**Лабораторная работа №8**

Создания пользовательской формы.

**1. Условие задачи:**

Создать приложение для работы по созданию списка.

• Заголовок списка формируется с помощью макроса.

• Для ввода данных используется пользовательская форма.

• После ввода данных таблица форматируется

На **рис. 1** показан рабочий лист и расположенный на нем заголовок таблицы.

Для заголовков таблицы установите полужирный шрифт, размер шрифта 12, тип шрифта Times New Roman.

Рис. 1

На **рис.2** приведена пользовательская форма, которую необходимо создать.

Экранная форма должна иметь следующие элементы управления и характеристики

**TextBox** для ввода имени, фамилии, города, адреса и индекса

• **ComboBox** для выбора области .

• Командную кнопку **Далее** для сохранения и подготовки формы для ввода

• Командную кнопку **Готово** сохранения данных, файла и закрытия формы

• Командную кнопку **Отмена –** игнорирование ввода и закрытие формы.

• Вводимая информация должна проверяться.

Рис. 2

**Решение задачи:**

1. Создать рабочую книгу **Адреса.xls**, используемую для хранения информации о группе людей. Сохранить файл в рабочей папке. Рабочий лист с данными также называем **Адреса.**
2. Создать макрос для построения строки заголовка создаваемой таблицы. Имя макроса **Заголовок**

Текст макроса дополнить кодами, которые необходимо вставить перед первым оператором. Назначение операторов проанализируйте.

Эти операторы актуальны, если макрос строит заголовок таблицы, начиная с ячейки **A2.**

1. Создать пользовательскую форму

*Проектирование формы представленной на рис.2*

1. Открыть редактор VBA (<Alt+F11> или..)

2. Вставить форму

3. Присвоить свойствам значения: **Name** ⇒**Address,**  а свойству **Caption**⇒**Ввод данных.**

*Добавление элементов управления*

1. Добавьте **5** элементов **TextBox** присвойте их свойствам **Name** значения **txtName**, **txtFIO**, **txtTown**, **txtAdres**, **txtIndex.** Поместите рядом c текстовыми окнами элементы **Label** и присвойте их свойствам **Caption** соответствующие значения (Имя, Фамилия и т.д. см. рис.2)
2. Добавьте к форме элемент управления **ComboBox** и присвойте егосвойству **Name** значение **Region.** Присвойте свойству **Style** этого комбинированного списка значение **fmStyleDropDownList.**
3. Поместите рядом cо списком элемент **Label** и присвойте его свойству **Caption** соответствующее значение.
4. Добавьте элемент управления **CommandButton.** Присвойте его свойству **Name** значение **cmdDone,** а свойству **Caption –** значение **Готово.**
5. Добавьте еще один элемент управления **CommandButton.** Присвойте его свойству **Name** значение **cmdNext,** а свойству **Caption –** значение **Далее** и свойству **Default** – значение **True.**
6. Добавьте третий элемент управления **CommandButton.** Присвойте его свойству **Name** значение **cmdCancel,** а свойству **Caption –** значение **Отмена**, свойству **Cancel –** значение **True.**
7. Используя форматирование с помощью курсора (вручную) и команды меню **Format** разместите элементы управления на форме в порядке примерно соответствующем приведенному на рис. 2.

*Имена элементов управления используются при создании ссылок на них в процедурах. Приведенные далее по тексту примеры процедур используют описанные имена.*

1. Создать процедуру, выполняющуюся при инициализации формы. Код этой процедуры выполняет одно действие, загружает список **ComboBox** c названиями областей. В этом списке привести названия: Брестская, Витебская, Гомельская, Гродненская, Минская, Могилевская.

*Порядок создания процедуры*:

1. Открыть окно редактирования кодов для экранной формы. Для этого дважды щелкните левую кнопку мыши, расположив курсор на свободном месте формы, или нажмите **<F7>,** или щелкните по кнопке **View Code** окна **Project.**
2. Из списка в левой верхней части окна ввода кодов выберите вариант **UserForm.**
3. Из списка в правой верхней части окна выберите вариант **Initialize**.
4. Наберите код

Заготовка процедуры приведена ниже. Текст процедуры дополнить строками для загрузки названий остальных областей.

Private Sub UserForm\_Initialize()

'Загружает комбинированный список с названиями областей

Region.AddItem "Брестская"

**…..**

**…..**

Region.AddItem "Могилевская"

End Sub

 **5**. Проверка корректности почтового индекса, значение которого может состоять только из набора цифр.

При нажатии клавиши элементу управления передается числовое значение, соответствующее коду клавиши. Если воспользоваться событием **KeyDown,** которое получает значение аргумента соответствующего нажатой клавише, можно оценить это значение, прежде чем оно передается элементу управления.

Известно также, что клавишам **0-9** соответствуют коды **48-57**.

С учетом этого создадим процедуру проверки корректности вводимого индекса

*Порядок создания процедуры*:

1. В окне редактора VBA активизируйте форму.
2. Дважды щелкните по окну ввода индекса.
3. В окне ввода кодов для элемента управления **txtIndex** из списка справа выберите **KeyDown.** В окно ввода кодов будет добавлена структура процедуры обработки события

Private Sub txtIndex\_KeyDown(ByVal KeyCode As MSForms.ReturnInteger, ByVal Shift As Integer)

End Sub

1. **Введите код**

Использование оператора **Beep** означает, что если будет нажата неправильная клавиша, система издает звуковой сигнал.

**Сохраните файл Адреса.xls**

**6. Добавление кодов, проверяющих корректность вводимых данных.**

Необходимо добавить коды, которые при нажатии клавиш **Далее** и **Готово** проверяют корректность вводимых данных. При этом должно проверяться, что:

• заполнены поля Имя, Фамилия, Город и Адрес;

• выбрана область;

• в поле почтового индекса содержится шесть символов.

После проверки, если все корректно, данные вводятся в ячейки рабочего листа, поля формы очищаются и она отображается для ввода следующего набора данных.

Процедуру можно вызывать из процедур обработки события **Click** кнопок **Далее** и **Готово.**

*Порядок создания процедуры*

1. Откройте окно редактора кодов формы.
2. Выберите команду **Вставить⇒ Процедуру.**
3. Наберите название процедуры **Проверка\_Данных,** в окне **Type** выберите вариант **Function.**
4. **OK.**

Код процедуры приведен ниже. Добавьте к своей программе представленный код и сохраните файл **Адреса.xls.**

Public Function Проверка\_Данных() As Boolean

If txtName.Value = "" Then

 MsgBox "Введите имя."

 Проверка\_Данных = False

 Exit Function

End If

Далее используя аналогичный алгоритм, введите операторы проверки ввода данных в поля:

**Фамилия**

**Город**

**Адрес**

If Region.Value = "" Then

 MsgBox "Вы должны выбрать область."

 Проверка\_Данных = False

 Exit Function

End If

If txtIndex.TextLength <> 6 Then

 MsgBox "Вы должны ввести 6 символов индекса."

 Проверка\_Данных = False

 Exit Function

End If

Проверка\_Данных = True

End Function

1. **Окончание проекта.**

Для завершения необходимо написать процедуры обработки событий **Click** для трех элементов управления командных клавиш **Отмена, Готово** и **Далее.**

Необходимо создать процедуру, которая сохраняет набранные в полях данные в ячейках рабочего листа. Назовем процедуру **EnterData.** Добавьте эту процедуру к кодам экранной формы

Следует также написать код для очистки полей экранной формы. Такие действия должны выполняться после щелчка по любой из командных клавиш. Код для удаления набранных данных в полях формы будет помещен в процедуру **ClearForm**.

Добавьте эту процедуру к своему проекту.

Таким образом, на этом этапе необходимо написать пять процедур.

*Пояснения к создаваемым процедурам:*

С помощью свойства **CurrentRegion** объекта **Range** определяется диапазон r, в котором размещена вся таблица, размер которой заранее может быть неизвестен.

Set r1 = r.Offset(r.Rows.Count, 0) – диапазон **r1** смещен вниз относительно диапазона **r** на величину равную количеству строк (r.Rows.Count) в диапазоне **r.**

*Процедура* **EnterData**

Public Sub EnterData ()

' Копирует данные из UserForm

' в следующую строку списка на рабочем листе.

Dim r As Range, r1 As Range

Set r = Worksheets("Адреса").Range("A2").CurrentRegion

Set r1 = r.Offset(r.Rows.Count, 0)

r1.Cells(1,1).Value = txtName.Value

r1.Cells(1,2).Value = txtFIO.Value

r1.Cells(1,4).Value = txtTown.Value

r1.Cells(1,5).Value = txtAdres.Value

r1.Cells(1,3).Value = Region.Value

r1.Cells(1,6).Value = txtIndex.Value

End Sub

*Процедура* **ClearForm**

Public Sub ClearForm()

' Clears all data from the form.

txtName.Value = ""

txtFIO.Value = ""

txtTown.Value = ""

txtAdres.Value = ""

txtIndex.Value = ""

Region.Value = ""

End Sub

**Сохраните файл Адреса.xls**

*Процедуры* обработки события **Click** для командных кнопок.

Коды этих процедур, показанные ниже добавьте к кодам экранной формы и проект будет готов к работе.

**Private** Sub **cmdCancel\_**Click()

ClearForm

Me.Hide

End Sub

**Private** Sub **cmdDone\_**Click()

If Проверка\_Данных = True Then

 EnterData

 ClearForm

 Me.Hide

End If

End Sub

**Private** Sub **cmdNext\_**Click()

If Проверка\_Данных = True Then

 EnterData

 ClearForm

End If

End Sub

**Сохраните файл Адреса.xls**

**8. Тестирование проекта.**

**Необходимо обеспечить автоматическое открытие формы при открытии книги (файла).**

Для этого щелкнем дважды по кнопке Эта книга и выбираем пункт WindowActivate.

В открывшемся окне запишем следующий текст процедуры (выделенный жирным шрифтом):

Private Sub Workbook\_windowactivate(ByVal wn As Excel.Window)

**Заголовок**

**Address.Show**

End Sub

**Сохраните и закройте файл Адреса.xls**

*Для тестирования созданного проекта*

*Запустите файл* **Адреса.xls** после появления на экране строки заголовка таблицы и экранной формы поработайте с ней, введите 5-7 строк. При вводе данных проверьте работоспособность процедур осуществляющих контроль корректности вводимых данных. Эти процедуры реагируют, если вы не заполняете все текстовые поля при вводе, а при вводе индекса используете не цифровые клавиши, или количество символов не равно шести.

**9. Форматирование таблицы**

*Создайте процедуру* ***FormatTable*** *для форматирования созданной вами таблицы.*

1. *Вставьте процедуру в ваш проект созданный в данной работе. Процедуру вставьте в модуль , в котором сохранен макрос* ***Заголовок****.*

*2. Измените коды в процедуре* ***FormatTable:***

*- цвет шрифта в таблице синий фон таблицы желтый*

*- фон первой строки зеленый, цвет шрифта коричневый, выравнивание по центру*

*-Обрамление таблицы двойная линия*

*3. Один из вариантов использования процедуры* ***FormatTable*** вызов ее из процедуры обработки события кнопки ***Готово***

Далее представлена заготовка текста программы, включающая основные структурные элементы. Приведены сведения о параметрах и операторах, которые можно использовать при форматировании

Тексты, помеченные *подчеркнутым курсивом* после копирования и использования в работе следует удалить или заменить операторами.

**Public Sub FormatTable()**

**' Форматирует созданную таблицу**

**Dim r As Range, r1 As Range**

**' Выделение области, содержащей всю таблицу**

**Set r = Worksheets("Адреса").Range("A2").CurrentRegion**

**' Добавление тонких границ ячеек во всем диапазоне**

**r.Borders.LineStyle = xlContinuous**

**r.Borders(xlInsideHorizontal).Weight = xlThin**

**' Выбор желтого фона, цвет шрифта красный**

**r.Interior.Color = vbYellow ‘1)**

**r.Font.Color = vbRed ‘2)**

**' Выделение текста жирным шрифтом выравнивание справа**

**r1.Font.Bold = True**

**r1.HorizontalAlignment = xlRight**

**' Выделение первой строки диапазона**

**Set r1 = r.Rows(1)**

**' Добавление толстой нижней границы строки.**

**r1.Borders(xlEdgeBottom).LineStyle = xlContinuous**

**r1.Borders(xlEdgeBottom).Weight = xlThick**

**' Выбор белого фона, цвет шрифта черный**

***Используйте операторы замените цвет фона***

**' Выделение текста жирным шрифтом выравнивание по центру**

**' Выделение всего диапазона и обрамление двойной линией по внешним границам**

**Set r = Selection.CurrentRegion**

**r.Borders(xlEdgeTop).LineStyle = xlDouble**

***Добавьте еще три таких же оператора, в которых отличается константы соответствующие верхней, левой и правой границам (см. таблица 9.3)***

**End Sub**

*Справочная информация*

Термин ***форматирование***относится к внешнему виду рабочего листа. Поскольку форматирование рабочего листа никак не влияет на особенности его функционирования, некоторые считают саму возможность форматирования несущественным дополнением, которому не стоит уделять особого внимания.

Это большая ошибка! Важным аспектом любого рабочего листа является его простота и дружелюбность по отношению к пользователю. Благодаря удачному применению форматирования рабочий лист не только становится более удобным в использовании, но при этом уменьшается также вероятность возникновения ошибок. В настоящем уроке рассматриваются наиболее важные аспекты форматирования рабочих листов.

**Форматирование шрифтов**

В VBA для форматирования шрифтов используется объект **Font**, имеющий ряд свойств, соответствующих различным параметрам шрифтов, основные из них описаны в таблице

Таблица 9.1.

|  |  |
| --- | --- |
| *Свойство* | *Описание* |
| *Name* | Название шрифта |
| *Color* | Цвет шрифта , указываемый в значениях RGB |
| *Size* | Размер шрифта в пунктах (1 пункт равен 1/72 дюйма |
| *Bold* | Полужирное начертание |
| *Italic* | Курсив |

Свойство *Color* использует значение RGB, которое определяет требуемый цвет путем фиксирования его красной (Red), зеленой (Green) и синей (Blue) составляющих.

Чтобы присвоить свойству *Color* значение используется функция RGB(r,g,b). Значение каждого аргумента может меняться от 0 до 255 и передать более 16 миллионов оттенков от RGB(0,0,0) – черный цвет

до RGB(255,255, 255) – белый цвет

В зависимости от значений аргументов, каждый из которых может меняться от 0 до 255, можно воспроизвести более 16 миллионов оттенков.

Очевидные цвета это:

RGB(255,0, 0) – красный

RGB(0,255, 0) – зеленый

RGB(0,0, 255) – синий

Например, показанная ниже строка кода делает шрифт выделенных ячеек желто-зеленым

**Selection.Font.Color = RGB(120,255, 0)**

В языке VBA предусмотрен ряд стандартных констант для представления некоторых цветов. Это vbBlack, vbRed, vbGreen, vbYellow, vbBlue, vbMagenta, vbCyan и vbWhite.

Эти константы можно использовать вместо функции RGB для изменения свойств Color.

Например **- Selection.Font.Color =** **vbYellow.**

**Выравнивание содержимого ячеек**

Чтобы изменить способ выравнивания, используйте следующие свойства объекта Range.

• HorizontalAlignment. Принимает значения xlLeft (горизонтальное выравнивание по левому краю), xlCenter (по центру) или xlRight (по правому краю).

• VerticalAlignment. Принимает значения xlTop (вертикальное выравнивание по верхнему краю), xlCenter (по центру) или xlBottom (по нижнему краю).

Например, показанная ниже процедура выравнивает содержимое выделенных ячеек

по горизонтали и вертикали

Public Sub Centr()

Dim r As Range

Set r = Selection

With Selection

.HorizontalAlignment = xlLeft

.VerticalAlignment = xlCenter

End With

End Sub

**Границы ячеек**

Excel может добавлять обрамление вокруг отдельных ячеек или их группы. Для диапазонов из двух и более ячеек такое обрамление может содержать как рамку по внешнему краю, так и границы между ячейками внутри диапазона. Каждый объект Range имеет коллекцию Borders, значения которой определяют характер используемых для данного диапазона границ. Каждая отдельная граница представлена своим объектом **Border**, ей соответствует определенная константа (табл. 9.2).

***Таблица 9.2 . Границы диапазона и соответствующие им константы***

|  |  |
| --- | --- |
| ***Граница*** | ***Константа*** |
| Нижняя граница диапазона | xlEdgeBottom |
| Верхняя граница диапазона | xlEdgeTop |
| Левая граница диапазона | xlEdgeLeft |
| Правая граница диапазона | xlEdgeRight |
| Вертикальные границы внутри диапазона | xllnsideVertical |
| Горизонтальные границы внутри диапазона | xlInsideHorizontal |

Работать с границами диапазона можно двумя способами.

- Если используется коллекция **Borders** без аргументов, изменению подвергаются все границы диапазона.

- Применяя коллекцию **Borders** с одной из констант, представленных в табл. 9.3, в качестве аргумента, вы воздействуете только на соответствующую границу.

Например, следующая строка добавляет двойную границу по всем четырем краям текущей выделенной области и внутри нее:

Selection.Borders.LineStyle = xlDouble

Напротив, эта строка кода добавляет двойную границу только по верхнему краю выделенной области:

Selection.Borders(xlEdgeTop).LineStyle = xlDouble

Параметры отображения границ, которые можно изменять в программе, приведены в табл. 9.3

***Таблица 9.3. Свойства объекта Border***

|  |  |  |
| --- | --- | --- |
| ***Свойство*** | ***Описание*** | ***Значения*** |
| Color | Цвет границы | Значение RGB (см. ранее); по умолчанию используется черный цвет |
| LineStyle | Стиль границы, например: непрерывная линия,пунктирная линия, линия из точек и т.д. | xlContinuous, xlDash, xlDashDot, xlDashDotDash, xlDot, xlDouble, xlSlantDashDot |
| Weight | Толщина границы: от очень тонкой до очень широкой | xlHairLine, xlThin, xlMedium, xlThick |

Чтобы удалить границу, присвойте ее свойству LineStyle значение xlLineStyleNone.

**Установка фона ячеек**

Фон ячейки или диапазона устанавливается изменением свойств объекта **Interior.** Свойству этого объекта **Color** присваивается значение RGB(рассмотрено ранее).

После ознакомления с инструментами форматирования, применим их для улучшения внешнего вида представляемых на рабочем листе данных