

## 14. Элементы управления в приложениях VBA

---

### **Кнопки как основа выполнения действий**

Наиболее простой путь обеспечить взаимодействие пользователя с приложением — предусмотреть наличие кнопок, нажатие которых будет вызывать необходимые действия. Для этих целей можно использовать элемент управления *Command Button* (Кнопка управления), предоставляемый системой VBA, или разработать собственную "кнопку", используя элемент управления *Image* (Изображение), содержащий графику (значок).

### **Кнопки управления**

Большинство приложений VBA имеет кнопки управления, простым нажатием которых пользователь может выполнять разнообразные действия. Когда пользователь нажимает кнопку, то не только выполняется соответствующее ей действие, но и она сама выглядит так, как будто ее вдавливают, а потом отпускают. При щелчке на кнопке вызывается событие *Click* (Щелчок). Для выполнения определенных действий при щелчке на кнопке в процедуру обработки события *Click* помещается соответствующий код.

---

Существует много способов "нажать" кнопку управления во время выполнения:

- *Использовать мышь, щелкнув на кнопке;*
- *Переместить фокус на кнопку, нажимая клавишу Tab, и затем выбрать эту кнопку, нажав клавишу Spacebar (пробел) или Enter (ввод);*
- *Нажать клавишу доступа (Alt+ <подчеркнутая буква в названии клавиши>) к данной для кнопки управления;*
- *Нажать кнопку управления программным путем, для чего любым способом (вручную с помощью окна свойств или с помощью оператора в программе) установить значение свойства Value (Значение) кнопки управления равным True (Истина), а затем с помощью оператора в программе вызвать процедуру обработки события click для этой кнопки;*
- *Если кнопка управления является кнопкой управления по умолчанию, выбрать эту кнопку, нажав клавишу Enter даже если в фокусе в данный момент оказывается другой элемент управления;*
- *Если кнопка управления по умолчанию является кнопкой отказа (cancel button), то, где бы не находился фокус, выбрать эту кнопку можно, нажав клавишу Esc.*


**Все** эти действия заставляют систему VBA вызывать процедуру обработки события Click. Понятие фокус рассматривается ниже в этой теме. Предварительно же отметим, что фокус означает выделенное, активное состояние объекта.

---

## **Элементы управления для отображения и ввода текста**

Элементы управления *Label* (Метка) и *TextBox* (Текстовое поле) применяются для отображения или ввода текста. Метки используются для отображения текста на форме, а текстовые поля — когда необходимо предоставить пользователю возможность вводить текст. Текст в метках можно только читать, тогда как текст в текстовых полях можно редактировать. Назначение элементов управления поясняет табл.

**Таблица.** <sup>1</sup> Назначение элементов управления для отображения и ввода текста

<i>Элемент управления</i>	<i>Назначение</i>
Текстовое поле ( <i>TextBox</i> ) 	<i>Ввод и отображение текста. Текст может редактироваться (изменяться) пользователем.</i>
Метка ( <i>Label</i> ) Метка 1	<i>Отображение не редактируемого текста, например, для обозначения поля на форме или отображения инструкции пользователю</i>

---

---

## **Подгонка размеров метки к ее содержимому**

*Однострочные надписи в метке можно задавать во время выполнения в окне `Properties`.*

*Но что делать, если необходимо вывести более длинные надписи, или надпись будет меняться во время выполнения? Метки имеют два свойства, которые помогают подгонять их размер для отображения надписей переменной длины. Это свойства `AutoSize` (Авторазмер) и `Wordwrap` (ПереносСлов).*

*Свойство `AutoSize` определяет, следует ли автоматически изменять размер элемента управления, чтобы он соответствовал выводимым данным. Если значение этого свойства равно `True`, то горизонтальный размер метки увеличивается, чтобы на ней поместилась надпись. Свойство `Wordwrap` увеличивает высоту метки, при этом ее ширина не меняется.*

---

---

## **Метки**

Элемент управления *Label* отображает текст, который пользователь не может изменять. Метки можно использовать для идентификации элементов управления, например, текстовых полей и полос прокрутки, не имеющих собственного свойства *caption* (Надпись). Текст, отображаемый в элементе *Label* (Метка), задается свойством *Caption* этого элемента. Оно может быть установлено во время разработки в окне *Properties* или во время выполнения оператором присваивания в программном коде.

По умолчанию надпись - единственная видимая часть элемента управления *Label*. Однако, если значение свойства *BorderStyle* установлено равным 1 (это можно сделать во время разработки), то метка появляется вместе с границей, что придает ей вид, похожий на вид текстового поля. Изменить внешний вид метки можно установкой свойств *BackColor* (Цвет фона), *BackStyle* (Стиль фона), *ForeColor* (Цвет букв) и *Font* (Шрифт).

---

---

## **Текстовые поля**

Текстовые поля являются гибкими элементами управления и применяются как для получения вводимых пользователем данных, так и для отображения текста. Их не следует использовать для отображения текста, который пользователь не должен изменять, если только значение свойства *Locked* (Заблокирован) не установлено равным *True*.

Текст, отображаемый в текстовом поле, управляется свойством *Text* (Текст). Его можно установить тремя различными способами: во время разработки в окне *Properties*, во время выполнения из программного кода или во время выполнения на основании данных, введенных пользователем.

Текущее содержимое текстового поля можно получить во время выполнения, считывая значение свойства *Text*.

---

## **Многострочные текстовые окна и перенос слов**

*По умолчанию в текстовом поле отображается одна строка текста, полосы прокрутки не отображаются. Если текст длиннее доступного пространства, будет видна только часть текста. Вид и поведение текстового поля можно изменить, установив значения свойств `MultiLine` и `ScrollBars`, которые доступны только во время разработки.*

### Примечание

*Свойство `ScrollBars` не следует путать с элементом управления `ScrollBar` (Полоса прокрутки), который не присоединяется к текстовым полям и имеет свой собственный набор свойств.*

*Установка значения свойства `MultiLine` равным `True` позволяет текстовому полю принимать и отображать много строк текста во время выполнения. Так как в многострочном текстовом поле нет горизонтальной полосы прокрутки, то оно автоматически управляет переходом на новую строку, если текст не помещается. По умолчанию значение свойства `ScrollBars` равно `None` (Нет).*

*Автоматический переход на новую строку избавляет пользователя от необходимости вводить в конце строк символ разрыва строки. Если строка текста длиннее строки текстового поля, то текстовое поле переносит вывод строки на следующую строку.*

---

*Значением свойства SelStart является число, которое указывает место вставки в строке текста, причем 0 соответствует крайней левой позиции. Если значением свойства SelStart является число, равное или больше, чем число символов в текстовом окне, точка ввода находится сразу за последним символом.*

*Число, равное значению свойства SelLength, задает ширину точки ввода. Если это число больше 0, то равное ему количество символов, начиная от текущей точки ввода, выбирается и выделяется.*

*Если пользователь начинает печатать при выделенном блоке текста, то последний будет замещен вводимым текстом. При необходимости можно заменить выделенный текст новым с помощью команды Paste (Вставить) меню Edit (Правка). Значением свойства SelText является строка текста, которую можно присвоить во время выполнения и которая заменит выделенный текст. Если такового нет, то в текущую точку ввода будет помещено значение свойства SelText.*

---



*Символ разрыва строки нельзя вводить в окне Properties во время разработки. В процедуре разрыв строки моделируется вводом символа возврата каретки, за которым следует символ перевода строки (символы с кодами ANSI 13 и 10 соответственно). Можно также использовать константу vbCrLf для ввода комбинации символов возврат каретки/перевод строки.*

*Например, следующая процедура обработки события вводит две строки текста в многострочное текстовое поле (Text1) при загрузке формы:*

***Sub Form1\_Load()***

***Text1.Text = "Here are two lines" & vbCrLf & "in a text box"***

***End Sub***

---

## **Работа с текстом в текстовом поле**

Управлять точкой ввода и поведением выделенного фрагмента (*Selection Behavior*) в текстовом поле можно с помощью свойств *SelStart*, *SelLength* и *SelText*. Эти свойства доступны только во время выполнения.

Когда текстовое окно впервые получает фокус, по умолчанию точка ввода (положение курсора) в текстовом поле находится слева от текста, находящегося в поле. Пользователь может его изменить с клавиатуры или с помощью мыши. Если текстовое окно теряет и затем снова приобретает фокус, точка ввода будет находиться там, где пользователь установил ее в последний раз.





В некоторых случаях, например, в приложениях обработки текстов пользователь предполагает, что новые символы должны появляться после существующего текста. В приложениях ввода данных пользователь предполагает, что новые данные замещают существующие. Свойства *SelStart* и *SelLength* позволяют модифицировать поведение текстового поля, приспособив его к конкретным целям.

---

## **Элементы управления, организующие выбор пользователя**

В большинстве приложений пользователю предоставляется возможность выбора, начиная от простейшего «да/нет», до сложного выбора из списка, состоящего из сотен возможностей. VBA содержит несколько стандартных элементов управления для организации выбора.

Таблица. Элементы управления для организации выбора

<b>Элемент управления</b>	<b>Возможности</b>
<i>Check boxes</i> (Флажки)  Флажок 11	Набор возможностей выбора, из которых пользователь может выбрать одну или более
<i>Option buttons</i> (Переключатели)  Перекл. 2	Набор возможностей, из которых пользователь может выбрать только одну
<i>Listbox</i> (Список) 	Прокручиваемый список возможностей выбора, из которого пользователь может выбрать только одну возможность
<i>Combobox</i> (Комбинированное окно) 	Прокручиваемый список возможностей выбора вместе с полем ввода. Пользователь может либо выбрать из списка, либо напечатать свой выбор в поле ввода

---

## **Флажки**

Флажок показывает, включено или нет определенное условие. В приложении флажки используются, чтобы дать пользователю возможность выбора типа «правда/ложь» или «да/нет». Так как флажки работают независимо друг от друга, пользователь может установить любое их число одновременно.

## **Группирование возможностей с помощью переключателей**

Переключатели предоставляют пользователю выбор из двух или более возможностей. В отличие от флажков, однако, они всегда работают как часть группы: выбор одного из них немедленно сбрасывает все другие переключатели группы. Объединение переключателей в группу означает для пользователя, что он может выбрать одну и только одну возможность из заданного набора.

---

---

## **Создание групп переключателей**

Существует два способа группировки переключателей:

1. При помощи рамки (*Frame*);
2. При помощи свойства *GroupName*.

Все переключатели, размещенные непосредственно на форме (т. е. не внутри рамки (*Frame*), составляют одну группу. Если необходимо создать дополнительные группы переключателей, некоторые из них надо поместить внутрь рамок.

Все переключатели внутри данной рамки составляют отдельную группу.

При создании отдельной группы переключателей всегда сначала следует создать рамку, а затем поверх них— переключатели.

Пользователь может выбрать только один переключатель в группе внутри рамки.

---

*Для группирования элементов управления с помощью рамки следует:*

- 1. Выбрать элемент управления `Frame` (Рамка) на Панели элементов управления (`Toolbox`), нарисовать рамку и поместить ее на форму.*
- 2. Выбрать переключатель на Панели элементов управления и нарисовать его внутри рамки.*
- 3. Повторить шаг 2 для остальных переключателей, добавляемых в группу.*

*Соблюдение последовательности создания группы элементов управления (сначала рисуется рамка, а затем внутри нее рисуется каждый элемент управления из группы) позволяет работать с ними как с группой. При перемещении рамки вместе с ней будут перемещаться и все элементы внутри нее. Если переместить существующий на форме элемент управления внутрь рамки, то он при перемещении рамки останется неподвижным, т. к. нарушена процедура создания группы.*

#### *Примечание*

*Чтобы сгруппировать существующие элементы управления, следует выбрать все элементы, которые будут составлять группу, вырезать их и вставить в рамку.*

*Можно сгруппировать переключатели, установив для них одинаковым свойство `GroupName` в окне свойств `Properties`.*

---

## **Контейнеры для элементов управления**

Элементы управления являются независимыми объектами, однако между ними и формами существуют отношения подчиненности (*parent and child relationships*).

Для понимания идеи контейнеров необходимо четко осознавать, что все элементы управления являются подчиненными по отношению к форме, на которой они нарисованы. Фактически большинство элементов управления поддерживает свойство *Parent* (Родитель) в режиме «только чтение» (*read-only*), т. е. значением этого свойства является форма, на которой расположен элемент управления, и это значение невозможно изменить. Положение элемента управления ограничено родительской формой, по отношению к которой он является подчиненным. Свойства *Left* и *Top* элемента управления связаны с родительской формой, и поэтому элемент управления нельзя переместить за границы этой формы. Перемещение контейнера также перемещает и элементы управления, и не изменяет значений свойств *Left* и *Top* элемента управления.

---

---

## **Выбор переключателей**

Переключатель можно выбрать:

- Щелчком на нем кнопкой мыши во время выполнения;
- Клавишей Tab выбрать группу переключателей и затем клавишами со стрелками выбрать переключатель в группе;
- Установив значение его свойства Value равным True в программном коде: **OptionButton1.Value = True;**
- С помощью клавиши быстрого доступа, определенной в надписи метки.

Сделать переключатель в группе переключателем по умолчанию можно во время разработки. Для этого надо установить значение его свойства Value равным True. Он остается выбранным, пока другой переключатель не будет выбран пользователем или программно.

Можно сделать переключатель недоступным, установив значение его свойства Enabled (Доступный) равным False. При выполнении программы такой переключатель выделен серым цветом, и это означает, что он не доступен пользователю.

---



## **Комбинированные окна и списки**

Списки и комбинированные окна предоставляют пользователю список возможных вариантов выбора, по умолчанию отображаемых вертикально в одну колонку, хотя возможно отображение и в несколько колонок. Если число элементов превышает возможности отображения в комбинированном окне или списке, автоматически появляется полоса прокрутки. Пользователь в этом случае может просматривать список элементов, прокручивая его вверх и вниз или влево и вправо.

Комбинированное окно совмещает возможности списка и текстового поля. В этом элементе управления пользователь может производить выбор либо вводом текста в поле ввода комбинированного списка, либо выбором элемента из его списка.

По сравнению с другими элементами управления, которые содержат одно значение, списки и комбинированные окна содержат набор значений. Для них существуют встроенные методы добавления, удаления и получения значений из этого набора во время выполнения. Программный код для добавления нескольких элементов в список с именем *List1* мог бы выглядеть следующим образом:

```
List1.AddItem "Paris"
```

```
List1.AddItem "New York"
```

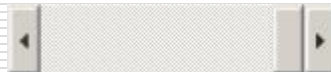
```
List1.AddItem "San Francisco"
```

Списки и комбинированные окна — это эффективный способ представления большого числа вариантов выбора для пользователя в ограниченном пространстве на форме.

---

## **Полосы прокрутки как устройства ввода**

*В основном полосы прокрутки используются в текстовых полях или окнах, но иногда их применяют в качестве устройств ввода. Полосы прокрутки показывают текущую позицию на масштабной линейке, поэтому их можно использовать самостоятельно для управления вводом в программу, например, регулировать цветовую палитру в картинке или уровень громкости.*



*Элементы управления `HScrollbar` (Горизонтальная полоса прокрутки) и `VScrollbar` (Вертикальная полоса прокрутки) функционируют независимо от других элементов управления и имеют собственные наборы событий, свойств и методов.*

*Элементы управления «Полоса прокрутки» — это не то же самое, что встроенные полосы прокрутки, присоединяемые к текстовым полям, спискам, комбинированным окнам.*

---

---

## **Графические элементы управления. Отображение картинок и графики**

### **Элемент управления Image**

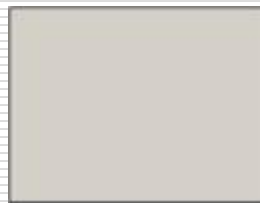
В состав VBA включен объект для работу с графикой: *Image* (Изображение).

Картинки загружаются в объект *Image*: во время разработки — назначением имени файла и пути свойству *Picture*, во время выполнения — посредством функции *LoadPicture*, например:

```
Private Sub Image1_Click()
```

```
Image1.Picture = LoadPicture("E:\Водяные лилии.jpg")
```

```
End Sub
```



---

*Объект Image имеет свойство AutoSize, которое, если его значение равно True, автоматически изменяет размеры окна вывода картинки, чтобы оно соответствовало размерам выводимой картинки.*



---

## **Создание кнопок**

*Объект Image распознает событие Click, что предоставляет удобный способ создания кнопок с картинками вместо кнопок с надписями. Объединение нескольких элементов управления Image в горизонтальном направлении вверху экрана в группу позволяет создавать панели инструментов приложения.*

### Примечание

*В отличие от кнопок управления, элементы управления Image не вдавливаются, если на них выполняется щелчок.*

---

---

## **Что такое фокус?**

Объект, имеющий фокус, может получать вводимую пользователем с помощью мыши и клавиатуры информацию. В системе Microsoft Windows одновременно могут выполняться несколько приложений, но только у приложения, имеющего фокус, будет активный заголовок окна (*active title bar* - выделен повышенной яркостью), и только оно может взаимодействовать с пользователем. На форме приложения VBA с несколькими полями ввода только поле ввода, имеющее фокус, будет отображать вводимый с клавиатуры текст.

Когда объект получает или теряет фокус, происходят соответственно события *GotFocus* и *LostFocus*. Формы и большинство элементов управления поддерживают эти события.

---

---

*Объекту можно передать фокус несколькими способами:*

- Выбрать объект с помощью мыши во время выполнения;*
- Выбрать объект с помощью клавиши доступа во время выполнения;*
- Из программного кода с помощью метода `Set Focus`.*

*Всегда можно определить, имеет ли объект фокус. Например, когда кнопка управления имеет фокус, обрамление ее надписи отображается с повышенной яркостью.*

*Объект может получить фокус, только если его свойства `Enabled` (Разрешено) и `Visible` (Видимый) установлены в `True`. Свойство `Enabled` (Разрешено) позволяет объекту реагировать на инициированные пользователем события, например, ввод с клавиатуры и нажатие кнопок мыши. Свойство `Visible` (Видимый) определяет, виден ли объект на экране.*

---

---

## **Событие *Validate* элементов управления**

Элементы управления имеют событие *Validate* (Проверка достоверности), которое происходит перед тем, как элемент управления теряет фокус. Однако это событие возникает, только если свойство *CausesValidation* элемента управления перед получением фокуса имеет значение *True*. Событие *Validate* происходит перед потерей фокуса, поэтому во многих случаях оно более подходит для проверки достоверности данных (*data validation*), чем событие *LostFocus*.

### **Элементы управления, которые не могут получить фокус**

Некоторые элементы управления не могут получать фокус. К ним относятся облегченные графические элементы управления:

*Frame* (Рамка)

*Image* (Изображение)

*Label* (Метка)

Кроме этих элементов управления не могут получить фокус и неотображаемые элементы управления.

---



---

## **Установка последовательности перехода**

Последовательность перехода (*tab order*) — это последовательность перехода от одного элемента управления к другому при нажатии клавиши *Tab*. У каждой формы своя последовательность перехода. Обычно она соответствует последовательности создания элементов управления.

Пусть создаются два поля ввода, *Text1* и *Text2*, а затем — кнопка управления *Command1*. При запуске приложения поле ввода *Text1* имеет фокус.

Нажатие клавиши *Tab* перемещает фокус между элементами управления в той последовательности, в которой они создавались.

Для изменения последовательности перехода необходимо установить свойство *TabIndex* элемента управления, которое определяет его позицию в последовательности перехода при нажатии клавиши *Tab*. По умолчанию значение свойства *TabIndex* элемента управления, созданного на форме первым, равно 0, элемента, созданного вторым — 1 и т. д.

---

*При изменении позиции элемента управления в последовательности переходов VBA автоматически перенумеровывает позиции других элементов управления, отражая вставку в другую позицию или удаление из последовательности перехода элемента управления.*

*Например, если кнопка управления Command1 в предыдущем примере становится первой в последовательности переходов, то значения свойства TabIndex других элементов управления автоматически перенумеровываются.*

*Наибольшее значение свойства TabIndex всегда на единицу меньше числа элементов в последовательности перехода, так как нумерация начинается с нуля. Даже если случайно значение свойства TabIndex окажется больше числа элементов управления, VBA переустановит это значение так, чтобы оно было на единицу меньше числа элементов управления.*

### *Примечание*

*Элементы управления, которые не могут получить фокус, как и недоступные и невидимые элементы управления, не имеют свойства TabIndex и не могут быть включены в последовательность перехода. Когда пользователь нажимает клавишу Tab, эти элементы управления пропускаются.*

---

## **Удаление элемента управления из последовательности перехода**

Обычно последовательное нажатие клавиши *Tab* во время выполнения выбирает каждый элемент управления из заданной последовательности перехода. Можно убрать элемент управления из последовательности перехода, установив значение его свойства *TabStop* равным *False* (или *0*).

Элемент управления, свойство *TabStop* которого равно *False*, сохраняет свою позицию в последовательности перехода, хотя при переходе от одного элемента управления к другому по клавише *Tab* он пропускается.

### Примечание

Группа переключателей имеет один табуляторный ограничитель (*tab stop*) Для выбранной кнопки, т.е. кнопки, значение свойства *Value* которой равно *True*, значение свойства *TabStop* автоматически равно *True*, тогда как значение этого свойства для других кнопок равно *False*.

---