Типы данных в VBA

Введение

- VBA, как и большинство других систем программирования, разделяет обрабатываемые данные на числа, даты, текст и другие типы.
- Тип данных (data type) – это термин, относящийся к определенным видам данных, которые VBA сохраняет и которыми может манипулировать.

Типы данных

Тип	Размер (байт)	Диапазон значений
Byte	1	От 0 до 255
Boolean	2	True или False
Integer	2	От -32 768 до 32 767
Long	4	От -2 1 47 483 648 до 2 1 47 483 647
Single	4	От -3,402823E38 до -1,401298E-45 для (-) от 1,401298E-45 до 3,402823E38 для (+)
Double	8	От -1 ,7976931 3486232E308 до -4,940656458 41247E-324 для (-) от 4,94065645841 247E-324 до 1, 7976931 3486232E308 для (+)
Currency	8	От -922 337 203 685 477,5808 до 922 337 203 685 477,5807

Типы данных

Тип данных	Размер (байт)	Диапазон значений
Decimal (масштабируемое целое число)*	**14**	+/-792281 6251 426433759354395033 5 с 28 знаками справа от запятой; минимальное ненулевое значение имеет вид +/-0,0000000000000000000000000001
Date (даты и время)	**8**	От 1 января 100 г. до 31 декабря 9999 г.
Object (объект)	**4**	Любой указатель объекта

*** - используется только внутри типа Variant через функцию CDec**

Типы данных

Тип данных	Размер (байт)	Диапазон значений
String (строка переменной длины)	10 + длина строки	От 0 до ~2 миллиардов
String (строка постоянной длины)	Длина строки	От 1 до ~65 400
Variant (числовые подтипы)	16	Любое числовое значение вплоть до границ диапазона для типа Double
Variant (строковые подтипы)	22 + длина строки	Как для строки (string) переменной длины
Тип данных, определяемый пользователем (с помощью ключевого слова Type)	Объем определяется элементами	Диапазон каждого элемента определяется его типом данных

Описание переменных

- Общий вид описание переменных:
- Dim Переменная As Тип данных
- Примеры: Dim i as integer, j as Byte, _
- strName as String, cMon as Currency
- Dim sHeight as Single, ch as Chart
- Dim wbk as Workbook
- Если пропустить описание переменной или не указать его, то переменной будет присвоен тип Variant. Однако, этого следует избегать.
- В объявлении Dim i, j as Byte – i будет типа **Variant(!)**
- Чтобы избежать такой проблемы рекомендуется в область описания помещать оператор **Option Explicit** (или включать соответствующий флажок в настройках)

Символы для описания переменных

Допустимо объявлять типы, добавляя специальные символы к имени переменной.

Пример: Dim Name$

Символ	Тип переменной	Символ	Тип переменной
%	Integer	#	Double
&	Long	@	Currency
!	Single	$	String

Допустимые имена

- Длина имени не должна превышать 255 символов
- Имя не может содержать точек, пробелов и следующих символов: %, &, !, #, @, $
- Имя может содержать любую комбинацию букв, цифр и символов, начинающуюся с буквы
- Имена должны быть уникальны внутри области, в которой они определены
- Не следует использовать имена, совпадающие с ключевыми словами VBA и именами встроенных функций и процедур
- Следует избегать использования l, O и c в качестве переменных. (L и o можно использовать без ограничений)

Обозначения в кодах

- […] - Код в скобках опциональный (т.е. может быть опущен)
- Public | Private - Public или Private
- <Инструкции> - произвольные инструкции внутри кода

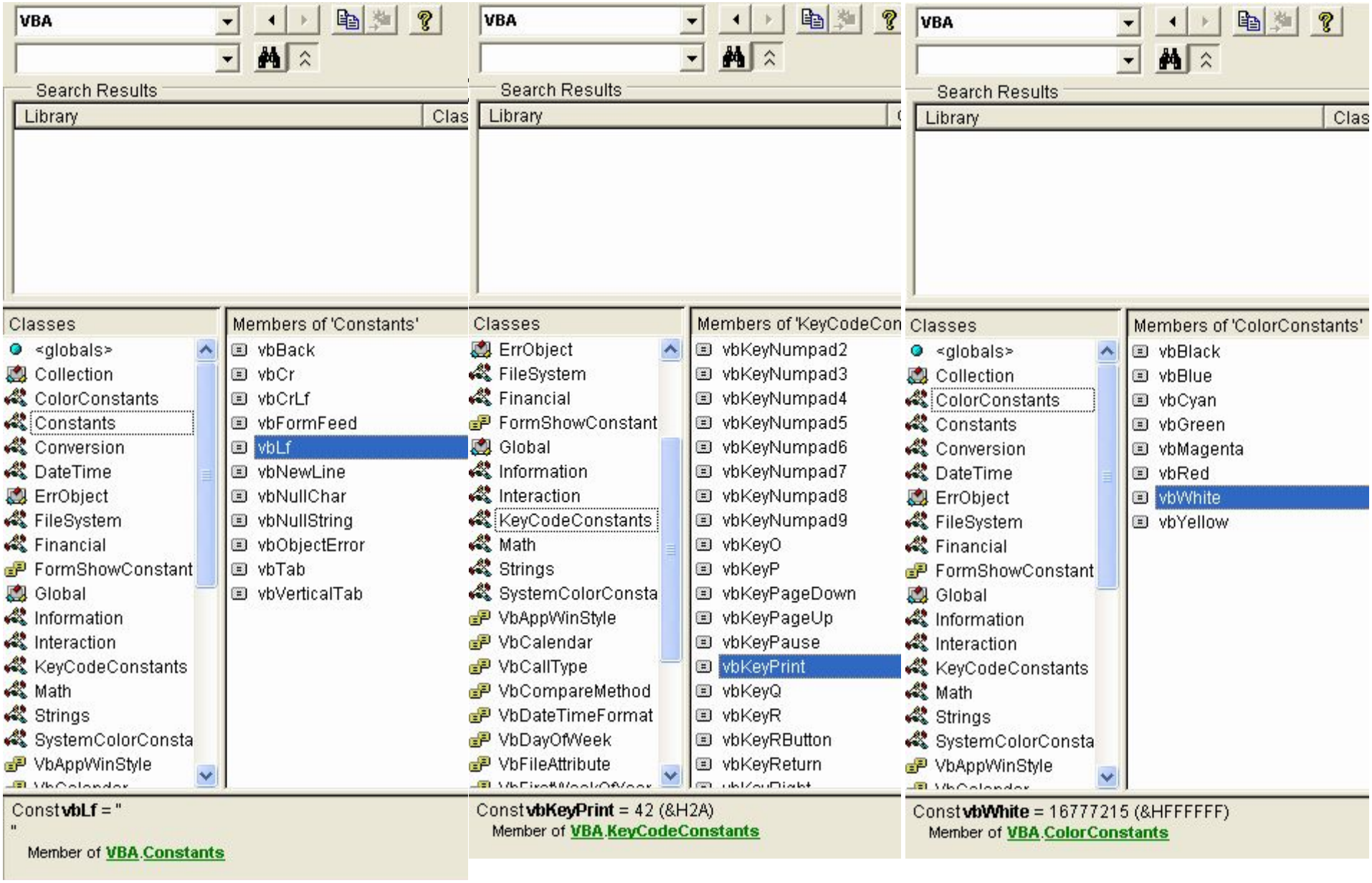
Константы

Константы, в отличие от переменных, не могут изменять свои значения. Использование констант делает программы легче читаемыми и позволяет проще вносить исправления.

Синтаксис:

- [Public | Private] Const ИмяКонстанты [As Тип] = Значение
- Примеры: Const Index As Single = 5
- Const strName As String = "Иван"
- Const i = 200
- Const j = 8.2 * 2, string1 = "строка "
- Const k = 8 + 5

- В Excel есть ряд встроенных констант. Их имена начинаются с букв vb.
- vbCrLf – Перенос строки
- vbTab – табуляция
- Подробный список см. в Object Browser


VBA
Search Results
Library
Clas
Classes
<globals>
Collection
ColorConstants
Constants
Conversion
DateTime
ErrObject
FileSystem
Financial
FormShowConstant
Global
Information
Interaction
KeyCodeConstants
Math
Strings
SystemColorConsta
VbAppWinStyle
Members of 'Constants'
vbBack
vbCr
vbCrLf
vbFormFeed
vbLf
vbNewLine
vbNullChar
vbNullString
vbObjectError
vbTab
vbVerticalTab
Const vbLf = "
"
Member of VBA.Constants
Classes
ErrObject
FileSystem
Financial
FormShowConstant
Global
Information
Interaction
KeyCodeConstants
Math
Strings
SystemColorConsta
VbAppWinStyle
VbCalendar
VbCallType
VbCompareMethod
VbDateTimeFormat
VbDayOfWeek
VbFileAttribute
Members of 'KeyCodeCon
vbKeyNumpad2
vbKeyNumpad3
vbKeyNumpad4
vbKeyNumpad5
vbKeyNumpad6
vbKeyNumpad7
vbKeyNumpad8
vbKeyNumpad9
vbKeyO
vbKeyP
vbKeyPageDown
vbKeyPageUp
vbKeyPause
vbKeyPrint
vbKeyQ
vbKeyR
vbKeyRButton
vbKeyReturn
Const vbKeyPrint = 42 (&H2A)
Member of VBA.KeyCodeConstants
Classes
<globals>
Collection
ColorConstants
Constants
Conversion
DateTime
ErrObject
FileSystem
Financial
FormShowConstant
Global
Information
Interaction
KeyCodeConstants
Math
Strings
SystemColorConsta
VbAppWinStyle
Members of 'ColorConstants'
vbBlack
vbBlue
vbCyan
vbGreen
vbMagenta
vbRed
vbWhite
vbYellow
Const vbWhite = 16777215 (&HFFFFFF)
Member of VBA.ColorConstants

Область действия переменных

- Термин область действия *(scope)* относится к области процедуры или модуля VBA, где данная переменная, процедура или другой идентификатор, являются доступными. Переменные, процедуры и идентификаторы, которые доступны только в процедуре, имеют область действия процедурного уровня, а те, которые доступны для всех процедур в модуле, имеют область действия модульного уровня.
- Переменная, объявленная в процедуре, является доступной только в этой процедуре. Эта переменная реально существует только во время выполнения этой процедуры.

```
Option Explicit 'Директива компилятору требующая явного
'задания всех переменных. Действует только на модуль.
Public ProjectVar As Boolean 'Переменная уровня проекта
Dim ModuleVar As Boolean 'Переменная модульного уровня
Const strMod As String = "Иван" 'константа модульного уровня
Public Const strProj As String = "Иван" 'константа проектного уровня
```

```
Sub Variables()
Dim ProcedureVar As Boolean 'Переменная уровня процедуры
'Dim ProcedureVar As Boolean 'Если переменная с именем,
'совпадающая с именем переменной модульного уровня объявлена в процедуре,
'То используется именно она, а не переменная модульного уровня!
End Sub
```

Строковые типы данных

- Строковые типы могут быть следующими:

- с фиксированной длиной
  - Декларация: Dim str as String*N
  - Разницу см. на скриншоте

- с произвольной длиной
  - Декларация: Dim str as String

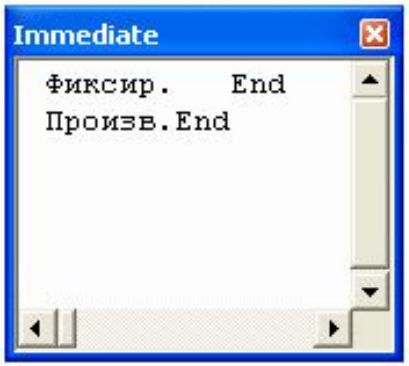
Строка, не помещающаяся в окно редактора может быть разбита с использованием оператора “&”

```
Sub Test()
Dim strFix As String * 10
Dim strFree As String
Dim str1 As String, str2 As String
Dim Str As String

strFix = "Фиксир."
strFree = "Произв."
str1 = strFix & "End"
str2 = strFree & "End"

Str = "Если надо разбить длинную текстовую строку," & _
"то её необходимо разбивать с использванием конкатенации строк"

Debug.Print str1
Debug.Print str2
End Sub
```



Операции VBA

- В VBA реализуются 3 основных типа операций:

- Математические - выполняются над числами, и их результатом являются числа
- Отношения - применяются не только к числам, и их результатом являются логические значения, например x>y
- Логические - используются в логических выражениях и их результатом являются логические значения, например Not x And y

Математические операции

Операнд1 + Операнд2	Сложение
Операнд1 - Операнд2	Вычитание
- Операнд1	Перемена знака
Операнд1 * Операнд2	Умножение
Операнд1 / Операнд2	Деление
Операнд1 \ Операнд2	Целочисленное деление
Операнд1 Mod Операнд2	Остаток от деления по модулю
Операнд1 ^ Операнд2	Возведение в степень

Типы данных результата выражения

- Порядок точности для численных типов данных VBA от наименее точного до наиболее точного следующий:
- Byte, Integer, Long, Single, Double, Currency
- Тип данных результата выражения сложения обычно тот же, что и наиболее точный тип в этом выражении. Например, если выражение содержит оба типа **Integer** и **Long**, результатом такого выражения будет тип **Long**. Однако существуют исключения, в частности, если выражение включает переменные типа **Variant**.

Исключения (сложение)

- Далее перечисляются эти исключения:

- Результатом сложения типа **Single** и **Long** является **Double.**
- Если складывать тип **Date** с любым другим типом данных, результатом выражения всегда будет тип **Date.**
- Если результат выражения сложения присваивается переменной **Variant**, имеющей в данный момент тип **Integer**, и если результат выражения больше, чем (переполняет) диапазон значений для типа **Integer**, то VBA преобразует результат в **Long.** После присваивания переменная **Variant** также имеет тип **Long.**
- Если результат выражения сложения присваивается переменной **Variant**, имеющей в данный момент тип **Long**, **Single** или **Date**, и если результат выражения переполняет диапазон численного типа, VBA преобразует результат в **Double.** После присваивания переменная типа **Variant** также имеет тип данных **Double.**
- Если любой операнд в выражении сложения является равным **Null** или вычисляется до **Null**, то результатом выражения сложения также будет **Null.**

- **(Null** – это особое значение, которое можно присваивать только переменным типа **Variant** для обозначения того, что они не содержат действительных данных.)

Типы данных результатов (-) (*)

• Вычитание

- VBA следует тем же правилам для определения типа данных результата выражения вычитания, что и для выражений, сложения, но имеются следующие дополнительные правила:
- Если один из операндов в выражении вычитания является типом Date, то результат выражения имеет тип Date.
- Если оба операнда в выражении являются типом Date, то результат выражения имеет тип Double.

• Умножение

- Оба операнда в выражении умножения должны быть численными выражениями или строками, которые VBA может преобразовать в число.
- Тип данных результата выражения умножения обычно тот же, что и наиболее точный тип в этом выражении. VBA следует тем же правилам для определения типа данных результата выражения умножения, что и для выражений, использующих сложение. В выражениях умножения все переменные Variant, которые содержат значения типа Date, преобразуются в численные значения.

Типы данных результатов деления (/)

- Если любой операнд в выражении деления имеет значение Null, то результатом выражения также является Null. Тип данных выражения со знаком деления с плавающей точкой – обычно Double, но имеется следующее исключение:
  - Если оба операнда в выражении деления имеют тип Integer или Single, то результат выражения деления с плавающей точкой имеет тип Single, если только результат выражения не переполняет диапазон значений для типа Single. Если результат переполняет диапазон для типа Single, то VBA преобразует результат в тип Double.

Возведение в степень

- Оба операнда в выражении возведения в степень должны быть численными выражениями или строками, которые VBA может преобразовать в числа. Операнд слева от знака возведения в степень может быть отрицательным числом, только если операнд справа является целым. Если какой-либо операнд является равным Null, то результатом выражения возведения в степень также будет Null, иначе результат выражения будет иметь тип Double.

Операции отношения

Операнд1 < Операнд2	Меньше
Операнд1 > Операнд2	Больше
Операнд1 <= Операнд2	Меньше или равно
Операнд1 >= Операнд2	Больше или равно
Операнд1 <> Операнд2	Не равно
Операнд1 = Операнд2	Равно
Операнд1 Is Операнд2	Сравнение двух операндов, содержащих ссылки на объекты
Операнд1 Like Операнд2	Сравнение двух строковых выражений

Знаки операций сравнения Is и Like

Is	E1 is E2	Оба операнда должны быть значения типа Object. True, если E1 ссылается на тот же объект, что и E2
Like	E1 Like E2	Подобие. Оба операнда должны быть типа String. True, если E1 совпадает с E2

Символы совпадения с образцом для оператора Like

Символ	Соответствие
#	Любая одиночная цифра от 0 до 9
*	Любое количество символов в любой комбинации или отсутствие символов
?	Любой одиночный символ
[list]	list – список определённых символов. Совпадение с любым одиночным символом
[!list]	Совпадение с любым одиночным символом не

Примеры использования оператора Like

"aBBBa" Like "a*a"	True
"F" Like "[A-Z]"	True
"F" Like "[!A-Z]"	False
"a2a" Like "a#a"	True
"aM5b" Like "a[L-P]#[!c-e]"	True
"BAT123khg" Like "B?T*"	True
"CAT123khg" Like "B?T*"	False

Логические операции

Операнд1 And Операнд2	Логическое умножение
Операнд1 Or Операнд2	Логическое сложение
Операнд1 Xor Операнд2	Исключающее ИЛИ
Not Операнд1	Логическое отрицание
Операнд1 Imp Операнд2	Импликация
Операнд1 Eqv Операнд2	Эквивалентность
Другие операции	
Строка1 & Строка2	Сцепление строк

Таблицы истинности логических операций

- Конъюнкция (логическое умножение)

Также называется «И» (AND)

a	b	$a \wedge b$
0	0	0
0	1	0
1	0	0
1	1	1

- Дизъюнкция (логическое сложение)

Также называется «ИЛИ» (OR)

a	b	$a \vee b$
0	0	0
0	1	1
1	0	1
1	1	1

- Сложение по модулю 2 (XOR)

Также называется исключающее «ИЛИ»

a	b	$a \oplus b$
0	0	0
0	1	1
1	0	1
1	1	0

- Отрицание (NOT)

a	$\neg a$
0	1
1	0

- Импликация (Из… следует…) (Imp)

a	b	$a \Rightarrow b$
0	0	1
0	1	1
1	0	0
1	1	1

- Равносильность (Эквивалентность) (Eqv)

a	b	$a \Leftrightarrow b$
0	0	1
0	1	0
1	0	0
1	1	1

Приоритеты операций

1	()	**8**	>,<,<=,>=,<>,=
2	^	**9**	Not
3	- (знак)	**10**	And
4	*, /	**11**	Or
5	\	**12**	Xor
6	Mod	**13**	Eqv
7	+, -	**14**	Imp

Обращение к ячейке по адресу

Допустим, у нас есть два открытых файла: «Книга1» и «Книга2», причем, файл «Книга1» активен и в нем находится исполняемый код VBA.
В общем случае при обращении к ячейке неактивной рабочей книги «Книга2» из кода файла «Книга1» прописывается полный путь:

```
1  Workbooks("Книга2.xlsm").Sheets("Лист2").Range("C5")
2  Workbooks("Книга2.xlsm").Sheets("Лист2").Cells(5, 3)
3  Workbooks("Книга2.xlsm").Sheets("Лист2").Cells(5, "C")
4  Workbooks("Книга2.xlsm").Sheets("Лист2").[C5]
```

Удобнее обращаться к ячейке через свойство рабочего листа Cells(номер строки, номер столбца), так как вместо номеров строк и столбцов можно использовать переменные. Обратите внимание, что при обращении к любой рабочей книге, она должна быть открыта, иначе произойдет ошибка. Закрытую книгу перед обращением к ней необходимо открыть.
Теперь предположим, что у нас в активной книге «Книга1» активны «Лист1» и ячейка на нем «A1». Тогда обращение к ячейке «A1» можно записать следующим образом:

```
1  ActiveCell
2  Range("A1")
3  Cells(1, 1)
4  Cells(1, "A")
5  [A1]
```

Запись информации в ячейку

Содержание ячейки определяется ее свойством «**Value**», которое в VBA Excel является свойством по умолчанию и его можно явно не указывать. Записывается информация в ячейку при помощи оператора присваивания «=»:

```
1  Cells(2, 4).Value = 15
2  Cells(2, 4) = 15
3  Range("A1") = "Этот текст записываем в ячейку"
4  ActiveCell = 28 + 10*36
```