VBA dla Excela

TABLICE INFORMATYCZNE

Helion

VISUAL BASIC FOR APPLICATIONS (VBA) — EXCEL

Elementy języka VBA

Zmienne

Zmienne służą do przechowywania danych określonego typu w pamięci komputera.
Zasady tworzenia nazw zmiennych:
- maksymalnie 255 znaków;
- pierwszy znak musi być literą;
- można stosować znaki alfanumeryczne, liczby i niektóre znaki interpunkcyjne;
- nie jest rozróżniana wielkość znaków;
- nie można stosować spacji, przecinka i znaku kropki;
- nie można umieszczać znaku deklarującego typ (%,&,^,!,#,@,$);
- nie można używać słów zastrzeżonych języka VBA (słowa kluczowe, instrukcje, funkcje, operatory).

Przykład: `zmienna1, Wynik_Dodawania, Next`

Słowa kluczowe

As	Binary	ByRef	ByVal	Date	Else
Empty	Error	False	For	Friend	Get
Input	Is	Len	Let	Lock	Me
Mid	New	Next	Nothing	Null	On
Option	Optional	ParamArray	Print	Private	Property
Public	PtrSafe	Resume	Seek	Set	Static
Step	String	Then	Time	To	True
WithEvents					

Deklarowanie zmiennych

Określenie nazwy zmiennej oraz typu przechowywanych danych (nie jest obowiązkowe).
Składnia deklaracji:

```
Dim [WithEvents] nazwa_zmiennej [([indeks_dolny])]
[As [New] typ_zmiennej]
```

gdzie:
`[WithEvents]` — określa, że *nazwa_zmiennej* jest zmienną obiektową używaną do odpowiedzi na zdarzenia wywoływane przez obiekt ActiveX;
`[New]` — zezwala na ukryte utworzenie obiektu.

Przykład:

```
Dim zmienna1 As String    'zmienna1 typu String
```

Skrócona deklaracja za pomocą znaków deklarujących typ:

```
Dim zmienna2%    'zmienna2 typu Integer
```

Deklaracja kilku zmiennych przy użyciu polecenia `Dim`:

```
Dim zmienna1 As Long, zmienna2 As String    'zmienna1 typu
Long, zmienna2 typu String
Dim zmienna1, zmienna2 As String    'nie można deklarować
grupy zmiennych tego samego typu, zmienna2 typu String, natomiast
zmienna1 typu domyślnego - Variant
```

Definiowanie typów danych

Typ danych określa postać danych zapisanych w pamięci. Brak deklaracji typu danych dla zmiennej oznacza, że zmienna jest typu `Variant` (domyślny typ danych).

Typy danych (wbudowane)

Typ danych		Liczba bajtów	Zakres wartości
Liczby całkowite	Byte	1	Od 0 do 255.
	Integer	2	Od –32 768 do 32 767.
	Long	4	Od –2 147 483 648 do 2 147 483 647.
	LongLong	8	Od –9 223 372 036 854 775 808 do 9 223 372 036 854 775 807 (dotyczy tylko środowiska 64-bitowego).
	Decimal	12	+/–79 228 162 514 264 337 593 543 950 335 (bez części dziesiętnej) +/–7,9228162514264337593543950335 (z 28 miejscami po przecinku).
Liczby rzeczywiste	Single	4	Od –3,402823E38 do –1,401298E–45 (dla wartości ujemnych); od 1,401298E–45 do 3,402823E38 (dla wartości dodatnich).
	Double	8	Od –1,79769313486231E308 do –4,94065645841247E–324 (dla wartości ujemnych); od 4,94065645841247E–324 do 1,79769313486232E308 (dla wartości dodatnich).
Ciągi znaków	String (zmienna długość)	10 + długość łańcucha	Od 0 do w przybliżeniu 2 miliardów.
	String (stała długość)	Długość łańcucha	Od 1 do w przybliżeniu 65 400.
LongPtr (typ aliasowy)		4 (8)	Zakres typu `Long` dla środowiska 32-bitowego (zakres typu `LongLong` dla środowiska 64-bitowego).
Boolean		2	`True` lub `False`.
Date		8	Od 1stycznia 0100 roku do 31 grudnia 9999 roku.
Currency		8	Od –922 337 203 685 477,5808 do 922 337 203 685 477,5807.
Object		4	Odwołanie do obiektu.
Variant (z liczbami)		16	Dowolna wartość numeryczna aż do maksymalnej wartości zakresu typu danych `Double`.
Variant (ze znakami)		22 + długość łańcucha	Taki sam zakres jak dla typu danych `String` (zmienna długość).
Definiowany przez użytkownika (użycie instrukcji `Type`)		Zmienna — wymagana przez elementy	Zakres każdego elementu jest taki sam jak zakres elementów jego typu danych (zmienny w zależności od elementów).

Wymuszenie deklarowania wszystkich zmiennych na poziomie modułu następuje poprzez użycie w pierwszej linii kodu modułu instrukcji `Option Explicit` (jawne deklarowanie zmiennych).

Znaki deklarujące typ danych

Typ danych	Znak	Typ danych	Znak
Integer	%	Double	#
Long	&	LongLong	^
Currency	@	Single	!
String	$		

Przykład:

```
Dim KomputerNazwa$    'zmienna KomputerNazwa typu String
```

Typ danych definiowany przez użytkownika

Może być używany tylko na poziomie modułu. Definiuje typ danych zawierający jeden lub więcej elementów oraz musi być definiowany w części deklaracyjnej modułu (przed kodem pierwszej procedury).
Składnia:

```
[Public | Private] Type nazwa_zmiennej
nazwa_elementu_1 As typ_danych
nazwa_elementu_2 As typ_danych
nazwa_elementu_n As typ_danych
End Type
```

Przykład:

```
Type KomputerInfo    'zmienna publiczna (wartość domyślna
Public)
KomputerNazwa As String
Model As String
ProcesorPredkosc As Long
End Type
```

Deklaracja zmiennej typu `KomputerInfo`:

```
Dim Komputery(1 To 10) As KomputerInfo
Komputery(1).KomputerNazwa = "OHP VAC"
```

Typ wyliczeniowy

Może być używany tylko na poziomie modułu. VBA automatycznie przypisuje pierwszej składowej wartość zero. Typ musi być definiowany w części deklaracyjnej modułu (przed kodem pierwszej procedury).
Składnia:

```
[Public | Private] Enum nazwa
nazwa_elementu_1 [= wartość_elementu]
nazwa_elementu_2 [= wartość_elementu]
nazwa_elementu_n [= wartość_elementu]
End Enum
```

Przykład:

```
Enum RodzajUmowy    'zmienna publiczna (wartość domyślna Public)
UmowaBezterminowa = 1
UmowaTerminowa
UmowaSpecjalna
End Enum
```

Deklaracja zmiennej typu `RodzajUmowy`:

```
Dim lngRodzajUmowy As RodzajUmowy
lngRodzajUmowy = UmowaSpecjalna    'lngRodzajUmowy = 3
```

Stałe

Nazwy zastępujące wartości niezmienne w kodzie programu, nazwy o określonym znaczeniu. Deklaracja stałych następuje przy użyciu instrukcji `Const`. Domyślnie stałe są prywatne (typu `Private`).
Składnia:

```
[Public | Private] Const nazwa_stałej [As typ_danej]
= wyrażenie
```

Przykład:

```
Const TLG = 40, Nazwa_Modulu As String = "CZAS PRACY"
Public Const Nazwa_Aplikacji As String = "Aplikacja
zarządzania ..."
```

Język VBA ma wiele predefiniowanych stałych m.in. stałe kolorów [lista wszystkich stałych dostępna za pomocą przeglądarki *Object Browser* (klawisz *F2*) edytora VBE].

Zasięg zmiennych

Rodzaj zmiennej	Zasięg (poziom zasięgu)	Miejsce i sposób deklaracji zmiennej	Czas życia zmiennych (trwałość zmiennej)
Zmienna lokalna	Pojedyncza procedura, w której została zadeklarowana (w innych procedurach mogą zostać zadeklarowane zmienne o identycznych nazwach).	Wewnątrz procedury (pomiędzy poleceniami `Sub`, `Function`, `Property` a `End Sub`, `End Function`, `End Property`), polecenie `Dim` lub `Static`.	Przestaje istnieć (zwalniana pamięć) po zakończeniu procedury (z wyjątkiem zmiennej deklarowanej jako `Static`).
Zmienna modułowa (dla całego modułu)	Pojedynczy moduł.	Przed pierwszą deklaracją procedury w module, polecenie `Dim` lub `Private`.	Zachowuje wartość po zakończeniu działania procedury poleceniem `End (Exit) Sub`, `End (Exit) Function`. Traci wartość po wykonaniu polecenia `End`.
Zmienna globalna	Wszystkie moduły (cały projekt).	W module standardowym VBA, przed pierwszą procedurą modułu (poza wszystkimi obszarami procedur), polecenie `Public`.	Traci wartość po wykonaniu polecenia `End`.
Zmienna statyczna	Pojedyncza procedura, w której została zadeklarowana.	Wewnątrz procedury (pomiędzy poleceniami `Sub` a `End Sub`), polecenie `Static`.	Zachowuje wartość po zakończeniu działania procedury. Traci wartość po wykonaniu polecenia `End`.

Funkcje dotyczące typów danych

Funkcje określające typ danych

Funkcja	Opis
TypeName	Zwraca łańcuch znaków typu `String` określający typ danej zmiennej.
VarType	Zwraca wartość typu `Integer` reprezentującą podtyp zmiennej.

Funkcje sprawdzające typ danych

Funkcja	Opis
IsArray	Zwraca `True`, jeśli zmienna jest tablicą.
IsDate	Zwraca `True`, jeśli wyrażenie może zostać przekonwertowane do typu `Date`.
IsEmpty	Zwraca `True`, jeśli zmienna nie została zainicjowana lub została ustalona jako `Empty`.
IsError	Zwraca `True`, jeśli wyrażenie ma wartość `Error`.
IsMissing	Zwraca `True`, jeśli do procedury nie przekazano opcjonalnego argumentu.
IsNull	Zwraca `True`, jeśli wyrażenie zawiera wartość `Null`.
IsNumeric	Zwraca `True`, jeśli wyrażeniu można nadać wartość liczby.
IsObject	Zwraca `True`, jeśli w wyrażeniu występuje odwołanie do obiektu.

Funkcje zwracają typ danych `Boolean`, określając, czy określone wyrażenie, występujące jako argument funkcji, jest danego typu i może zostać przekonwertowane na określony typ danych.

Funkcje konwersji typów danych

Funkcja	Zakres dla wyrażenia (argumentu)
CBool	Przekształca wyrażenie na typ danych `Boolean`.
CByte	Przekształca wyrażenie na typ danych `Byte`.
CCur	Przekształca wyrażenie na typ danych `Currency`.
CDate	Przekształca wyrażenie na typ danych `Date`.
CDbl	Przekształca wyrażenie na typ danych `Double`.
CDec	Przekształca wyrażenie na typ danych `Decimal`.
CInt	Przekształca wyrażenie na typ danych `Integer`.
CLng	Przekształca wyrażenie na typ danych `Long`.
CLngLng	Przekształca wyrażenie na typ danych `LongLong`.
CLngPtr	Przekształca wyrażenie na typ `LongPtr`.
CSng	Przekształca wyrażenie na typ danych `Single`.
CStr	Przekształca wyrażenie na typ danych `String`.
CVar	Przekształca wyrażenie na typ danych `Variant`.
CVErr	Zwraca typ danych `Variant` (podtyp `Error`) zawierający numer błędu zdefiniowany przez użytkownika.

Nazwa funkcji określa zwracany typ danych. Wyrażenie jako argument funkcji jest ciągiem znaków lub wyrażeniem numerycznym. Jeżeli wyrażenie przekazywane do funkcji wykracza poza zakres dla danego typu danych, wówczas podczas konwersji generowany jest błąd.

Funkcje konwersji pomiędzy różnymi typami danych

Funkcja	Opis
Int, Fix	Zwraca całkowitą część liczby (typu `Integer`). Różnica pomiędzy funkcją `Int` a `Fix` polega na różnicy konwersji liczb ujemnych.
Hex	Zwraca ciąg znaków (typ `String`) reprezentujący liczbę w postaci szesnastkowej.
Oct	Zwraca daną typu `Variant` (`String`) reprezentującą liczbę w postaci ósemkowej.

Funkcja	Składnia, przykład	Opis

Format	Składnia:
	`Format(wyrażenie[, format[, pierwszy_dzień_tygodnia[, pierwszy_tydzień_roku]]])`
	Przykład:
	`Format(Date, "yyyy-mm-dd")  'zwraca datę systemową w formacie rok-miesiąc-dzień`
	`Format(101.5 , "##0.00")  'zwraca "101.50"`
	`Format(9 , "0.00%")  'zwraca "900%"`
	`Format(32)  'zwraca "32"`
	`Format("Opis" , ">")  'zwraca "OPIS"`
		Zwraca daną typu `Variant` (`String`) zawierającą wyrażenie sformatowane zgodnie z instrukcjami zawartymi w wyrażeniu *format*. Argumenty *pierwszy_dzień_tygodnia* oraz *pierwszy_tydzień_roku* określone są przez wartości i stałe.

Ciąg dalszy na stronie 2 →

Dalsza część książki dostępna w wersji pełnej.

woblink