**Создание собственных диалоговых окон средствами VBA**

В этой заметке описываются методы создания пользовательских диалоговых окон, которые существенно расширяют стандартные возможности Excel. Диалоговые окна – это наиболее важный элемент пользовательского интерфейса в Windows. Они применяются практически в каждом приложении Windows, и большинство пользователей неплохо представляет, как они работают. Разработчики Excel создают пользовательские диалоговые окна с помощью пользовательских форм (UserForm). Кроме того, в VBA имеются средства, обеспечивающие создание типовых диалоговых окон.[[1]](#footnote-1)

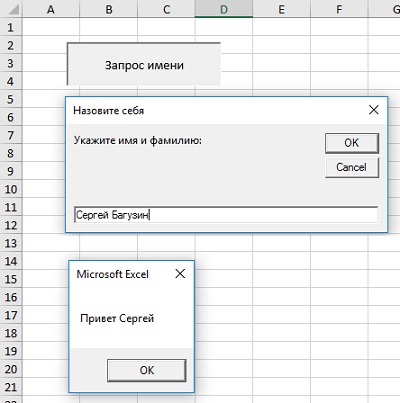


Рис. 1. Работа процедуры GetName

Перед тем как приступить к изучению тонкостей создания диалоговых окон на основе пользовательских форм, следует научиться использовать некоторые встроенные инструменты Excel, предназначенные для вывода диалоговых окон.

### Использование окон ввода данных

*Окно ввода данных* — это простое диалоговое окно, которое позволяет пользователю ввести одно значение. Например, можно применить окно ввода данных, чтобы предоставить пользователю возможность ввести текст, число или диапазон значений. Для создания окна ввода предназначены две функции InputBox: одна— в VBA, а вторая является методом объекта Application.

### Функция InputBox в VBA

Функция имеет следующий синтаксис:

InputBox(запрос [, заголовок] [, по\_умолчанию] [, xpos] [, ypos] [, справка, раздел])

* Запрос. Указывает текст, отображаемый в окне ввода (обязательный параметр).
* Заголовок. Определяет заголовок окна ввода (необязательный параметр).
* По\_умолчанию. Задает значение, которое отображается в окне ввода по умолчанию (необязательный параметр).
* xpos, ypos. Определяют координаты верхнего левого угла окна ввода на экране (необязательные параметры).
* Справка, раздел. Указывают файл и раздел в справочной системе (необязательные параметры).

Функция InputBox запрашивает у пользователя одно значение. Она всегда возвращает строку, поэтому результат нужно будет преобразовать в числовое значение. Текст, отображаемый в окне ввода, может достигать 1024 символов (длину допускается изменять в зависимости от ширины используемых символов). Если определить раздел справочной системы, то в диалоговом окне будет отображена кнопка *Справка*.

Процедура GetName запрашивает у пользователя полное имя (имя и фамилию). Затем программа выделяет имя и отображает приветствие в окне сообщения (см. рис. 1; код функции можно найти в файле *VBA inputbox.xlsm*).

Sub GetName()

Dim UserName As String

Dim FirstSpace As Integer

Do Until UserName <> ""

UserName = InputBox("Укажите имя и фамилию: ", \_

"Назовите себя")

Loop

FirstSpace = InStr(UserName, " ")

If FirstSpace <> 0 Then

UserName = Left(UserName, FirstSpace - 1)

End If

MsgBox "Привет " & UserName

End Sub

Обратите внимание: функция InputBox вызывается в цикле Do Until. Это позволяет убедиться в том, что данные введены в окно. Если пользователь щелкнет на кнопке *Отмена* или не введет текст, то переменная UserName будет содержать пустую строку, а окно ввода данных появится повторно. Далее в процедуре будет предпринята попытка получить имя пользователя путем поиска первого символа пробела (для этого применяется функция InStr). Таким образом, можно воспользоваться функцией Left для получения всех символов, расположенных слева от символа пробела. Если символ пробела не найден, то используется все введенное имя.

Если строка, предоставленная в качестве результата выполнения функции InputBox, выглядит как число, ее можно преобразовать с помощью функции VBA Val.

В процедуре GetWord пользователю предлагается ввести пропущенное слово (рис. 2). Этот пример также иллюстрирует применение именованных аргументов (*р* и *t*). Текст запроса выбирается из ячейки А1 рабочего листа.

Sub GetWord()

Dim TheWord As String

Dim p As String

Dim t As String

p = Range("A1")

t = "Какое слово пропущено?"

TheWord = InputBox(prompt:=p, Title:=t)

If UCase(TheWord) = "ВОДОКАЧКУ" Then

MsgBox "Верно."

Else

MsgBox "Не верно."

End If

End Sub

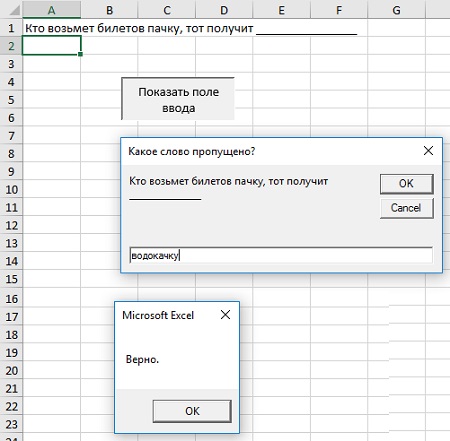


Рис. 2. Использование функции VBA inputBox, отображающей запрос

### Метод Excel InputBox

Метод Excel InputBox по сравнению с функцией VBA InputBox предоставляет три преимущества:

* возможность задать тип возвращаемого значения;
* возможность указать диапазон листа путем выделения с помощью мыши;
* автоматическая проверка правильности введенных данных.

Метод InputBox имеет следующий синтаксис.

InputBox(запрос, [, заголовок], [, по\_умолчанию], [, слева], [, сверху], [, справка, раздел], [, тип])

* Запрос. Указывает текст, отображаемый в окне ввода (обязательный параметр).
* Заголовок. Определяет заголовок окна ввода (необязательный параметр).
* По\_умолчанию. Задает значение, которое отображается в окне ввода по умолчанию (необязательный параметр).
* Слева, сверху. Определяют координаты верхнего левого угла окна ввода на экране (необязательные параметры).
* Справка, раздел. Указывают файл и раздел в справочной системе (необязательные параметры).
* Тип. Указывает код типа данных, который будет возвращаться методом (необязательный параметр; значения см. рис. 3).

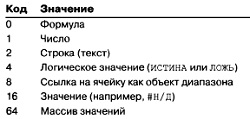


Рис. 3. Коды типов данных, возвращаемые методом Excel InputBox

Используя сумму приведенных выше значений, можно возвратить несколько типов данных. Например, для отображения окна ввода, которое принимает текстовый или числовой тип данных, установите код равным 3 (1 + 2 или *число* + *текст*). Если в качестве кода типа данных применить значение 8, то пользователь сможет ввести в поле адрес ячейки или диапазона ячеек. Пользователь также можент выбрать диапазон на текущем рабочем листе.

В процедуре EraseRange используется метод InputBox. Пользователь может указать удаляемый диапазон (рис. 4). Адрес диапазона вводится в окно вручную, или выделяется мышью на листе. Метод InputBox с кодом 8 возвращает объект Range (обратите внимание на ключевое слово Set). После этого выбранный диапазон очищается (с помощью метода Clear). По умолчанию в поле окна ввода отображается адрес текущей выделенной ячейки. Если в окне ввода щелкнуть на кнопке *Отмена*, то оператор On Error завершит процедуру.

Sub EraseRange()

Dim UserRange As Range

On Error GoTo Canceled

Set UserRange = Application.InputBox \_

(Prompt:="Удаляемый диапазон:", \_

Title:="Удаление диапазона", \_

Default:=Selection.Address, \_

Type:=8)

UserRange.Clear

UserRange.Select

Canceled:

End Sub

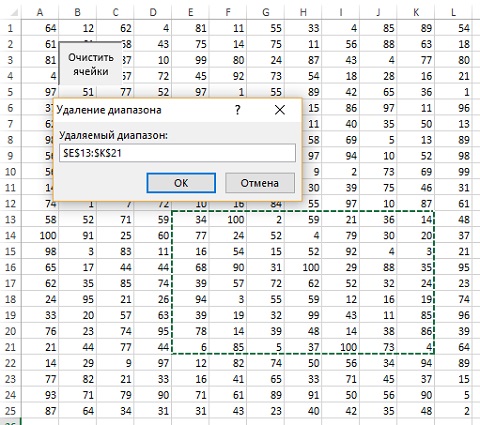


Рис. 4. Пример использования метода InputBox для выбора диапазона

Если в процедуре EraseRange ввести не диапазон адресов, то Excel отобразит сообщение (рис. 5) и позволит пользователю повторить ввод данных.

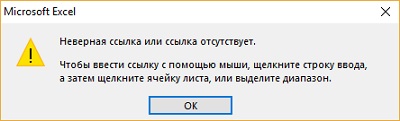


Рис. 5. Метод InputBox автоматически проверяет вводимые данные

### Функция VBA MsgBox

Функция VBA MsgBox служит для отображения сообщения. Также она передает результат щелчка на кнопке ОК или Отмена). Синтаксис функции:

MsgBox(запрос[, кнопки][, заголовок][, справка, раздел])

* Запрос. Определяет текст, который будет отображаться в окне сообщения (обязательный параметр).
* Кнопки. Содержит числовое выражение (или константу), которое определяет кнопки, отображаемые в окне сообщения (необязательный параметр; рис. 6). Также можно задать кнопку по умолчанию.
* Заголовок. Содержит заголовок окна сообщения (необязательный параметр).
* Справка, раздел. Указывают файл и раздел справочной системы (необязательные параметры).

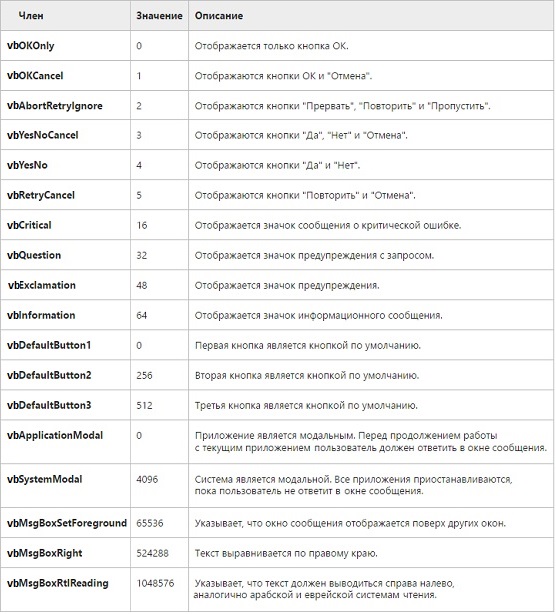


Рис. 6. Константы и значения, используемые для выбора кнопок в функции MsgBox

Первая группа значений (0–5) описывает номер и тип кнопок в диалоговом окне. Вторая группа (16, 32, 48, 64) описывает стиль значка. Третья группа (0, 256, 512) определяет, какая кнопка назначена по умолчанию. Четвертая группа (0, 4096) определяет модальность окна сообщения. Пятая указывает, показывать ли окно сообщений поверх других окон, устанавливает выравнивание и направление текста. В процессе сложения чисел для получения окончательного значения аргумента Buttons следует использовать только одно число из каждой группы.

Можно использовать функцию MsgBox в качестве процедуры (для отображения сообщения), а также присвоить возвращаемое этой функцией значение переменной. Функция MsgBox возвращает результат, представляющий кнопку, на которой щелкнул пользователь. В следующем примере отображается сообщение и не возвращается результат (код функций, приведенных в этом разделе см. также в файле *VBA msgbox.xlsm*).

Sub MsgBoxDemo()

MsgBox "При выполнении макроса ошибок не произошло."

End Sub

Чтобы получить результат из окна сообщения, присвойте возвращаемое функцией MsgBox значение переменной. В следующем коде используется ряд встроенных констант (рис. 7), которые упрощают управление возвращаемыми функцией MsgBox значениями.

Sub GetAnswer ()

Dim Ans As Integer

Ans = MsgBox("Продолжать?", vbYesNo)

Select Case Ans

Case vbYes

' ... [код при Ans равно Yes]

Case vbNo

' ... [код при Ans равно No]

End Select

End Sub



Рис. 7. Константы, возвращаемые MsgBox

Функция MsgBox возвращает переменную, имеющую тип Integer. Вам необязательно использовать переменную для хранения результата выполнения функции MsgBox. Следующая процедура представляет собой вариацию процедуры GetAnswer.

Sub GetAnswer2 ()

If MsgBox("Продолжать?", vbYesNo) = vbYes Then

' ... [код при Ans равно Yes]

Else

' ... [код при Ans равно No]

End If

End Sub

В следующем примере функции используется комбинация констант для отображения окна сообщения с кнопками Да, Нет и знаком вопроса (рис. 8). Вторая кнопка (*Нет*) используется по умолчанию. Для простоты константы добавлены в переменную Config.

Private Function ContinueProcedure() As Boolean

Dim Config As Integer

Dim Ans As Integer

Config = vbYesNo + vbQuestion + vbDefaultButton2

Ans = MsgBox("Произошла ошибка. Продолжить?", Config)

If Ans = vbYes Then ContinueProcedure = True \_

Else ContinueProcedure = False

End Function

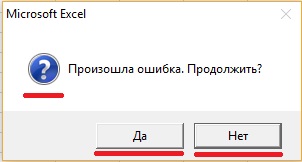


Рис. 8. Параметр *Кнопки* функции MsgBox определяет кнопки, которые отображаются в окне сообщения

В файле *VBA msgbox.xlsm* функция ContinueProcedure в демонстрационных целях представлена в виде процедуры. Функция ContinueProcedure может вызываться из другой процедуры. Например, оператор

If Not ContinueProcedure() Then Exit Sub

вызывает функцию ContinueProcedure (которая отображает окно сообщения). Если функция возвращает значение ЛОЖЬ (т.е. пользователь щелкнул на кнопке *Нет*), то процедура будет завершена. В противном случае выполняется следующий оператор.

Если в сообщении необходимо указать разрыв строки (рис. 9), воспользуйтесь константой vbCrLf (или vbNewLine):

Sub MultiLine()

Dim Msg As String

Msg = "Это первая строка." & vbCrLf & vbCrLf

Msg = Msg & "Вторая строка." & vbCrLf

Msg = Msg & "Третья строка."

MsgBox Msg

End Sub

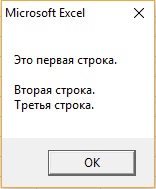


Рис. 9. Разбиение сообщения на несколько строк

Для включения в сообщение символа табуляции применяется константа vbTab. В процедуре ShowRange окно сообщения используется для отображения диапазона значений размером 10 строк на 3 столбца — ячейки А1:С10 (рис. 10). В этом случае столбцы разделены с помощью константы vbTab. Новые строки вставляются с помощью константы vbCrLf. Функция MsgBox принимает в качестве параметра строку, длина которой не превышает 1023 символов. Такая длина задает ограничение на количество ячеек, которое можно отобразить в сообщении.

Sub ShowRange()

Dim Msg As String

Dim r As Integer, c As Integer

Msg = ""

For r = 1 To 10

For c = 1 To 3

Msg = Msg & Cells(r, c).Text

If c <> 3 Then Msg = Msg & vbTab

Next c

Msg = Msg & vbCrLf

Next r

MsgBox Msg

End Sub

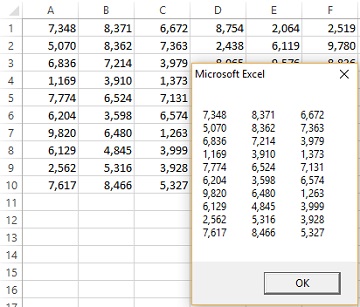


Рис. 10. Текст в этом окне сообщения содержит символы табуляции и разрыва строк

### Метод Excel GetOpenFilename

Если приложению необходимо получить от пользователя имя файла, то можно воспользоваться функцией InputBox, но этот подход часто приводит к возникновению ошибок. Более надежным считается использование метода GetOpenFilename объекта Application, который позволяет удостовериться, что приложение получило корректное имя файла (а также его полный путь). Данный метод позволяет отобразить стандартное диалоговое окно *Открытие документа*, но при этом указанный файл не открывается. Вместо этого метод возвращает строку, которая содержит путь и имя файла, выбранные пользователем. По окончании данного процесса с именем файла можно делать все что угодно. Синтаксис (все параметры необязательные):

Application.GetOpenFilename(фильтр\_файла, индекс\_фильтра, заголовок, множественный\_выбор)

* Фильтр\_файла. Содержит строку, определяющую критерий фильтрации файлов (необязательный параметр).
* Индекс\_фильтра. Указывает индексный номер того критерия фильтрации файлов, который используется по умолчанию (необязательный параметр).
* Заголовок. Содержит заголовок диалогового окна (необязательный параметр). Если этот параметр не указать, то будет использован заголовок *Открытие документа*.
* Множественный\_выбор. Необязательный параметр. Если он имеет значение ИСТИНА, можно выбрать несколько имен файлов. Имя каждого файла заносится в массив. По умолчанию данный параметр имеет значение ЛОЖЬ.

Аргумент Фильтр\_файла определяет содержимое раскрывающегося списка *Тип файлов*, находящегося в окне *Открытие документа*. Аргумент состоит из строки, определяющей отображаемое значение, а также строки действительной спецификации типа файлов, в которой находятся групповые символы. Оба элемента аргумента разделены запятыми. Если этот аргумент не указывать, то будет использовано значение, заданное по умолчанию: "Все файлы (\*.\*),\*.\*". Первая часть строки Все файлы (\*.\*) – то текст, отображаемый в раскрывающемся списке тип файлов. Вторая часть строки \*.\* указывает тип отображаемых файлов.

В следующих инструкциях переменной Filt присваивается строковое значение. Эта строка впоследствии используется в качестве аргумента фильтр\_файла метода GetOpenFilename. В данном случае диалоговое окно предоставит пользователю возможность выбрать один из четырех типов файлов (кроме варианта *Все файлы*). Если задать значение переменной Filt, то будет использоваться оператор конкатенации строки VBA. Этот способ упрощает управление громоздкими и сложными аргументами.

Filt = "Текстовые файлы (\*.txt),\*.txt," & \_

"Файлы Lotus (\*.prn),\*.prn," & \_

"Файлы, разделенные запятой (\*.csv),\*.csv," & \_

"Файлы ASCII (\*.asc),\*.asc," & \_

"Все файлы (\*.\*),\*.\*"

В следующем примере у пользователя запрашивается имя файла. При этом в поле типа файлов используются пять фильтров (код содержится в файле *prompt for file.xlsm*).

Sub GetImportFileName()

Dim Filt As String

Dim FilterIndex As Integer

Dim Title As String

Dim FileName As Variant

' Настройка списка фильтров

Filt = "Text Files (\*.txt),\*.txt," & \_

"Lotus Files (\*.prn),\*.prn," & \_

"Comma Separated Files (\*.csv),\*.csv," & \_

"ASCII Files (\*.asc),\*.asc," & \_

"Все файлы (\*.\*),\*.\*"

' По умолчанию используется фильтр \*.\*

FilterIndex = 3

' Заголовок окна

Title = "Выберите импортируемый файл"

' Получение имени файла

FileName = Application.GetOpenFilename \_

(FileFilter:=Filt, \_

FilterIndex:=FilterIndex, \_

Title:=Title)

' При отмене выйти из окна

If FileName = False Then

MsgBox "Файл не выбран."

Exit Sub

End If

' Отображение полного имени и пути

MsgBox "Вы выбрали " & FileName

End Sub

На рис. 11 показано диалоговое окно, которое выводится на экран после выполнения этой процедуры (по умолчанию предлагается фильтр \*.csv).

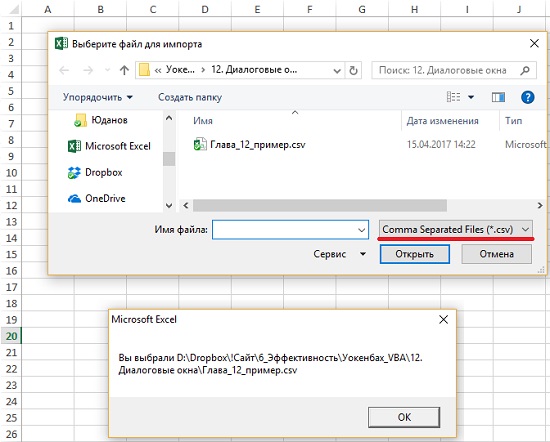


Рис. 11. Метод GetOpenFilename отображает диалоговое окно, в котором выбирается файл

В следующем примере пользователь может, удерживая нажатыми клавиши <Shift> и <Ctrl>, выбрать в окне несколько файлов. Обратите внимание, что событие использования кнопки *Отмена* определяется по наличию переменной массива FileName. Если пользователь не щелкнул на кнопке *Отмена*, то результирующий массив будет состоять как минимум из одного элемента. В этом примере список выбранных файлов отображается в окне сообщения.

Sub GetImportFileName2()

Dim Filt As String

Dim FilterIndex As Integer

Dim FileName As Variant

Dim Title As String

Dim i As Integer

Dim Msg As String

' Установка списка фильтров файлов

Filt = "Text Files (\*.txt),\*.txt," & \_

"Lotus Files (\*.prn),\*.prn," & \_

"Comma Separated Files (\*.csv),\*.csv," & \_

"ASCII Files (\*.asc),\*.asc," & \_

"All Files (\*.\*),\*.\*"

' Отображает \*.\* по умолчанию

FilterIndex = 5

' Настройка заголовка диалогового окна

Title = "Выберите файл для импорта"

' Получение имени файла

FileName = Application.GetOpenFilename \_

(FileFilter:=Filt, \_

FilterIndex:=FilterIndex, \_

Title:=Title, \_

MultiSelect:=True)

' Выход в случае отмены работы с диалоговым окном

If Not IsArray(FileName) Then

MsgBox "Файл не выбран."

Exit Sub

End If

' Отображение полного пути и имени файлов

For i = LBound(FileName) To UBound(FileName)

Msg = Msg & FileName(i) & vbCrLf

Next i

MsgBox "Было выбрано:" & vbCrLf & Msg

End Sub

Обратите внимание: переменная FileName определена как массив переменного типа (а не как строка в предыдущем примере). Причина заключается в том, что потенциально FileName может содержать массив значений, а не только одну строку.

### Метод Excel GetSaveAsFilename

Данный метод отображает диалоговое окно *Сохранение документа* и дает пользователю возможность выбрать (или указать) имя сохраняемого файла. В результате возвращается имя файла, но никакие действия не предпринимаются. Синтаксис (все параметры необязательные):

Application.GetSaveAsFilename(начальное\_имя, фильтр\_файла, индекс\_фильтра, заголовок, текст\_кнопки)

* Начальное\_имя. Указывает предполагаемое имя файла.
* Фильтр\_файла. Содержит критерий фильтрации отображаемых в окне файлов.
* Индекс\_фильтра. Код критерия фильтрации файлов, который используется по умолчанию.
* Заголовок. Определяет текст заголовка диалогового окна.

### Получение имени папки

Для того чтобы получить имя файла, проще всего воспользоваться описанным выше методом GetOpenFileName. Но если нужно получить лишь имя папки (без названия файла), лучше воспользоваться методом объекта Excel FileDialog. Следующая процедура отображает диалоговое окно, в котором можно выбрать папку (см. также файл *get directory.xlsm*). С помощью функции MsgBox отображается имя выбранной папки (или сообщение *Отменено*).

Sub GetAFolder()

With Application.FileDialog(msoFileDialogFolderPicker)

.InitialFileName = Application.DefaultFilePath & "\"

.Title = "Выберите местоположение резервной копии."

.Show

If .SelectedItems.Count = 0 Then

MsgBox "Отменено"

Else

MsgBox .SelectedItems(1)

End If

End With

End Sub

Объект FileDialog позволяет определить начальную папку путем указания значения свойства InitialFileName. В примере в качестве начальной папки применяется путь к файлам Excel, заданный по умолчанию.

### Отображение диалоговых окон Excel

Создаваемый вами код VBA может вызывать на выполнение многие команды Excel, находящиеся на ленте. И если в результате выполнения команды открывается диалоговое окно, ваш код может делать выбор в диалоговом окне (даже если само диалоговое окно не отображается). Например, следующая инструкция VBA эквивалентна выбору команды *Главная* –> *Редактирование* –> *Найти и выделить* –> *Перейти* и указанию диапазона ячеек А1:СЗ с последующим щелчком на кнопке ОК. Но само диалоговое окно *Переход* при этом не отображается (именно это и нужно).

Application.Goto Reference:=Range("А1:СЗ")

Иногда же приходится отображать встроенные окна Excel, чтобы пользователь мог сделать свой выбор. Для этого используется коллекция Dialogs объекта Application. Учтите, что в настоящее время компания Microsoft прекратила поддержу этого свойства. В предыдущих версиях Excel пользовательские меню и панели инструментов создавались с помощью объекта *CommandBar*. В версиях Excel 2007 и Excel 2010 этот объект по-прежнему доступен, хотя и работает не так, как раньше. Начиная с версии Excel 2007 возможности объекта *CommandBar* были существенно расширены. В частности, объект *CommandBar* можно использовать для вызова команд ленты с помощью VBA. Многие из команд, доступ к которым открывается с помощью ленты, отображают диалоговое окно. Например, следующая инструкция отображает диалоговое окно *Вывод на экран скрытого листа* (рис. 12; см. также файл *ribbon control names.xlsm*):

Application.CommandBars.ExecuteMso("SheetUnhide")

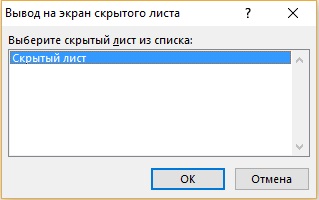


Рис. 12. Диалоговое окно, отображаемое в результате выполнения указанного выше оператора

Метод ExecuteMso принимает лишь один аргумент, idMso, который представляет элемент управления ленты. К сожалению, сведения о многих параметрах в справочной системе отсутствуют.

В файле *ribbon control names.xlsm* описаны все названия параметров команд ленты Excel. Поэкспериментируйте с параметрами, перечисленными в этой рабочей книге. Многие из них вызывают команды немедленно (без промежуточных диалоговых окон). Но большинство из них генерирует ошибку при использовании в неправильном контексте. Например, Excel отображает сообщение об ошибке, если команда Functionwizard вызывается в случае выбора диаграммы.

В результате выполнения следующего оператора отображается вкладка *Шрифт* диалогового окна *Формат ячеек*:

Application.CommandBars.ExecuteMso("FormatCellsFontDialog")

На самом деле пользоваться объектами CommandBar не стоит, поскольку вряд ли они будут поддерживаться в будущих версиях Excel.

### Отображение формы ввода данных

Многие пользователи применяют Excel для управления списками, информация в которых ранжирована по строкам. В Excel поддерживается простой способ работы с подобными типами данных с помощью встроенных форм ввода данных, которые могут создаваться автоматически. Подобная форма предназначена для работы как с обычным диапазоном, так и с диапазоном, оформленным в виде таблицы (с помощью команды *Вставка* –> *Таблицы* –> *Таблица*). Пример формы ввода данных показан на рис. 13 (см. также файл *data form example.xlsm*).

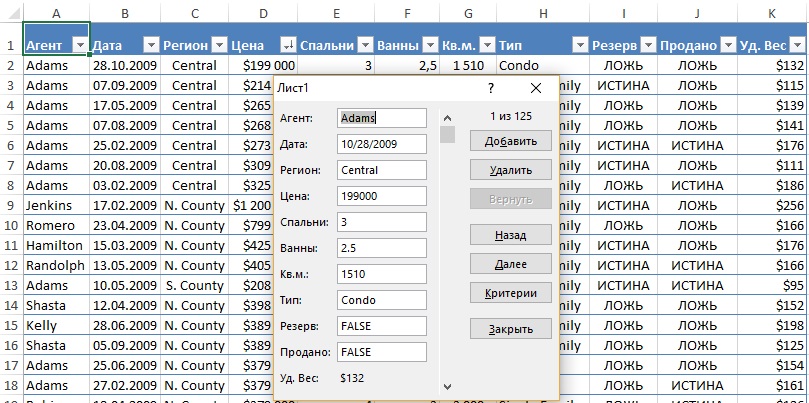


Рис. 13. Некоторые пользователи предпочитают применять встроенные формы ввода данных Excel для ввода сведений

В силу каких-то неизвестных причин на ленте Excel отсутствует команда, обеспечивающая доступ к форме ввода данных. Подобную команду можно добавить на панель быстрого доступа. Для этого выполните следующие действия.

1. Щелкните правой кнопкой мыши на панели быстрого доступа и в контекстном меню выберите параметр *Настройка панели быстрого доступа*.
2. На экране появится вкладка *Панель быстрого доступа* диалогового окна *Параметры Excel*.
3. В раскрывающемся списке *Выбрать команды из* выберите параметр *Команды не на ленте*.
4. В появившемся списке выберите параметр *Форма*.
5. Щелкните на кнопке *Добавить* для добавления выбранной команды на панель быстрого доступа.
6. Щелкните на кнопке ОК для закрытия диалогового окна *Параметры Excel*.

После выполнения перечисленных выше действий на панели быстрого доступа появится новый значок.

Для работы с формой ввода данных следует структурировать данные таким образом, чтобы Excel распознавал их в виде таблицы. Начните с указания заголовков столбцов в первой строке диапазона вводимых данных. Выделите любую ячейку в таблице и щелкните на кнопке *Форма* панели быстрого доступа. Excel отображает диалоговое окно, в котором будут вводиться данные. Для перемещения между текстовыми полями в целях ввода информации используйте клавишу <Tab>. Если ячейка содержит формулу, результат вычислений отображается в виде текста (а не в формате поля ввода данных). Другими словами, невозможно изменить формулы с помощью формы ввода данных.

По завершении ввода данных в форму щелкните на кнопке *Создать*. После этого Excel вводит данные в строку рабочего листа, а также очищает диалоговое окно для ввода следующей строки данных.

Используйте метод ShowDataForm для отображения формы ввода данных Excel. Единственное требование заключается в том, что активная ячейка должна находиться в диапазоне. Следующий код активизирует ячейку А1 (в таблице), а затем отображает форму ввода данных.

Sub DisplayDataForm()

Range("A1").Select

ActiveSheet.ShowDataForm

End Sub

1. По материалам книги [Джон Уокенбах. Excel 2010. Профессиональное программирование на VBA](http://baguzin.ru/wp/?p=16255). – М: Диалектика, 2013. – С. 387–403. [↑](#footnote-ref-1)