Podstawy Informatyki – cz. 1 – Visual Basic w Windows Script Host

Instrukcja laboratoryjna

Windows Script Host (WSH)

Microsoft® Windows® Script Host jest narzędziem pozwalającym na wykonywanie ciągów poleceń (skryptów) napisanych w wielu językach. Standardowo system Windows dostarcza języki VBScript oraz Jscript, ale dostępne są także: PerlScript, TCL, REXX i Python, dostarczane przez inne firmy. WSH jest idealny do pisania skryptów administracyjnych, skryptów logowania oraz wszelkiej automatyzacji pracy. Dzięki współpracy z technologią ActiveX pozwala na automatyzację wielu czynności samego środowiska Windows jak i aplikacji dla niego przeznaczonych.

Do najważniejszych zalet WSH należą:

- Możliwość uruchamiania zarówno przy pomocy środowiska graficznego jak i linii poleceń.
- Możliwość jednoczesnego uruchomienia wielu skryptów (przy pomocy jednego wywołania pliku .WSF).
- Niskie wymagania w odniesieniu do ilości pamięci operacyjnej.
- Rozpoznawanie użytego języka na podstawie rozszerzenia pliku.

Środowisko skryptów WSH jest zorientowane obiektowo. Dostarcza elementów, obiektów, metod i własności pozwalających na realizację następujących zadań:

- Wyświetlanie wiadomości na ekranie.
- Operacje na obiektach.
- Obsługa dysków sieciowych.
- Obsługa drukarek sieciowych.
- Pobieranie i modyfikowanie zmiennych środowiskowych.
- Dostęp do rejestru systemowego.

Obsługa WSH

Skrypty WSH można pisać przy pomocy dowolnego edytora tekstowego lub zewnętrznego środowiska IDE. System Windows nie dostarcza wyspecjalizowanych narzędzi do edycji skryptów. Rozszerzenia plików zawierających skrypty wskazują na język zapisu:

- VBS – **Visual Basic** Script
- JS – Java Srcipt
- WSF – format swobodny ze specyfikacją XML, mogący zwierać skrypty w dowolnych językach

Aby pokazać obsługę narzędzi WSH, musimy przygotować pierwszy skrypt. W tym celu należy utworzyć (np. przy pomocy Notatnika Windows/ Notepad++) plik Hello.vbs i umieścić w nim kod w Visual Basicu o treści:

```
WScript.Echo "Hello World!"
```

lub utworzyć plik Hello.js o treści:

```
WScript.Echo("Hello World!");
```

Do uruchamiania skryptów służą dwa polecenia: WScript (wersja WSH z interfejsem graficznym oraz CScript (wersja przeznaczona do uruchamiania skryptów z linii poleceń). Oba polecenia mają podobną składnię. Podstawowa różnica polega na tym, że CScript wyświetla komunikaty na konsoli, zaś WScript pokazuje je w postaci okienek.

Składnia polecenia:

```
CScript|Wscript nazwa_skryptu.rozszerzenie [opcja...] [argumenty...]
```

//B
: Tryb wsadowy - pomija wyświetlanie błędów i monitów w skryptach

//D
: Włącza debugowanie aktywne

//E:aparat
: Wykonuje skrypt przy użyciu aparatu

//H:CScript
: Ustawia program CScript.exe jako domyślny host skryptów

//H:Wscript
: Ustawia program WScript.exe jako domyślny host skryptów (ta wartość jest domyślna)

//I
: Tryb interaktywny (domyślny, przeciwieństwo trybu //B)

//Job:xxxx
: Wykonuje zadanie WSF

//Logo
: Wyświetla logo (domyślnie)

//Nologo
: Nie wyświetla logo: podczas wykonywania skryptu nie będzie wyświetlany transparent

//S
: Zapisuje bieżące opcje wiersza poleceń dla tego użytkownika

//T:nn
: Limit czasu w sekundach: maksymalny dozwolony czas wykonywania skryptu

//X
: Wykonuje skrypt w debugerze

//U
: Używa standardu Unicode dla przekierowań We/Wy z konsoli (tylko przy komendzie CScript)

Przykłady: `CScript test1.vbs //nologo`

Wykonuje skrypt nie wyświetlając na konsoli tekstu powitalnego (informującego o wersji WSH), zaś informacje o błędach wyświetla na konsoli.

```
CScript test1.vbs //T:10
```
Polecenie wykona skrypt i zatrzyma go, jeżeli czas pracy przekroczy 10 sekund.

Uruchomienie polecenia WScript bez wyspecyfikowania parametrów powoduje wyświetlenie okna pozwalającego na podstawową konfigurację WSH.

Podstawy składni Visual Basic

Instrukcja Visual Basic

Skrypt WSH napisany VB jest ciągiem instrukcji (programem). Pisanie programów dla WSH nie rózni się niczym od innych języków programowania. Poniżej postaram się zwrócić uwagę na najważniejsze różnice pomiędzy VB a popularnymi językami programowania. W VB każda instrukcja znajduje się w oddzielnej linii. Oznacza to, ze w przeciwieństwie do innych języków, w których instrukcje oddziela określony separator, nie możemy dowolnie przenosić linii. Jeśli wymagane jest przeniesienie części długiej instrukcji do następnej linii należy umieścić na końcu niedokończonej instrukcji znak "_":

```
początek długiej instrukcji _
oraz jej dokończenie
```

W przeciwieństwie do wielu innych języków VB **nie jest czuły** na wielkość liter - nazwa zapisana małymi literami będzie równoważna napisanej wielkimi:

```
NowaZmienna
NOWAZMIENNA
nowazmienna
```

Komentarze, czyli linie nieinterpretowane, należy rozpoczynać znakiem "`".

Najważniejszymi składnikami jeżyka VB są:

- zmienne
- operatory
- procedury
- funkcje

Zmienne

W VB nie ma potrzeby jawnego deklarowania zmiennych. Zmienna zostaje utworzona w chwili jej pierwszego przypisania. Istnieje jednak możliwość jawnej deklaracji zmiannej. Służy do tego słowo kluczowe *Dim*:

```
Dim NazwaZmiennej
```

Wersja VB dostępna w WSH ma ograniczone możliwości definiowania typów zmiennych. W przeciwieństwie do innych implementacji VB nie można jawnie określić typu danych przy pomocy słowa *As*.

VisualBasic pozwala także na deklaracje tablic czyli zbiorów zmiennych tego samego typu. Aby zadeklarować tablicę wpisujemy:

```
Dim nazwa_tablicy (rozmiar)
```

Odwołanie do elementów tablicy można wykonać następująco:

```
nazwa_tablicy (3) = 45
```

Dla tablicy zadeklarowanej Dim tablica(5) można się odwoływać indeksami od 0 do 4. Inny zakres indeksów można uzyskać operatorem to np.:

```
Dim tablica (5 to 10)
```

Operatory

Operatory to składniki języka programowania wykonujące proste operacje arytmetyczne, logiczne i inne. Zazwyczaj mają postać jednego lub pary znaków. Najczęściej używanym operatorem jest przypisanie "=". W VB ma ono postać:

```
zamienna = wartość
```

Najważniejszymi operatorami arytmetycznymi w VB są:

	dodawanie
/	dzielenie
*	mnożenie
^	potęgowanie
\	dzielenie całkowite
mod	modulo (reszta z dzielenia)

Najważniejszymi operatorami logicznymi w VB są:

=, >, <, =>, =<	porównania
and	koniunkcja
or	alternatywa
not	negacja
xor	alternatywa wykluczająca

Inne ważne operatory to:

=	przypisanie
&	łączenie łańcuchów znaków

Procedury

Za funkcjonalność danego programu odpowiadają wywołane w nim procedury. Wywołanie procedury w VB może mieć trzy różne składnie:

```
nazwa_proceedury(wartość)
```
lub
```
nazwa_procedury wartość
```
lub
```
nazwa_procedury nazwa_parametru:=wartość
```

Cechą charakterystyczną VB jest możliwość deklarowania parametrów domyślnych, których nie trzeba podawać podczas wywołania. Jeśli chcemy pominąć parametr po prostu nie podajemy jedo wartości. Np.:

```
procedura(wartość1, , wartość3)
```
lub

```
procedura parametr10:=wartość
```

Funkcje

Jedyna różnica pomiędzy procedurą a funkcją jest to, iż funkcja przekazuje wartość (może pełnić rolę r-wyrażenia). Wywołanie funkcji może mieć postać:

```
zmienna = funkcja(parametr)
```

VB w WSH dysponuje szeregiem funkcji i procedur standardowych. Najważniejsze z nich to:

CDate()	Przekształca wartość w datę
CInt()	Przekształca wartość w liczbę całkowitą
CStr()	Przekształca wartość w ciąg znaków
Date()	Zwraca datę systemową
DateDiff()	Wyznacza okres pomiędzy dwiema datami
DatePart()	Wyznacza cześć daty (rok, miesiąc, dzień, etc.)
GetObject()	Zwraca wskaźnik na obiekt automatyzacji (szczegóły dalej)
InputBox()	Pobiera dane od użytkownika
InStr()	Odszukuje dany ciąg w innym ciągu znaków
LCase()	Zmienia na małe litery
Left()	Pobiera zadaną liczbę znaków z lewej strony ciągu
Len()	Podaje długość ciągu znaków
MsgBox()	Wyświetla komunikat lub dialog z przyciskami
Now()	Zwraca bieżący czas i datę
Replace()	Zamienia podciąg znaków na inny
Right()	Pobiera zadaną liczbę znaków z prawej strony ciągu
Split()	Ciąg znaków podzielonych określonym separatorem zamienia w tablicę
Time()	Zwraca bieżący czas
UCase()	Zamienia na wielkie litery

Pytania:

- Jaki znak jest separatorem instrukcji w VB
- Jakie typy liczbowe występują w VB
- Wymień funkcje i procedury VB operujące na dacie i czasie
- Jakie operatory użyjesz aby sprawdzić podzielność dwóch liczb

Komunikacja z użytkownikiem

Większość programów posiada jakiś interfejs użytkownika. W najprostszym przypadku zbudowany jest on z instrukcji pobierających tekst od użytkownika oraz wyświetlający komunikaty. W WSH do dyspozycji mamy instrukcje wyświetlające proste okna dialogowe.

Aby wyświetlić tekst na ekranie należy skorzystać z instrukcji MsgBox:

```
MsgBox "Wyświetlany tekst"
```

Do wprowadzania danych do programu służy instrukcja InputBox:

```
zmienna = InputBox("Tekst zachęty: ")
```

Przykład wykorzystania obu poleceń:

```
Dim sMyName
sMyName = InputBox("Type your name")
MsgBox "Hello, " & sMyName
```

Innym sposobem wyświetlania komunikatów jest polecenie:

```
WScript.Echo "Hello World"
```

W przeciwieństwie do MsgBox działanie metody jest zróżnicowane w zależności od interpretera w którym został skrypt uruchomiony. W CScript zostanie wyświetlony w postaci tekstu na konsoli, zaś w Wscript pokazane zostanie okno z komunikatem.

Istnieje także możliwość wyświetlenia okienka z zapytaniem wyposażonego w przyciski umożliwiające odpowiedź. Służy do tego wymieniona wcześniej instrukcja MsgBox. Aby wyświetliła ona zestaw przycisków należy podać jej drugi parametr:

```
iResponse = MsgBox("Czy kontynuować?", 4)
```

Szczegóły dotyczące wartości drugiego parametru można znaleźć w dokumentacji.

Zadania – blok 1

1. Napisz program obliczający wynik dodawania dwóch liczb wprowadzonych przez użytkownika.
2. Napisz program obliczający średnią arytmetyczną trzech liczb wprowadzonych przez użytkownika.
3. Napisz program zwracający datę systemową oraz bieżacą datę i czas.
4. Napisz program wyznaczający okres pomiędzy dwiema datami.
5. Napisz program, który oblicza pole powierzchni wybranej przez Ciebie figury geometrycznej.

Struktury języka Visual Basic

Pętla For

Pętla For nazywana jest również pętlą For-Next. Pozwala ona na wielokrotne powtarzanie bloku kodu. W przeciwieństwie do pętli typu Do wykonuje to ściśle określoną ilość razy. Powtórzenia pętli zwane są również iteracjami pętli.

Składnia instrukcji For:

```
For ZmiennaLicznik = WartośćPoczątkowa To WartośćKońcowa [Step WartośćKroku]
   Blok instrukcji VB wykonywany
Next
```

Wyrażenia w nawiasach [] są opcjonalne i nie muszą występować w instrukcji. W przypadku, gdy pominięta zostanie część rozkazu określająca WartośćKroku, Visual Basic przyjmie domyślną WartośćKroku=1.

ZmiennaLicznik musi być zmienną (nie może być np. elementem sterującym). Jest ona wykorzystywana przez pętlę jako licznik powtórzeń.

Program wyświetlający liczby od 1 do 10 będzie miał postać:

```
Dim i
For i = 1 To 10
    WScript.Echo (CStr(i) + " ")
Next
```

W kodzie dodatkowo pojawiła się instrukcja przekształcająca zmienną typu liczbowego w zmienną tekstowa CStr. Operację odwrotną można wykonać funkcjami CInt lub CDbl. Poza tym wykonana jest operacja dodawania na dwóch wyrażeniach tekstowych. Operacja ta powoduje połączenie dwóch ciągów.

Warunek If

If jest najprostszą instrukcją wyboru. Dokonuje ona sprawdzenia podanego warunku i jeżeli jest on prawdziwy (ma wartość TRUE) wykonywany jest podany blok rozkazów. W przypadku gdy warunek jest fałszywy blok ten jest omijany.

Składnia instrukcji If:

```
If Warunek Then
   Blok instrukcji VB wykonywany gdy Warunek jest prawdziwy
End If
```

Przykładem wykorzystania If może być:

```
imie = InputBox("Jak się nazywasz?")

If imie = "Hubert" Then
   MsgBox ("Witaj Hubert")
End If
```

Instrukcja If wykonuje część programu tylko w przypadku, gdy sprawdzanny warunek ma wartość TRUE. Nie wykonuje go jednak w przypadku, gdy ma on wartość FALSE. Rozbudowaną wersją instrukcji If jest If ...Else. W przypadku gdy sprawdzany warunek ma wartość TRUE wykonuje ona blok rozkazów zawarty

między Then i Else, gdy zaś warunek ma wartość FALSE wykonuje blok rozkazów zawarty między Else i End If. Może ona więc "reagować" zarówno jeżeli warunek jest spełniony jak i gdy nie jest spełniony.

Instrukcja If...Else nazywana jest wyrażeniem wzajemnie wykluczającym się, ponieważ może zostać wykonany tylko jeden z zawartych w nim bloków rozkazów.

Składnia instrukcji If...Else:

```
If Warunek Then
   Blok instrukcji VB wykonywany gdy Warunek jest prawdziwy
Else
   Blok instrukcji VB wykonywany gdy Warunek nie jest prawdziwy
End If
```

Przykładem wykorzystania If..Else może być:

```
wynik = MsgBox("Wszystko w porządku?", vbYesNo)

If wynik = vbYes Then
   WScript.Echo ("Uzytkownik czuje się dobrze")
Else
   WScript.Echo ("Uzytkownik czuje się źle")
End If
```

Możliwość zagnieżdżania instrukcji If..Else (umieszczania jednej instrukcji If wewnątrz bloku rozkazów innej instrukcji If) wymusiło powstanie konstrukcji If...ElseIf. Jest to rozbudowana wersja instrukcji If ...Else. Pozwala ona na kolejne sprawdzanie wielu warunków.

Składnia instrukcji If...ElseIf:

```
If Warunek1 Then
   Blok instrukcji
ElseIf Warunek2 Then
   Blok instrukcji
Else
   Blok instrukcji
End If
```

Pętla Do

Pętla typu Do While jest chyba najbardziej powszechną pętlą Visual Basica. Wymaga ona zastosowania wyrażenia porównania. Wykorzystuje do tego celu operatory porównania. Pętla te jest ograniczona słowami kluczowymi Do i Loop. W ciele pętli może znajdować się jedna instrukcja lub cały blok instrukcji VB. Blok ten jest wykonywany tak długo jak długo podany warunek jest prawdziwy. Ważne jest więc, aby w ciele pętli umieścić instrukcje, które doprowadzą do tego, że podany warunek kiedyś stanie się fałszywy. W przeciwnym razie pętla się nie zakończy.

Składnia instrukcji Do While:

```
Do While (Warunek)
   Blok instrukcji VB wykonywany gdy Warunek jest prawdziwy
Loop
```

Blok instrukcji umieszczony w ciele pętli wykonywany jest tak długo, jak długo warunek pętli jest prawdziwy. Kiedy tylko stanie się fałszywy, to pętla kończy swoje działanie i przechodzi do rozkazu

następnego za pętlą (za Loop). Jeżeli warunek jest fałszywy już na samym początku przed wykonaniem pętli, to blok instrukcji w ciele pętli nie zostanie wykonany ani razu.Charakterystyczne dla pętli Do While jest to, że jej ciało może nie zostać wykonane ani razu. Nieco inaczej działa pętla Do...Loop While, której ciało jest zawsze wykonywane przynajmniej jeden raz.

Najczęściej w Warunku pętli wykorzystywana jest jakaś zmienna. Należy zadbać, aby w ciele pętli wartość tej zmiennej zmieniała się, aby doprowadzić do fałszywości Warunku i zakończyć działanie pętli.

Przykład użycia instrukcji Do While:

```
intLiczba=0
Do While (intLiczba<100)
   intLiczba=InputBox("Wpisz liczbę całkowitą", "Podaj liczbę")
Loop
```

Pętla typu Do Until jest pętlą podobną do Do While. Jedyna różnica polega na tym, że pętla Do Until wykonuje zawarty w niej blok rozkazów tak długo, jak długo podany warunek jest fałszywy. Wymaga ona zastosowania wyrażenia porównania. Wykorzystuje do tego celu operatory porównania. Pętla te jest ograniczona słowami kluczowymi Do i Loop. W ciele pętli może znajdować się jedna instrukcja lub cały blok instrukcji VB. Blok ten jest wykonywany tak długo jak długo podany warunek jest fałszywy. Ważne jest więc, aby w ciele pętli umieścić instrukcje, które doprowadzą do tego, że podany warunek kiedyś stanie się prawdziwy. W przeciwnym razie pętla się nie zakończy.

Składnia instrukcji Do Until:

```
Do Until (Warunek)
   Blok instrukcji VB wykonywany gdy Warunek jest fałszywy
Loop
```

Zadania – blok 2

6. Napisz program wykonujący jedno z czterech działań (dodawanie, odejmowanie, mnożenie i dzielenie) na dwóch liczbach. Skorzystaj z InputBox do wyboru działania. W przypadku dzielenia zaimplementuj zabezpieczenie dzielenia przez zero.
7. Rozbuduj poprzedni program 6 w taki sposób aby po wykonaniu zadania program prosił o podanie nowego działania. Program ma kończyć działanie po wpisaniu przez użytkownika słowa „koniec”.
8. Napisz program oparty na pętli for wypisujący następujący sformatowany blok znaków:

```
oooooooooo
oooooooooo
oooooooooo
oooooooooo
oooooooooo
```

9. Zmodyfikuj program 8 tak by wypisał następujący sformatowany blok znaków:

```
o
oo
ooo
oooo
ooooo
```

10. Napisz program wyświetlający liczby pierwsze mniejsze od 60. Czy stosując tablice można przyspieszyć działanie programu?
11. Napisz program wczytujący do tablicy 10 liczb i wyświetlających i sumę oraz średnią arytmetyczną.

12. Napisz program obliczający sumę ciągu $0^2 + 1^2 + 2^2 + 3^2 + ... + n^2$ dla wczytanego z klawiatury n.
13. Dana jest tablica ciągu liczb zakończonego zerem, którego nie wliczamy do ciągu. Obliczyć średnią arytmetyczną z wyrazów dodatnich ciągu oraz ustalić, ile wyrazów dzieli się bez reszty.
14. Napisz program, który oblicza wartość wyrażenia W(n), gdzie n jest liczbą naturalną.

$$W(n) = \begin{cases} \frac{1}{2}n + n^2 & dla\ n\ parzystych \\ (n-5)n & dla\ n\ nieparzystych \end{cases}$$