# Основы VBScript

Visual basic и VBS – ЭТО НЕ ОДНО И ТО ЖЕ.

**VBScript** (Visual Basic Script) - это скриптовый язык, разработанный компанией Microsoft. Он широко используется для автоматизации задач в Windows, создания небольших программ и веб-скриптов.

## Info

1)Русский текст не поддерживается.

Чтобы исправить нужно, создать блокнот, сохранить его в формате Unicode с расширением vbs.

## Комментирование:

' Комментарий в одной строке

' ' Комментарий в одной строке

'Многострочный

' комментарий

## Флаги

Option Explicit в VBScript (а также в VBA) — это оператор, который требует явного объявления всех переменных перед их использованием. Если вы не объявили переменную, а затем попытаетесь ее использовать, скрипт выдаст ошибку во время выполнения, а не просто создаст неявно переменную с нулевым значением.

## **Переменные**:

**VBS не поддерживает явное указание типов**

Dim i =5 нельзя

Dim i - только так

I =5

Dim имя\_переменной - объявление переменной

имя\_переменной = значение - присваивание значения

Option Explicit - обязательное объявление всех переменных

### Типы переменных:

String: строковый тип

Integer: целое число

Long: длинное целое число

Single: число с плавающей запятой

Double: число с плавающей запятой двойной точности

Boolean: логический тип (True/False)

Variant: универсальный тип

' Присвоение значений переменным

myString = "Это строка"

myInteger = 10

myDouble = 3.14159

myBoolean = True

' Вывод значений переменных

MsgBox("String: " & myString)

MsgBox("Integer: " & myInteger)

MsgBox("Double: " & myDouble)

MsgBox("Boolean: " & myBoolean)

## Структуры данных в VBS

### Узнать тип объекта TypeName()

Dim strText, intNumber, objFile

strText = "Hello, world!"

intNumber = 123

Set objFile = CreateObject("Scripting.FileSystemObject")

MsgBox TypeName(strText) ' Выведет "String"

MsgBox TypeName(intNumber) ' Выведет "Integer"

MsgBox TypeName(objFile) ' Выведет "Scripting.FileSystemObject"

### Массив

Создание массива

Получение элемента из массива:

Element = array\_name(\*number\_of\_element\*) // a = coords(5)

При создании массива индексация элементов начинается с **1.**

Узнать длину массива

Dim myArray(5)

myArray(0) = "Первый элемент"

myArray(1) = "Второй элемент"

myArray(3) = "Четвертый элемент" ' Обратите внимание, что индексы не обязательно должны быть последовательными

Dim arrayLength

arrayLength = UBound(myArray)

MsgBox "Длина массива: " & arrayLength ' Выведет 5

Чтобы получить длину нужно UBound(myArray) +1

### Динамический массив

#### 1)

Dim myArray() ' Объявляем массив без размера

ReDim myArray(0) ' Устанавливаем начальный размер 1 элемент (индекс 0)

myArray(0) = "Элемент 1"

ReDim Preserve myArray(1) ' Увеличиваем размер на 1 (индексы 0 и 1)

myArray(1) = "Элемент 2"

ReDim Preserve myArray(2) ' Увеличиваем размер на 1 (индексы 0, 1 и 2)

myArray(2) = "Элемент 3"

' Теперь myArray содержит 3 элемента: "Элемент 1", "Элемент 2", "Элемент 3"

\_\_\_\_\_\_\_\_\_

Важно: ReDim без Preserve удалит все данные массива.

Медленный и неэффективный при большом количестве изменений размера.

#### 2) ArrayList()

Добавляем объект через COM подключение (более эффективный)

<https://learn.microsoft.com/ru-ru/dotnet/api/system.collections.arraylist?view=net-8.0>

Function CreateArrayList()

Set CreateArrayList = CreateObject("System.Collections.ArrayList")

End Function

Dim arrList

Set arrList = CreateArrayList()

arrList.Add "Элемент 1"

arrList.Add "Элемент 2"

arrList.Add "Элемент 3"

' Теперь arrList содержит 3 элемента: "Элемент 1", "Элемент 2", "Элемент 3"

' 1. Узнать длину ArrayList

Dim listLength

listLength = arrList.Count ' Используем свойство .Count

' 2. Получить доступ к элементам ArrayList

' Индексы начинаются с 0

Dim firstElement

firstElement = arrList.Item(0)

' Выводим все элементы, используя цикл

app.PutMessage "Все элементы:"

Dim i

For i = 0 To arrList.Count - 1

app.PutMessage " " & arrList.Item(i)

Next

' Удалить по индексу

arrList.RemoveAt 1

' Удалить по значению

arrList.Remove "Элемент 3" ' Удаляет элемент со значением "Элемент 3"

' 4. Переназначить значения элементов ArrayList

' Заменяем элемент с индексом 0 новым значением

arrList.Item(0) = "Новый элемент 1"

' 5. Очистить ArrayList (удалить все элементы)

arrList.Clear

' Освобождаем объект (важно для COM-объектов в VBScript)

Set arrList = Nothing

### Классы

### Ссылки

### Dictation (словарь)

Set dict = CreateObject("Scripting.Dictionary")

dictpins.Add X, Y

Где X – ключ - стринг или int или флоат

Y – значение - любая структура данных

Получение данных из словаря

MSGBOX dictcons(X)

Проверка на наличие ключа

If dictcons.Exists(con.GetName) Then

Вывод всех ключей словаря

keysArray = dict.Keys ' Преобразуем коллекцию Keys в массив

For i = 0 To UBound(keysArray)

MsgBox keysArray(i) ' Выведет каждый ключ в отдельном окне сообщения

Next

Цикл вывода всех элементов словаря

x = 3

y = 1

out = print\_dictcons\_excl(dictcons, x, y, app)

app.PutMessage out

Function print\_dictcons\_excl(dictcons, x, y, app)

'массив ключей (все имена соединителей)

cons = dictcons.keys

'app.PutMessage "количество cons: " & ubound(cons) & " " & cons(ubound(cons))

for each key in cons

Set cur\_dictpins = dictcons(key)

for each keypin in cur\_dictpins.keys

cur\_pininfo = cur\_dictpins(keypin)

for each val in cur\_pininfo

app.PutMessage key & " " & keypin & " " & val

next

x = x + 1

next

next

print\_dictcons\_excl = 1

end function

## Объекты классов Set

Set a – создаёт экземпляр объекта класса.

Set fso = CreateObject("Scripting.FileSystemObject")

fso – экземпляр объекта класса

CreateObject("Scripting.FileSystemObject") – функция создает ссылку на объект класса

"Scripting.FileSystemObject" – название класса.

ИЛИ

Set e3 = ConnectToE3

ConnectToE3 – класс

Можно и так:

Dim e3

'Start out connection to E3

Set e3 = ConnectToE3

Set – ключевое слово в VBScript, которое используется для создания объектов и присвоения их переменным.

Что такое объекты?

Объекты – это элементы, которые представляют собой определенный тип данных и обладают свойствами и методами. В VBScript, например, FileSystemObject – это объект, который предоставляет доступ к файловой системе.

' Создание объекта FileSystemObject и присвоение его переменной fso

Set fso = CreateObject("Scripting.FileSystemObject")

fso.CreateTextFile("C:\temp\test.txt", True)

Важно:

Тип переменной: Переменная, к которой вы присваиваете объект с помощью Set, должна быть объявлена как Object.

Ссылочный тип: В VBScript, объекты хранятся не как значения, а как ссылки. Set создает ссылку на объект, а не копию объекта.

Пример:

Set obj1 = CreateObject("Scripting.FileSystemObject")

Set obj2 = obj1

' Изменение свойства объекта obj1

obj1.Path = "C:\newPath"

' Проверка свойства объекта obj2

MsgBox obj2.Path ' Выведет "C:\newPath"

' Объект obj2 также изменился, так как он является ссылкой на объект obj1.

Примеры использования Set:

Работа с файлами:

Set fso = CreateObject("Scripting.FileSystemObject")

Set file = fso.OpenTextFile("C:\test.txt", 1)

' Чтение содержимого файла

MsgBox file.ReadAll

Работа с реестром:

Set reg = CreateObject("Wscript.Shell")

MsgBox reg.RegRead("HKEY\_LOCAL\_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\ProductName")

### Просмотр информации об существующем классе.

Set obj = GetObject(, "CT.Application")

' Теперь, если "CT.Application" существует,

' вы можете получить доступ к его свойствам и методам:

' obj.SomeProperty = ...

' obj.SomeMethod()

## Операторы:

= - присваивание

+, -, \*, /, ^ - арифметические

& - конкатенация строк

=, <>, >, <, >=, <= - сравнения

And, Or, Not – логические

## Конструкции языка

### Условные операторы:

If...Then...Else...End If

If условие Then

[ElseIf условие Then]

[Else]

End If

### Циклы

For...Next

For i = начальное\_значение To конечное\_значение [Step шаг]

[код]

Next i

While...Wend

While условие

[код]

Wend

Do...Loop

Do [While/Until условие]

[код]

Loop

For i = 1 To 5

MsgBox(i)

Next i

For Each symId In dictSymbolIds.Keys()

' Установка идентификатора символа

sym.SetId symId

Next

### Функции

Sub имя\_функции()

[код]

End Sub

**Sub:**

* **Предназначение:** Процедуры Sub используются для выполнения набора инструкций, но **не возвращают никаких значений**. Они используются для организации кода в модульные блоки, чтобы сделать его более читаемым и управляемым.
* **Синтаксис:**

Sub имя\_процедуры()

' Код процедуры

End Sub

* **Пример:**

Sub PrintMessage()

MsgBox("Hello, world!")

End Sub

* **Вызов:**

Call PrintMessage ' Вызов процедуры Sub

Function имя\_функции()

[код]

имя\_функции = значение

End Function

**Function:**

* **Предназначение:** Функции Function используются для выполнения набора инструкций и **возврата значения**. Они могут принимать аргументы и возвращать результат, который может быть использован в других частях кода.
* **Синтаксис:**

Function имя\_функции()

' Код функции

имя\_функции = значение ' Возврат значения

End Function

* **Вызов:**

result = AddNumbers(2, 3) ' Вызов функции и присвоение возвращаемого значения переменной

**Код**

Dim a, b, c

a = 10

b = 5

c = a + b

Function Sum(a, b)

Sum = a + b

End Function

Dim result

result = Sum(3, 7)

MsgBox(result)

### Возвращение функцией 2х значений

x=1

y=1

dim arr

arr = one(x,y)

app.PutMessage "x = " & arr(0) & " y = " & arr(1)

function one(x,y)

dim arr(1)

arr(0) = x

arr(1) = y

one = arr

end function

### With - End With

Это ключевые слова VBA, которые используются для упрощения кода при работе с одним и тем же объектом. Вместо того чтобы постоянно писать

Worksheets("Sheet1").Range("B2").Borders(xlEdgeBottom).свойство/метод ,

мы можем один раз написать With Worksheets("Sheet1").Range("B2").Borders(xlEdgeBottom) и затем, внутри блока With ... End With, использовать . для доступа к свойствам и методам этого объекта.

Пример

With Worksheets("Sheet1").Range("B2").Borders(xlEdgeBottom)

.LineStyle = xlContinuous

.Weight = xlThin

.ColorIndex = 3

End With

## Utility функции

MsgBox("Hello, world!")

WScript.Sleep 5000

### Мой тройной цикл словарей

Каркас моего цикла (силовые машины поиск наконечников)

'цикл соединителей

For u=1 to cntcon

'Подключение к соединителю

con.SetId(conids(u))

'Все пины соединителя

cntpins = con.GetPinIds ( pinids)

Set pin = job.CreatePinObject()

'Цикл пинов

Set dictpins = CreateObject("Scripting.Dictionary")

For k=1 to cntpins

'Подключение к пину

pinid = pinids(k)

out = get\_dictpins(pininfo, pinid, pin, nameatt\_nak\_name, nameatt\_nak\_sap)

'добавление нового pininfo как новый элемент словаря пинов

dictpins.Add pin.GetName, pininfo

next

### Просмотр всех полей класса

Не работает

Dim obj

Set obj = CreateObject("Scripting.FileSystemObject")

' Вывод имен всех свойств объекта

For Each prop In obj

WScript.Echo prop

Next

ИЛИ

Dim obj

Set obj = GetObject("winmgmts:\\\\.\\root\\cimv2")

' Вывод информации о компоненте

WScript.Echo obj.Path

WScript.Echo obj.Name

Или

Set objFSO = CreateObject("Scripting.FileSystemObject")

Dim strScriptPath : strScriptPath = objFSO.GetAbsolutePathName(WScript.ScriptFullName)

strScriptPath = objFSO.GetParentFolderName(strScriptPath)

Dim strFilePath : strFilePath = strScriptPath & "\filesystemobject\_properties.txt"

Set objFile = objFSO.OpenTextFile(strFilePath, 2, True)

Dim objProperties : Set objProperties = objFSO.Properties

objFile.WriteLine "Свойства объекта FileSystemObject:"

For Each objProperty In objProperties

objFile.WriteLine objProperty.Name & " = " & objProperty.Value

Next

objFile.Close

ИЛИ

' Функция для вывода всех возможных свойств и методов класса

Sub PrintClassAttributesAndMethods(className, outputFileName)

Dim fso, outputFile, obj, prop, method

Dim scriptPath

' Получаем путь к папке скрипта

scriptPath = WScript.ScriptFullName

scriptPath = Left(scriptPath, InStrRev(scriptPath, "\") - 1)

' Инициализация FileSystemObject для записи в файл

Set fso = CreateObject("Scripting.FileSystemObject")

Set outputFile = fso.CreateTextFile(scriptPath & "\" & outputFileName, True)

' Создаем объект класса

Set obj = CreateObject(className)

' Получаем тип объекта

Set classType = obj.GetType()

' Выводим все возможные свойства

outputFile.WriteLine "Свойства класса " & className & ":"

For Each prop In classType.Properties

outputFile.WriteLine "Свойство: " & prop.Name

Next

' Выводим все возможные методы

outputFile.WriteLine "Методы класса " & className & ":"

For Each method In classType.Methods

outputFile.WriteLine "Метод: " & method.Name

Next

outputFile.Close

Set outputFile = Nothing

Set fso = Nothing

End Sub

' Пример использования функции

PrintClassAttributesAndMethods "Scripting.FileSystemObject", "output.txt"

### Вывод всех объектов массива

' Функция для вывода атрибутов и методов объекта

Sub PrintAttributesAndMethods(obj, outputFileName)

Dim fso, outputFile, prop

Dim scriptPath

' Получаем путь к папке скрипта

scriptPath = WScript.ScriptFullName

scriptPath = Left(scriptPath, InStrRev(scriptPath, "\") - 1)

' Инициализация FileSystemObject для записи в файл

Set fso = CreateObject("Scripting.FileSystemObject")

Set outputFile = fso.CreateTextFile(scriptPath & "\" & outputFileName, True)

outputFile.WriteLine "Атрибуты и методы объекта:"

' Перечисление свойств объекта

For Each prop In obj

outputFile.WriteLine "Свойство/Метод: " & prop

Next

outputFile.Close

Set outputFile = Nothing

Set fso = Nothing

End Sub

' Пример использования функции

' Создайте объект для тестирования

Set exampleObj = CreateObject("Scripting.Dictionary")

exampleObj.Add "ключ1", "значение1"

exampleObj.Add "ключ2", "значение2"

' Вызываем функцию, передавая имя файла для сохранения

PrintAttributesAndMethods exampleObj, "output.txt"

## Основные рабочие классы

### Scripting.FileSystemObject

<https://learn.microsoft.com/ru-ru/office/vba/language/reference/user-interface-help/filesystemobject-object>

Set fs = CreateObject("Scripting.FileSystemObject")

Функции:

fs.FileExists(‘path\obj’)

fs.DeleteFile (‘path\obj’, True)

True – это необязательный аргумент, который, если указан, означает, что файл должен быть удален даже если он является только для чтения.

### WScript

<http://www.script-coding.com/WSH/WScript.html>

WScript - это объект в VBScript (Visual Basic Scripting Edition), предоставляющий доступ к различным функциям и свойствам, связанным с интерпретатором скриптов Windows Script Host (WSH).

Управление скриптом:

WScript.Echo: Выводит текст в консоль или в окно сообщения (в зависимости от того, как запущен скрипт).

WScript.Sleep: Останавливает выполнение скрипта на заданное время (в миллисекундах).

WScript.Quit: Прекращает выполнение скрипта.

WScript.Arguments: Предоставляет доступ к аргументам, переданным скрипту при его запуске.

Информация о скрипте:

WScript.FullName: Полный путь к файлу скрипта.

WScript.ScriptName: Имя файла скрипта.

WScript.Version: Версия WSH.

Взаимодействие с пользователем:

WScript.CreateObject: Создает объект COM (Component Object Model).

WScript.Interactive: Возвращает True, если скрипт запущен в интерактивном режиме (в окне командной строки).

## Ресурсы:

[Официальная документация Microsoft по VBScript](https://docs.microsoft.com/en-us/previous-versions/windows/internet-explorer/ie-developer/scripting/aa905122(v=vs.85))

[W3Schools - VBScript Tutorial](https://www.w3schools.com/vbscript/default.asp)

[VBScript Tutorial - Learn VBScript From Scratch](https://www.tutorialspoint.com/vbscript/index.htm)

## Запуск кода

1) Сохранить код из notepad++ как hello.vbs.

При клике на файл Windows запустит интерпретатор VBScript, который выполнит ваш код.

**2) Из командной строки:** Откройте командную строку (cmd) и перейдите в папку, где вы сохранили файл hello.vbs. Введите команду cscript hello.vbs и нажмите Enter.

3) Через notepad++.

В Notepad++ вставьте код MsgBox("Привет, мир!")

Сохраните файл с расширением .vbs. Например, hello.vbs.

**Запуск через Run:**

В Notepad++ нажмите **“Run”** -> **“Run…”**

В появившемся окне в поле “Command” вставьте: cscript "C:\путь\к\файлу\hello.vbs".

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# Основы VBApplication

Язык использующий Visual Basic, но библиотеки классов для работы Microsoft Office приложений уже встроены в этот язык

Отличие от VBS – заключается в том, что VBA работает внутри приложений офиса, а vbs требует COM подключения.

В целом VBS может выполнять функции VBA подключаясь к приложению следующим образом:

Set objExcel = CreateObject("Excel.Application")

Но библиотеки могут быть ограничены.

## Документация

<https://learn.microsoft.com/ru-ru/office/vba/api/overview/>

## Excel

<https://learn.microsoft.com/ru-ru/office/vba/api/overview/excel>

### Инфо/Определения

Если надо вывести число с нулями вначале в эксель

fit = 123

pininfo(3) = Chr(160) & CStr(fit)

Чтобы выводилась кириллица нужно сохранить файл VBS в формате ANSI или UTF-16 LE

1) Найти все открытые книги excel

Set excl = CreateObject("Excel.Application")

excl.Visible = True

Set file = excl.Workbooks

Workbooks возвращает класс, хранящий все книги excel открытые сейчас

2) Работать с конкретной книгой

Set book = excl.Workbooks(1) или excl.Workbooks.Add

Возвращает объект одной конкретной книги эксель

3) Листы одной книги

Set sheets = Book.Worksheets

Worksheets возвращает объекты всех листов

3) Конкретный лист

Set sheets = Book.Worksheets(1)

Worksheets(1) возвращает объект листа

### Программное управление excel

#### Создание файла эксель

' Создаем объекты Excel

Set objExcel = CreateObject("Excel.Application")

objExcel.Visible = True

Set objWorkbook = objExcel.Workbooks.Add

Set objWorksheet = objWorkbook.Worksheets(1)

#### Заполнение ячеек

Заполнение первой ячейки

sheet.Cells(1, 1).Value = 24

sheet.Cells(1, 2).Value = "Hello EXCEL!!"

#### Cтили ячейки

Объект range (ячейка)

<https://learn.microsoft.com/ru-ru/office/vba/api/excel.range(object)>

Высота и ширина

sheet.Cells(1, 1).Value = "Зона"

sheet.Cells(1, 1).RowHeight = 50

sheet.Cells(1, 1).ColumnWidth = 20

Границы

Значения стиля линии(LineStyle), толщины линии(Weight) задаются константами, в коде ниже нужно определить эти константы

Значения констант можно посмотреть здесь, если перейти по нужному свойству

<https://learn.microsoft.com/ru-ru/office/vba/api/excel.borders#properties>

With Worksheets("Sheet1").Range("B2").Borders

.LineStyle = xlContinuous

.Weight = xlThin

.ColorIndex = 3

End With

Если нужно изменить конкретную границу, то

Worksheets("Sheet1").Range("B2").Borders(**xlEdgeRight**) .LineStyle = xlContinuous

Ориентация

sheet.Cells(1, 1).Orientation = 90

Шрифт

Свойства Sheet.Cells(1,1).Font

<https://learn.microsoft.com/ru-ru/office/vba/api/excel.font(object)>

With sheet.Cells(1, 1)

.Font.Size = 14

.Font.Italic = True

.Font.Name = "GOST type A"

.Font.Bold = True

End With

Выравнивание текста по ячейке

Константы можно найти в horizontal aligment и vertical aligment

<https://learn.microsoft.com/ru-ru/office/vba/api/excel.range(object)>