# Основы программирования.

## Глава 4. Модели приложений

##### Что есть общего в приложениях?

Приложения Microsoft Office имеют некоторые общие черты. Очевидно в первую очередь это интерфейс. Интерфейсом мы называем то, что видим в окне того или иного приложения. Интерфейсы приложений Microsoft Office похожи: это меню и панели инструментов, полосы прокрутки, причём многие иконки на панелях также одинаковы. И в Word, и в Excel есть одноимённые панели.

Так может быть один и тот же макрос, работающий с одинаковыми объектами, может быть создан для одного приложения и будет работать в другом? Попробуем изучить этот вопрос и начнём с наших макросов, работающих с графическими элементами. Действительно стоит попробовать, ведь в Excel тоже есть панель рисования!

Запускаем Excel. Редактор VBA в Excel открывается также как в Word: Строка меню 🡪 Сервис 🡪 Макрос 🡪 Макросы 🡪 имя макроса: Макрос# 🡪 кнопка Создать (# - поставьте любую цифру, не использующуюся в именах макроса, если макросов в списке нет вообще, то поставьте 1). Через буфер обмена скопируйте любой макрос, например, рисующий улыбающиеся лица, в редактор VBA приложения Excel. Если вы попытаетесь запустить макрос, то появится сообщение об ошибке. Не торопитесь расстраиваться. Сообщение говорит о том, что вызываемого объекта не существует. Действительно, теперь мы работаем на листе электронной таблице, а не в документе. Замените слово ActiveDocument на слово ActiveSheet (лист), и макрос будет работать! Например:

Sub Макрос#()

 For i = 1 To 7

 ActiveSheet.Shapes.AddShape(msoShapeSmileyFace, 30, 30, \_

 100 - i \* 10, 100 - i \* 10).Select

 Next

End Sub

Теперь проделаем ещё один опыт. Многие элементы интерфейса («внешнего вида») программы устанавливаются из меню Строка меню 🡪 Сервис 🡪 Параметры. Некоторые установки и настройки, общие для обоих приложений, расположены на разных вкладках и немного по-разному. Посмотрим же на некоторых примерах, какие из установок одинаковы, а какие нет для разных приложений не только на пользовательском уровне, но на уровне организации объектов и их свойств.

В Microsoft Word c помощью макрорекордера запишите процесс отключения горизонтальной и вертикальной полосы прокрутки (Сервис 🡪 Параметры 🡪 Вид 🡪 🞎 горизонтальная … 🞎 вертикальная полосы прокрутки).

Откройте записанный макрос – вы увидите десятка два операторов. Не удивительно, ведь мы открывали диалоговое окно и, хотя изменили состояние только двух флажков, состояние остальных параметров также записано макрорекордером. Отыщите нижеприведенные четыре строки. Их оставьте, остальные аккуратно удалите:

Sub Макрос#()

 With ActiveWindow

 .DisplayHorizontalScrollBar = False

 .DisplayVerticalScrollBar = False

 End with

End Sub

СтрокаWith ActiveWindowозначает, что дальнейшие операторы будут относиться к активному окну. Поскольку таких операторов несколько, то, чтобы не писать вначале каждой строчки ActiveWindow, все операторы заключаются внутрь программной конструкции With ..... End With**.** Принцип действия этой конструкции аналогичен математическим скобкам (т.е. 2\*(a+b) = 2\*a+2\*b), то есть тот же программный код можно было бы записать как

Sub Макрос#()

 ActiveWindow.DisplayHorizontalScrollBar = False

 ActiveWindow.DisplayVerticalScrollBar = False

End Sub

После выполнения этого задания логично было бы создать и второй макрос, который будет восстанавливать удалённые элементы окна. Для этого не надо записывать новый макрос. Достаточно скопировать уже имеющийся макрос (Внимание! Не забудьте дать ему другое имя!) и изменить в скопированном макросе оба значения False (ложь, отключение) на True (истина, включено).

Скопируем оба макроса в редактор VBA Excel. Они будут работать и в этом приложении!

Таким образом, мы убедились, что приложения имеют не только одинаковый синтаксис языка VBA, но и общие объекты.

Задания для самостоятельной работы

1. В Microsoft Word запишите макрос, убирающий из интерфейса программы панели инструментов Стандартную, Форматирования и Рисования (Строка меню 🡪 Вид 🡪 Панели инструментов).
2. Копируя и изменяя записанный макрос, создайте новый макрос, который должен восстановить панели. Скопируйте оба макроса в редактор VBA Excel. Проверьте их работоспособность.
3. Проверьте в Ms Word и в Ms Excel работоспособность макроса, устанавливающего количество недавно открытых документов, которые «помнит» программа. Сервис 🡪 Параметры 🡪 Общие 🡪 🗹 помнить список из [ 9 ] файлов. Запишите макрос в Ms Word. Скопируйте макрос в редактор VBA Excel. Проверьте его работу.

##

##### Приложение и другие объекты

Все объекты приложений построены по иерархическому принципу.

Объект приложение Application является главным в любом приложении. Другие объекты являются его свойствами и лежат на следующем уровне иерархии. Это панели инструментов (CommandBars), окна (Windows), выделенный диапазон (Selection), документы (Documents) и другие объекты. Некоторые из этих объектов также имеют подчинённые объекты-свойства. У объекта из семейства CommandBars, т.е. у конкретной панели, есть свойство, так же являющееся объектом Controls (кнопки).

 Например, объект документ имеет подчинённый уровень, в котором находятся объекты: Paragraphs (абзацы), Sentences (предложения), Words (слова). Следующая строка кода демонстрирует три уровня вложенности:

Application.Documents(1).Words(4). \_ Font.Color=wdColorBlueGray

Этот оператор можно интерпретировать следующим образом: цвету шрифта четвертого слова первого открытого документа приложения присвоить серо-голубой цвет.

Полностью модель можно увидеть, вызвав клавишей F1 помощь Microsoft Word Objects. Важным средством для получения информации об объектах служит Просмотр объектов (F2).

На рисунке показаны фрагменты модели объектов, подчинённых семейству Documents.

##### Основные объекты Word

##### Свойства и методы.

Каждый объект имеет свойства. Как видим, эти свойства сами могут быть объектами, а могут быть просто свойствами, которым можно присвоить значение. Например, ActiveDocument.Name - имя активного документа, и ему можно присвоить значение. У объекта также могут быть методы, т.е. действия которые он совершает или совершаются над ним. Например:

Charts.Add

Этот метод добавление диаграммы, он не имеет параметров.

Selection.MoveDown Unit:=wdParagraph, Count:=1

Как помните, это действие состоит в продвижении на 1 абзац вниз. У метода есть параметры, в данном случае это тип единицы перемещения и количество этих единиц. Параметры в операторе могут помещаться как со своим названием, так и без него:

Selection.MoveDown wdParagraph, 1

В этом случае необходимо помнить о месте параметра и, если он отсутствует, то ставить запятую:

Selection.MoveDown wdLine, , wdExtend

Этот оператор пометит одну (по умолчанию) строчку.

Во всплывающем списке-подсказке, который появляется после точки при вводе операторов в окне кода, указующая рука обозначает свойство, а зелёный параллелепипед – метод.

##### Семейства объектов

Объектов, а также свойств и методов очень много. Этого не стоит пугаться. В работе обычно нужна лишь небольшая их часть. Мы уже научились разбираться в том, как узнать, какой объект и его свойства и методы нам нужны для решения конкретной задачи. Получив начальные сведения из записи макрорекодера, можно далее воспользоваться просмотром объектов, управляющими конструкциями, помощью и примерами из неё и преобразовать макрос.

Важно знать, что многие объекты объединяются в семейства (Collection). Семейство – это упорядоченный набор компонентов, на который можно ссылаться как на единое целое. Это может быть семейство объектов в активном документе или в выделенном диапазоне. Documents – это семейство открытых документов.

У каждого семейства есть свойство Count (количество). Так можно узнать, сколько документов открыто:

MsgBox Documents.Count

, или сколько фигур в активном документе

MsgBox ActiveDocument.Shapes.Count

Вспомните наши задачи с обработкой абзацев, слов или предложений. Если воспользоваться свойством Count, то можно в цикле по числу объектов в семействе обработать весь документ. В таблице 1 описаны некоторые семейства Word и их элементы

Таблица 1.

|  |  |  |  |
| --- | --- | --- | --- |
| Семейство |  | элемент | Обращение |
| Documents | Все открытые документы | Document | По имени или номеру, активный документ |
| Words | Все слова документа или выделенного диапазона | Word | По номеру (индексу) |
| Sentences | Все предложения документа или выделенного диапазона | Sentence | По номеру (индексу) |
| Characters | Символы | Character | По номеру (индексу) |
| Shapes | Все фигуры документа или выделенного диапазона | Shape | По номеру (индексу) |

К отдельным объектам можно обратиться через семейство по индексу, т.е. по номеру компонента в семействе. Например, следующий макрос «перебирает» все слова в выделенном диапазоне (Selection) и присваивает им свойства: цвет шрифта – синий, размер – 12, жирность – истина:

Sub Макрос()

For i = 1 To Selection.Words.Count

‘Цикл по i от 1 до количества слов в выделении

With Selection.Words(i)

‘Изменяем свойства каждого i-ого слова в выделенном диапазоне

 .Font.Color = wdColorBlue

 .Font.Size = 12

 .Font.Bold = True

End With

Next

End Sub

К документам можно обращаться не только по индексу, но и по имени. Следующий оператор сообщит количество слов в документе «Пример»:

MsgBox Application.Documents ("Пример").Words.Сount

Аналогичным образом можно работать со строками или ячейками таблицы, с фигурами и т.д.

##### Использование активного документа

Ещё один способ обращения к отдельному документу, который мы видели в записи макрорекодера – это обращение к активному документу. Но тогда мы заранее находились в нужном документе. Если на протяжении работы макроса нужно переходить от одного документа к другому, то надо активизировать документ с помощью метода Activate. Например, то же сообщение о количестве слов в документе «Привет» можно получить так:

Documents("Пример ").Activate

 MsgBox ActiveDocument.Words.Count

Обращаясь таким образом к документам, можно просмотреть и изменить объекты, подчинённые документу. Например, следующий макрос сделает все фигуры активного документа одинаковой высоты:

Sub Figure()

 For i = 1 To ActiveDocument.Shapes.Count

‘Цикл по i от 1 до количества фигур в активном документе

 With ActiveDocument.Shapes(i)

‘выводим имя и изменяем высоту каждой i-ой фигуры

 MsgBox .Name

 .Height = 150

 End With

 Next

End Sub

Позже, рассматривая объекты и коллекции Excel, покажем специальный вид цикла для обработки всех элементов семейства.