Школа современного программирования

Паньгина Нина Николаевна

ИЗУЧЕНИЕ VBA В ШКОЛЕ

Покажем использование языка программирования Visual Basic for Applications (VBA), встроенного в среду программ Microsoft Office. Он позволяет провести нестандартную для данного пакета обработку данных, автоматизировать рутинную работу, дополнить интерфейсное содержание и форму Ваших специальных приложений.

Предполагается знание (на простейшем уровне) языка Visual Basic (VB), использующего технологию объектно-ориентированного программирования [1].

Понятие объекта объединяет в себе как некоторые данные и их свойства, так и определенные действия (методы) над этими данными. Объект может также:

– состоять из коллекции однотипных объектов, различаемых по порядковому номеру или по имени, например, Documents(1) в MS Word или Worksheets («Лист2») в MS Excel;

– включаться иерархически в другой объект, например, Excel-приложение (Application) содержит открытые книги (Workbooks), в которых заданы наборы листов (Worksheets) с массивом ячеек (Range);

– распознавать специфичные события (типа щелчка мыши), для которых имеется стандартный системный или запрограммированный пользователем отклик.

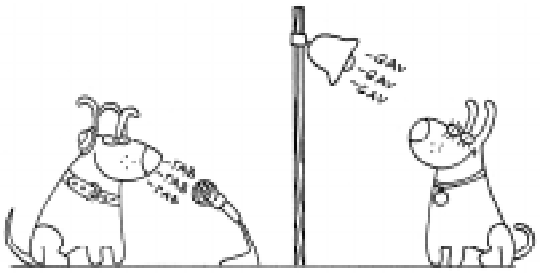
Полная однозначная ссылка на нужный объект состоит из перечисления всех имен объектов, стоящих иерархически выше (по аналогии с файловой системой), при этом разделителем служит точка. Ссылки на объекты, которые активны в данный момент, как правило, можно опускать. Синтаксис использования методов и свойств объекта такой же, как в VB. Не углубляясь в описание элементов VBA (для этого существует хорошая справочная система и интеллектуальные подсказки редактора кода), приведем два урока для демонстрации использования средств VBA в офисных программах Word и Excel.

VBA – ЭТО МИФ?

Самая, пожалуй, распространенная ошибка, когда при работе в MS Word мы забываем «переключиться на нужный язык» и, не глядя на экран монитора, успеваем набрать целую фразу абракадабры. Огорчение по поводу потерянного времени и напрасных усилий велико – ведь нужно все стирать и набирать заново. По этой же причине вопрос, заданный перед этим абзацем, имеет различное толкование.

Для автоматического переключения раскладки клавиатуры можно использовать программу **Punto Switcher** (http://www.punto.ru/switcher), однако, установив ее, обнаружили, что повсеместно это «самоуправство» нежелательно.

Закрепим свои знания программирования на VB и создадим на языке VBA макрос «Перевод с Латиницы в Кириллицу и



обратно». Для начала запустим в среде Windows приложение MS Word, выберем последовательно пункты меню «Сервис» → «Макрос» → «Макросы...». В появившемся диалоговом окне зададим имя макроса «Латиница_Кириллица» и нажмем кнопку «Создать». В окне редактора кода Visual Basic наберем нижеследующие операторы внутри тела нашей процедуры, смысл действий которых следующий:

– составляется строка Lat$ из набора латинских символов клавиатуры и строка Cyr$ соответствующих символов кириллицы;

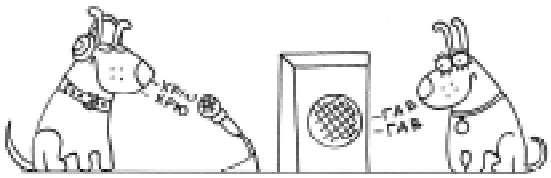
– в строковой переменной sel$ хранится выделенный в Word-документе текст (объект Selection), для которого требуется конвертация;

– далее посимвольно определяется принадлежность одному из наборов и при успешном поиске выполняется соответствующее преобразование;

– результирующая строка **Change$** сцепляется с преобразованным символом;

– по окончании анализа всех символов выделенная строка в тексте заменяется новой результирующей строкой (см. листинг 1).

Назначим данному макросу специальную кнопку на панели инструментов, которая может превращать выделенные «иероглифы» ошибочного набора в осмысленный вид. Для этого необходимо выполнить следующие действия:



Листинг 1

```
Sub Латиница_Кириллица()
 Lat$ = "qwertyuiop[]asdfghjkl;'zxcvbnm,./QWERTYUIOP[]ASDFGHJKL:" + _
       Chr$(34) + "ZXCVBNM<>"
 Cyr$ = "йцукенгшщзхъфывапролджэячсмитьбю.ЙЦУКЕНГШЩЗХЪФЫВАПРОЛДЖ" + _
                "ЭЯЧСМИТЬБЮ"
 nCyr = Len(Cyr$)
 sel$ = Selection.Text
 Change$ = ""
 For i = 1 To Len(sel$)
    z = Asc(Mid$(sel$, i, 1))
    Flag = 0
    For j = 1 To nCyr
       c1 = Asc(Mid$(Lat$, j, 1))
       c2 = Asc(Mid$(Cyr$, j, 1))
            If c1 = z Then
                z = c2: Exit For
            End If
            If c2 = z Then
                z = c1: Exit For
            End If
     Next j
     Change$ = Change$ + Chr$(z)
 Next i
 Selection.Text = Change$
End Sub
```

– выбрать пункт «Настройка» в меню «Сервис»;

– открыть вкладку «Команды», щелкнуть в категории «Макросы» на команде с названием нового макроса «Латиница_Кириллица»;

– удерживая клавишу мыши, перетащить кнопку на панель инструментов (положение ее указывает знак «I»);

– нажать правой клавишей мыши на новой кнопке – в открывшемся контекстном меню можно изменить текст на поверхности кнопки (он является одновременно контекстной подсказкой при наведении мыши на кнопку);

– выбрать основной стиль изображения кнопки – можно назначить пиктограмму для кнопки из предлагаемого набора, однако лучше выбрать опцию «изменить значок»;

– в появившемся редакторе кнопок (рисунок 1) очистить стандартный рисунок и создать свой (можно выбирать цвет и рисовать, щелкая на квадратиках сетки; стирать точку можно, щелкнув на ней снова, используя тот же цвет или выбрав фоновый цвет в окне «Удалить»; с помощью стрелок секции «Перемещение» можно сдвигать все изображение; образец рисунка на кнопке отображается в секции «Просмотр»);

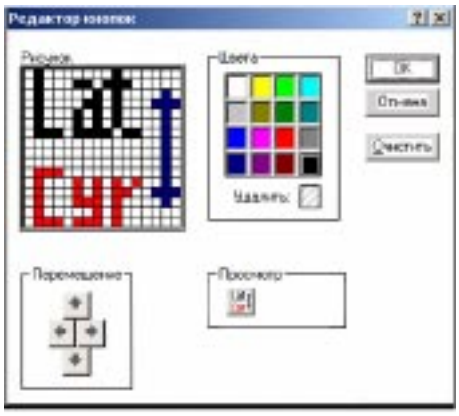


Рисунок 1.

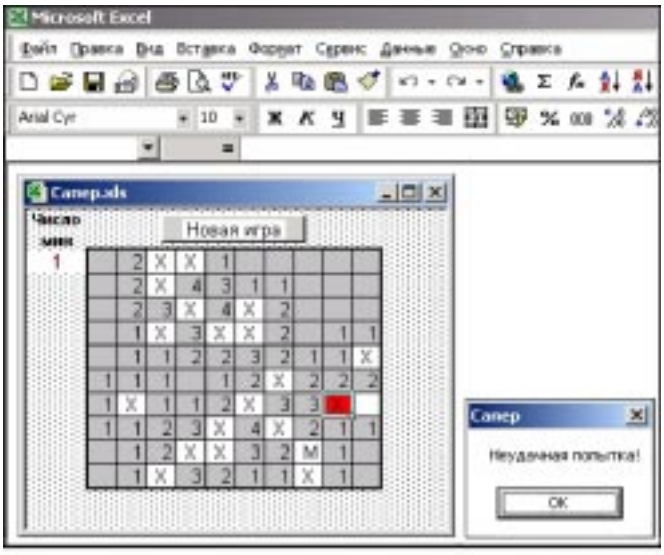
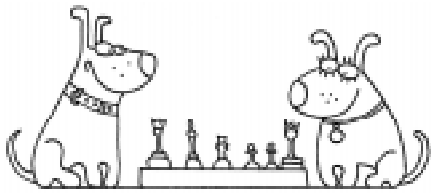


Рисунок 2.

– по окончании оформления щелкнуть кнопку «OK» и закрыть диалоговое окно настройки.

Проверим правильность работы созданной кнопки, при необходимости исправим ошибки в тексте макроса. Не исключено, что возникнет желание усовершенствовать предлагаемый макрос.

ЛОГИЧЕСКАЯ ИГРА С EXCEL



Функционально Excel многогранен: от использования в бухгалтерском деле, обработки информации в научных целях до употребления его в качестве «навороченного» калькулятора. Попробуем освоить расширенные возможности Excel на основе встроенного языка Visual Basic for Applications (VBA). Для этого выберем что-нибудь простое, знакомое и не очень скучное. Создадим в Excel аналог стандартной игры Windows «Сапер» (это экзотично, но мы преследуем свои интересы). Напомним правила игры.

Игровое пространство включает кнопку инициализации игры, счетчик мин и минное поле (см. рисунок 2).

Цель игры состоит в том, чтобы обезвредить все мины, следуя ***правилам***:

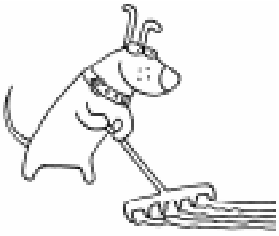
1) Игрок имеет возможность вскрыть любую ячейку, щелкнув на ней ПРАВОЙ кнопкой мыши. Если ячейка содержит мину, он проигрывает.

2) Если мины нет, в ячейке появится цифра, которая указывает, сколько мин находится в восьми смежных с ней ячейках.

3) Чтобы пометить флажком «M» ячейку, в которой, по мнению игрока, находится мина, необходимо щелкнуть на ней ДВАЖДЫ ЛЕВОЙ кнопкой мыши. Для снятия отметки операцию повторить.

Листинг 2.

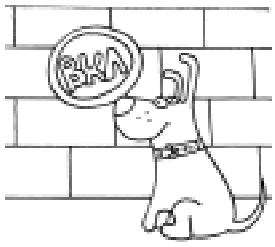
```
Public Mine As Range

Sub NewGame()
  Range("B2:K11").ClearContents
  Range("B2:K11").Interior.ColorIndex = xlNone
  MineSet MineN:= 9, ColN:= 10, RowN:= 10
  Range("A2") = Mine.Count
End Sub
```



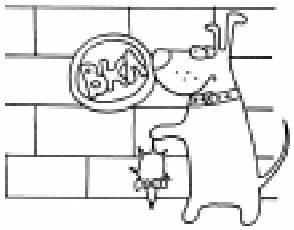
ПОДГОТОВИМ ИГРОВОЕ ПОЛЕ

Определим границы и выровняем (примерно до квадрата) размеры ячеек B2:K11 минного поля величиной 10×10. Такое расположение поля удобно для алгоритма анализа краевых ячеек игрового поля. Увеличим высоту первой строки примерно в два раза. В ячейке A1 введем текст «Число мин» в качестве метки для ячейки A2, в которой будем указывать разницу исходного числа мин и числа отмеченных игроком ячеек (флажком «M»). Выигрыш в игре означает, что открыты все ячейки, кроме помеченных флажком «M», и число не обезвреженных мин в ячейке A2 равно нулю. Отформатируем остальные ячейки листа желаемым образом, уберем в параметрах окна (на вкладке «Вид» пункта меню «Параметры») лишние галочки.

Подключим из Excel-меню «Вид» панель инструментов «Формы». Выберем на ней щелчком мыши объект Кнопку и «нарисуем» ее на рабочем листе. В возникшем диалоговом окне назначим имя макроса NewGame для объекта и «создадим» тело макроса в открывшемся окне встроенного редактора Visual Basic (это можно сделать и с помощью меню Сервис → Макрос → Редактор Visual Basic → Insert → Module) (см. листинг 2).



Макрос NewGame вызывается по щелчку мыши на созданной кнопке (изменим ее заголовок на «Новая игра»). Происходит очистка значений (метод **ClearContents**) и цвета фона (**ColorIndex = xlColorIndexNone**) всех ячеек минного поля (объект Range), затем новая случайная расстановка девяти мин – вызов процедуры MineSet. В ней создается объект Mine, являющийся регионом «минных» ячеек. Объектная переменная описывается как глобальная для использования ее в разных процедурах. В ячейку A2 заносится фактическое (с учетом наложения) количество мин (свойство Count объекта Mine). Для простоты изложения выбраны фиксированные значения числа мин и размеров минного поля, но возможно предусмотреть их задание по желанию игрока с помощью процедуры InputBox.

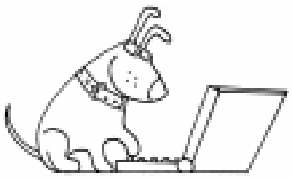


Напишем процедуру MineSet (с параметрами MineN – количество мин, RowN – число строчек, ColN – число колонок игрового поля), которая «по Rnd расставляет мины», то есть определяет VBA-объект типа Range с именем Mine из случайно выбранных ячеек игрового поля. Присвоение

Листинг 3.

```
Sub MineSet(MineN, RowN, ColN)
    For k = 1 To MineN
      i = 2 + Int(Rnd * RowN )     'номер строки
      j = 2 + Int(Rnd * ColN )       'номер столбца
      If k = 1 Then Set Mine = Range(Chr$(j + 64) & i)
      Set Mine = Union(Mine, Cells(i, j))
    Next
     'Mine.Select
End Sub
```

объектной переменной должно осуществляться с помощью инструкции Set. Для создания несвязного диапазона «минных» ячеек служит VBA-процедура Union, которая добавляет в объект Mine очередную ячейку. Обратите внимание, как для задания отдельной ячейки – частного случая диапазона ячеек – можно использовать синтаксис вида Range("A1") или вида Cells(i,j). Оператор Select показывает расположение мин (в модуле он закомментирован) (см. листинг3).



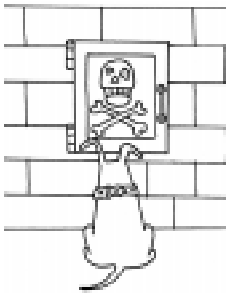
ПРОГРАММИРУЕМ ПРАВИЛА ИГРЫ

Воспользуемся событийной процедурой VBA для обработки щелчка правой кнопки мыши на любой ячейке рабочего листа Excel. Для этого в редакторе VBA выполним двойной щелчок в окне проекта на выбранном для игры листе (Лист1). В открывшемся окне редактирования кода выберем из списка (вверху слева) управляющий элемент Worksheet. Появится заготовка обработчика Worksheet_SelectionChange (изменение выбранной активной ячейки), которую далее следует удалить за ненадобностью. В правом же списке выберем обработчик BeforeRightClick. Данная процедура имеет параметрами Target для указания целевой ячейки и булевскую переменную Cancel для отмены (Cancel=True) или срабатывания (по умолчанию при Cancel=False) стандартного события Excel показа контекстного меню (справку об объекте, его свойствах и методах можно получить, вызвав из контекстного меню окно Object Browser).

Запишем в теле процедуры обработку ***правила 1***. Она заключается в следующем:

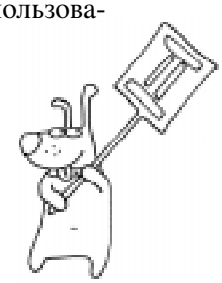
1. Проверяется принадлежность целевой ячейки диапазону ячеек игрового поля. То есть, если их пересечение (Intersect) не пусто, то проверяется далее, входит ли целевая ячейка в набор ячеек с именем Mine.

2. Если ячейка не «минная», то для удобства дальнейшего анализа по правилу 2 соответствующие свойства (Property) целевой ячейки Target – ее номер строки (Row) и колонки (Column) – передаются в процедуру Wave (ее запишем далее в том же окне кода).



3. Иначе показываем все мины (в цикле присваиваем значения Value = " X" каждой ячейке из набора Mine) и отображаем диалоговое окно сообщения о неудачной игре (см. листинг 4).

Помимо событийных процедур, в VBA возможно использование пользовательских процедур.



Рекурсивная процедура Wave(i, j) реализует ***правило 2*** игры.

1. Проверяется, что анализируемая ячейка Cells(i, j) не «закрашена»

Листинг 4.

```
Private Sub Worksheet_BeforeRightClick(ByVal Target As Range, Cancel As Boolean)
   If Not Intersect(Target, Range("B2:K11")) Is Nothing Then
       If Intersect(Target, Mine) Is Nothing Then
        Call Wave(Target.Row, Target.Column)
       Else
         Target.Interior.ColorIndex = 3
         For Each c In mine
           c.Value = " X"
         Next c
         MsgBox "Неудачная попытка!", , "Сапер"
       End If
       Cancel = True
   End If
 End Sub
```

(или не пустая) и лежит в поле (важно для рекурсивного просмотра соседних с граничными ячейками поля). При этих условиях в ячейке закрашивается фон (свойство ячейки Interior) серым цветом (свойство фона ColorIndex = 15) и заполняется стандартно «пробелом».

2. Определяется пересечение (объект типа Range, именованный isect) для текущей ячейки и восьми соседних с ней с набором «минных» ячеек. Ключевое слово Set обязательно для идентификатора объекта в правой части присваивания.

3. Для пустого пересечения (isect Is Nothing) рекурсивная волна анализа переходит на соседние ячейки, в противном случае в текущей ячейке указывается число мин на границе (заметим, что свойство Count объекта isect применимо лишь для непустого объекта) (см. листинг 5).

Для реализации ***правила 3*** создадим событийную процедуру VBA обработки двойного щелчка левой кнопки мыши на любой ячейке

Листинг 5.

```
Sub Wave(i, j)
 If Intersect(Cells(i, j), Range("B2:K11")) Is Nothing _
 Or Not IsEmpty(Cells(i, j)) Then Exit Sub
 Cells(i, j).Interior.ColorIndex = 15
 Cells(i, j) = " "
  Set isect = Intersect(Range(Cells(i - 1, j - 1), Cells(i + 1, j + 1)), Mine)
  If isect Is Nothing Then
    Call Wave(i - 1, j - 1)
    Call Wave(i - 1, j)
    Call Wave(i - 1, j + 1)
    Call Wave(i, j - 1)
    Call Wave(i, j + 1)
    Call Wave(i + 1, j - 1)
    Call Wave(i + 1, j)
    Call Wave(i + 1, j + 1)
  Else
   Cells(i, j) = isect.Count
 End If
End Sub
```

Листинг 6.

```
Sub Worksheet_BeforeDoubleClick(ByVal Target As Range, Cancel As Boolean)
 Select Case Target
  Case " M"         'убираем флажок "M"
    [A2] = [A2] + 1
    Target.ClearContents
  Case Else
    Target = " M"
    Range("A2") = Range("A2") - 1
  End Select
Cancel = True
End Sub
```

выбранного листа. Действие данной процедуры изменяет значение ячейки A2 (демонстрируются различные способы такого действия) и распространяется на все ячейки рабочего листа (ограничение на игровой регион можно настроить так же, как и в предыдущих процедурах). Анализ содержимого целевой ячейки показан с помощью оператора Select Case (см. листинг 6).

Следует заметить, что в нераскрытые ячейки игрового поля игрок может вводить символы или формулы, чтобы с их помощью построить логическую цепочку возможных вариантов расположения мин, а не держать эти варианты в «голове».

При открытии рабочей книги автоматически запускается специальный макрос с именем Auto_Open. В теле данного макроса вызовем макрос инициализации игры NewGame и сообщение об этом (см. листинг 7).

Приложение готово! Активизируйте лист Excel, для начала новой игры кликните мышкой на созданной кнопке. Чтобы протестировать нужный макрос или событийные процедуры объектов, установите курсор мыши в любое место тела макроса и нажмите клавишу F5, или пункт меню «Run», или кнопку }. Для трассировки модуля используйте клавишу F8. Создатель игры может заложить подсказку координат мин по нажатию клавиши F2, для этого достаточно раскомментировать последний оператор и добавить следующий макрос (см. листинг 8).

Используемая константа vbCrLf означает перевод позиции вывода на новую строку.

Игра может стать более совершенной, если в ней использовать счетчик времени игры (примените метод OnTime для объекта Application). Для этого можно создать пользовательскую процедуру, например, TimeSet, внутри которой происходит анализ состояния игры, подсчет времени игры, а также последующие вызовы этой же процедуры через каждую одну секунду. Функция Now определяет текущее время (см. листинг 9).

Листинг 7.

```
Sub Auto_Open()
  NewGame
  MsgBox "Игра Сапер"
   'Application.OnKey "{F2}", "promt"
End Sub
```

Листинг 8.

```
Sub promt()
MsgBox "Координаты мин:" + vbCrLf + mine.Address,, "Сапер"
End Sub
```

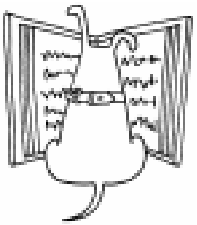
Листинг 9.

```
Application.OnTime Now + TimeValue("00:00:01"), "TimeSet"
```

РЕЗЮМЕ

Невозможно на примере двух уроков показать все многообразие средств Visual Basic for Applications. Однако данная статья дает представление о расширении функциональных возможностей и области применения офисных приложений благодаря этому встроенному языку программирования.

Много полезной информации и ответы на вопросы можно получить на русскоязычном форуме VBA на сайте www.relib.com.



Литература.

1. *Паньгина Н.Н.* Занятия по Visual Basic. Журнал «Компьютерные инструменты в образовании», №№ 1–6, 2001, и № 1, 2002.

Паньгина Нина Николаевна,
преподаватель ОИ и ВТ
школы лицея № 8, г. Сосновый Бор
Ленинградской области.