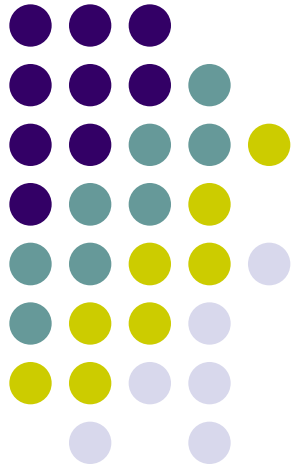
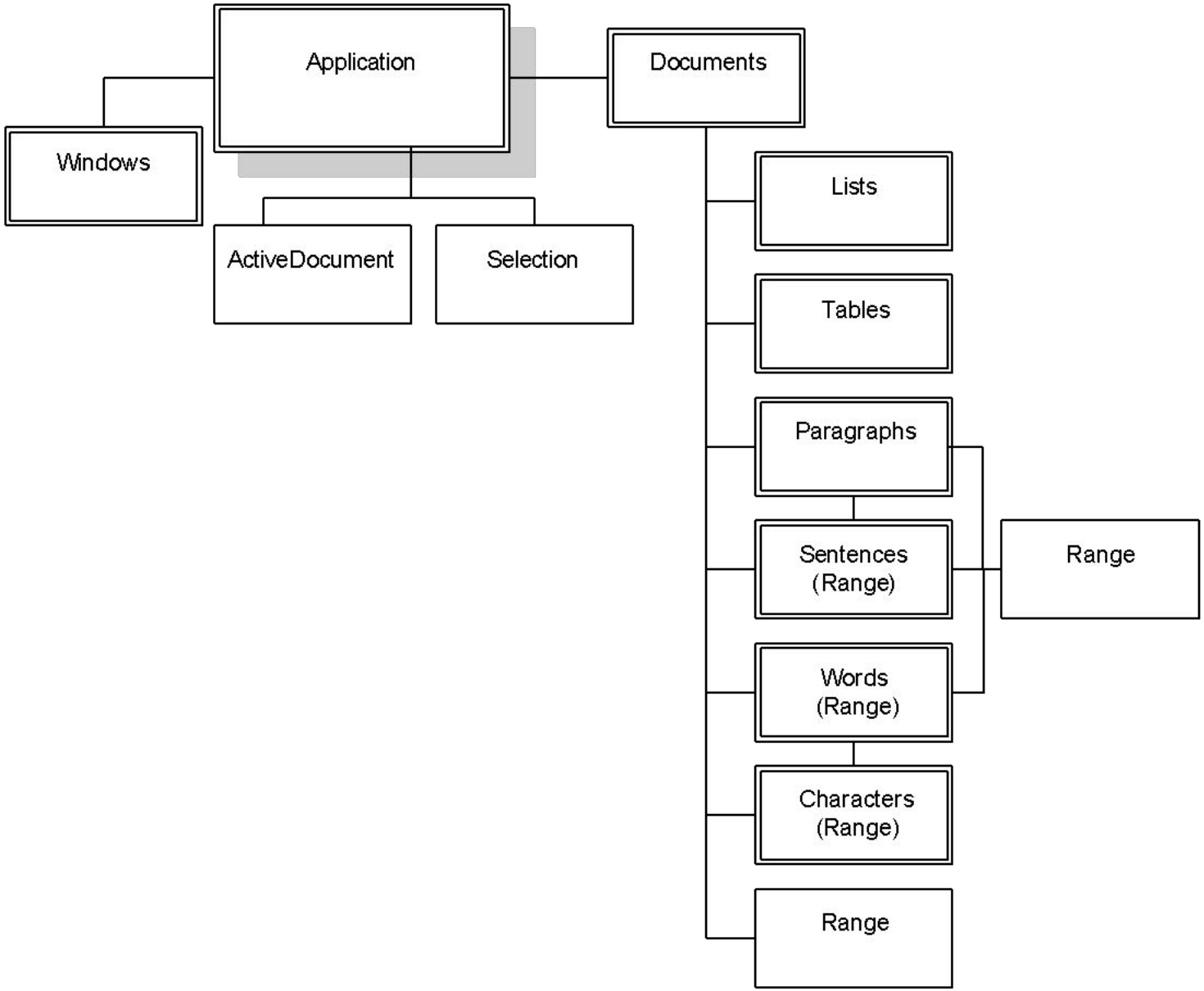
VBA в Word

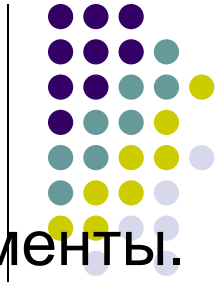


Структура объектов Word



Среди объектов, вложенных в объект Document на первом уровне иерархии, коллекции составляют явное большинство — их 36. Каждая коллекция содержит элементы одного класса. Как правило, имя класса коллекции строится как множественное число (по правилам английского языка) от имени класса элемента коллекции. Например, коллекция Documents содержит объекты класса *Document,* а коллекция Paragraphs содержит объекты класса *Paragraph.* В тех случаях, когда это правило не выполняется, в скобках указано имя класса для объектов, входящих в коллекцию. Заметьте, например, что коллекции предложений, слов и символов документа Word состоят из объектов класса *Range.*

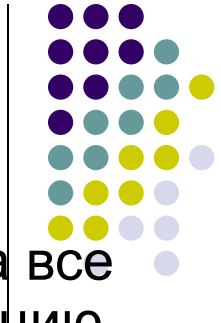
Работа с документами



Когда открывается приложение, создается коллекция документов Documents, содержащая открытые документы. Если приложение Word создается в момент открытия документа Word, (объект Word.Application может быть создан и в программном проекте другого приложения, например Excel), то в начальный момент коллекция содержит минимум один новый или ранее существовавший документ. Программно новый документ добавляется в коллекцию методом **Add**, а уже существующий — методом **Open** объекта Documents. Применяя метод Open, надо обязательно указывать имя, а точнее путь к открываемому файлу.

Чтобы добраться до нужного документа в коллекции, достаточно указать его индекс — имя файла, хранящего документ, — или его порядковый номер в коллекции. Для той же цели можно использовать и метод Item, но обычно он опускается. Метод **Save** позволяет сохранить документ, а метод **Close**, сохраняя документ в файле, закрывает его и

Доступ к объекту Document



В Word каждый документ является объектом типа **Document,** а все открытые документы текущей сессии Word образуют коллекцию **Documents.** Для ссылок на конкретный документ следует использовать коллекцию **Documents** объекта Application

Documents*(Index)*

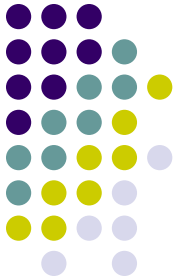
Index может быть:

- *Численным выражением, представляющим документ, который необходимо использовать. Число 1 означает первый документ, открытый в ходе текущей рабочей сессии, 2 — второй открытый документ и так далее.*
- *Строковым выражением, представляющим имя открытого документа, который нужно использовать.*

Наиболее распространенным и в большинстве случаев наиболее удобным способом использования Index является текстовая строка.

Documents(“Мой документ”).Activate

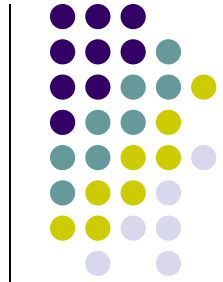
Создание нового документа



Documents.Add([*Template*] , [*NewTemplate*])

Оба аргумента метода **Documents.Add** не являются обязательными. *Template* представляет строковое выражение, задающее имя шаблона документа, на котором должен основываться новый документ. Аргумент *Template* наряду с именем файла может включать имя диска и путь к папке. Включать полный путь необходимо, если шаблон документа расположен в папке, отличной от той, в которой по умолчанию хранятся шаблоны Word. Если вы опустите аргумент *Template,* Word создаст новый документ, основываясь на шаблоне Normal.dot. Аргумент *NewTemplate* может быть любым выражением типа **Boolean.** Если он равен значению True, Word создает новый документ в виде шаблона. По умолчанию значение этого аргумента равно значению False.

Сохранение нового документа



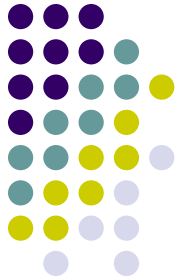
Object.SaveAs(*[FileName], [FileFormat], . . .*)

Object — любая допустимая ссылка на открытый документ типа **Document.** Необязательный аргумент *FileName* — строковое выражение, указывающее новое имя, под которым должен быть сохранен документ.

Необязательный аргумент *FileFormat* определяет формат, в котором будет сохранен документ. Аргумент *FileFormat* может быть любой из встроенных констант (определенных в классе **wdSaveFormat): wdFormatDocument, wdFormat-DOSText, wdFormatDOSTextLineBreaks, wdFormatEncodedText, wdFormat-FilteredHTML, wdFormatHTML, wdFormatRTF, wdFormatTemplate, wdFormatText, wdFormatTextLineBreaks, wdFormatUnicodeText** или **wdFormatWebArchive.**

```
ActiveDocument.SaveAs FileName:="D:\Документ", _
                      FileFormat:=wdFormatDocument
```

Открытие и закрытие документов



Documents.Open*(Filename)*

Аргумент *Filename* является текстовой строкой, представляющей полный путь к документу: логический диск, папка и имя файла.

```
Sub OpenTest()
 ' Предлагает открыть документ и затем его открывает
   Dim DocumentName As String
   DocumentName = _
              InputBox("Введите полный путь к документу")
   If DocumentName <> "" Then
       Documents.Open FileName:=DocumentName
   End If
 End  Sub
```

Открытие и закрытие документов

Завершив работу с документом, надо закрыть его для освобождения памяти и других системных ресурсов. Закрыть документ можно, используя одну из следующих форм метода **Close.**

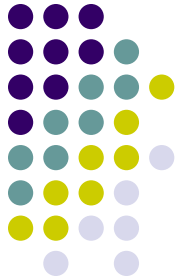
Object.Close(*[SaveChanges][,OriginalFormat] [,. . .]*)

Аргумент SaveChanges может быть равен одной из перечисленных ниже встроенных констант (определенных в классе ***wdSaveOptions): wdDoNotSaveChanges*** (документ закрывается без сохранения изменений), ***wdPromptToSaveChanges*** *(*Word запрашивает, нужно ли сохранять изменения), ***wdSaveChanges*** (изменения сохраняются). Аргумент *OriginalFormat* определяет формат сохранения для документа и может быть одной из следующих ***wdOriginalFormat***-констант***: wdOriginalDocument Format, wdPromptUser*** или ***wdWordDocument.***

Текст — это основа большинства документов. Его можно структурировать, оперируя различными единицами при решении тех или иных задач преобразования текста. Минимальная единица текста — символ. Дальше идут слова, предложения, абзацы. Более крупными частями текста являются параграфы. Количественными единицами текста являются страницы.

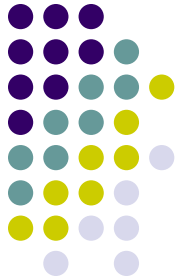
Классы *Characters,* Words, *Sentences, Paragraphs* позволяют работать с последовательностями (коллекциями) символов, слов, предложений, абзацев. Может показаться удивительным, но классов, соответствующих таким элементам, как символ, слово или предложение, нет. Элементом коллекций Characters, Words и *Sentences* является объект класса *Range.* Это один из самых важных объектов, необходимых для понимания работы с текстами. Объект Range позволяет работать как с одним символом, так и с их последовательностью. Документы, абзацы, разделы, — все они имеют метод или свойство Range, возвращающее объект Range, представляющий область, связанную с объектом, вызвавшим метод (свойство) Range. Эту область можно рассматривать как интервал, задаваемый первым и последним символом текста данной области. Поэтому работа с текстом, так или иначе, ведется через методы и свойства объекта Range.

Characters



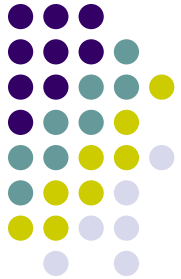
Коллекция символов. В зависимости от того, из какого объекта вы получили доступ к коллекции, она может содержать некоторые или все символы области документа. Объект для *одного символа* отсутствует. Каждый элемент коллекции **Characters** является объектом типа **Range,** включающим только один символ.

Words



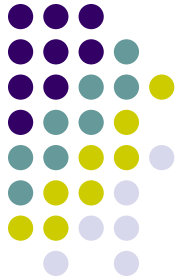
Коллекция слов. В зависимости от того, из какого объекта вы получили доступ к коллекции **Words,** она может содержать некоторые или все слова области документа. В Word отсутствует объект *слово.* Каждый элемент коллекции **Words** является объектом типа **Range,** включающим только одно слово. (Слово — это группа символов, отделенная пробелами или знаками пунктуации.)

Sentences



Коллекция предложений. В зависимости от того, из какого объекта вы получили доступ к коллекции **Sentences,** она может содержать некоторые или все предложения области документа. Подобно коллекциям **Characters и Words** объект *одно предложение отсутствует.* Каждый элемент коллекции **Sentences** является объектом типа **Range** , включающим единственное предложение.

Paragraphs



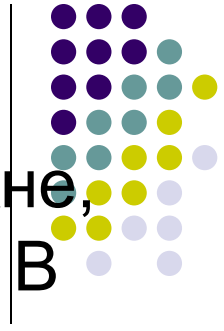
Коллекция абзацев. В зависимости от того, **Paragraphs-** коллекция какого именно объекта используется, может содержать некоторые или все абзацы области документа. Каждый элемент коллекции **Paragraphs** является объектом типа **Paragraph.** Вы можете использовать свойства и методы объекта **Paragraph** для изменения стиля, увеличения или уменьшения интервалов и других, связанных с абзацем задач. Каждый объект типа **Paragraph,** в свою очередь, содержит объект типа **Range,** который содержит текст параграфа.

Range

Представленный обзор коллекций показывает, что доступ к тексту для каждой из этих коллекций, в конечном счете, осуществляется с помощью объекта **Range.** Объект **Range** является одним из основных объектов Word и появляется в виде свойства во многих других объектах Word.

Каждый объект **Range** представляет непрерывную часть области документа, определяемую положениями начального и конечного символов. Объект **Range** может не содержать ни одного, содержать один или много символов и может представлять весь документ или любую его часть. Используя методы и свойства объекта **Range,** можно добавлять текст, форматировать его, добавлять поля или выполнять любые другие действия. Объект **Range** также предоставляет методы, позволяющие задавать или изменять диапазон символов, на который ссылается конкретный объект **Range,** и определять, в каком текстовом блоке документа находится диапазон.

Selection



Объект Selection представляет курсор вставки в окне, отображающем определенную часть документа. В каждом окне документа может находиться только один объект Selection, и только один объект **Selection** может быть активным в любой конкретный момент времени. Объект **Selection** используется для того, чтобы добавлять текст в место, на которое указывает курсор вставки, применять форматирование, выделять текст для копирования или вырезания, или любой другой задачи, которую можно выполнить интерактивно с помощью курсора вставки, вводя текст, щелкая, или перетаскивая его. Объект **Selection** может быть как непрерывной областью документа, так и сжатым до курсора вставки.

Ссылка на диапазон

Существует несколько различных способов сослаться на конкретный диапазон документа. Можно задать диапазон непосредственно или получить его при помощи одной из коллекций документа: **Paragraphs, Sentences** или **Words.**

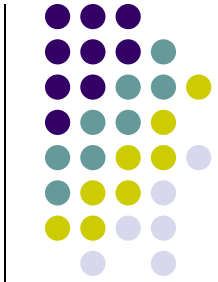
Для того чтобы задать диапазон в области основного текста документа непосредственно, воспользуйтесь методом **Range,** применив следующую синтаксическую конструкцию:

Object.Range(Start, End)

Здесь *Object* — любая допустимая ссылка на объект **Document.** *Start* и *End* — целые значения типа **Long,** задающие положение начального и конечного сим волов диапазона, соответственно. Метод **Range** возвращает объект **Range.**

Set AnyRange = ActiveDocument.Range(Start:=0, End:=20)

Ссылка на диапазон



Ссылку на диапазон можно получить и с помощью свойства **Range** любой из коллекций диапазона, упоминавшихся ранее: **Paragraphs, Sentences** или **Words.**

ActiveDocument.Sentences(3)
ActiveDocument.Paragraphs(2)
ActiveDocument.Words(10)

ActiveDocument.Paragraphs(2).Range.Words(1)
ActiveDocument. Sentences(1).Words(1)

Методы объекта Range

Изменение области, на которую ссылается диапазон **SetRange** :

Object.SetRange(Start, End)

Здесь *Object* — любая допустимая ссылка на объект **Range** или **Selection.** Оба аргумента *Start* и *End* являются обязательными, каждый из них является целым числом типа **Long,** представляющим новые позиции начального и конечного символов диапазона, соответственно.

```
Set MyRange = ActiveDocument.Words(1)
AnyRange.SetRange Start:=AnyRangeStart, _
 End:= ActiveDocument.Words(2).End
```

Методы объекта Range

Метод **MoveStart** изменяет положение начального символа диапазона, в то время как метод **MoveEnd** изменяет положение конечного символа диапазона. Для увеличения или уменьшения области, на которую ссылается диапазон, вы можете перемещать конечную точку диапазона, используя методы **MoveEnd** или **Move**Start при помощи следующих синтаксических форм:

*Object.*MoveEnd(*[Unit], [Count]*) *Object*.MoveStart([Unit], *[Count]*)

Object — любая допустимая ссылка на объект **Range.** Как для метода **MoveEnd,** так и для метода **MoveStart** аргументы *Unit* и *Count* являются необязательными. Аргумент *Unit* задает элемент, для которого будет изменяться начало или конец диапазона. *Unit* может быть одной из констант **wdUnits.** Если вы опустите аргумент *Unit,* методы MoveStart и MoveEnd по умолчанию будут использовать wdCharacter.

Метод MoveStart и MoveEnd

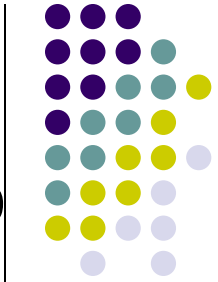
Аргумент *Count* — число элементов (units), на которое должна быть сдвинута конечная точка. Если вы используете для аргумента *Count* положительное число, конечная точка перемещается в документе вперед, если вы используете для аргумента *Count* отрицательное число, конечная точка перемещается назад.

Пример:

(AnyRange является объектной переменной, содержащей ссылку на объект **Range):**

```
AnyRange.MoveStart  Unit:=wdParagraph,   Count:=  -1
AnyRange.MoveEnd Unit:=wdWord,   Count:=-2
AnyRange.MoveEnd Count:=6
AnyRange.MoveStart  Count:=8
```

Методы объекта Range

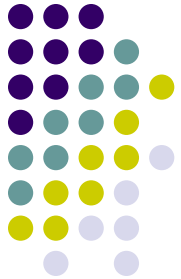


Чтобы расширить диапазон, от его текущего значения, до размеров, включающих следующий, больший по величине компонент документа, используется метод **Expand**

*Object.*Expand*([Unit]*)

Здесь *Object* — представляет любую допустимую ссылку на объект Range или Selection. Необязательный аргумент *Unit* определяет элемент, до которого расширяется диапазон, и может быть равен одной из констант WdUnits. Если вы опустите аргумент *Unit,* метод Expand использует по умолчанию значение константы wdWord.

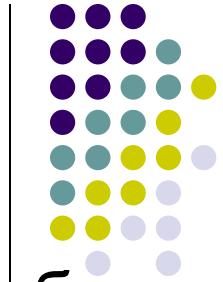
Методы объекта Range



Фрагмент кода, выделяющий десятое предложение документа и затем расширяющий диапазон для включения в него всего абзаца, в котором находится десятое предложение (MyRange объектная переменная):

```
Set MyRange = ActiveDocument.Sentences(10)
MyRange.Expand Unit:=wdParagraph
```

Объект Selection



Используя объект **Selection** можно создать VBA-программу, выполняющую с курсором вставки любые операции, которые можно выполнить в Word интерактивно: добавить текст, переместить кур сор вставки, выделить текст и другие.

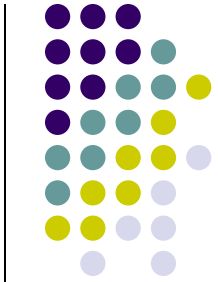
Выполнить ссылку на объект **Selection** очень просто: необходимо только использовать свойство **Selection** объектов **Application.**

В любой момент может быть активным только один объект Selection и в любом окне, отображающем документ, может находиться только один объект Selection.

Можно связать объект **Selection** с любым диапазоном, используя метод Select объекта **Range:**

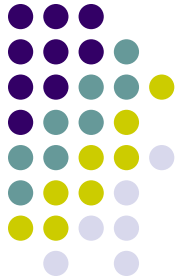
MyRange.Select

Примеры макросов



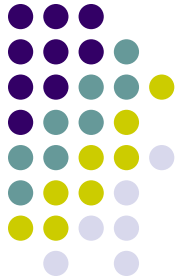
```
Sub КоличествоКомпонентов()
MsgBox "В документе " & ActiveDocument.Paragraphs.Count & _
" абзацев " & Chr(13) & ActiveDocument.Sentences.Count & _
" предложений " & Chr(13) & ActiveDocument.Characters.Count _
& " символов "
End Sub
```

Вставка разрыва страницы после третьего абзаца



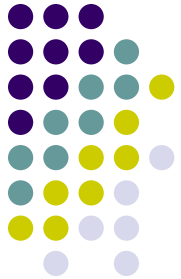
```
Sub разрыв()
Selection.Move Unit:=wdParagraph, Count:=3
Selection.InsertBreak Type:=wdPageBreak
End Sub
```

Заменить символа ↵ на конец абзаца



```
Sub замена()
n = ThisDocument.Range.Characters.Count
For i = 1 To n
c = ThisDocument.Range.Characters(i)
If Asc(c) = 11 Then
ThisDocument.Range.Characters(i) = Chr(13)
End If
Next
End Sub
```

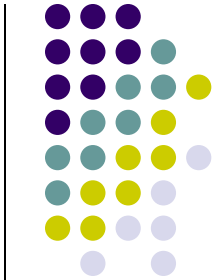
Удаление пустых абзацев



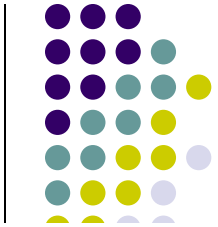
```
Sub abzacs()
Dim k As Integer, i As Long
k = ActiveDocument.Paragraphs.Count
i = 1
Do While i <= k
If ActiveDocument.Paragraphs(i).Range.Characters.Count = 1 _
    Then
ActiveDocument.Paragraphs(i).Range.Characters(1).Delete
k = ActiveDocument.Paragraphs.Count
Else
i = i + 1
End If
Loop
End Sub
```

Макрос FromEToR, переводящего «английский ошибочный» текст в правильный русский

```
Public Sub FromEToR()
'Translation of Symbols: England -> Russian
Const ALU = "ФИСВУАПРШОЛДЬТЩЗЙКЫЕГМЦЧНЯ"
Const AL = "фисвуапршолдьтщзйкыегмцчня"
Dim Sym As String, Sym1 As Range
Dim Index As Byte
Dim Result As String
Dim Pravka As Boolean
Dim Pravkal As Boolean
Pravka = False
Pravkal = False
Result = ""
For Each Sym1 In Selection.Characters
Sym = Sym1
'Исправление ошибочной автокорректировки
If Pravka And (Sym <> " ") Then Sym = LCase(Sym): Pravka = False
Select Case Sym
Case "A" To "Z" 'английская буква верхнего регистра
Index = Asc(Sym) - Asc("A") + 1
Sym = Mid(ALU, Index, 1)
Case "a" To "z" 'английская буква нижнего регистра
Index = Asc(Sym) - Asc("a") + 1
Sym = Mid(AL, Index, 1) 'Символы, переходящие в символы
```



```
Case "?": Sym = ","
Case "/": Sym = "."
Case "^": Sym = ":"
Case "$": Sym = ";"
Case "&": Sym = "?"
Case "@": Sym = """"
Case "#": Sym = "№"
'Символы, переходящие в буквы
Case ",": Sym = "б"
Case "<": Sym = "Б"
Case ".": Sym = "ю"
Case ">": Sym = "Ю"
Case ";": Sym = "ж"
Case ":": Sym = "Ж"
Case "'": Sym = "э"
Case """": Sym = "Э"
Case "[": Sym = "х"
Case "]": Sym = "ъ"
Case "{": Sym = "Х"
Case "}": Sym = "Ъ"
Case "`": Sym = "ё"
Case "~": Sym = "Ё"
'другие виды кавычек
Case Chr(145): Sym = "э"
Case Chr(146): Sym = "э"
Case Chr(147): Sym = "Э"
Case Chr(148): Sym = "Э"
Case Chr(171): Sym = "Э"
Case Chr(187): Sym = "Э"
Case Else: 'Кодировки совпадают
End Select
```



```
'Обнаружение ошибочной автокорректировки
If Sym = "," Then Pravka = True
If Pravkal And (Sym = " ") Then
Pravka = True
Else
Pravkal = False
End If
If Sym = "ю" Then Pravkal = True
'формирование результата
Result = Result + Sym
Next
Selection.LanguageID = wdRussian
Selection.TypeText Result
End Sub
```