**Примеры использования элемента управления ListBox**

Ранее я рассмотрел [методы создания пользовательских форм и основы работы с ними](http://baguzin.ru/wp/?p=16994) (если вы никогда не работали с пользовательскими формами, рекомендую для начала прочитать указанную заметку). Далее привел целый ряд практически полезных [примеров пользовательских диалоговых окон](http://baguzin.ru/wp/?p=17036). В настоящей заметке подробнее рассказывается об использовании элемента управления *ListBox*.[[1]](#footnote-1)



Рис. 1. Установка свойства *RowSource* на этапе разработки

Элемент управления *ListBox* довольно гибкий в использовании, но работа с ним может оказаться достаточно сложной. В большинстве случаев методы, описанные здесь, могут применяться и для работы с элементом управления *ComboBox*.

При работе с элементом управления *ListBox* следует учитывать, что:

* Опции списка элемента управления *ListBox* могут извлекаться из диапазона ячеек (определяемого свойством *RowSource*) или добавляться с помощью VBA (для этого используется метод *Addltem*).
* Элемент управления *ListBox* может быть применен для выделения одной или нескольких опций. Соответствующее поведение определяется значением свойства *MultiSelect*.
* Если элемент управления *ListBox* не настроен на выделение нескольких опций, то значение элемента управления *ListBox* может связываться с ячейкой листа с помощью свойства *ControlSource*.
* Элемент управления *ListBox* может отображаться без предварительно выбранной опции (для этого необходимо установить свойство *Listlndex* равным –1). Но как только пользователь выделит одну из опций списка, отменить выделение будет невозможно. Исключение из этого правила— значение свойства *MultiSelect* равно *True*.
* Элемент управления *ListBox* может содержать несколько столбцов (что указывается в свойстве *ColumnCount*) и даже описательные заголовки (для этого используется свойство *ColumnHeads*).
* Вертикальный размер элемента управления *ListBox*, помещенного в пользовательское диалоговое окно, не всегда совпадает с вертикальным размером объекта *UserForm* на экране.
* Опции списка элемента управления *ListBox* могут отображаться в виде флажков, если разрешено выделение нескольких элементов, или в виде переключателей, если поддерживается только единичное выделение. Это поведение определяется значением свойства *ListStyle*.

### Добавление опций в элемент управления *ListBox*

Перед отображением пользовательского диалогового окна, которое содержит элемент управления *ListBox*, следует заполнить элемент управления *ListBox* необходимыми опциями. Элемент управления *ListBox* заполняется на этапе разработки проекта посредством указания диапазона ячеек, которые содержат необходимые данные. Его также можно заполнить на этапе выполнения, воспользовавшись кодом VBA для добавления всех опций списка.

В двух примерах этого раздела предполагается следующее:

* используется диалоговое окно *UserForm* с именем *UserForm1*;
* диалоговое окно *UserForm1* содержит элемент управления *ListBox*, который называется *ListBox1*;
* рабочая книга содержит лист *Лист1*, в диапазоне А1:А12 которого определены опции, отображаемые в элементе управления *ListBox*.

### Добавление опций в элемент управления *ListBox* на этапе разработки

Для добавления опций в элемент управления *ListBox* на этапе разработки необходимо, чтобы они хранились в диапазоне ячеек рабочей книги. Воспользуйтесь свойством *RowSource* для указания диапазона, который содержит опции элемента управления *ListBox*. На рис. 1 показано окно *Properties* для элемента управления *ListBox* (см. также файл *listbox fill.xlsm*) Свойство *RowSource* установлено равным Лист1!А1: А12. Когда диалоговое окно *UserForm* отображается на экране, элемент управления *ListBox* содержит двенадцать опций из этого диапазона. Опции добавляются в элемент управления *ListBox* на этапе разработки, сразу после того, как диапазон определяется в качестве значения свойства *RowSource*.

Удостоверьтесь, что в значении свойства *RowSource* присутствует имя листа. В противном случае элемент управления *ListBox* будет применять указанный диапазон в активном рабочем листе. Иногда следует идентифицировать диапазон максимально точно, указав даже имя рабочей книги. Лучше всего сначала определить имя для диапазона, а затем включить его в состав разработанного кода. Это гарантирует использование требуемого диапазона, даже если были добавлены или удалены строки, не входящие в диапазон.

### Добавление опций в элемент управления *ListBox* на этапе выполнения

Чтобы добавить опции элемента управления *ListBox* на этапе выполнения, необходимо:

* с помощью кода определить значение свойства *RowSource*, чтобы указать диапазон, хранящий необходимые данные;
* создать код, использующий метод *Addltem* для добавления опций в элемент управления *ListBox*.

Следующая процедура устанавливает значения свойства *RowSource* элемента управления *ListBox* перед тем, как отображается диалоговое окно *UserForm*. В этом случае опции состоят из значений в ячейках диапазона *Categories* рабочего листа *Budget*:

UserForml.ListBoxl.RowSource = "Budget!Categories"

UserForml.Show

Если опции элемента управления *ListBox* не содержатся в диапазоне ячеек листа, то можно создать специальный код VBA для заполнения элемента управления *ListBox* перед кодом отображения диалогового окна. Следующая процедура заполняет окно списка элемента управления *ListBox* названиями месяцев года с помощью метода *Addltem*:

Sub ShowUserForm2()

‘ Заполнение списка

 With UserForm1.ListBox1

 .RowSource=""

 .Addltem "Январь"

 .Addltem "Февраль"

 .Addltem "Март"

 .Addltem "Апрель"

 .Addltem "Май"

 .Addltem "Июнь"

 .Addltem "Июль"

 .Addltem "Август"

 .Addltem "Сентябрь"

 .Addltem "Октябрь"

 .Addltem "Ноябрь"

 .Addltem "Декабрь"

 End With

 UserForm1.Show

End Sub

В предыдущем коде свойство *RowSource* сначала приравнивалось к пустой строке. Таким образом предотвращалась потенциальная ошибка, которая появляется в том случае, когда в окне *Properties* свойство *RowSource* элемента управления *ListBox* имеет определенное значение. Если попытаться добавить опции в элемент управления *ListBox* с ненулевым значением свойства *RowSource*, появится сообщение об ошибке *permission denied* (отсутствуют права доступа).

Можно также использовать метод *Addltem* для выборки опций элемента управления *ListBox*, хранящихся в диапазоне ячеек. Ниже рассмотрен пример заполнения элемента управления *ListBox* содержимым диапазона А1:А12 листа *Лист1*:

 For row = 1 To 12

 UserForm1.ListBox1.AddItem Sheets("Лист1").Cells(row, 1)

 Next row

Использование свойства *List* еще больше облегчает решение задачи. Приведенный ниже оператор дает тот же эффект, что и предыдущий цикл *For Next*.

UserForm1.ListBox1.List = Application.Transpose(Sheets( \_

 "Лист1").Range("A1:A12"))

Обратите внимание, что использовалась функция *Transpose*, поскольку свойство *List* настроено на ввод массива-строки, в то время как диапазон ячеек представляет собой массив-столбец.

Если данные хранятся в одномерном (векторном) массиве, можно воспользоваться свойством *List*. Например, предположим, что имеется массив *MyList*, который содержит 50 элементов. Приведенный ниже оператор создает 50-элементный список в элементе управления *ListBox1*.

UserForml.ListBoxl.List = MyList

### Добавление в элемент управления *ListBox* только уникальных элементов

В определенных случаях возникает необходимость в заполнении элемента управления *ListBox* уникальными (неповторяющимися) опциями из существующего списка. Предположим, у нас есть лист, который содержит данные о заказчиках. В одном из столбцов могут содержаться названия штатов (рис. 2). Необходимо заполнить элемент управления *ListBox* названиями штатов, в которых проживают потребители, исключив при этом дублирование названий штатов.

Один из методов заполнения предполагает использование объекта *Collection*. Элементы в объект *Collection* добавляются с помощью следующего синтаксиса:

object.Add item, key, before, after

Аргумент *key* (если он используется) содержит уникальную текстовую строку, которая необходима для получения доступа к элементам коллекции. Если к коллекции добавить неуникальный ключ, то возникнет ошибка, и в результате элемент добавлен не будет. Этим можно воспользоваться и создать коллекцию, которая содержит только уникальные элементы. Представленная далее процедура демонстрирует использование этого способа. Процедура начинается с объявления нового объекта коллекции — *NoDupes*. Предполагается, что диапазон, называющийся *Data*, содержит список элементов, часть которых повторяется.



Рис. 2. Объект *Collection* применяется для заполнения элемента управления *ListBox* уникальными значениями из столбца В

Sub RemoveDuplicates1()

 Dim AllCells As Range, Cell As Range

 Dim NoDupes As New Collection

 On Error Resume Next

 For Each Cell In Range("State")

 NoDupes.Add Cell.Value, CStr(Cell.Value)

 Next Cell

 On Error GoTo 0

' Добавление уникальных элементов в список

 For Each Item In NoDupes

 UserForm1.ListBox1.AddItem Item

 Next Item

' Результаты подсчета

 UserForm1.Label1.Caption = \_

 "Уникальные записи: " & NoDupes.Count

' Показать окно UserForm

 UserForm1.Show

End Sub

В коде циклически просматриваются ячейки диапазона, и в коллекцию *NoDupes* добавляются значения только уникальных ячеек. Кроме того, значение ячейки (преобразованное в строку) используется в качестве значения аргумента *key*. Применение оператора *On Error Resume Next* приводит к тому, что в коде VBA игнорируется ошибка, которая возникает при добавлении в коллекцию неуникального ключа. Если возникает ошибка, элемент в коллекцию не добавляется — это именно то поведение, которого необходимо добиться. Затем процедура передает элементы коллекции *NoDupes* в элемент управления *ListBox*. В диалоговом окне *UserForm* также содержится подпись, которая указывает количество уникальных элементов коллекции.

В файле *listbox unique items2.xlsm* находится более сложный пример, выводящий на экран отсортированные элементы.

### Определение выделенного элемента списка

В примерах, рассмотренных в предыдущих разделах, отображалось диалоговое окно *UserForm* с элементом управления *ListBox*, который содержит список из нескольких элементов. Эти процедуры не включают главной функции: определения опции или опций, которые выбраны пользователем (в дальнейшем будет рассматриваться элемент управления *ListBox* с одним выделенным элементом списка — его свойство *Multiselect* должно иметь значение 0). Чтобы определить, какой элемент списка выбран, необходимо узнать значение свойства *Value* элемента управления *ListBox*. Оператор, показанный ниже, отображает текст выделенного в объекте *ListBoxl* элемента (если не выбран ни один элемент списка, то выполнение оператора приведет к возникновению ошибки):

MsgBox ListBoxl.Value

Чтобы узнать расположение выделенного элемента в списке (а не только его содержимое), воспользуйтесь значением свойства *Listlndex* элемента управления *ListBox*. В следующем примере демонстрируется простое окно сообщения, в котором указан номер выделенной позиции элемента управления *ListBox*:

MsgBox "Вы выбрали позицию #" & ListBox1.Listlndex

Если не выделен ни один элемент списка, свойство Listlndex возвращает значение –1. Нумерация позиций в элементе управления *ListBox* начинается с 0, а не c 1. Таким образом, значение свойства *Listlndex* для первого элемента будет 0, а для последнего элемента значение свойства соответствует значению свойства *ListCount* минус 1.

### Определение нескольких выделенных элементов списка

Свойство *MultiSelect* элемента управления *ListBox* может принимать одно из трех значений:

* 0 (fmMulti Select Single). Может быть выбран только один элемент. Эта установка задана по умолчанию.
* 1 (fmMultiSelectMulti). Нажмите клавишу пробела или щелкните мышью для выделения (отмены выделения) элементов в списке.
* 2 (fmMultiSelectExtended). Удерживая клавишу <Shift>, щелкайте мышью для расширения области выделения от предыдущего до текущего элемента. Для расширения выделенной области можно также удерживать нажатой клавишу <Shift> и нажимать одну из клавиш управления курсором.

Если элемент управления *ListBox* разрешает выделение нескольких элементов (свойство *MultiSelect* равно 1 или 2), попытка доступа к свойству *Listlndex* либо *Value* приведет к ошибке. В этом случае лучше использовать свойство *Selected*, возвращающее массив, первый элемент которого имеет индекс 0. Например, следующий оператор возвращает значение *True*, если выделен первый элемент в списке *ListBox:*

MsgBox ListBox1.Selected(0)

Следующий код, размещеннный в файле *listbox selected items.xlsm*, демонстрирует циклический обход каждой опции элемента управления *ListBox*. Если опция выделена, соответствующий ей текст добавляется в переменную *Msg*. Названия всех выделенных опций отображаются в окне сообщения (рис. 3):



Рис. 3. В окне сообщения отображается список выделенных опций элемента управления *ListBox*

Private Sub OKButton\_Click()

 Dim Msg As String

 Dim i As Integer

 If ListBox1.ListIndex = -1 Then

 Msg = "Ничего не выделено"

 Else

 Msg = ""

 For i = 0 To ListBox1.ListCount - 1

 If ListBox1.Selected(i) Then \_

 Msg = Msg & ListBox1.List(i) & vbCrLf

 Next i

 End If

 MsgBox "Вы выбрали: " & vbCrLf & Msg

 Unload Me

End Sub

### Несколько списков в одном элементе управления ListBox

В следующем примере элемент управления *ListBox* изменяет свое содержимое в зависимости от того, какие переключатели *OptionButton* установил пользователь (рис. 4). *ListBox* получает список значений из диапазона на листе. Процедуры, обрабатывающие событие *Click* для элементов управления *OptionButton*, устанавливают значение свойства *RowSource* элемента управления *ListBox* равным необходимому диапазону (см. файл *listbox multiple lists.xlsm*).



Рис. 4. Содержимое элемента управления *ListBox* зависит от того, какой элемент управления *OptionButton* выбран в настоящий момент

Private Sub obMonths\_Click()

 ListBoxl.RowSource = "Лист1!Месяцы"

End Sub

Щелчок на элементе управления *OptionButton*, называющемся *obMonths*, приводит к изменению значения свойства *RowSource* элемента управления *ListBox*, что заставляет его использовать диапазон *Месяцы* на листе *Лист1*.

### Передача опций элемента управления *ListBox*

В некоторых приложениях требуется выбрать несколько элементов списка. Зачастую следует создать список на основе выделенных элементов. Примером такой ситуации может быть вкладка *Панель быстрого доступа* в диалоговом окне *Параметры Excel*. На рис. 5 показано диалоговое окно с двумя элементами управления *ListBox*. Кнопка *Добавить* добавляет элемент, выделенный в левом элементе управления *ListBox*, в правый элемент управления *ListBox*. Кнопка *Удалить* удаляет выделенный элемент из правого списка. Флажок определяет поведение при добавлении в список повторяющихся элементов. Если флажок *Разрешить дублирование* не установлен, то в случае, если пользователь попытается добавить элемент, который уже присутствует в списке, ничего не произойдет (см. файл *listbox item transfer.xlsm*).



Рис. 5. Построение одного списка на основе другого

Следующая процедура выполняется после щелчка на кнопке *Добавить*.

Private Sub AddButton\_Click()

 Dim i As Integer

 If ListBox1.ListIndex = -1 Then Exit Sub

 If Not cbDuplicates Then

' Проверка существования элемента списка

 For i = 0 To ListBox2.ListCount - 1

 If ListBox1.Value = ListBox2.List(i) Then

 Beep

 Exit Sub

 End If

 Next i

 End If

 ListBox2.AddItem ListBox1.Value

End Sub

Код для управления кнопкой *Удалить*:

Private Sub RemoveButton\_Click()

 If ListBox2.ListIndex = -1 Then Exit Sub

 ListBox2.RemoveItem ListBox2.ListIndex

End Sub

Обратите внимание, что обе процедуры проверяют существование выделенного элемента. Если значение свойства *Listlndex* элемента управления *ListBox* равно –1, значит, не выделен ни один элемент. В результате процедура завершается.

Этот пример также включает две дополнительные процедуры, которые проверяют состояние кнопки *Удалить* (активна или неактивна). При этом вызываются соответствующие события в результате ввода данных в элемент управления *ListBox* (с помощью клавиатуры или щелчка мышью). В результате кнопка *Удалить* активизируется только в том случае, когда пользователь работает с элементом управления *ListBox2*.

Private Sub ListBox1\_Enter()

 RemoveButton.Enabled = False

End Sub

Private Sub ListBox2\_Enter()

 RemoveButton.Enabled = True

End Sub

### Перемещение опций в списке элементов управления *ListBox*

Можно перемещать опции вверх или вниз в списке элемента управления *ListBox*. В VBE подобная техника применяется для контроля порядка активизации опций в окне *UserForm* (щелкните правой кнопкой мыши в окне *UserForm* и в контекстном меню выберите команду *Tab Order*). На рис. 6 показано диалоговое окно с элементом управления *ListBox* и двумя элементами управления *CommandButton*. Щелчок на кнопке *Вверх* приведет к перемещению выделенной опции вверх по списку элемента управления *ListBox*. Щелчок на кнопке *Вниз* приведет к перемещению выделенной опции вниз по списку (см. файл *listbox move items.xlsm*).



Рис. 6. Перемещаться вверх или вниз по списку можно с помощью специально предназначенных для этого кнопок

Следующая процедура обрабатывает клик на кнопке *Вверх*:

Private Sub MoveUpButton\_Click()

 Dim NumItems As Integer

 Dim i As Integer

 Dim ItemNum As Integer

 Dim TempItem As String

 Dim TempList()

 If ListBox1.ListIndex = 0 Then Exit Sub

 NumItems = ListBox1.ListCount

 ReDim TempList(0 To NumItems - 1)

' Заполнение массива элементами списка

 For i = 0 To NumItems - 1

 TempList(i) = ListBox1.List(i)

 Next i

' Выделенные элементы

 ItemNum = ListBox1.ListIndex

' Обмен элементами

 TempItem = TempList(ItemNum)

 TempList(ItemNum) = TempList(ItemNum - 1)

 TempList(ItemNum - 1) = TempItem

 ListBox1.List = TempList

' Изменение индекса элементаChange the list index

 ListBox1.ListIndex = ItemNum - 1

End Sub

### Работа с многоколоночными элементами управления *ListBox*

Как правило, элемент управления *ListBox* содержит один столбец, в котором отображается один список. Однако можно создать элемент управления *ListBox*, который содержит несколько столбцов, а иногда — даже несколько столбцов с заголовками. На рис. 7 отображен элемент управления *ListBox* с несколькими столбцами, который получает данные из диапазона ячеек рабочего листа (см. файл *listbox multicolumn1.xlsm*).



Рис. 7. Элемент управления *ListBox* отображает трехколоночный список с заголовками колонок

Для того чтобы создать элемент управления *ListBox* с несколькими столбцами, в которые заносятся данные, хранимые в диапазоне ячеек листа, выполните следующие действия:

1. Удостоверьтесь, что свойству *ColumnCount* элемента управления *ListBox* присвоено правильное значение, которое соответствует количеству столбцов в элементе управления.
2. Укажите правильный исходный диапазон данных из нескольких столбцов, присвоив соответствующее значение свойству *RowSource* элемента управления *ListBox*.
3. Если необходимо отобразить заголовки столбцов, присвойте свойству *ColumnHeads* значение *True*. Не включайте заголовки столбцов в диапазон рабочего листа, указанный в свойстве *RowSource*. VBA автоматически использует для них строку, которая находится сразу над строкой, указанной в значении свойства *RowSource*.
4. Измените ширину столбцов, присвоив свойству *ColumnWidths* значения, которые указываются в пунктах (1/72 часть дюйма) и разделены точками с запятой. Например, следующее значение свойства *ColumnWidths* определяет ширину трех столбцов списка элемента управления *ListBox* — 100; 40; 30.
5. Укажите столбец в качестве значения свойства *BoundColumn*. Это свойство определяет столбец, на который указывает ссылка при обращении к свойству *Value* элемента управления *ListBox*.

Чтобы заполнить элемент управления *ListBox* данными из нескольких столбцов без использования диапазона, необходимо создать двумерный массив, а затем присвоить массив свойству *List* элемента управления *ListBox*. Следующие операторы демонстрируют применение двумерною массива (размером 12x2) под названием *Data*. Двухколоночный список *ListBox* отображает названия месяцев в столбце 1, а количество дней — в столбце 2 (рис. 8). Обратите внимание, что процедура присваивает свойству *ColumnCount* значение 2 (см. файл *listbox multicolumn2 .xlsm*).



Рис. 8. Двухколоночный список *ListBox*, заполненный данными двумерного массива

Private Sub UserForm\_Initialize()

 Dim i As Integer

 Dim Data(1 To 12, 1 To 2) As Variant

 Dim TheYr As Integer

 TheYr = Year(Now)

' Заполнение списка

 For i = 1 To 12

 Data(i, 1) = Format(DateSerial(TheYr, i, 1), "mmmm")

 Next i

 For i = 1 To 12

 Data(i, 2) = Day(DateSerial(TheYr, i + 1, 1) - 1)

 Next i

 ListBox1.ColumnCount = 2

 ListBox1.List = Data

End Sub

Не существует способа определить заголовки столбцов в свойстве *ColumnHeads*, когда данные списка находятся в массиве VBA.

### Использование элемента управления *ListBox* для выделения строк на листе

Следующий пример позволяет отображать элемент управления *ListBox*, который состоит из элементов заполненного диапазона на текущем листе (рис. 9). Пользователь может выбрать несколько опций списка в элементе управления *ListBox* (или щелкнуть на кнопке *Все*). Щелчок на кнопке ОК приводит к выделению строк, которые соответствуют выделенным опциям элемента управления *ListBox*. Конечно, можно выделить несколько несмежных диапазонов непосредственно на листе. Эта задача выполняется с помощью клавиши <Ctrl>. Но со временем становится понятно, что метод, предложенный в этом разделе, намного удобнее (см. файл *listbox select rows.xlsm*).



Рис. 9. Элемент управления *ListBox* облегчает выделение строк в рабочем листе

Выбор нескольких опций возможен, поскольку свойству *MultiSelect* элемента управления *ListBox* присвоено значение 1 – *fmMultiSelectMulti*. Установка свойства *ListStyle* элемента управления *ListBox* равным 1 (*fmListStyleOption*) приводит к отображению флажков для каждого элемента.

Следующая процедура объекта *UserForm* применяется для обработки события *Initialize*. Эта процедура создает объект *rng*, который состоит из используемого диапазона активного листа. Дополнительный код устанавливает свойства *RowSource* и *ColumnCount* элемента управления *ListBox*, а также изменяет значение свойства *ColumnWidths*, чтобы столбцы элемента управления *ListBox* по ширине соответствовали столбцам активного рабочего листа.

Private Sub UserForm\_Initialize()

 Dim ColCnt As Integer

 Dim rng As Range

 Dim cw As String

 Dim c As Integer

 ColCnt = ActiveSheet.UsedRange.Columns.Count

 Set rng = ActiveSheet.UsedRange

 With ListBox1

 .ColumnCount = ColCnt

 .RowSource = rng.Address

 cw = ""

 For c = 1 To .ColumnCount

 cw = cw & rng.Columns(c).Width & ";"

 Next c

 .ColumnWidths = cw

 .ListIndex = 0

 End With

End Sub

Кнопки *Все* и *Сброс* (называющиеся *SelectAllButton* и *SelectNoneButton*) имеют следующие процедуры обработки событий:

Private Sub SelectAllButton\_Click()

 Dim r As Integer

 For r = 0 To ListBox1.ListCount - 1

 ListBox1.Selected(r) = True

 Next r

End Sub

Private Sub SelectNoneButton\_Click()

 Dim r As Integer

 For r = 0 To ListBox1.ListCount - 1

 ListBox1.Selected(r) = False

 Next r

End Sub

Процедура обработки события *OKButton\_Click* создает объект *Range*, называющийся *RowRange*. Он состоит из строк, соответствующих выделенным опциям в элементе управления *ListBox*. Для того чтобы определить факт выделения опции, в коде проверяется значение свойства *Selected* элемента управления *ListBox*. Обратите внимание на использование функции *Union* для добавления дополнительных диапазонов к объекту *RowRange*.

Private Sub OKButton\_Click()

 Dim RowRange As Range

 Dim RowCnt As Integer

 Dim r As Integer

 RowCnt = 0

 For r = 0 To ListBox1.ListCount - 1

 If ListBox1.Selected(r) Then

 RowCnt = RowCnt + 1

 If RowCnt = 1 Then

 Set RowRange = ActiveSheet.UsedRange.Rows(r + 1)

 Else

 Set RowRange = Union(RowRange, ActiveSheet.UsedRange.Rows(r + 1))

 End If

 End If

 Next r

 If Not RowRange Is Nothing Then RowRange.Select

 Unload Me

End Sub

### Использование элемента управления *ListBox* для активизации листа

В следующем примере использован элемент управления *ListBox* с несколькими столбцами (еще его называют многоколоночным) для отображения списка рабочих листов активной рабочей книги. В столбцах содержатся следующие данные: имя листа, тип листа, количество непустых ячеек в листе, состояние листа (рис. 10; см. файл *listbox activate sheet.xlsm*).



Рис. 10. С помощью этого диалогового окна пользователь активизирует лист

Код процедуры *UserForm\_Initialize* создает двумерный массив и собирает информацию, циклически просматривая листы активной рабочей книги. После этого массив передается в элемент управления *ListBox*.

Private Sub UserForm\_Initialize()

 Dim SheetData() As String

 Dim ShtCnt As Integer

 Dim ShtNum As Integer

 Dim Sht As Object

 Dim ListPos As Integer

 Set OriginalSheet = ActiveSheet

 ShtCnt = ActiveWorkbook.Sheets.Count

 ReDim SheetData(1 To ShtCnt, 1 To 4)

 ShtNum = 1

 For Each Sht In ActiveWorkbook.Sheets

 If Sht.Name = ActiveSheet.Name Then \_

 ListPos = ShtNum - 1

 SheetData(ShtNum, 1) = Sht.Name

 Select Case TypeName(Sht)

 Case "Worksheet"

 SheetData(ShtNum, 2) = "Лист"

 SheetData(ShtNum, 3) = \_

 Application.CountA(Sht.Cells)

 Case "Диаграмма"

 SheetData(ShtNum, 2) = "Диаграмма"

 SheetData(ShtNum, 3) = "Не определено"

 Case "DialogSheet"

 SheetData(ShtNum, 2) = "Диалоговое окно"

 SheetData(ShtNum, 3) = "Не определено"

 End Select

 If Sht.Visible Then

 SheetData(ShtNum, 4) = "Да"

 Else

 SheetData(ShtNum, 4) = "Нет"

 End If

 ShtNum = ShtNum + 1

 Next Sht

 With ListBox1

 .ColumnWidths = "100 pt;30 pt;40 pt;50 pt"

 .List = SheetData

 .ListIndex = ListPos

 End With

End Sub

Значение элемента управления *CheckBox* (с названием *cbPreview*) определяет необходимость предварительного просмотра листа после того, как пользователь щелкнет на соответствующей опции списка элемента управления *ListBox*. Щелчок на кнопке *ОК* (объект *OKButton*) приводит к выполнению процедуры *OKButton\_Click*:

Private Sub OKButton\_Click()

 Dim UserSheet As Object

 Set UserSheet = Sheets(ListBox1.Value)

 If UserSheet.Visible Then

 UserSheet.Activate

 Else

 If MsgBox("Отобразить лист?", \_

 vbQuestion + vbYesNoCancel) = vbYes Then

 UserSheet.Visible = True

 UserSheet.Activate

 Else

 OriginalSheet.Activate

 End If

 End If

 Unload Me

End Sub

Процедура *OKButton\_Click* создает объектную переменную, которая представляет выделенный лист. Если лист отображается, то он активизируется. Если лист скрыт, то на экран выводится сообщение, в котором предлагается сделать лист видимым. Если пользователь даст утвердительный ответ на запрос, то лист будет отображен и активизируется. В противном случае активизируется исходный лист (который хранится в переменной *OriginalSheet*).

Двойной щелчок на опции списка в элементе управления *ListBox* приводит к тому же результату, что и щелчок на кнопке ОК. Процедура *ListBoxl\_DblClick* вызывает процедуру *OKButton\_Click*.

Private Sub ListBox1\_DblClick(ByVal Cancel As MSForms.ReturnBoolean)

 Call OKButton\_Click

End Sub

1. По материалам книги [Джон Уокенбах. Excel 2010. Профессиональное программирование на VBA](http://baguzin.ru/wp/?p=16255). – М: Диалектика, 2013. – С. 449–466. [↑](#footnote-ref-1)