Примеры пользовательских форм в Excel, построенных с помощью VBA

Ранее я рассмотрел методы создания пользовательских форм и основы работы с ними (если вы никогда не работали с пользовательскими формами, рекомендую для начала прочитать указанную заметку). В настоящем материале приводится целый ряд практически полезных примеров пользовательских диалоговых окон.[1]

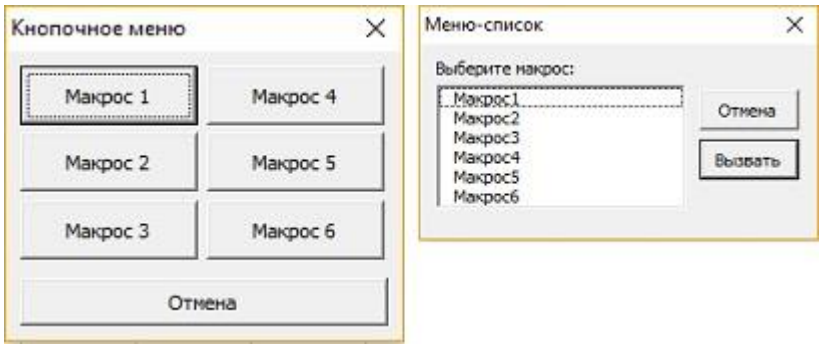

Рис. 1. Меню на основе элементов управления *CommandButton* или *ListBox*

Создание меню

Меню можно создать с помощью элементов управления *CommandButton* (рис. 1; см. файл *userform menus.xlsm*) и с помощью элемента управления *ListBox*. Каждый элемент управления *CommandButton* имеет собственную процедуру обработки событий. Например, представленная ниже процедура выполняется после щелчка на кнопке *CommandButton1*:

```
Private Sub CommandButton1_Click()
    Me.Hide
    Call Macro1
    Unload Me
End Sub
```

Эта процедура приводит к вызову макроса *Macro1* и закрытию диалогового окна *UserForm*. После щелчка на других кнопках (отличных от *CommandButton1*) вызываются похожие процедуры обработки событий.

В случае использования элемента управления *ListBox*, перед отображением пользовательского диалогового окна вызывается процедура обработки события *Initialize*. В следующей процедуре используется метод *AddItem* для добавления шести опций в элемент управления *ListBox*:

```
Private Sub UserForm_Initialize()
    With ListBox1
        .AddItem "Макрос1"
        .AddItem "Макрос2"
        .AddItem "Макрос3"
        .AddItem "Макрос4"
        .AddItem "Макрос5"
        .AddItem "Макрос6"
    End With
End Sub
```

Процедура обработки события привязывается к кнопке *Выполнить:*

```
Private Sub ExecuteButton_Click()
    Select Case ListBox1.ListIndex
        Case -1
            MsgBox "Выберите макрос из списка."
            Exit Sub
        Case 0: Me.Hide: Call Macro1
        Case 1: Me.Hide: Call Macro2
```

[1] По материалам книги Джон Уокенбах. Excel 2010. Профессиональное программирование на VBA. – М: Диалектика, 2013. – С. 439–449, 466–472.

```
        Case 2: Me.Hide: Call Macro3
        Case 3: Me.Hide: Call Macro4
        Case 4: Me.Hide: Call Macro5
        Case 5: Me.Hide: Call Macro6
    End Select
    Unload Me
End Sub
```

Данная процедура проверяет значение свойства *ListIndex* элемента управления *ListBox*, чтобы определить, какой элемент выбран в списке (если свойство *ListIndex* равно –1, то не выбран ни один из элементов). После этого запускается соответствующий макрос.

Выбор диапазона в пользовательской форме

Некоторые встроенные диалоговые окна Excel предоставляют пользователю возможность выбирать диапазон. Например, диалоговое окно *Подбор параметра*, для вызова которого следует пройти по меню *Данные –> Работа с данными –> Анализ "что если " –> Подбор параметра*, запрашивает у пользователя два диапазона. Пользователь может или ввести имя диапазона непосредственно в окне, или применить мышь для выделения диапазона на листе.

Пользовательское диалоговое окно также обеспечивает подобную функциональность. Это достигается с помощью элемента управления *RefEdit*. Данный элемент выглядит иначе, чем элемент выбора диапазона во встроенных диалоговых окнах Excel, однако работает точно так же. Если пользователь щелкнет на небольшой кнопке в правой части элемента управления, то диалоговое окно временно исчезнет, а на экране будет отображен небольшой указатель выбора диапазона.

К сожалению, элемент управления *RefEdit* не позволяет использовать специальные клавиши при выделении диапазона (например, невозможно выделить ячейки до конца столбца, нажав комбинацию клавиш <Shift+↓). Кроме того, после щелчка мышью на маленькой кнопке в правой части элемента управления (для временного сокрытия диалогового окна) можно применять только выделения с помощью мыши. Клавиатуру в этом случае применять нельзя.

На рис. 2 представлено пользовательское диалоговое окно с добавленным элементом управления *RefEdit* (см. файл *range selection demo.xlsm*). Это диалоговое окно выполняет простую математическую операцию над всеми не содержащими формул и непустыми ячейками указанного диапазона. Выполняемая операция задается активным переключателем *OptionButton*.

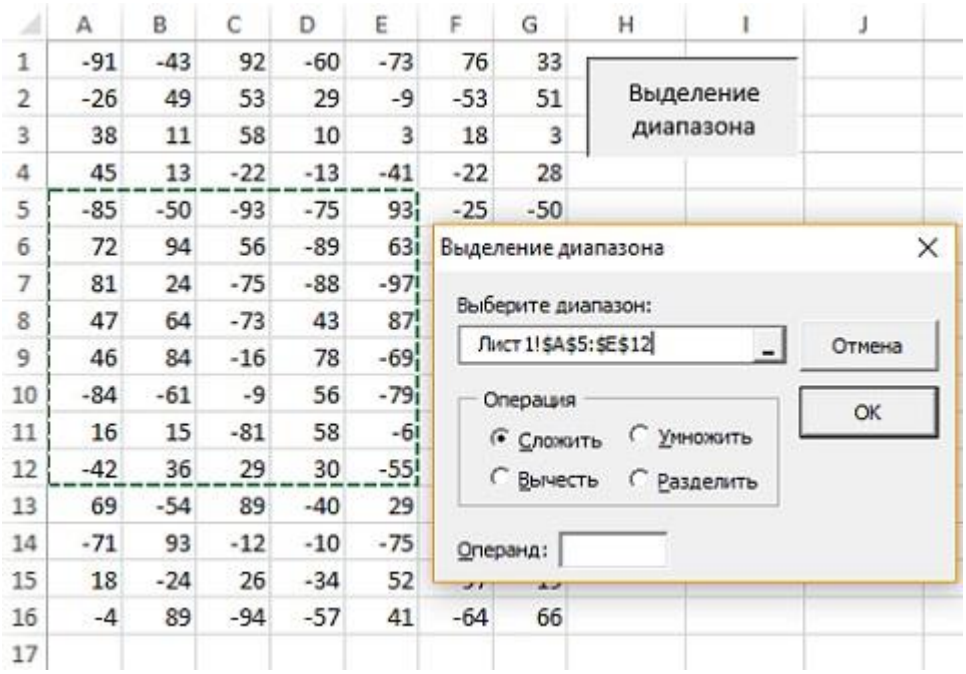

Рис. 2. С помощью элемента управления *RefEdit* можно выбрать диапазон

Элемент управления *RefEdit* возвращает текстовую строку, которая представляет выбранный диапазон. Можно преобразовать эту строку в объект *Range*. Для этого используется оператор:

```
Set UserRange = Range(RefEditl.Text)
```

Удачной практикой считается инициализация элемента управления *RefEdit* для представления текущего выделения. Для этого в процедуре *UserForm_Initialize* воспользуйтесь оператором:

```
RefEditl.Text = ActiveWindow.RangeSelection.Address
```

Для достижения наилучших результатов не помещайте элемент управления *RefEdit* внутри элемента *Frame* либо *MultiPage*. Это может привести к сбою в работе Excel. Элемент управления *RefEdit* не всегда возвращает действительный диапазон. Выделение диапазона указателем мыши — это один из способов присвоения значения данному элементу управления. Пользователь может ввести в поле любой текст, а также отредактировать или удалить уже отображаемый текст. Таким образом, предварительно необходимо убедиться, что диапазон является допустимым.

Следующий код — это пример одного из способов проверки допустимости введенного значения. Если определено, что значение неправильное, то пользователю выдается сообщение, а элемент управления *RefEdit* становится активным, предоставляя возможность ввести корректный диапазон.

```
    On Error Resume Next
    Set UserRange = Range(RefEdit1.Text)
    If Err <> 0 Then
        MsgBox "Выбран некорректный диапазон"
        RefEdit1.SetFocus
        Exit Sub
    End If
    On Error GoTo 0
```

Пользователь может щелкнуть на вкладке одного из листов при выборе диапазона, применив элемент управления *RefEdit*. Поэтому не всегда выбранный диапазон находится на активном рабочем листе. Если пользователем выбран другой лист, то адрес диапазона указывается после имени листа, на котором этот диапазон находится. Если необходимо получить от пользователя выделение в виде одной ячейки, то можно указать верхнюю левую ячейку выделенного диапазона. Воспользуйтесь следующим оператором:

```
Set OneCell = Range(RefEditl.Text).Range("Al")
```

Создание заставки

Некоторые разработчики предпочитают отображать определенную вступительную информацию при запуске приложения. Эта методика называется заставкой. Без сомнения, все пользователи видели заставку Excel, которая отображается несколько секунд при запуске программы. В приложении Excel заставку можно создать с помощью пользовательского диалогового окна. В приведенном ниже примере реализуется автоматическое отображение заставки, которое исчезает по истечении пяти секунд (рис. 3; см. файл *splash screen.xlsm*). Для создания заставки выполните следующие действия:

1. Создайте рабочую книгу.
2. Активизируйте редактор VBE и вставьте пользовательское диалоговое окно в проект. Код в этом примере предполагает, что объект *UserForm* называется *UserForm1*.
3. Поместите любые необходимые элементы управления в только что созданное диалоговое окно *UserForm1*. Например, вам может понадобиться расположить элемент управления *Image*, который будет содержать логотип компании (рис. 3).
4. Вставьте процедуру в модуль кода для объекта *ЭтаКнига* (*ThisWorkbook*):

```
Private Sub Workbook_Open()
    UserForm1.Show
End Sub
```

5. Вставьте следующую процедуру в модуль кода для объекта *UserForm1* (эта процедура обеспечивает пятисекундную задержку). Если нужно другое время задержки, измените значение аргумента функции *TimeValue*.

```
Private Sub UserForm_Activate()
    Application.OnTime Now + TimeValue("00:00:05"), "KillTheForm"
```

```
End Sub
```

Рис. 3. Эта заставка на короткое время появляется на экране после открытия рабочей книги

6. В общий модуль VBA вставьте следующую процедуру.

```
Private Sub KillTheForm()
    Unload UserForm1
End Sub
```

При открытии рабочей книги будет выполнена процедура *Workbook_Open* и появится диалоговое окно *UserForm* (п. 4). В этот момент генерируется событие *Activate*, которое приводит к выполнению процедуры *UserForm_ Activate* (п. 5). Данная процедура использует метод *OnTime* объекта *Application* для выполнения процедуры *KillTheForm* в определенный момент времени. Однако предварительно определена задержка в пять секунд с момента возникновения события *Activate*. Процедура *KillTheForm* просто выгружает диалоговое окно *UserForm* из памяти.

7. В качестве необязательного действия можно добавить элемент управления *CommandButton* с именем *CancelButton*, установить его свойство *Cancel* равным *True* и добавить представленную ниже процедуру обработки события в модуль кода формы *UserForm*.

```
Private Sub CancelButton_Click()
    KillTheForm
End Sub
```

Таким образом, пользователь сможет закрыть заставку, прежде чем пройдет указанное время задержки. Окно будет закрыто также в результате нажатия клавиши <Esc>. Эту небольшую кнопку можно разместить за другим объектом, чтобы она не была видна.

Помните о том, что заставка не будет отображаться, если рабочая книга загружена не полностью. Другими словами, если нужно отобразить заставку только для того, чтобы пользователь не скучал во время загрузки рабочей книги, описанная выше методика не годится.

Для того чтобы выполнить VBA-процедуру при открытии документа, нужно так отобразить пользовательское диалоговое окно в немодальном режиме, чтобы код продолжал выполняться. Для этого измените процедуру *WorkbookOpen* следующим образом.

```
Private Sub Workbook_Open()
    UserForml.Show vbModeless
```

```
    ' другой код
End Sub
```

Отключение кнопки закрытия пользовательского диалогового окна

Если пользовательское диалоговое окно уже отображено на экране, щелчок на кнопке *Закрыть* в правом верхнем углу приведет к выгрузке формы *UserForm* из памяти. Иногда этого допускать нельзя. Например, иногда требуется, чтобы диалоговое окно *UserForm* закрывалось только после щелчка на специальной кнопке *CommandButton*. Несмотря на то что реально отключить кнопку *Закрыть* невозможно, вы вправе предотвратить закрытие диалогового окна, вызванное щелчком на этой кнопке. Для этого воспользуйтесь обработчиком события *QueryClose* (см. файл *gueryclose demo.xlsm*). Следующая процедура, которая расположена в модуле кода диалогового окна *UserForm*, выполняется перед закрытием формы (т.е. в момент возникновения события *QueryClose*).

```
Private Sub UserForm_QueryClose _
  (Cancel As Integer, CloseMode As Integer)
    If CloseMode = vbFormControlMenu Then
        MsgBox "Щелкните на кнопке OK для закрытия формы."
        Cancel = True
    End If
End Sub
```

Процедура *UserForm_QueryClose* использует два аргумента. Аргумент *CloseMode* содержит значение, которое указывает на причину возникновения события *QueryClose*. Если значение аргумента *CloseMode* равно *vbFormControlMenu* (встроенная константа), значит, пользователь щелкнул на кнопке *Закрыть*. В таком случае будет отображено сообщение (рис. 4); аргумент *Cancel* устанавливается равным *True*, и форма не закрывается.

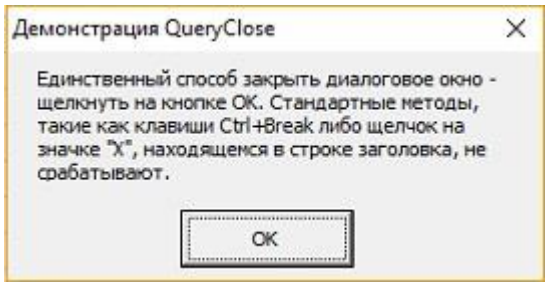

Рис. 4. Процедура перехватывает закрытие окна, и оставляет его открытым

Имейте в виду, что пользователь может нажать клавиши <Ctrl+Break>, прекратив тем самым выполнение макроса. В рассматриваемом примере нажатие клавиш <Ctrl+Break> во время отображения формы *UserForm* на экране приведет к тому, что пользовательское диалоговое окно будет закрыто. Во избежание этого выполните следующий оператор до отображения пользовательского диалогового окна:

```
Application.EnableCancelKey = xlDisabled
```

Прежде чем добавить этот оператор, удостоверьтесь, что в приложении нет ошибок. В противном случае возникает опасность формирования бесконечного цикла.

Изменение размера диалогового окна

Во многих приложениях используются окна, которые могут изменять собственные размеры, добавляя кнопки и опции. Например, высота диалогового окна Excel *Найти и заменить*, которое отображается после выбора команды *Главная –> Редактирование –> Найти и выделить –> Заменить*, увеличивается после щелчка на кнопке *Параметры*.

Изменение размеров пользовательского диалогового окна осуществляется путем изменения значений свойств *Width* и *Height* объекта *UserForm*. На рис. 5а показано первоначальное диалоговое окно, а на рис. 5б показано это же окно после щелчка на кнопке *Параметры*. Обратите внимание на то, что надпись на кнопке изменяется в зависимости от размера диалогового окна (см. файл *change userform size.xlsm*).

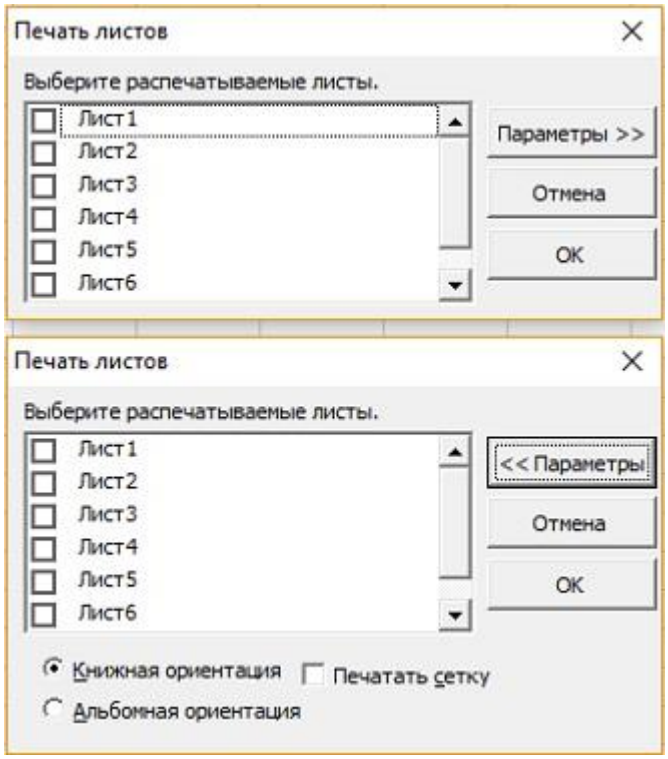

Рис. 5. Диалоговое окно: вверху – в стандартном режиме; внизу – после нажатия на кнопке *Параметры*

Создавая пользовательское диалоговое окно, определите его максимальный размер, чтобы получить доступ ко всем элементам управления. После этого воспользуйтесь процедурой *UserForm_Initialize* для установки размеров диалогового окна по умолчанию.

В коде применяются две константы, определенные в верхней части модуля.

```
Const SmallSize As Integer = 124
Const LargeSize As Integer = 164
```

Пример предназначен для печати рабочих листов активной книги. Он позволяет пользователю указать листы, которые необходимо распечатать. Ниже приведена процедура обработки события, которая выполняется после щелчка на кнопке *Параметры*.

```
Private Sub OptionsButton_Click()
    If OptionsButton.Caption = "Параметры >>" Then
        Me.Height = LargeSize
        OptionsButton.Caption = "<< Параметры"
    Else
        Me.Height = SmallSize
        OptionsButton.Caption = "Параметры >>"
    End If
End Sub
```

Эта процедура проверяет значение свойства *Caption* объекта *CommandButton* и устанавливает значение свойства *Height* объекта *UserForm* в соответствии с полученным значением свойства *Caption*.

Если элементы управления не отображаются из-за того, что находятся за пределами границы диалогового окна, соответствующие этим элементам управления комбинации клавиш будут продолжать функционировать. В рассматриваемом примере пользователь может нажать клавиши <Alt+L> (для выбора альбомной ориентации страницы), даже если соответствующий элемент управления не отображается на экране.

Масштабирование и прокрутка листа в пользовательском диалоговом окне

Для прокрутки и масштабирования листа при активном диалоговом окне можно применить элемент управления *ScrollBar* (рис. 6; см. файл *zoom and scroll sheet.xlsm*). В этом примере масштаб

можно изменять в диапазоне от 100 до 400%. Два элемента управления *ScrollBar* в нижней части диалогового окна позволяют прокручивать лист по горизонтали и по вертикали.

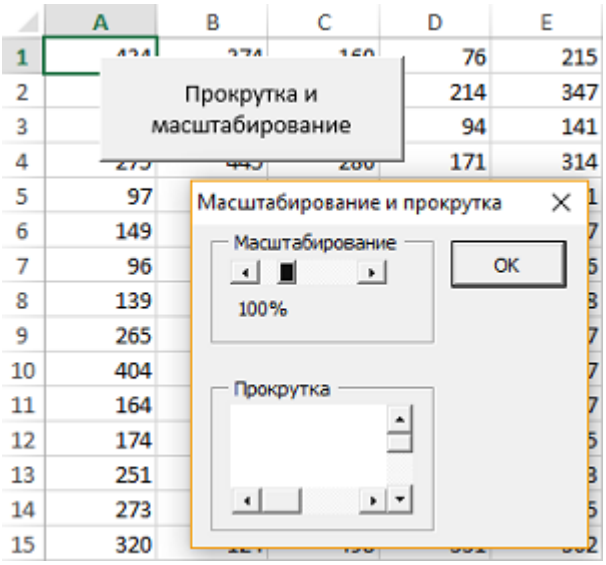

Рис. 6. Элементы управления *ScrollBar* позволяют прокручивать лист и изменять его масштаб

Элементы управления инициализируются в процедуре *UserForm_Initialize*:

```
Private Sub UserForm_Initialize()
    LabelZoom.Caption = ActiveWindow.Zoom & "%"
'   Масштабирование
    With ScrollBarZoom
        .Min = 10
        .Max = 400
        .SmallChange = 1
        .LargeChange = 10
        .Value = ActiveWindow.Zoom
    End With

'   Прокрутка по горизонтали
    With ScrollBarColumns
        .Min = 1
        .Max = ActiveSheet.UsedRange.Columns.Count
        .Value = ActiveWindow.ScrollColumn
        .LargeChange = 25
        .SmallChange = 1
    End With

'   Прокрутка по вертикали
    With ScrollBarRows
        .Min = 1
        .Max = ActiveSheet.UsedRange.Rows.Count
        .Value = ActiveWindow.ScrollRow
        .LargeChange = 25
        .SmallChange = 1
    End With
End Sub
```

Эта процедура позволяет устанавливать значения различных свойств элементов управления *ScrollBar*. Значения определяются на основе данных, полученных из активного окна. При использовании элемента управления *ScrollBarZoom* выполняется процедура *ScrollBarZoom_Change*. Она устанавливает значение свойства *Zoom* объекта *ActiveWindow* равным значению свойства *Value* элемента управления *ScrollBar*. Кроме того, изменяется текст подписи, которая представляет текущий масштаб рабочего листа.

```
Private Sub ScrollBarZoom_Change()
```

```
    With ActiveWindow
        .Zoom = ScrollBarZoom.Value
        LabelZoom = .Zoom & "%"
        .ScrollColumn = ScrollBarColumns.Value
        .ScrollRow = ScrollBarRows.Value
    End With
End Sub
```

Прокрутка листа осуществляется с помощью двух процедур. Эти процедуры устанавливают значение свойств *ScrollRow* и *ScrollColumns* объекта *ActiveWindow* равными значениям свойств *Value* элементов управления *ScrollBar*.

```
Private Sub ScrollBarColumns_Change()
    ActiveWindow.ScrollColumn = ScrollBarColumns.Value
End Sub
```

```
Private Sub ScrollBarRows_Change()
    ActiveWindow.ScrollRow = ScrollBarRows.Value
End Sub
```

При нажатии на кнопку Ok пользовательская форма закрывается:

```
Private Sub OKButton_Click()
    Unload Me
End Sub
```

Применение элемента управления MultiPage

Элемент управления *MultiPage* применяется при отображении в пользовательских диалоговых окнах множества элементов управления. Элемент управления *MultiPage* позволяет группировать опции, а также размещать каждую группу на отдельной вкладке (рис. 7; см. файл *multipage control demo.xlsm*). Панель инструментов *Toolbox* также включает элемент управления *TabStrip*, напоминающий элемент управления *MultiPage*. Однако в отличие от *MultiPage*, элемент управления *TabStrip* не может включать другие объекты. Поскольку элемент управления *MultiPage* является более гибким, вряд ли вам придется обращаться к элементу управления *TabStrip*.

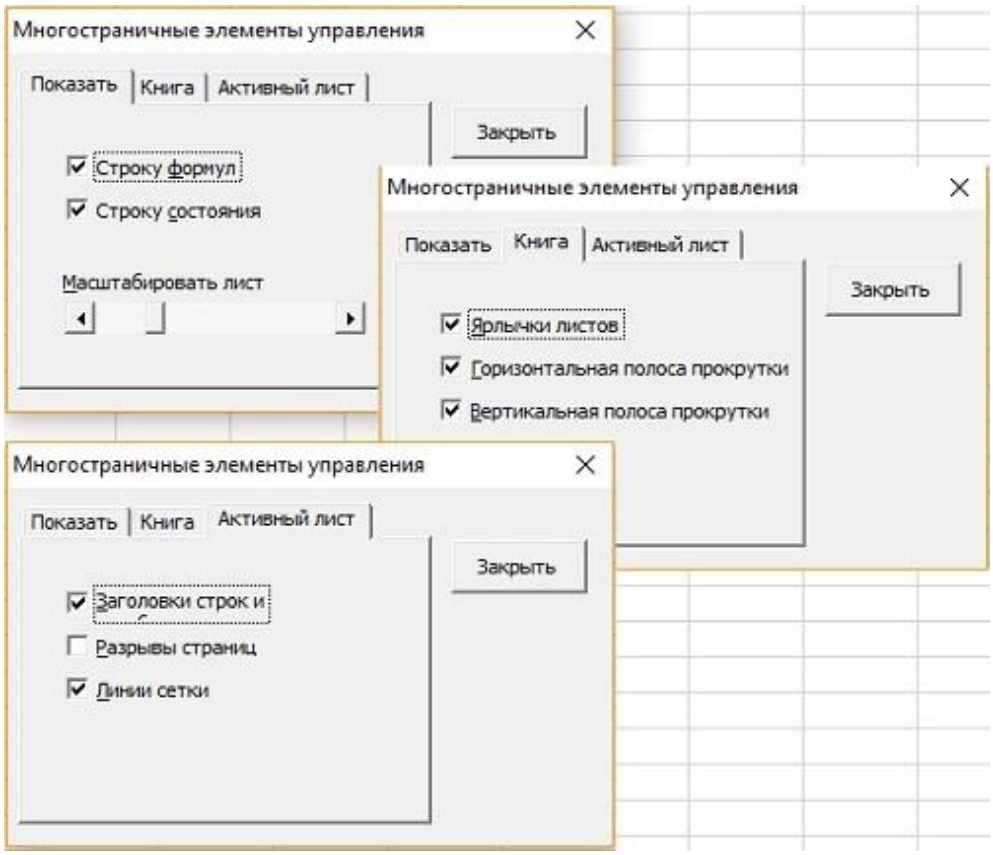

Рис. 7. Элемент управления *MultiPage* группирует элементы управления на страницах, доступ к которым обеспечивается с вкладки

Вкладка (или страница), которая отображается поверх всех остальных, определяется значением свойства *Value* элемента управления *MultiPage*. Значение 0 соответствует первой вкладке. Значение 1 вызывает отображение второй вкладки и т.д. По умолчанию элемент управления *MultiPage* состоит из двух страниц. Для того чтобы добавить дополнительные вкладки, щелкните на любой вкладке правой кнопкой мыши и в контекстном меню выберите пункт *New Page*.

При работе с элементом управления *MultiPage* щелкните на вкладке, чтобы установить свойства страницы. В окне *Properties* отобразятся свойства, значения которых можно изменить. Иногда сложно выделить сам элемент управления *MultiPage*, так как щелчок на нем приводит к выделению страницы элемента управления. Для того чтобы выделить только элемент управления, щелкните на его границе. Кроме того, можете воспользоваться клавишей <Tab> для циклического перемещения между элементами управления. Еще одним вариантом выделения элемента управления является выбор пункта *MultiPage* из раскрывающегося списка окна *Properties*.

Если элемент управления *MultiPage* содержит много вкладок, то присвойте его свойству *MultiRow* значение *True*, чтобы отобразить вкладки в несколько строк. Если необходимо, то вместо вкладок можно отображать кнопки. Достаточно изменить значение свойства *Style* на 1. Если значение свойства *Style* равно 2, элемент управления *MultiPage* не будет отображать ни вкладки, ни кнопки.

Свойство *TabOrientation* определяет расположение вкладок на элементе управления *MultiPage*. Для каждой страницы можно установить эффект перехода. Для этого воспользуйтесь свойством *TransitionEffect*. Например, щелчок на вкладке приведет к тому, что новая страница «отодвинет» старую. Применяйте свойство *TransitionPeriod*, чтобы задать скорость эффекта перехода.

Использование внешних элементов управления

Пример, рассматриваемый в этом разделе, основан на элементе управления *Microsoft Windows Media Player*. Несмотря на то что он не является элементом управления *Excel* (настраивается при установке *Windows*), он прекрасно работает с формами *UserForm*. Для того чтобы воспользоваться элементом управления *Microsoft Windows Media Player* выполните следующие действия:

1. Активизируйте среду VBE.
2. Создайте новую пользовательскую форму: *Insert –> UserForm*.
3. Щелкните правой кнопкой мыши на панели *Toolbox* и в контекстном меню выберите параметр *Additional Controls* (Дополнительные элементы управления). Если окно *Toolbox* не отображается, выполните команду *View –> Toolbox*.
4. В окне *Additional Controls* установите флажок *Windows Media Player* (рис. 8).
5. Щелкните на кнопке ОК. Набор инструментов пополнится новым элементом управления.

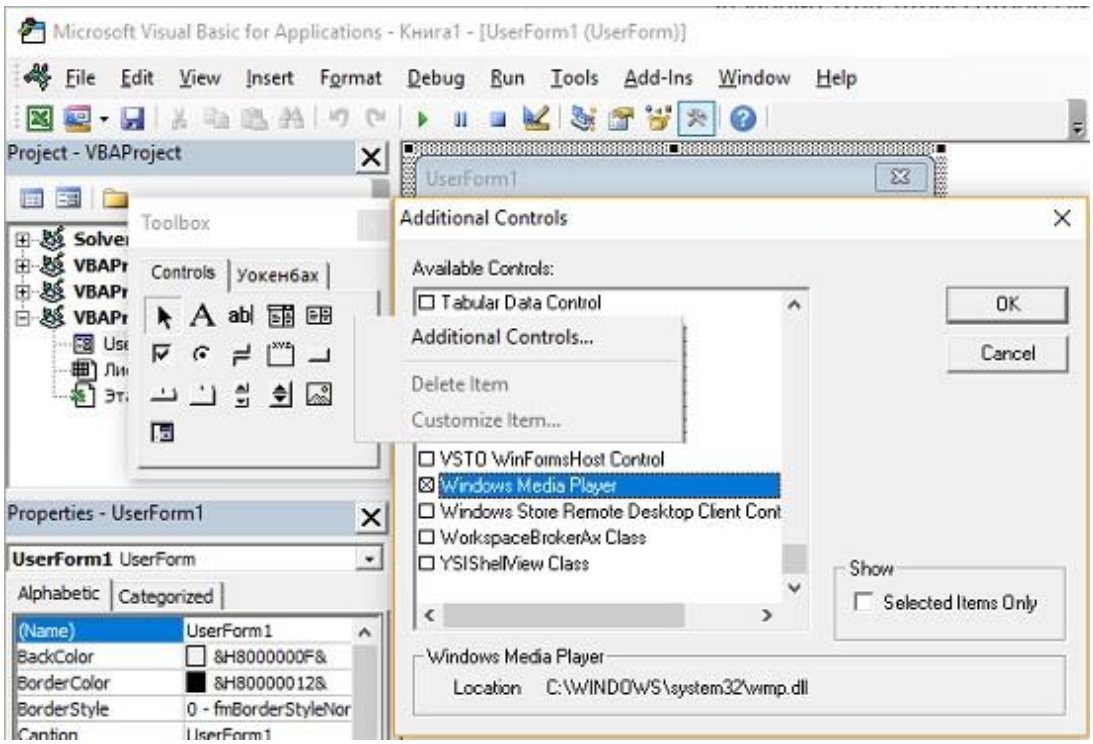

Рис. 8. Подключение элемента управления *Microsoft Windows Media Player*

На рис. 9 показаны элемент управления *Windows Media Player*, встроенный в форму *UserForm*, а также окно *Properties* (см. также файл *mediaplayer.xlsm*, расположенный в отдельной папке). Свойство URL определяет URL-ссылку воспроизводимой композиции (музыкальная запись или видеоролик). Если композиция находится на жестком диске вашего компьютера, свойство URL определяет полный путь и имя соответствующего файла.

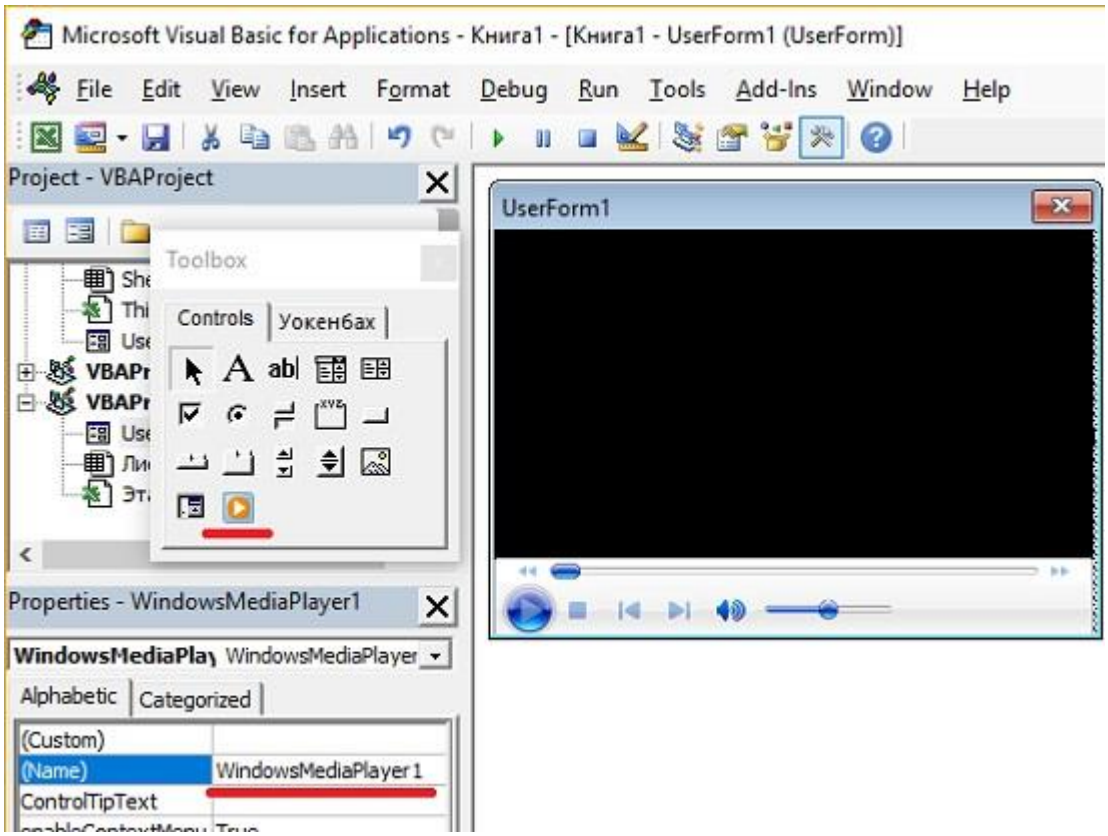

Рис. 9. Элемент управления *Windows Media Player*, встроенный в форму

На рис. 10 показан пример использования этого элемента управления. Для сокрытия окна, предназначенного для отображения видеороликов, была уменьшена высота окна элемента управления *Windows Media Player*. Также был добавлен список, созданный на основе элемента управления *ListBox*, в котором отображаются аудиофайлы MP3. После щелчка на кнопке *Пуск* начинается воспроизведение выбранного файла. Щелчок на кнопке *Закрыть* приведет к прекращению воспроизведения и к закрытию окна *UserForm*. Форма UserForm отображается в немодальном режиме, поэтому можно продолжать работу во время отображения диалогового окна.

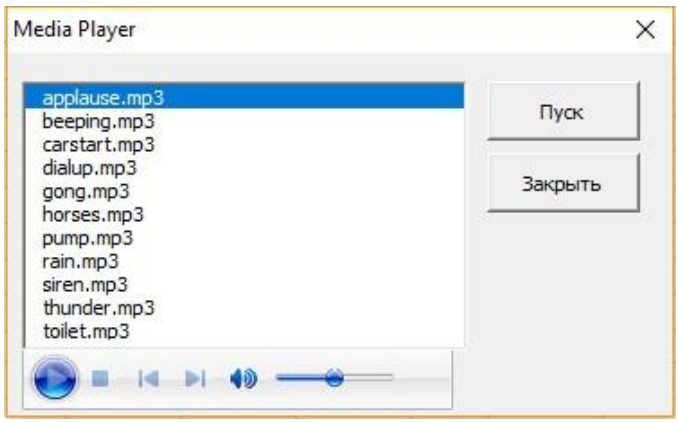

Рис. 10. Элемент управления *Windows Media Player* в действии

Названия MP3-файлов в окне списка отображаются с помощью процедуры *UserForm_Initialize*. В целях упрощения алгоритма аудофайлы находятся в той же папке, что и рабочая книга. Можно реализовать и более гибкий подход, предусматривающий выбор пользователем папки, содержащей требуемые аудиофайлы.

```
Private Sub UserForm_Initialize()
    Dim FileName As String
'   Заполнение списка MP3-файлами
    FileName = Dir(ThisWorkbook.Path & "\*.mp3", vbNormal)
    Do While Len(FileName) > 0
        ListBox1.AddItem FileName
        FileName = Dir()
    Loop
    ListBox1.ListIndex = 0
End Sub
```

Код обработчика событий *PlayButton_Click* включает единственный оператор, который присваивает выбранное имя файла свойству *URL* объекта *WindowsMediaPlayer1*.

```
Private Sub PlayButton_Click()
'   Свойство URL загружает трек и запускает плеер
    WindowsMediaPlayer1.URL = _
      ThisWorkbook.Path & "\" & ListBox1.List(ListBox1.ListIndex)
End Sub
```

Анимация элемента управления Label

Форма *UserForm* (рис. 11) представляет собой интерактивный генератор случайных чисел. Два элемента управления *TextBox* содержат верхнее и нижнее значения для случайного числа. Элемент управления *Label* изначально отображает четыре знака вопроса, а после щелчка мышью на кнопке *Пуск* отображаются анимированные случайные числа. При этом кнопка Пуск превращается в кнопку *Остановить*, а щелчок на ней мышью приводит к прекращению анимации и отображению случайного числа. Код, связанный с кнопкой генератора случайных чисел можно найти в модуле UserForm1 файла *random number generator.xlsm*.

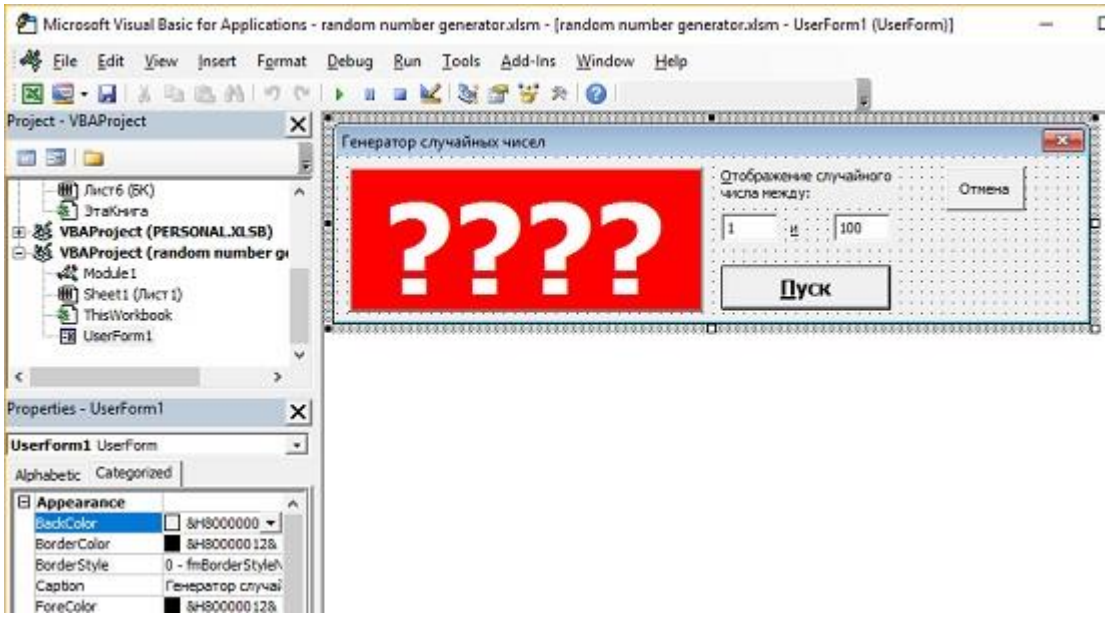

Рис. 11. Генератор случайных чисел

Поскольку кнопка выполняет две функции (запуск и остановка анимации), в процедуре используется общедоступная переменная *Stopped*, с помощью которой отслеживается состояние кнопки. Первая часть процедуры состоит из двух структур *If-Then*, с помощью которых проверяется содержимое элементов управления *TextBox*. Два другая подобных оператора позволяют удостовериться в том, что меньшая величина действительно не превосходит большей величины. В следующем разделе кода осуществляется настройка размера шрифта элемента управления *Label* на основании максимальной величины. Цикл *Do Until loop* генерирует и отображает случайные числа. Обратите внимание на оператор *DoEvents*. Он позволяет Excel использовать все возможности операционной системы. Если бы его не было, элемент управления *Label* не смог бы отобразить каждое генерируемое случайное число. Другими словами, именно оператор *DoEvents* делает возможной анимацию.

Форма *UserForm* также включает элемент управления *CommandButton*, который выполняет функции кнопки *Отмена*. Этот элемент управления находится за пределами окна *UserForm*, поэтому невидим. Свойству *Cancel* элемента управления *CommandButton* присвоено значение *True*. Вследствие этого нажатие клавиши <Esc> дает тот же эффект, что и щелчок на кнопке *Отмена*. Соответствующая процедура обработки событий присваивает переменной *Stopped* значение *True* и выгружает форму *UserForm*.