostępna dla wszystkich procedur i funkcji we wszystkich aplikacjach uruchomionych w danym momencie.

Przypisanie wartości do zmiennej:

Aby zmienna mogła przechowywać pewne określone wartości musimy to wartość przypisać do zmiennej. Operacje przypisania wartości do zmiennej nazywamy instrukcją przypisania. Instrukcja przypisania składa się z nazwy zmiennej, znaku równości oraz wartości (lub wyrażenia określającego wartość), która ma być przypisana do zmiennej.

Poniżej kilka przykładów przypisania wartości do zmiennej.

```
MojaWartosc = 3 'Zmiennej o nazwie MojaWartosc
przypisujemy wartość 3.

Przywitanie = "Pozdrawiam wszystkich" 'W tym przypadku
instrukcja przypisania przypisuje tekst umieszczony z
prawej strony znaku równości do zmiennej Powitanie.
Łańcuchy znakowe przypisywane do zmiennych należy ujmować
w cudzysłów.

Nazwisko = InputBox("Jak się nazywasz?") 'W instrukcji
przypisania możemy użyć funkcji. W przykładzie
przypisujemy wartość zwróconą przez funkcje InputBox,
zmiennej Nazwisko.
```

Nazwy zmiennych

Tworzenie kodu źródłowego stanie się prostsze, jeżeli wyrobimy w sobie nawyk nadawania zmiennym jak najbardziej zrozumiałych nazw. Język VBA narzuca jednak w stosunku do nazw zmiennych klika zasad:

- W nazwach zmiennych można stosować znaki alfanumeryczne, liczby i niektóre znaki interpunkcji, ale pierwszy znaj musi być literą.
- W języku VBA nie jest rozróżniana wielkość znaków. Aby nazwy zmiennych były bardziej czytelne, programiści często stosują znaki różnej wielkości (na przykład PierszaZmienna zamiast pierwszazmienna).
- W nazwach zmiennych nie można stosować spacji lub kropek. Aby nazwy były bardziej czytelne, programiści często używają znaku podkreślenia (Pierwsza_Zmienna).
- W nazwach zmiennych nie można umieszczać znaków specjalnego typu używanych w deklaracjach (#, $, %, & lub !).
- Nazwy zmiennych mogą zawierać maksymalnie 254 znaki, ale w praktyce tworzenie zmiennych o tak długich nazwach nie jest polecane.

Operatory wykonują operacje logiczne, matematyczne, łańcuchowe, podstawienia lub porównania. VBA zapewnia pełen zestaw operatorów, co przedstawiono w poniższej tabeli:

Operatory matematyczne	
^	Operator potęgowania
-	Operator negacji
*	Operator mnożenia
/	Operator dzielenia

****	Operator dzielenia całkowitego
Mod	Operator modulo
+	Operator dodawania
-	Operator odejmowania
	Operatory porównania
<	Operator mniejszości
<=	Operator mniejszości lub równości
>	Operator większości
>=	Operator większości lub równości
=	Operator równości
<>	Operator różności
Is	Operator porównania referencji
Like	Operator porównania łańcuchów
	Operatory łańcuchowe
+	Operator dodawania
&	Operator konkatenacji
	Operatory logiczne
Not	Operator negacji
And	Operator iloczynu logicznego, koniunkcji
Or	Operator sumy logicznej, alternatywy
Xor	Operator wyłączenia
Eqv	Operator równości
Imp	Operator implikacji, jeśli..to..

Operatory – kolejność

- Kolejność wykonywania operatorów: zmiana znaku, tożsamość, potęgowanie, mnożenie (*,/,\,mod), dodawanie (+/-), przypisanie (=);
- Nawiasy – kolejność wykonywania od najgłębiej zagnieżdżonego – nie ma ograniczeń w zagnieżdżeniu! Warto używać nawiasów dla pewności, że operacje będą wykonywane w żądanej kolejności;

Operacje na tekstach

Konkatenacja – łączenie tekstów: + lub &

Np.: `"Jan" + "Kowalski" => "JanKowalski„`

`"Jan" & Chr(32) & "Kowalski" => „Jan Kowalski”`

`"A" & Chr(98) & "c" => „Abc”`

`1 + 2 = "3"`

`1 & 2 = "12"`

(gdzie liczby mogą być też w postaci zmiennych typu string/variant)

Uwaga, operator + może być używany tylko do zmiennych tego samego typu (liczbowy/tekstowy), na potrzeby konkatenacji Variant liczy się jak string, o ile dodawany jest do stringa.

Operator logiczne – tablica prawdy:

x	y	Not x	Not y	x And y	x Or y	x Xor y	x Eqv y	x Imp y
0	0	1	1	0	0	0	1	1
0	1	1	0	0	1	1	0	1
1	0	0	1	0	1	1	0	0
1	1	0	0	1	1	0	1	1

Przykłady:

```
Dim Wynik
 Wynik = 10 * 3 ' Wynikiem jest 30
 MsgBox Wynik
```

```
Dim Wynik
 Wynik = 10 Mod 3 ' Wynikiem jest 1
 MsgBox Wynik
```

```
Dim Wynik
 Wynik = 10 - 3 ' Wynikiem jest 7
 MsgBox Wynik
 Wynik = -Wynik ' Wynikiem jest -7
 MsgBox Wynik
```

```
' W przykładzie poniższym wyświetlane jest okno dialogowe, w
którym użytkownik powinien wpisać swoje imię. Następnie
wyświetlone jest okno komunikatu z tekstem powitania.
Dim imię, powitanie
 imię = InputBox("Podaj swoje imię")
 powitanie = "Witaj " & imię & " miłej zabawy"
 MsgBox powitanie
```

Instrukcja warunkowa IF i jej warianty

Instrukcja warunkowa IF może przyjmować kilka wariantów, oto ważniejsze z nich:

```
If [warunek] Then [działanie jeśli prawda] Else [działanie
jeśli fałsz]
```

```
If warunek Then
     [polecenia jeżeli warunek jest spełniony]
     [polecenia jeżeli warunek jest spełniony]
Else
     [polecenia jeżeli warunek nie jest spełniony]
     [polecenia jeżeli warunek nie jest spełniony]
End If
```

Iteracje

Iteracja jest to powtarzanie tej samej instrukcji (bloku instrukcji) w pętli;

Pętla musi mieć podaną ilość powtórzeń, niebezpośrednio – tworzy się ją przez zakres od liczby do liczby (przy rozpoczęciu sprawdzany jest warunek wykonania);

```
For <licznik> = <pocz> To <koniec> Step <krok>
     <instrukcje>
Next <licznik>
```

Człon Step jest opcjonalny;

Komenda „For i = 1 To 5" powtórzy następującą po niej (aż do komendy Next) sekwencję instrukcji 5 razy;

Komenda „For i = 0 To 10 Step 5" powtórzy czynności 3 razy (0,5,10), a po zakończeniu pętli i=15 (wartość ta nie spełniła warunku, więc pętla zakończyła działanie;

Wykonywanie pętli można przerwać z wewnątrz instrukcją Exit For – wywoływaną na przykład przez instrukcję warunkową; <początek>, <koniec> i <krok> mogą być zmiennymi, nawet zmienianymi w trakcie wykonywania pętli; Stan zmiennych określających rozpoczęcie powtarzania sprawdzane jest na początku każdej pętli; Komenda Next dodaje liczbę <krok> do zmiennej <licznik> (inkrementacja zmiennej); <początek> może być większy od <koniec>, ale albo musi być to rozwiązane w trakcie pętli, albo <krok> musi mieć wartość ujemną;

Składnia:

```
For Licznik = Początek To Koniec Step Krok
     [blok instrukcji]
Next Licznik
```

Inna składnia iteracji specjalnie dla tablic:

```
For Each <element> In <tablica>
     <blok instrukcji>
Next
```

Inne iteracje:

```
While <wyrażenie logiczne>
     <instrukcje jeśli wyrażenie = prawda>
Wend

Do Until <wyrażenie logiczne>
     <instrukcje wykonywane póki wyrażenie = fałsz>
Loop (Until <wyrażenie logiczne>)

Do While <wyrażenie logiczne>
     <instrukcje wykonywane póki wyrażenie = prawda>
Loop (While <wyrażenie logiczne>)
```

Umiejscowienie warunku decyduje czy warunek jest sprawdzany po każdej pętli czy przed nią.

Instrukcja SELECT CASE

```
Select Case Wyrażenie
     Case Wartość1
     [blok kodu wykonywany, jeżeli Wyrażenie równa się
Wartość1]
     Case Wartość2
     [blok kodu wykonywany, jeżeli Wyrażenie równa się
Wartość2]
     ...
     Case Else
     [blok kodu wykonywany, jeżeli Wyrażenie nie równa się
     żadnej z wartości określonej przez instrukcje Case]
End Select
```

Instrukcja skoku

GoTo <etykieta>

Gdzies w kodzie: <etykieta>: <Instrukcje>

```
Sub test()
Dim x As Byte
petla: x = InputBox("Podaj liczbę:")
     If x <> 0 Then
          GoTo petla
     Else
          MsgBox ("Koniec programu!")
     End If
End Sub
```

Skierniewicka 10A
01-230 Warszawa
tel.: +48 22 465 88 88
biuro@expose.pl
www.expose.pl