

# **СОЗДАНИЕ ФОРМ В VBA И ВКЛЮЧЕНИЕ ИХ В ПРОЕКТЫ**

---

к.п.н., доцент Суханова Н.Т.

# СОЗДАНИЕ ФОРМ

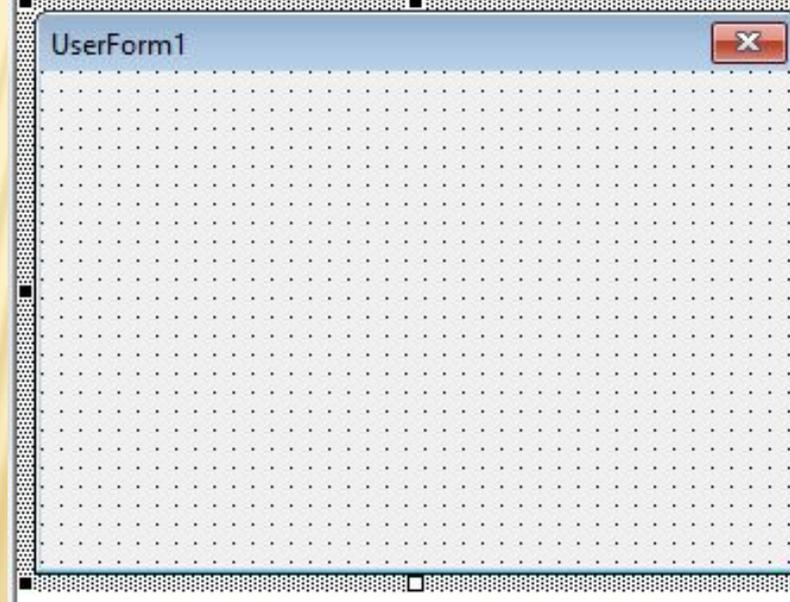
---

**Форма** – это глобальный объект, образующий визуальную основу приложения. По своей сути форма представляет собой окно, в котором можно размещать различные управляющие элементы при создании приложений.

# СОЗДАНИЕ ФОРМ

---

Для создания формы необходимо выполнить Команду **Insert – UserForm**. В конструкторе форм  
появится форма, имеющая стандартный  
вид.



# СВОЙСТВА ФОРМЫ

Как и любой объект VBA форма имеет набор свойств, основные из которых приведены в таблице.

Свойство	Описание
<b>BackColor</b>	Цвет фона для формы
<b>BorderStyle</b>	Определяет тип границы, окружающей форму
<b>Caption</b>	Текст, который выводится в заголовке формы
<b>Font</b>	Определяет тип и вид шрифта в форме
<b>Height</b>	Определяет высоту формы в твипах
<b>Name</b>	Имя объекта, для программы VBA
<b>Width</b>	Определяет ширину формы в твипах

# СВОЙСТВА ФОРМЫ

---

Свойства можно изменять в режиме конструирования в окне свойств, либо программно в режиме выполнения.

*Например:*

Изменить заголовок формы можно командой:

**frmForm1.Caption = “Привет”**

# СОБЫТИЯ ФОРМЫ

---

Программы управляются событиями. Каждый раз, когда нажимается кнопка, перемещается мышь, изменяются размеры формы и т.д.

Операционная система генерирует сообщение,

Которое доставляется соответствующему объекту, например форме, а та генерирует соответствующее **Событие**.

# СОБЫТИЯ ФОРМЫ

---

Следовательно, можно составить фрагмент программы, в котором объект будет реагировать на событие определенным образом, т.е. любому стандартному соответствует определенная процедура.

Чтобы просмотреть события связанные с формой,

Необходимо в режиме конструирования дважды

щелкнуть на ней – появится окно программы, в котором щелкнуть на списке **Процедура**.

# ОСНОВНЫЕ СОБЫТИЯ ФОРМ

Событие	Описание
<b>Initialize</b>	Происходит во время конфигурации и до загрузки формы в память
<b>Activate</b>	Происходит после загрузки формы в память
<b>Deactivate</b>	Происходит, если форма перестает быть активной
<b>Click</b>	Происходит при нажатии левой кнопки мыши на форме

**Пример** в котором происходит изменение заголовка формы при активизации, и уменьшает размер формы после щелчка левой кнопки мыши по форме.

```
Private Sub UserForm_Activate()  
frmForm1.Caption = “Щелчок на форме  
уменьшает её размеры”
```

```
End Sub
```

```
Private Sub UserForm_Click()  
frmForm1.Width = frmForm1.Width/2  
frmForm1.Height = frmForm1.Height/2  
frmForm1.Caption = “Сделай это еще раз!”
```

```
End Sub
```

# МЕТОДЫ ФОРМЫ

Форма обладает набором методов и инструкций. Метод определяет действие, которое может быть выполнено с объектом. Инструкция инициирует действия. Она может выполнить метод или функцию.

Метод	Описание
<b>Hide</b>	Скрывает объект UserForm, но не выгружает его
<b>Show</b>	Выводит на экран объект UserForm

Инструкция	Описание
<b>Load</b>	Загружает объект UserForm, но не отображает его на экране
<b>UnLoad</b>	Удаляет объект UserForm из памяти

# ПРИМЕР

---

Предполагается что созданы две формы **frmForms**. При запуске проекта происходит событие **Initialize** для формы **frmForm1**, форма **frmForm2** загружается и выводится на экран.

**Когда** при помощи мыши выбирается **frmForm2**, она делается невидимой, и появляется форма **frmForm1**.

Если же выбирается **frmForm1**, **frmForm2** появляется вновь.

# ПРИМЕР

---

‘ Событие Initialize формы UserForm1

**Private Sub UserForm\_Initialize()**

**Load frmForm2**

**frmForm2.Show**

**End Sub**

‘ Событие Click для формы UserForm2

**Private Sub UserForm\_Click ()**

**frmForm2.Hide**

**End Sub**

‘ Событие Click для формы UserForm1

**Private Sub UserForm\_Click ()**

**frmForm1.Show**

**End Sub**

# ИСПОЛЬЗОВАНИЕ УПРАВЛЯЮЩИХ ЭЛЕМЕНТОВ

---

Создание управляющих элементов на форме выполняется с помощью *Панели инструментов* (**Вид – Панель элементов (View – Toolbox)**).

С помощью кнопок этой панели можно поместить в форму необходимый элемент управления.

Каждый элемент управления характеризуется набором свойств, событий и методов.

Для каждого объекта проекта необходимо определить его имя. В соответствии с общепринятыми соглашениями об именах объектов первые три символа имени должны отражать вид элемента, а остальные символы

# РЕКОМЕНДУЕМЫЕ СОЧЕТАНИЯ ПЕРВЫХ ТРЕХ СИМВОЛОВ ИМЕН

Объект	Первые три символа имени	Пример имени
Форма	<b>frm</b>	<b>frmMyForm</b>
Надпись	<b>lbl</b>	<b>lblInfo</b>
Текстовое поле	<b>txt</b>	<b>txtInput</b>
Командная кнопка	<b>cmd</b>	<b>cmdExit</b>
Флажок	<b>chk</b>	<b>chkSound</b>
Переключатель	<b>opt</b>	<b>optLevel</b>
Список	<b>lsb</b>	<b>lsbTypes</b>
Рамка	<b>fra</b>	<b>fraChoices</b>
Полоса прокрутки	<b>veb</b>	<b>vcbSpeed</b>
Рисунок	<b>pic</b>	<b>picChema</b>

# ЭЛЕМЕНТ УПРАВЛЕНИЯ КОМАНДНАЯ КНОПКА

---

Командная кнопка самый распространенный элемент управления.

Может использоваться для организации выполнения вычислений и других действий (вызов процедур и функций пользователя, открытие форм и т.д.)

# СВОЙСТВА КОМАНДНЫХ КНОПОК

Свойство	Описание
<b>BackColor</b>	Цвет фона кнопки
<b>Caption</b>	Текст, который выводится на кнопке
<b>Enabled</b>	Значение False делает кнопку недоступной
<b>Font</b>	Определяет тип и вид шрифта на кнопке
<b>ForeColor</b>	Определяет цвет шрифта на кнопке
<b>Name</b>	Имя объекта для программы VBA
<b>Picture</b>	Добавляет рисунок на кнопку
<b>PicturePosition</b>	Определяет расположение текста и рисунка на кнопке
<b>Visible</b>	Значение False делает кнопку невидимой

# СОБЫТИЯ И МЕТОДЫ КОМАНДНОЙ КНОПКИ

---

Основным событием командной кнопки является **Click**.

Наиболее полезным методом командной кнопки является **SetFocus**, позволяющий вернуться к кнопке (передать ей фокус).

*Например:* команда, позволяющая вернуться к кнопке по умолчанию после ввода данных в текстовое поле: **cmdMyButtum.SetFocus**

# ТЕКСТОВОЕ ПОЛЕ

Текстовое поле применяется для ввода или вывода информации.

Свойство	Описание
<b>Enabled</b>	Значение False делает поле недоступным
<b>Font</b>	Определяет тип и вид шрифта в текстовом поле
<b>ForeColor</b>	Определяет цвет шрифта в текстовом поле
<b>Name</b>	Имя объекта для программы VBA
<b>MaxLength</b>	Определяет количество вводимых символов в текстовое поле
<b>PasswordChar</b>	Определяет символ, отображаемый при вводе в текстовое поле
<b>Text</b>	Определяет содержимое текстового поля

**Например:** для очистки содержимого текстового поля нужно ввести команду **txtResult.Text = “ “**

Основным событием текстового поля является текстового поля **Change** происходящее при вводе или удалении символов.

*Например,* команду **cmdMyButtum.SetFocus** можно

поместить в процедуру события **Change** текстового поля.

# НАДПИСЬ

---

*Надпись* применяется как самостоятельно для вывода справочной информации, так и в виде «подсказок» для текстового поля, списка и другого элемента.

Главное её отличие от текстового поля в том, что пользователь не может изменить текст надписи.

# СВОЙСТВА НАДПИСИ

Свойство	Описание
<b>Caption</b>	Определяет текст, содержащийся в надписи
<b>Font</b>	Определяет тип и вид шрифта надписи
<b>ForeColor</b>	Определяет цвет шрифта надписи
<b>Name</b>	Имя объекта для программы VBA
<b>Picture</b>	Добавляет рисунок в надпись
<b>PicturePosition</b>	Определяет расположение текста и рисунка надписи

# СПИСОК

---

*Список* позволяет работать с перечнем из нескольких вариантов. Пользователь может просмотреть содержимое списка и выбрать один из вариантов для последующей обработки.

Прямое редактирование содержимого списка невозможно.

Если в списке помещаются не все строки, то автоматически добавляется вертикальная полоса прокрутки.

# СВОЙСТВА СПИСКА

свойство	описание
<b>Name</b>	Имя объекта для программы VBA
<b>ListIndex</b>	Возвращает номер текущей выделенной строки списка -1
<b>Text</b>	Содержимое текущей выделенной строки списка

Для списка чаще всего используется событие **Click** и **DbClick** во втором случае пользователь одновременно выделяет строку и начинает её обработку.

# МЕТОДЫ СПИСКА

---

Работа со списком начинается с его заполнения методом **AddItem**, который может вызываться несколько раз подряд.

Часто метод **AddItem** помещается в процедуру **UserForm)Initialize()**, чтобы список заполнялся при загрузке формы.

Метод **RemoveItem** удаляет строки из списка.

Метод **Clear** очищает сразу весь список.

---

**Пример:** показывает, как работают списки, при этом предполагается, что в проекте создана форма с двумя списками (**List1** и **List2**).  
Двойной щелчок на любой строке одного списка перемещает её в другой список. Строка включается в другой список до того, как она будет удалена из текущего.

## **Private Sub UserForm\_Initialize()**

```
List1.AddItem "Стол"  
List1.AddItem "Стул"  
List1.AddItem "Диван"  
List1.AddItem "Кресло"  
List1.AddItem "Кровать"
```

## **End Sub**

## **Private Sub List1\_DblClick()**

```
List2.AddItem List1.Text  
List1.RemoveItem  
List1.ListIndex
```

## **End Sub**

## **Private Sub List2\_dblClick()**

```
List1.AddItem List2.Text  
List2.RemoveItem  
List2.ListIndex
```

## **End Sub**

# ПЕРЕКЛЮЧАТЕЛИ

---

*Переключатели* позволяют выбрать один вариант из группы.

Обычно они группируются в рамках, но их можно располагать прямо на форме, Если используется только одна группа переключателей.

# СВОЙСТВА ПЕРЕКЛЮЧАТЕЛЯ

свойство	описание
<b>Caption</b>	Задаёт текст, определяющий назначение переключателя
<b>Name</b>	Имя объекта для программы VBA
<b>Value</b>	Значение True указывает, что переключатель выбран

Наиболее важным является свойство **Value** значение **True** (переключатель находится в установленном состоянии), Которого в режиме конструирования задается только у одного переключателя в группе. В режиме выполнения это свойство чаще всего проверяется в процессе события **Click** кнопки, нажатой после установки нужного переключателя, что позволяет проверить перед вызовом следующей процедуры некоторое условие.

Однако определенные действия можно выполнять сразу же после выбора переключателя в процедуре его события **Click**.

# ФЛАЖОК

---

**Флажок** частично аналогичен переключателю, но в отличие от него может использоваться как отдельный самостоятельный элемент. Даже объединенные в группу флажки работают независимо друг от друга. Основные же свойства флажков такие же, как и у переключателя. Однако свойство **Value** может принимать три значения (флажок находится в установленном состоянии, снятом или неопределенном).

Наиболее часто используемым событием флажков является **Click**, В процедуре которого можно проверять состояние флажка по свойству **Value**. Следующий пример иллюстрирует работу флажков, при этом предполагается, что в проекте создана форма с двумя флажками (**ChkBold** и **ChkItalic**) и текстовым полем **TxtExam**. После ввода символов в текстовое поле, с помощью флажков можно делать текст полужирным или курсивом. Свойства **FontBold** и **FontItalic** текстового поля устанавливают способы начертания текста.

```
Private Sub Chkbold_Click()  
    If ChkBold.Value = True Then  
        TxtExam.FontBold = True  
    Else  
        TxtExam.FontBold = False  
    End If  
End Sub  
Private Sub ChkItalic_Click()  
    If ChkBold.Value = True Then  
        TxtExam.FontItalic = True  
    Else  
        TxtExam.FontItalic= False  
    End If  
End Sub
```

# РАМКА

---

*Рамка* используется для группировки переключателей или флажков, и помещается на форму раньше элементов, находящихся внутри неё. Переключатели находящиеся внутри рамки, работают как самостоятельная группа и не влияют на состояние переключателей в других рамках. Основным свойством рамки является **Caption**, которое задает текст, определяющий назначение элементов в рамке

# РИСУНОК

Рисунок используется для простейшего вывода изображения на форме. Он может отображать растровые файлы (.bmp), значки (.ico), метафайлы (.wmf), а также файлы в формате jpeg (.jpg), gif (.gif).

События и методы рамок и рисунков практически не используются.

свойство	описание
<b>Autosize</b>	Значение True подгоняет размер элемента под размер содержимого
<b>Name</b>	Имя объекта для программы VBA
<b>Picture</b>	Задаёт файл для рисунка