Word con Visual Basic Para Aplicaciones (VBA)

Parte 1: Objeto Application

VISUAL BASIC PARA APLICACIONES

El objeto Application de Word

Aparte de los distintos métodos y propiedades que el objeto Application tiene en común para todas las aplicaciones que emplean VBA, Word posee bastantes propiedades y métodos exclusivos de este programa.

Propiedades del objeto Application

ActivePrinter: devuelve el valor o determina el nombre de la impresora predeterminada (debe ser una impresora ya instalada en Windows). La siguiente sentencia instala la impresora predeterminada:

```
ActivePrinter = "HP LaserJet 5P/5MP Postcript local on LPT1:"
```

Application.CapsLock: devuelve el valor True si está activada la tecla BloqMayús.

Application.DefaultSaveFormat: devuelve el valor o establece el formato de archivo por defecto que usa Word para grabar un documento. Como ejemplo, la siguiente sentencia establece el formato por defecto a "Word 6.0/95".

```
Application.DefaultSaveFormat = "MSWord6Exp"
```


Escuela Superior de Informatica
Universidad de Castilla-La Mancha
http://www.inf-cr.uclm.es
Word con VBA

VISUAL BASIC PARA APLICACIONES

Propiedades del objeto Application

¿Cómo saber qué cadena emplear par un formato determinado? Word tiene ocho nombres internos de convertidores de clase de archivo:

Nombre de formato de archivo	Nombre de clase de convertidor
Documento Word	<null string>
Plantilla de Word	Dot
Sólo texto	Text
Sólo texto con saltos de línea	CRText
Texto MS-DOS	8Text
Text MS-DOS con saltos de línea	8CRText
Formato RTF	Rtf
Texto codificado	unicode

Application.MouseAvailable: devuelve el valor True si el sistema tiene ratón.

Application.Template: devuelve un objeto Template que representa a la plantilla Normal de Word.

Application.NumLock: devuelve True si la tecla Bloq Num está activada.

VISUAL BASIC PARA APLICACIONES

Propiedades del objeto Application

Application.StartupPath: devuelve el valor o determina la ruta de la carpeta de inicio de Word.

Application.UserInitials: devuelve el valor o establece las iniciales del usuario. Word usa estas iniciales en las marcas de comentarios.

Métodos del objeto Application

Application.ChangeOpenDirectory: este método especifica la carpeta que aparecerá por defecto en el cuadro de diálogo Abrir la próxima vez que el usuario elija esta opción del menú archivo. La sintaxis es

ApplicationChangeOpenDirectory (*Ruta*)

Ruta es la ruta de la carpeta que aparecerá en el cuadro de diálogo Abrir.

En el siguiente ejemplo, se usa primero la función Dir par comprobar si existe la carpeta.

```
If Dir("C:\Mis Documentos\") <> "" Then
        ApplicationChangeOpenDirectory "C:\Mis Documentos\"
End If
```


VISUAL BASIC PARA APLICACIONES

Métodos del objeto Application

Application.OnTime: ejecuta un procedimiento a una hora especificada usando la siguiente sintaxis:

Application.OnTime(*When, Name, Tolerancia*)

When: La hora (y fecha si es necesario) en que quiere que se ejecute el procedimiento. Introduzca la fecha y hora con n número serial.

Name: El nombre (introducido como texto) del procedimiento a ejecutar cuando llegue la hora indicada en When.

Tolerancia: Si Word no está disponible en el momento de ejecutar el procedimiento, puede intentarlo durante el número de segundos indicados en Tolerancia. Si se omite este argumento, VBA espera hasta que Word esté listo.

La manera más fácil de introducir un número serial es es emplear la función TimeValue:

TimeValue(*Hora*)Hora: una cadena que representa la hora que quiere usar (como "5:00PM" o "17:00")

```
Application.OnTime When:=TimeValue("5:00PM"), Name:="MakeBackup"
```

VISUAL BASIC PARA APLICACIONES

Métodos del objeto Application

Si quiere que el método OnTime se ejecute tras un intervalo de tiempo especificado, utilice Now + TimeValue(*Tiempo*) par *When*, (donde *Tiempo* es el intervalo).

Por ejemplo, la siguiente sentencia programa un procedimiento para ejecutarse cada 30 segundos:

```
Application.OnTime When:=Now + TimeValue("00:30PM"),
                   Name:="MakeBackup"
```

Application.Move: mueve la ventana de la aplicación Word de acuerdo a la siguiente sintaxis:

Application.Move(*Left, Top*)

Left: La posición horizontal en la pantalla, en puntos, del borde izquierdo de la ventana de la aplicación.

Top: La posición vertical en la pantalla, en puntos, del borde superior de la ventana de la aplicación.

VISUAL BASIC PARA APLICACIONES

Métodos del objeto Application

Ejemplo: un procedimiento que comprueba si la ventana está maximizada o minimizada y, en caso negativo, mueve la ventana a la esquina superior izquierda.

```
Sub TopLeftCorner()
  With Application
    If .WindowState <> wdWindowStateMaximize and wdWindowStateMinimize Then
      .Move 0,0
  End With
End Sub
```

Application.Resize: cambia el tamaño de la ventana de la aplicación de Word. La sintaxis es como sigue:

Application.Resize(*Ancho, Alto*)

Ancho: El nuevo ancho de la ventana de la aplicación en puntos.

Alto: El nuevo alto de la ventana de la aplicación en puntos.

Igual que sucedía antes este método produce un error cuando la ventana esta maximizada o minimizada.

VISUAL BASIC PARA APLICACIONES

Métodos del objeto Application

Application.Quit: Cierra Word. Si hay documentos abiertos con cambios sin grabar, Word preguntará si guarda los cambios. Si quiere evitar esta pregunta, almacene los documentos antes de ejecutar el método Quit o establezca la propiedad DisplayAlerts en False (en este caso, Word no guardará los cambios).

Parte 2:
Objeto Document

VISUAL BASIC PARA APLICACIONES

Objeto Document

El objeto Document aparece inmediatamente debajo de Application en el árbol jerárquico de objetos. Puede usar VBA para crear nuevos documentos, guardarlos, cerrarlos, etc.

Especificar un documento Document

Si necesita hacer algo con un documento o trabajar con un objeto contenido en un documento específico (como una sección de texto), debe indicar a Word el documento que quiere usar. VBA proporciona 3 maneras de hacerlo:

- Usar el objeto Documents. Este objeto es la colección de todos los documentos abiertos. Para especificar uno determinado utilice su número de índice (donde 1 es el primero) o el nombre del archivo entre comillas.

VISUAL BASIC PARA APLICACIONES

Especificar un documento Document

- Por ejemplo, si Memo.doc fuera el primer documento abierto, las siguientes sentencias serían equivalentes:

  Documents(“Memo.doc”) Documents(1)

- Usar el objeto ActiveDocument. Este objeto representa el documento activo.
- Usar el documento ThisDocument. Representa el documento donde el código VBA se esta ejecutando. Si el código sólo maneja objetos que se encuentran en el mismo documento que el código, puede utilizar el objeto ActiveDocument. Pero si también maneja otros documentos, use ThisDocument para asegurarse que el código sólo afecte al documento donde se encuentra el procedimiento.

Abrir un documento

- Para abrir un documento, use el método Open de la colección Documents. Este método tiene una docena de argumentos que puede emplear para refinar las operaciones de apertura, pero sólo uno es obligatorio. Esta es la sintaxis simplificada:

  Documents.Open(*NombreArchivo*)

VISUAL BASIC PARA APLICACIONES

Abrir un documento

- Por ejemplo, para abrir un documento llamado Carta.doc en la unidad y carpeta actuales, use la siguiente sentencia:

Documents.Open "Carta.doc"

Crear un nuevo documento

- Si necesita crear un nuevo documento, use el método Add de la colección Documents:

Documents.Add(*Plantilla, NuevaPlantilla, TipoDocumento, Visible*)

- Estos argumentos son opcionales.
  - *Plantilla.* Especifica el archivo de plantilla a utilizar en el nuevo archivo. Debe ser una cadena que contenga la rutay el nombre del archivo .DOT. Si se omite este argumento, Word utilizará la plantilla Normal.
  - *NuevaPlantilla.* Si se da a este argumento el valor True, Word crea una nueva plantilla.
  - *TipoDocumento.* Determina el tipo de documento que se crea. Use una de las siguientes constantes:

    wdNewBlankDocument. Crea un nuevo documento de Word en blanco (opción por defecto).

Escuela Superior de Informatica
Universidad de Castilla-La Mancha
http://www.inf-cr.uclm.es
Word con VBA

VISUAL BASIC PARA APLICACIONES

Crear un nuevo documento

wdNewEmailMessage. Crea un nuevo mensaje de correo.
wdNewFrameset. Crea un nuevo marco de página Web.
wdNewWebPage. Crea una nueva página Web.

- Visible. Un valor Boolean que determina si Word muestra el nuevo documento en una ventana visible. El valor por defecto es True. Utilice False para crear un documento y no mostrarlo en la ventana visible.

El Objeto RecentFiles

- Otra manera de abrir documentos de Word es usar el objeto RecentFiles, que es la colección de archivos usados recientemente que se muestra al final del menú Archivo de Word. Cada elemento es un objeto RecentFile.
- Se puede especificar un objeto RecentFile mediante RecentFiles(*Indice*), donde *Indice* es el número entero que especifica el archivo con que se quiere trabajar. El más reciente será el 1, el siguiente el 2 y así sucesivamente. El valor máximo (modificable) lo da la propiedad RecentFiles.Maximum.

VISUAL BASIC PARA APLICACIONES

El Objeto RecentFiles

•Veamos algunas propiedades de este objeto que se deben considerar:

ArchivoReciente.Name: devuelve el nombre del archivo reciente especificado.

ArchivoReciente.Path: devuelve la ruta del archivo reciente escpecificado.

```
With RecentFiles(1)
        Documents.Open  .Path & "\" & .Name
End With
```

Propiedades del objeto Document

La mayoría de las propiedades del objeto Document devuelven colecciones de otros objetos. Por ejemplo la propiedad Words es la colección de todas las palabras del documento.

Documento.Characters: devuelve el objeto Characters, que son todos lo caracteres del documento especificado. Cada elemento es un objeto Range.

Documento.EndNotes: devuelve la colección EndNotes, que incluye todas las notas al final del documento especificado. Cada elemento de la colección es un objeto EndNote.

Escuela Superior de Informatica
Universidad de Castilla-La Mancha
http://www.inf-cr.uclm.es
Word con VBA

VISUAL BASIC PARA APLICACIONES

Propiedades del objeto Document

Documento.Envelope: devuelve un objeto Envelope, que representa el sobre del dcoumento especificado.

Documento.Footnotes: devuelve la colección Footnotes, que consiste en todas las notas al pie del dcoumento especificado. Cada elemento de la colección será un objeto Footnote.

Documento.FullName: devuelve la ruta completa del documento especificado, que incluye la ruta y el nombre del archivo.

Documento.GrammarChecked: devuelve True si el documento especificado tiene activada la revisión gramatical.

Documento.Name: devuelve el nombre del archivo del documento especificado.

Documento.Paragraphs: devuelve el objeto Paragraphs, que contiene los párrafos del documento especificado. Cada elemento es un objeto Paragraph.

Documento.Path: devuelve la ruta del documento especificado (la cadena resultante no lleva una barra invertida al final). La propiedad path de un documento nuevo no guardado devuelve una cadena vacía ("").

VISUAL BASIC PARA APLICACIONES

Propiedades del objeto Document

Documento.Sentences: devuelve el objeto Sentences, que consta de las oraciones incluidas en el documento especificado. Cada elemento es un objeto Range.

Documento.ReadOnly: retorna true si el documento especificado se abrió con atributo de sólo lectura.

Documento.Saved: devuelve si se han realizado cambios desde que se grabó por última vez.

Documento.SaveFormat: devuelve el formato de archivo especificado. Consulte en el examinador de objetos la clase wdSaveFormat para obtener una lista de constantes predefinidas.

Documento.Sections: devuelve el objeto Sections, que contiene las secciones del documento especificado. Cada elemento es un objeto Section.

Documento.SpellingChecked: devuelve True si el documento especificado tiene la revisión ortográfica activada.

Documento.Styles: recupera el objeto Styles que contiene los estilos incorporados y definidos por el usuario en el documento especificado. Cada elemento es un objeto Style.

Escuela Superior de Informatica
Universidad de Castilla-La Mancha
http://www.inf-cr.uclm.es
Word con VBA

VISUAL BASIC PARA APLICACIONES

Propiedades del objeto Document

•Por ejemplo, el siguiente fragmento de código crea un nuevo estilo de párrafo llamado “PagTitle” y establece algunas propiedades para el nuevo objeto Style.

```
Set newStyle = ActiveDocument.Styles.Add(“PagTitle”, wdStyleTypeParagrapgh)
With newStyle
  .Font.Bold = True
  .Font.Underline = True
  .Font.Size = 34
  .Font. Name = Arial
  .ParagraphFormat.Alignement = wdAlignParagraph = wdAlignParagraphCenter
  .ParagraphFormat.SpaceAfter = 12
  .NextParagraphStyle = wdStyleNormal
End With
```

VISUAL BASIC PARA APLICACIONES

Propiedades del objeto Document

Documento.Tables: retorna el objeto Tables, que contiene las tablas del documento especificado. Cada elemento es un objeto Table.

Documento.Words: devuelve el objeto Words, que contiene las palabras del documento especificado. Cada elemento es un objeto Range.

VISUAL BASIC PARA APLICACIONES

Métodos del objeto Document

•Los objetos Documents tienen docenas de métodos que le permiten hacer de todo, desde guardar un documento hasta cerrarlo. Aquí están los métodos que usará con más frecuencia:

Documento.Activate: activa el documento abierto especificado.

Documents("Tirade.dc").Activate

Documento.CheckGrammar: comprueba la gramática del documento especificado.

Documento.CheckSpelling: comprueba la ortografía del documento especificado. Este método incluye una serie de argumentos que le permiten establecer distintas opciones de comprobación.

Documento.Close: cierra el documento especificado. Este método usa la siguiente sintaxis:

Documento.Close(*GuardarCambios*, *FormatoOriginal*, *RutaDocumento*)

Documento: es el objeto Document que se desea cerrar.

GuardarCambios: Si se ha modificado el documento, el argumento determina si Word guarda estos cambios:

VISUAL BASIC PARA APLICACIONES

Métodos del objeto Document

wdSaveChanges: guarda los cambios antes de cerrar

wdDoNotsaveChanges: no guarda los cambios antes de cerrar

wdPromptToSave: pregunta si se guardan los cambios

FormatoOriginal: especifica el formato que se usará para guardar el documento:

wdOriginalFormat: almacena el documento en su formato original

wdWordDocument: guarda el documento en formato Word

wdPromptUser: pregunta al usuario si desea grabar el documento en su formato original

RutaDocumento: si tiene el valor True, este argumento dice a Word que envíe el documento al siguiente destinatario.

Documento.CopyStyesFromTemplate: copia los estilos de una plantilla a un documento especificado. Esta es la sintaxis:

Documento.CopyStylesFromTemplate(*Plantilla*)

Documento: es el objeto Document donde se copiarán los estilos.

Escuela Superior de Informatica
Universidad de Castilla-La Mancha
http://www.inf-cr.uclm.es
Word con VBA

VISUAL BASIC PARA APLICACIONES

Métodos del objeto Document

Plantilla: son la ruta y nombre de archivo de la plantilla de la que quiere copiar los estilos.

Documento.GoTo: devuelve un objeto Range que representa el comienzo de una posición específica en el documento. La sintaxis es como sigue:

Documento.GoTo(*What, Which, Count, Nombre*)

Documento: es el objeto Document con el que se quire trabajar.

What: El tipo de elemento al que desplazarse. Word emplea diecisiete constantes diferentes para este argumento. Estas son las más comunes:

wdGoToBookmark,wdGoToEndNote, wdGoToGraphic, wdGoToObjet, wdGotoSection, wdGoToComment, wdGoToFotNote, wdGoToLine, wdGoToPage, wdGoToTable

Which: una constante que indica a Word cómo desplazarse a la nueva ubicación:

wdGoToAboslute: usa la posición absoluta del elemento

wdGoToFirst: Va al primer elemento del rango

wdGoToLast: va al último elemento del rango

wdGoToNext: va al siguiente elemento del rango

VISUAL BASIC PARA APLICACIONES

Métodos del objeto Document

wdGoToPrevious: se desplaza al elemento anterior del rango

wdGoToRelative: usa la posición relativa del elemento

Count: un valor positivo que representa el número del elemento.

Nombre: el nombre del elemento, es decir, si *What* es wdGToBookMark, wdGoToComment, wdGoToField o WdGoToObject.

Por ejemplo, la siguiente sentencia se desplaza a la segunda línea del documento activo:

ActiveDocument.GoTo What:=wdGoToLine, Which:= wGoToAbsolute, Count:=2

En este otro caso, la siguiente sentencia avanza dos líneas desde la posición actual del cursor:

ActiveDocument.GoTo What:=wdGoToLine, Which:= wGoToRelative, Count:=2

Documento.PrintOut: imprime el docmento especificado. La sintaxis completa de este método tiene no menos de dieciocho argumentos. Aquí están los doce primeros:

Escuela Superior de Informatica
Universidad de Castilla-La Mancha
http://www.inf-cr.uclm.es
Word con VBA

VISUAL BASIC PARA APLICACIONES

Métodos del objeto Document

Documento.PrinOut(*SegundoPlano, Append, Rango, Archivosalida, From, To, Item, copias, Páginas, TipoPáginas, ImprimirAArchivo, Intercalar*)

Documento: es el objeto Document que se quiere imprimir.

SegundoPlano: cuando su valor es True, el procedimiento continúa mientras Word imprime en segundo plano. Si el valor es False, el procedimiento espera hasta que el documento haya ido a la cola de impresión.

Append: Si su valor es True, la salida se añade al final del archivo especificado en el argumento*ArchivoSalida*. En caso de valor False, la salida sobrescribe el contenido del archivo.

Rango: especifica el rango del texto a imprimir, como sigue:

wdPrintAllDocument: imprime el documento entero

wdPrintCurrentPage: imprime sólo la página actual

wdprintFromTo: imprime un rango de páginas desde el número inicial (véase argumento *From*) hasta el final (véase argumento *To*)

wdPrintRangeOfPages: imprime un rango de páginas(veáse argumento páginas)

VISUAL BASIC PARA APLICACIONES

Métodos del objeto Document

wdPrintPrintSelection: imprime sólo una selección.

ArchivoSalida: la ruta y el nombre del Archivo donde quiere imprimir. (Vea que el argumento *ImprimirAArchivo* debe ser true).

From: si rango es wdPrintFromTo, este argumeto especifica desde donde comenzar la impresión.

Item: una constante que representa el elemento que quiere imprimir. Use uno de los siguientes valores:

wdPrintAutoTextEntries, wdPrinComments, wdPrintDocmentContent, wdPrintKeyAssignments, wdPrintProperties o wdPrintStyles

To: Si rango es wdPrintFromTo, este argumento especifica la página donde finalizar la impresión.

Copias: el número de copias para imprimir. El valor por defecto es 1.

Páginas: si rango es wdPrintRangeOfpages, este argumento especifica el rango de páginas (por ejemplo, 4-8, 10)

TipoPáginas: especifica qué páginas se van a imprimir usando una de las siguientes constantes: wdPrintAllPages, wdPrintEvenPagesOnly, wdPrintOddPagesOnly.

Escuela Superior de Informatica
Universidad de Castilla-La Mancha
http://www.inf-cr.uclm.es
Word con VBA

VISUAL BASIC PARA APLICACIONES

Métodos del objeto Document

ImprimirAArchivo: con el valor True, Word imprime el documento a un archivo y pide al usuario el nombre para el mismo.

Intercalar: si el valor es True y Copias es mayor que 1 imprime todo el documento y después la segunda copia. Si es False y Copias es mayor que 1, imprime las distintas copias de la primera página, luego las de la segunda y así sucesivamente.

Documento.PrintPreview: muestra el documento especificado en la ventana Vista preliminar.

Documento.ReDo: rehace la última acción o acciones que se deshicieron:

Documento.ReDo(*Veces*)

Documento: es el objeto Document sobre el que quiera trabajar.

Veces: el número de acciones que quiere rehacer.

Documento.Save: guarda el documento especificado. Si el documento es nuevo use el método SaveAs en su lugar.

Documento.SaveAs: guarda el documento especificado en un archivo diferente. Esta es la sintaxis simplificada para este método (para la sintaxis completa, consulte la ayuda en línea de VBA):

VISUAL BASIC PARA APLICACIONES

Métodos del objeto Document

Documento.SaveAs(*NombreArchivo, FormatoArchivo*)

Documento: es el objeto Document que se quiere guardar en un archivo diferente.

NombreArchivo: el nombre completo del nuevo archivo de documento, incluyendo la unidad y carpeta de destino.

FormatoArchivo: el formato de archivo en que se grabará el documento.

La siguiente sentencia almacena el documento en formato Word 6.0/95.

ActiveDocument.Save “Memo.doc”, FileConverters(“MSWord6Exp”).SaveFormat

Documento.Select: selecciona todo el texto del documento especificado.

Documento.UnDo: deshace la última acción o acciones llevadas a cabo:

Documento.Undo(*Veces*)

Documento: es el objeto Document con el que se desea trabajar.

Veces: el número de acciones que se quieren deshacer.

Escuela Superior de Informatica
Universidad de Castilla-La Mancha
http://www.inf-cr.uclm.es
Word con VBA

VISUAL BASIC PARA APLICACIONES

Métodos del objeto Document

Ejemplo: un procedimiento que realiza una copia de seguridad de documento activo en un disquete.

```
Sub MakeBackup()
  Dim backupFile As String
  Dim currFile As String
with ActiveDocument
  If .Saved Or .Path =“” Then Exit Sub
  .Bookmarks.Add Name:=“LastPosition”
  .Save
  currFile = .FullName
  backupFile = “A:\” + .Name
  .SaveAs FileName:=backupFile
End With
ActiveDocument.Close
Documents.Open FileName:=currFile
Selection.GoTO what:=wdGoToBoomark,
Name:=“LastPosition”
End Sub
```

VISUAL BASIC PARA APLICACIONES

Eventos del objeto Document

Estos objetos responden a tres eventos diferentes: Close, Open y New (Cerrar, Abrir y Nuevo)

Recuerde que no puede crear manipuladores de eventos en un módulo normal VBA. En vez de eso, siga los siguientes pasos:

1) En el Examinador de Proyectos del editor de Visual Basic destaque el objeto Document con el que quiere trabajar. Para el documento que contiene el código VBA, escoja ThisDocument.

2) Seleccione Ver, Código, pulse F7 o el botón Ver código en el examinador de proyectos.

3) En la ventana de código use la lista desplegable de objetos (la de la izquierda) para seleccionar el objeto Document.

4) Use la lista desplegable de procedimientos (la de la derecha) para seleccionar el evento con que se quiere trabajar. VBA añadirá el procedimiento manipulador del evento en la ventana de código.

5) Introduzca aquí su código manipulador.

Escuela Superior de Informatica
Universidad de Castilla-La Mancha
http://www.inf-cr.uclm.es
Word con VBA

VISUAL BASIC PARA APLICACIONES

Eventos del objeto Document

Close: este evento se activa cuando el usuario elige Cerrar en el menú Archivo o cuando el código ejecuta el método Close del objeto Document. Ésta sería la estructura de este manipulador de eventos:

```
Private Sub Document-Close()
        <el código manipulador iría aquí>
End Sub
```

New: este evento se aplica sólo a las plantillas y sucede cuando el usuario crea un nuevo documento basado en la plantilla o cuando el código ejecuta el método Add del objeto Document y especifica esta planitlla. El procedimiento quedará así

```
Private Sub Document_New()
        <el código manipulador iría aquí>
End Sub
```

Open: este evento se activa cuando el usuario elige Abrir del menú Archivo o cuando el código ejecuta el mérodo Open del objeto Document. Ésta sería la estructura de este manipulador de eventos:

```
Private Sub Document_Open()
        <el código manipulador iría aquí>
End Sub
```

Parte 3:
Objeto Range y Selection

VISUAL BASIC PARA APLICACIONES

El objeto Range

- Word no tiene objetos separados para sus unidades de texto más fundamentales, el carácter y la palabra.Word considera estos elementos partes de una clase genérica, la clase Range. Este objeto se define como una sección de texto contiguo, por lo que puede ser un simple carácter o el documento completo.
- Hay dos técnicas básicas para emplear un objeto Range: el método Range del objeto Document y la propiedad Range.

El método Range

- El objeto Document tiene un método Range que le permite especificar el punto de partida y final de un rango. Ésta es su sintaxis:

Documento.Range(*comienzo, final*)

Documento: es el objeto Document con que se quiere trabajar.

comienzo: la posición del carátcer de inicio. El primer carácter en un documento está en la posición 0.

Final: la posición del carácter final.

VISUAL BASIC PARA APLICACIONES

El método Range

•Por ejemplo, las siguientes sentencias usan la variable del objeto myRange para almacenar los primeros 100 caracteres del documento activo:

```
Dim myRange As Range
myRange = ActiveDocument.Range(0,99)
```

La propiedad Range

•Muchos objetos Word tienen una propiedad Range que emplea un objeto de tipo Range, incluso objetos Paragraph y selection. Esto es importante porque tales objetos carecen de ciertas propiedades y métodos que son útiles para manipular texto. Por ejemplo, el objeto paragraph no tiene una propiedad Font, pero el objeto Range sí, por lo que puede cambiar la fuente del párrafo desde su programa refiriéndose a su propiedad range:

```
ActiveDocument.Paragraphs(1).Range.Font.Italic = True
```

•Esta sentencia da formato al primer párrafo del documento activo con letra cursiva.

VISUAL BASIC PARA APLICACIONES

Propiedades del objeto Range

•Las propiedades del objeto Range incluyen muchos de los comandos habituales de formato de texto.

Range.Bold: devuelve True si el rango especificado tiene en su totalidad formato negrita; si ninguna parte del rango está en negrita devuelve False; si parte de él está en negrita devuelve wsUndefined. Puede establecer esta propiedad con True (negrita), false (quitar negrita) o wdToggle (alterna entre True y False).

Range.Case: devuelve el valor o establece las mayúsculas o niúsculas del rango especificado. Esta propiedad usa diversas constantes wdCharactercase, incluyendo wdLowerCase, wdUpperCase, wdTitleSentece, wdTitleWord, wdToggleCase.

•El siguiente procedimiento pone el primer párrafo como Tipo Título, las primeras letras de cada palabra en mayúscula.

```
Sub Tipo_Título()
   ActiveDocument.Paragraphs(1).Range.Case = wdTitleWord
End Sub
```

VISUAL BASIC PARA APLICACIONES

Propiedades del objeto Range

Range.Characters: devuelve una colección de characters que representa todos los careacteres del rango.

Range.End: La posición del último carácter del rango.

Range.Start: la posición del primer carácter del rango.

Range. Font: devuelve el valor o establece un objeto Font que especifica el tipo de letra usado en el rango especificado.

Range.Italic: devuelve True si el rango especificado tiene en su totalidad formato cursiva, de otro modo funciona de la misma manera que Range.Bold.

Range.Paragraphs: devuelve una colección Paragraphs que representa a todos los objetos párrafo del rango.

Range.Sentences: devuelve una colección que representa a todos los objetos oración del rango.

Range.Words: devuelve una colección que contiene todas las palabras del rango.

Range. Text: devuelve el valor o establece el texto del rango.

VISUAL BASIC PARA APLICACIONES

Propiedades del objeto Range

```
Sub Ejemplo()
  With ThisDocument.Paragrphs(1).Range
        .Bold = true
        .Italic = True
        .Case = wdUpperCase
  End With
End Sub
```

Métodos del objeto Range

•El objeto Range incluye un gran número de métodos que se pueden utilizar para manipular texto. A continuación se exponen los más frecuentes:

Range.CheckGrammar: comprueba la gramática en el rango especificado.

Range. CheckSpelling: comprueba la ortografía del rango especificado.

Range.ConvertToTable: convierte el rango especificado en una tabla. La sintaxis simplificada es como sigue (para conocer sus 16 argumentos, consulte la ayuda en línea VBA):

VISUAL BASIC PARA APLICACIONES

Métodos del objeto Range

Range.ConvertToTable(*Separator, NumFilas, NumColumnas*)

Separator: una constante que especifíca el carácter que se va usar para separar las columnas: wdSeparateByCommas, wdSeparateByParagraphs, wdSeparateByTabs, wdSeparateByDefaultListSeparator. Si se omite ese argumento, VBA usa el valor de la propiedad del objeto ApplicationDefaultTableSeparator.

NumFilas: El número de filas que se quiere en la tabla. Si se omite este valor, VBA determina el número de filas automáticamente de acuerdo al texto y al valor de Separator.

NumColumnas: el número de columnas que se quiere en la tabla. Si omite este valor, VBA determina el número de columnas automáticamente de acuerdo al texto y al valor de Separator.

Range.Copy: copia el rango al portapapeles.

Range.Cut: corta el rango del documento y lo coloca en el portapapeles.

Range.Paste: pega el contenido del portapapeles en la posición actual del rango. Si no quiere sobrescribir el rango, use el método Collapse antes de pegar el texto.

Escuela Superior de Informatica
Universidad de Castilla-La Mancha
http://www.inf-cr.uclm.es
Word con VBA

VISUAL BASIC PARA APLICACIONES

Métodos del objeto Range

Range.Delete(*Unit, Count*)

Range: es el objeto range que contiene el texto a borrar.

Unit: una constante que especifica si se están borrando caracteres (wdCharacter) o palabras enteras (wdWord). Si se omite este argumento, VBA asume que se están borrando caracteres.

Count. El número de unidades a borrar; use un número positivo para borrar hacia delante y un número negativo para borrar hacia atrás.

Range.InsertAfter: inserta texto después del rango especificado.

Range.InserAfter(texto)

Range.InsertBefore: inserta texto antes del rango especificado.

Range.InserBefore(texto)

Range.InsertParagraph: inserta un párrafo que reemplaza al rango especificado.

Range.InsertParagraphAfter: inserta un párrafo después de un rango especificado.

Range.InsertParagraphBefore: inserta un párrafo antes de un rango especificado.

Range.select: selecciona el rango especificado.

VISUAL BASIC PARA APLICACIONES

El objeto Selection

Este objeto siempre hace referencia a una de dos cosas:

•El texto seleccionado

•La posición del cursor

Ya que mucho de lo que se hace en Word implica uno de estos elementos (dar formato al texto, insertar texto en la posición del cursor, etc.), el objeto Selection es uno de los más importantes de Word. Para referirse al texto seleccionado o al cursor se puede usar la propiedad Selection sin un cualificador de objeto. Por ejemplo, la siguiente sentencia cambia a negrita el texto seleccionado:

```
Selection.Range.Bold = True
```

Para especificar un objeto Selection, use el método Select, que está disponible en una serie de objetos Word, incluidos Document, range, Bookmark y table. Como ejemplo la siguiente sentencia selecciona el primer párrafo del documento activo.

```
ActiveDocument.Paragraphs(1).Range.Select
```


VISUAL BASIC PARA APLICACIONES

Propiedades del objeto Selection

El objeto Selection tiene bastantes propiedades, incluidas muchas que ya hemos visto en los objetos Document y Range (como Bookmarks, Characters, End, Start, etc.). Veamos algunas que son únicas para el objeto Selection.

Information(*Type*): devuelve información sobre el objeto Selection. Los datos devueltos dependen del valor del argumento *Type*. Por ejemplo, si es wdOverType, esta propiedad devuelve True si Word está en modo de sobrescribir. Consulte la ayuda en línea de VBA para ver la lista completa de constantes *Type*.

IPAtEndOfLine: devuelve True si el cursor se encuentra al final de una línea en un medio de un párrafo y False si no se encuentra al final de la línea, si está al final de un párrafo o si hay algún carácter seleccionado.

IsEndOfRowmark: devuelve True si el cursor se encuentra al final de una fila en una tabla. Si no, devuelve False.

Type: devuelve el tipo de selección:

wdNoSelection: no hay ninguna selección.

wdSelectionColumn: se selecciona una columna de una tabla.

VISUAL BASIC PARA APLICACIONES

Propiedades del objeto Selection

wdSelectionIP: la selección del cursor.

wdSelectionNormal: hay texto seleccionado.

wdSelectionRow: hay una fila seleccionada en una tabla.

Métodos del objeto Selection

El objeto Selection comparte mucho de sus métodos con el objeto range, como Copy, Cut, delete, InserAfter, InserBefore, InserParagraphAfter, InsertParagraphBefore y Paste.

Collapse: quita la selección y coloca el cursor de acuerdo con la siguiente sintaxis:

Selection.Collapse *Dirección*

Direccion: especifica el lugar donde quiere que vaya el cursor. Use wdCollapseEnd para colocar el cursor al final de la selección.

EndKey: extiende la selección o mueve el cursor al final de una unidad especificada (como una línea). En otras palabras, este método es equivalente a pulsar la tecla Fin. Vea, además que este método devuelve el número de caracteres que se movió la selección. Ésta es la sintaxis:

Escuela Superior de Informatica
Universidad de Castilla-La Mancha
http://www.inf-cr.uclm.es
Word con VBA

VISUAL BASIC PARA APLICACIONES

Métodos del objeto Selection

Selection.EndKey(*Unit, Extend*)

Unit: especifica al final de qué se va a mover la selección. Para texto normal, use wdLine (opción por defecto) o wdStory. En una tabla, wdColumn o wdRow.

Extend: especifica lo que ocurrirá a la selección. Para extender la selección al final de Unit, use wdExtend; para borrar la selección hasta la posición del cursor, use wdMove (opción por defecto).

EndOf: extiende la selección o mueve el cursor al final de una unidad especificada (como un párrafo). Este método devuelve el número de caracteres que se desplazó la selección. Este método es un poco más flexible que el método EndKey. Ésta es la sintaxis:

Selection.EndOf(*Unit, Extend*)

Unit: especifica la unidad en la que se va a mover la selección. Para texto normal, use wdCharacter, wdWord(opción por defecto), wdLine, wdSentence, wdParagraph, wdSection, wdStory. En una tabla, emplee wdCell, wdClumn, wdRow o wdTable.

VISUAL BASIC PARA APLICACIONES

Métodos del objeto Selection

Extend: especifica lo que le ocurrirá a la selección. Para extender la selección hasta el final de Unit, use wdExtended; para borrar la selección hasta la posición del cursor, wdMove (ésta es la opción por defecto).

Expand: expande la selección usando la siguiente sintaxis:

Selection.Expand *Unit*

Unit: especifica cómo se expandirá la selección. Para texto normal, use wdCharacter, wdWord (opción por defecto), wdLine, wdSentence, wdParagraph, wdsection o wdSotry. En una tabla emplee wdCell, wdColumn, wdRow o wdTable.

Por ejemplo, el siguiente código selecciona la primera palabra del primer párrafo. Con el objeto selection en su lugar, cambia la primera palabra a mayúsculas, extiende la selección a todo el párrafo, da formato al texto en negrita y cancela la selección para poner el cursor después del párrafo.

VISUAL BASIC PARA APLICACIONES

Métodos del objeto Selection

```
ActiveDocument.Paragraphs(1).Range.Words(1).Select
with Selection
        .Range.Case = wdUpperCase
        .Expand wdParagraph
        .Range.Bold = True
        .Collapse wdCollapsed
End with
```

Extend: activa el modo de Extender Selección de Word y extiende la selección actual. Cuando se usa sin argumento, este método extiende primero la selección a la palabra. Si se invoca a Extend en esta nueva selección, la selección se extiende a toda la frase. Llamadas subsiguientes a este método seleccionan párrafo, sección y documento.Como alternativa puede usar este método con un argumento:

Selection.Extend *Carácter*

VISUAL BASIC PARA APLICACIONES

Métodos del objeto Selection

Carácter: un único carácter de cadena que especifica el carácter al que se va a extender la selección. Vea que este argumento distingue entre mayúsculas y mnúsculas, por lo que las siguientes sentencias producen resultados diferentes:

Selection.Extend “t” ´Extiende la selección a la siguiente “t” minúscula

Selection.Extend “T” ´Extiende la selección a la siguiente “T” mayúscula

TypeBackspace: borra el carácter a la izquierda del cursor. Si existe una selección, este métdo borra el texto seleccionado. (Éste método es equivalente a pulsar la tecla Retroceso).

TypeParagrapgh: inserta un párrafo nuevo en la posición del cursor. Si existe texto seleccionado, este método eliminará dicho texto. (Este método equivale a pulsar la tecla Intro).

TypeText: inserta texto en la posición actual del cursor. Si existe texto seleccionado y la propiedad Options.ReplaceSelection tiene el valor True, este método reemplaza la selección con el texto especificado.

Escuela Superior de Informatica
Universidad de Castilla-La Mancha
http://www.inf-cr.uclm.es
Word con VBA

VISUAL BASIC PARA APLICACIONES

Métodos del objeto Selection

Ésta es la sintaxis:

Selection.TypeText *Texto*

Texto: la cadena que se quiere insertar.

Ejemplo. Un procedimiento que pregunta al usuario si se debe reemplazar el texto seleccionado

```
Sub AskToReplaceText(9
Dim result As Integer
´¿Estamos en la posición del cursor o está desactivado ReplaceSelections?
If selection.Type = wdSelectionIP Or Not ReplaceSelection Then
  ´Se inserta el texto
  Selection.TypeText “Texto”
Else
  ´Si no, pregunta al usuario si se reemplaza el texto
  result = MsgBox(“¿Reemplazar el texto seleccionado? ”, VbYesNo + vbQuestion))
```

VISUAL BASIC PARA APLICACIONES

Métodos del objeto Selection

```
  If result = vbNo Then
    ´Desactiva ReplaceSelection para insertar sin reemplazar
    Options.replaceSelection = False
    Selection.TypeText “Texto”
    Options.replaceSelection = True
  Else
    ´Reemplaza el texto seleccionado
     Selection.TypeText “Texto”
  End If
 End If
End Sub
```


VISUAL BASIC PARA APLICACIONES

Ejemplo: Formato Fuente

```
Sub FormatoCaracteres()
  With Selection.Font
     .Name = "Arial"
     .Size = 14
     .Bold = True
     .Italic = False
     .Underline = wdUnderlineNone
     .StrikeThrough = False
     .DoubleStrikeThrough = False
     .Outline = False
     .Emboss = False
     .Shadow = False
     .Hidden = False
     .SmallCaps = True
     .AllCaps = False
     .ColorIndex = wdAuto
     .Engrave = False
     .Superscript = False
     .Subscript = False
     .Spacing = 0
     .Scaling = 100
     .Position = 0
     .Kerning = 0
     .Animation = wdAnimationNone
  End With
End Sub
```

Parte 4:
Objetos Characters, Words, Sentences y Paragraphs

VISUAL BASIC PARA APLICACIONES

El objeto Characters

•El objeto Characters es una colección que representa todos los cartacteres en el objeto especificado. Por ejemplo, ActiveDocumen.Paragraphs(1).Range.Characters es la colección de todos los caracteres del objeto Range (el primer párrafo del documento activo). Otros objetos que tienen la propiedad Characters son Document y Selection.

•Como Characters es una colección, para referirse a cada carácter individual hay que incluir el índice numérico. La siguiente sentencia establece el tamaño del primer carácter en 20 puntos.

```
ActiveDocument.Characters(1).Font.Size = 20
```

•Para contar el número de caracteres en un objeto específico, use la propiedad Count:

```
totalChars = Documents("Chap06.doc").Characters.Range.Count
```

Ejemplo: una función que cuenta el número de coincidencias de un caracter específico en un objeto.

VISUAL BASIC PARA APLICACIONES

El objeto Characters

```
Function CountCharacters(countObject As Object, letter As String) As long
Dim i As long, char As Range
i = 0
For each char In countObect.Characters
  If char = letter Then i = i + 1
Next char
CountCharacters = i
End Function

Sub TestCountCharacters()
        MsgBox CountCharacters(ActiveDocument, “e”)
End Sub
```


VISUAL BASIC PARA APLICACIONES

El objeto Words

•El objeto Words es una colección que representa todas las palabras de un objeto especificado. Por ejemplo, ActiveDocument.Words es la colección de todas las palabras del documento activo. Los objetos Paragraph, Range y Selection tienen también la propiedad Words..

•Para referirse a las palabras individuales se usa el índice numérico con la colección Words. Sin embargo, no devuelve un objeto “Word”, pues tal cosa no existe en el universo VBA. En vez de ello, cada palabra se clasifica como un objeto Range.

•La siguiente sentencia da formato negrita a la primera palabra del documento activo:

```
ActiveDocument.Words(1).Font.Bold = True
```

•Para contar el número de palabras del objeto especificado, use la propiedad Count.

```
TotalWords = Documents(“Article.doc”).Words.Count
```

•Vea, sin embargo, que el objeto Words incluye los signos de puntuación y las marcas de párrafo dentro del objeto.

•Si quiere saber el número real de palabras de un objeto, use la siguiente función:

VISUAL BASIC PARA APLICACIONES

El objeto Words

```
Function CountWords(countObject As Object) As Long
Dim i As Long, word As Range
i = 0
For each word in counObject.Words
  Select  Case Asc(Left(word,1))
          Case 48 To 90, 97 To 122  ´letra o número
                  i = i +1
  End Select
Next word
CountWords = i
End Function

Sub TestCountWords()
  With ActiveDoucument
          MsgBox "Total de palabras" & CountWords(.Range)
  End With
End Sub
```


VISUAL BASIC PARA APLICACIONES

El objeto Sentences

•Es una colección de todas las oraciones de un objeto especificado, ya sea Document, Range o Selection. Cada miembro de esta colección se referencia mediante un índice numérico y el objeto resultante es de tipo Range. Por ejemplo la siguiente sentencia almacena la primera oración del documento activo en la variable: firstSentence:FirstSentence = ActiveDocument.Sentences(1)

•En el siguiente fragmento de procedimiento, la propiedad Count se usa para determinar la última oración de un documento:

```
With Documents("Remarks.doc")
          totalSentences = .Sentences.Count
          LastSentence = .Sentences(.totalSentences)
End With
```

VISUAL BASIC PARA APLICACIONES

El objeto Paragraph

•Un objeto Paragraph es miembro de la colección Paragraphs, que representa a todos los párrafos de un objeto Document, Range o Selection especificado. Al igual que con los demás objetos de texto, hay que usar un índice numérico para especificar un párrafo individual.

Propiedades del objeto Paragraph

Párrafo.KeepTogether: indica o establece si el párrafo especificado debe permanecer en la misma página cuando Word vuelva a paginar el documento.

Párrafo.KeepWithNext: devuelve el valor o establece si el párrafo especificado debe permanecer en la misma página junto al siguiente cuando Word vuelva a paginar el documento.

Párrafo.LeftIndent: devuelve el valor o establece la sangría izquierda (en puntos) del párrafo especificado.

Párrafo.RightIndent: devuelve el valor o establece la sangría derecha (en puntos) del párrafo especificado.

Párrafo.LineSpacing: indica o establece el interlineado (en puntos) del párrafo especificado.

Escuela Superior de Informatica
Universidad de Castilla-La Mancha
http://www.inf-cr.uclm.es
Word con VBA

VISUAL BASIC PARA APLICACIONES

Propiedades del objeto Paragraph

Párrafo.SpaceAfter: informa o establece el espaciado posterior (en puntos) del párrafo especificado.

Párrafo.SpaceBefore: informa o establece el espaciado anterior (en puntos) del párrafo especificado.

Párrafo.Style: devuelve el valor o bien establece el estilo del párrafo especificado. Word tiene una inmensa cantidad de constantes que representan sus estilos predefinidos. Consulte al Examinador de objetos para conocer todas estas constantes.

El siguiente fragmento de procedimiento aplica algunas proiedades al párrafo activo:

```
With Selection.Range
        .LeftIndent = InchesToPoints(1)
        .LineSpacing = 12
        .SpaceAfter = 6
        .Style = wdStyleNormal
End With
```

VISUAL BASIC PARA APLICACIONES

Ejemplo: Formato Párrafo

```
Sub FormatoParrafo()
With Selection.ParagraphFormat
    .LeftIndent = InchesToPoints(1)
    .RightIndent = InchesToPoints(1)
    .SpaceBefore = 6
    .SpaceAfter = 6
    .LineSpacingRule = wdLineSpace1pt5
    .Alignment = wdAlignParagraphJustify
    .WidowControl = True
    .KeepWithNext = False
    .KeepTogether = False
    .PageBreakBefore = False
    .NoLineNumber = False
    .Hyphenation = True
    .FirstLineIndent = InchesToPoints(0.49)
    .OutlineLevel = wdOutlineLevelBodyText
  End With
End Sub
```


VISUAL BASIC PARA APLICACIONES

Métodos del objeto Paragraph

Párrafo.Indent: aplica una sangría hasta la siguiente marca de tabulación en el párrafo especificado.

Párrafo.Next: se mueve hacia delante en el documento desde el párrafo especificado hasta devolver un objeto Paragraph:

Párrafo.Next(*Count*)

Párrafo: el párrafo desde el que se quiere realizar el movimiento.

Count: el número de párrafo para moverse hacia delante.

Párrafo.Outdent: reduce una sangría hasta la marca de tabulación anterior en el párrafo especificado.

Párrafo.Previous: se mueve hacia atrás en el documento desde el párrafo especificado hasta devolver un objeto Paragraph.

Párrafo.Previous(*Count*)

Párrafo: el párrafo desde el que se quiere realizar el movimiento.

Count: el número de párrafo para moverse hacia atrás.

VISUAL BASIC PARA APLICACIONES

Métodos del objeto Paragraph

Párrafo.Space1: establece el interlineado del párrafo especificado en “sencillo”.

Párrafo.Space15: establece el interlineado del párrafo especificado en “1,5 líneas”.

Párrafo.Space2: establece el interlineado del párrafo especificado en “doble”.

Parte 5:
Objeto Table

VISUAL BASIC PARA APLICACIONES

El objeto Table

•Una tabla en Word se representa por el objeto Table. Dado que un documento puede contener múltiples tablas es obvio que debe existir una colección Tables que las contiene. De hecho cualquier objeto rango es en s´un texto que puede contener tablas, de forma que en Range tenemos también una colección Tables.

•Cada tabla está compuesta de celdas representadas por la propiedad Cells(fila, columna).

•Cada celda contiene a su vez texto, representado por la propiedad Range de la celda.

•Por ejemplo, para asignar un contenido a la celda situada en la fila 1 columna 2 de una tabla referenciada por la variable Tabla escribimos:

Tabla.Cells(1, 2).Range.Text = “Contenido”

Insertar una tabla en el texto

•Para insertar una tabla en un texto representado por un objeto rango podemos usar el método Add de la colleccion Tables del rango, que tiene la siguiente sintaxis:

rango.Tables.Add(*lugar, filas, columnas*)

VISUAL BASIC PARA APLICACIONES

Insertar una tabla en el texto

•El primer parámetro es un objeto rango que especifica dónde se ubicará la tabla. Los otros dos parámetros dan el número de filas y columnas que tendrá inicialmente la tabla.

Ejemplo: ubicar el cursor y crear una tabla de 10 filas y 5 columnas

```
Sub CrearTabla()
 ActiveDocument.Tables.Add Range:=Selection.Range, NumRows:=10, NumColumns :=5
End Sub
```

Ejemplo: Poner como título de cada columna, ENERO, FEBRERO, MARZO ABRIL, MAYO, poner los títulos en negrita, tamaño 12 y con alineación centrada.

VISUAL BASIC PARA APLICACIONES

Insertar una tabla en el texto

```
Sub DatosTabla()
   Selection.TypeText Text:="ENERO"
   Selection.MoveRight Unit:=wdCharacter, Count:=1
   Selection.TypeText Text:="FEBRERO"
   Selection.MoveRight Unit:=wdCharacter, Count:=1
   Selection.TypeText Text:="MARZO"
   Selection.MoveRight Unit:=wdCharacter, Count:=1
   Selection.TypeText Text:="ABRIL"
   Selection.MoveRight Unit:=wdCharacter, Count:=1
   Selection.TypeText Text:="MAYO"
   Selection.MoveLeft Unit:=wdCharacter, Count:=8, Extend:=wdExtend
   Selection.ParagraphFormat.Alignment = wdAlignParagraphCenter
   Selection.Font.Bold = wdToggle
   Selection.Font.Size = 12
   Selection.MoveLeft Unit:=wdCharacter, Count:=1
End Sub
```


Escuela Superior de Informatica
Universidad de Castilla-La Mancha
http://www.inf-cr.uclm.es
Word con VBA

VISUAL BASIC PARA APLICACIONES

Insertar o eliminar filas y columnas

•Una vez creada la tabla podemos añadir filas teniendo en cuenta que existe una colección Rows que representa a las filas y que tiene su correspondiente método Add.

•De igual forma eliminaremos una fila de la tabla con el método Delete.

•Por ejemplo, para eliminar la tercera fila de una Tabla escribiremos:

Tabla.Rows(3).Delete

•La siguiente instrucción eliminaría toda la tabla, ya que al no especificar un número de fila se asume que borra toda la colección:

Tabla.Rows.Delete

•En este caso sería equivalente a:

Tabla.Delete

•Las mismas observaciones se aplican al caso de las columnas de la tabla, representadas por la colección Columns.

VISUAL BASIC PARA APLICACIONES

Dar formato a la tabla

•Usaremos el método AutoFormat del objeto Tabla que equivale a la opción Tabla autoformato del menú Word. Para cada tipo de formato tenemos definida una constante de VBA, por ejemplo :wdTableFormat3DEffects1, wdTableFormat3DEffects2, wdTableFormat3DEffects3, wdTableFormatClassic1, wdTableFormatClassic2, wdTableFormatClassic3, wdTableFormatClassic4, etc.

•El método AutoFormat tiene varios métodos que podrá encontrarlos en la ayuda en línea de VBA.

•Como ejemplo tenemos a continuación dos procedimientos. En el primero se crea una tabla de una fila y dos columnas situadas al final del documento activo y se inicializa asignándole un formato y un contenido a su primera fila, que realiza la función de cabecera de la tabla. Este primer procedimiento llama repetidas veces a la segunda que se encarga de ir añadiendo filas a la tabla conforme el usuario va introduciendo datos que se colocan en la tabla.

VISUAL BASIC PARA APLICACIONES

```
Sub Crear_Tabla
  Dim Tabla As Table, texto As Range
  Dim Rng As Range, seguir As String * 1
  ´--- crear la tabla
  Set texto = ActiveDocument.Range
  Set Rng = texto.Paragraphs.Last.Range
  Set Tabla = texto.Tables.Add(Rng, 1, 2)
  Tabla.AutoFormat Format:=wdTableFormatClassic3
  Tabla.Cell(1,1).Range.Text = “Nombre”
  Tabla.Cell(1, 2).Range.text = “Teléfono”
  ´---- Añadir filas y datos a la tabla
  Do
    Añadir_Datos_Tabla Tabla
    seguir = UCase(InputBox(“Seguir S/N” ))
  Loop Until seguir = “N”
End Sub
```

VISUAL BASIC PARA APLICACIONES

```
Sub Añadir_Datos_Tabla(Tabla As Table)
  Dim nm As String, tel As String
  nm = InputBox("Nombre: ")
  Tel = InputBox("Teléfono: ")
  Tabla.Rows.Add
  Tabla.Cell(Tabla.Rows.Count, 1).Range = nm
  Tabla.Cell(Tabla.Rows.Count, 1).Range = Tel
End Sub
```

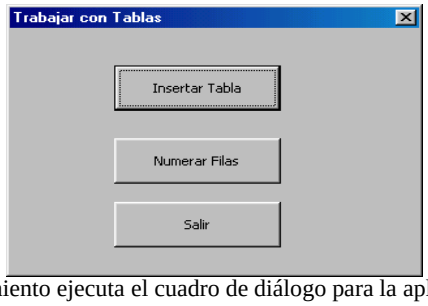
•Observe en el segundo procedimiento que Add añade la fila al final de la tabla, y que para acceder a ella usamos la propiedad Count de la colección Rows que nos devuelve el número de elementos de la colección, que en este caso será el número de filas de la tabla.

Parte 6: Ejemplos

VISUAL BASIC PARA APLICACIONES

Ejemplo 1:

Implementar una aplicación que permita crear una tabla, y numere las filas.

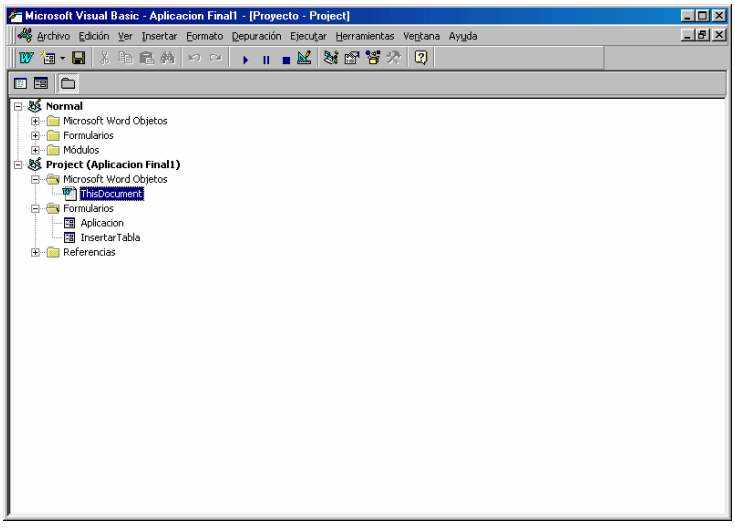


El siguiente procedimiento ejecuta el cuadro de diálogo para la aplicación.

```
Sub Inicio_Aplicacion()
  Aplicacion.Show  ´El Cuadro de diálogo se llama Aplicación
End Sub
```

Este código deberá aparecer en el módulo ThisDocument, del proyecto en el cual está trabajando.

VISUAL BASIC PARA APLICACIONES

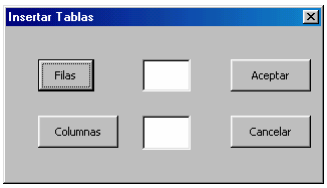


VISUAL BASIC PARA APLICACIONES

Ejemplo 1:

El botón insertar tabla debe preguntar en otro cuadro de diálogo el número de filas y columnas de cada tabla.

Al hacer clic sobre el botón Insertar Tabla aparecerá el siguiente cuadro de diálogo:




```
Private Sub CommandButton1_Click() ´Click sobre Insertar Tabla
  InsertarTabla.Show
End Sub
```


VISUAL BASIC PARA APLICACIONES

Ejemplo 1:

Al hacer clic sobre el botón Aceptar del cuadro de diálogo Insertar Tabla se ejecutará el siguiente código, que inserta la tabla y vuelve al cuadro de diálogo principal:

```
Private Sub CommandButton3_Click()
ActivateDocument.Tables.Add Range:=Selection.Range, NumRows:=TextBox1,
                                                    NumColumns:=TextBox2
End Sub
```

Al hacer sobre el botón Cancelar, no inserta la tabla y vuelve al cuadro de diálogo principal.

```
Private Sub CommandButton4_Click()
  Unload InsertarTabla
End Sub
```

VISUAL BASIC PARA APLICACIONES

Ejemplo 1:

Ahora veamos el botón Numerar líneas del cuadro de diálogo principal. A continuación mostramos el código que se ejecutará al hacer click sobre este botón:

```
Private Sub CommandButton2_Click()
  TotalFilas = ActiveDocument.Tables(1).Rows.Count
  For Fila = 2 To TotalFilas
    ActiveDocument.Tables(1).Cell(Fila, 1).Range.Text = Fila
  Next
End Sub
```

Ahora veamos el botón Sali del cuadro de diálogo principal. A continuación mostramos el código que se ejecutará al hacer click sobre este botón:

```
Private Sub CommandButton3_Click()
 ActiveDocument.Save ´graba el trabajo realizado
 Unload Aplicación ´termina la aplicación
End Sub
```


VISUAL BASIC PARA APLICACIONES

Ejemplo 2 (Objeto Range de Word):

El siguiente procedimiento muestra la primera palabra, la primer oración y el primer párrafo del documento.

```
Sub Palabra_Línea_y_Párrafo
  Dim Rng As Range ´creamos el objeto Range
  Dim txt As String
  Set Rng = ActiveDocument.Range
  txt = ActiveDocument.Name
  MsgBox Rng.Words(1), , “primera palabra del texto” + txt
  MsgBox Rng.Sentences(1), , “primera oración del texto” + txt
  MsgBox Rng.Paragraphs(1), , “primer párrafo del texto” + txt
End Sub
```

VISUAL BASIC PARA APLICACIONES

Ejemplo 3 (Añadir texto al final del documento)

La propiedad Last de la colección Paragraphs contiene una referencia al último párrafo del documento, que a su vez es un objeto Range.

Cada elemento de un texto puede ser referenciado como un objeto rango a través de la propiedad Range. El siguiente código usa Last y range conjuntamente para obtener el objeto rango correspondiente al último párrafo del documento activo. Posteriormente utiliza el método InsertAfter para añadir al final el nuevo texto:

```
Sub Añadir_Texto(texto As String)
  Dim Parr as Paragrapgh
  Dim Rng As Range, RngParr As Range
  ´--- referencia al documento activo
  Set Rng = ActiveDocment.Range
  ´--- referencia al último párrafo
  Set Parr = Rng.Paragraphs.Last
  ´--- objeto rango del párrafo
  SetParr = Parr.Range
  RngParr.InsertAfter texto
End Sub
```


VISUAL BASIC PARA APLICACIONES

Ejemplo 3 (Añadir texto al final del documento)

El texto quedará incorporado al último párrafo. Si quisiéramos añadir el texto como un nuevo párrafo independiente, entonces antes de invocar a InsertAfter, deberemos crear el párrafo con el método InsertParagraphAfter

```
´--- insertar un párrafo nuevo al finañ
RngParr.InsertParagrapghAfter
´--- añadir el texto en el nuevo párrafo
Rng.Parr.InsertAfter texto
```

Observe que al insertar el párrafo automáticamente la variable RngParr pasa a referirse al nuevo párrafo. Por otra parte el nuevo párrafo “hereda” el formato del último párrafo, ya que se ha creado a partir de éste.

VISUAL BASIC PARA APLICACIONES

Ejemplo 4 (Dar formato al texto)

Tenemos para ello los objetos Font (fuente del texto)y ParagraphFormat (formato de párrafo). Ambos proporcionan acceso desde código a las mismas opciones que tenemos en los menús Formato Fuente y Formato Párrafo de Word a través de sus propiedades.

Por ejemplo el siguiente código asigna el atributo cursiva, tamaño de la letra 10 puntos y tipo de fuente "Arial" al texto de un párrafo, al cual además le aplica formato de párrafo justificado y espaciado de 20 puntos.

```
Sub Formato(Parr As Paragrapgh)
  Dim Rng As Range
  Set Rng = Parr.Range
            With Rng
              .Font.Italic = True
              .Font.Size = 10
              .Font.Name = "Arial"
            End With
            With  Rng.ParagraphFormat
              .Alignment = wdAlignparagraphJustify
              .LineSpacing = 20
            End With
End Sub
```


VISUAL BASIC PARA APLICACIONES

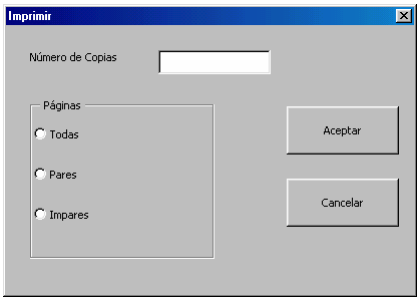
Ejemplo 5:

Crear una barra de herramientas : IMPRESION que tenga tres botones (la barra de herramientas deberá estar presente siempre):

1) N_PÁGINAS :deberá ejecutar un procedimiento que permita poner el número a las páginas, abajo a la derecha.

2) MARGENES: deberá definir los márgenes, SUPERIOR = 5 cm, INFERIOR = 3 cm, IZQUIERDO = 2, DERECHO = 2

3) IMPRESIÓN: deberá mostrar un Cuadro de diálogo que tendrá el siguiente aspecto:



VISUAL BASIC PARA APLICACIONES

Ejemplo 5:

1) Crear una barra de herramientas “Impresión”: HERRAMIENTAS PERSONALIZAR BARRA DE HERRAMIENTAS NUEVA

2) Escribir todos los procedimientos que deberá asignar luego a los tres botones de la barra de herramientas.

3) Añadir un botón a la barra de herramientas: HERRAMIENTAS PERSONALIZAR COMANDOS MACROS, y arrastrar la macro que desee a la barra de herramientas que acaba de crear. Para cambiar el aspecto del botón elija MODIFICAR SELECCIÓN, y luego NOMBRE para cambiarle el nombre y CAMBIAR IMAGEN DEL BOTON, si quiere cambiarle el diseño del botón que aparece en la barra de herramientas.

Procedimiento para poner número a las páginas (botón N_PAGINAS):

```
Sub NPagina()
   Selection.Sections(1).Footers(1).PageNumbers.Add PageNumberAlignment:=
                                wdAlignPageNumberRight, FirstPage:=True
End Sub
```


VISUAL BASIC PARA APLICACIONES

Ejemplo 5:

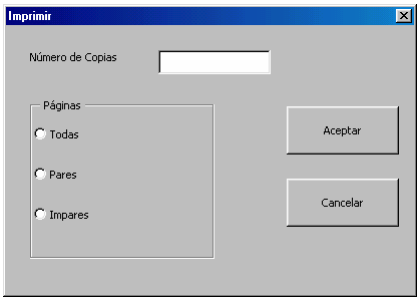
Procedimiento para definir los márgenes(botón MARGENES):

```
Sub MargenesCM()
   With ActiveDocument.PageSetup
      .TopMargin = CentimetersToPoints(5)
      .BottomMargin = CentimetersToPoints(3)
      .LeftMargin = CentimetersToPoints(2)
      .RightMargin = CentimetersToPoints(2)
    End With
End Sub
```

VISUAL BASIC PARA APLICACIONES

Ejemplo 5:

Cuadro de diálogo Imprimir (botón Impresión)



El procedimiento asociado al botón IMPRESIÓN tendrá el siguiente código:

```
Sub MenuImpresion()
   Imprimir.Show
End Sub
```


VISUAL BASIC PARA APLICACIONES

Ejemplo 5:

El botón ACEPTAR del cuadro de diálogo IMPRIMIR tendrá asociado el siguiente código:

```
Private Sub Aceptar_Click()
  If Todas Then QueImprimir = wdPrintAllPages
  If Pares Then QueImprimir = wdPrintEvenPagesOnly
  If Impares Then QueImprimir = wdPrintOddPagesOnly
  NroCopias = Copias
   Application.PrintOut FileName:="", Copies:=NroCopias, PageType:=QueImprimir
  Unload Imprimir
 End Sub
```

El botón CANCELAR tendrá el siguiente código asociado:

```
Private Sub Cancelar_Click()
   Unload Imprimir
End Sub
```

VISUAL BASIC PARA APLICACIONES

Ejemplo 5: