

Capítulo 5

Los objetos y colecciones en VBA

1. Noción de objeto

VBA es un lenguaje que permite hacer programación orientada a objetos (POO): un objeto representa una idea, concepto o entidad del mundo real, como un avión, un individuo e incluso una película. Tiene una estructura interna y un comportamiento, y se puede comunicar con sus homólogos. Los elementos que permiten describir un objeto forman lo que se llama una clase. Cada objeto que proviene de una clase es una instancia de clase. Las clases tienen propiedades, métodos y eventos.

1.1 Propiedades

El objeto es una entidad que podemos distinguir gracias a sus propiedades (su color o dimensiones, por ejemplo). Si tomamos como ejemplo un libro, este se caracteriza por sus propiedades: número de páginas, título, número de capítulos, editor, contenido, etc. Cada una de sus propiedades se puede especificar en cada libro, pero todos los libros tienen fundamentalmente las mismas propiedades. Algunas propiedades de los objetos se pueden modificar (velocidad de un coche, por ejemplo) y otras no (una marca de coche).

En programación con VBA, la sintaxis general de acceso a las propiedades de un objeto es la siguiente:

■ `UnObjeto.SuPropiedad`

Aquí se accede a la propiedad `SuPropiedad` del objeto `UnObjeto`. La combinación `UnObjeto.SuPropiedad` tendrá el mismo comportamiento que una variable clásica, que puede tomar un valor o devolver un valor con el uso del operador `=`.

Por ejemplo, es posible leer el contenido SQL de una consulta de Access y visualizarla con el siguiente código:

```
Sub LeerSQLConsulta()  
    Dim UnaConsulta As QueryDef  
    UnaConsulta.SQL = "SELECT * FROM MiTabla"  
    ...  
    MsgBox UnaConsulta.SQL  
End Sub
```

En el capítulo Los objetos de acceso a los datos DAO y ADO, volveremos al objeto `QueryDef` con más detalle.

Una propiedad de un objeto puede ser un objeto ella misma, por lo que se podría utilizar más adelante en el código.

1.2 Métodos

Además de los elementos que caracterizan un objeto, también es posible realizar acciones con estos objetos, como por ejemplo "sentarse", "acostarse", "arreglarse", etc. Estas acciones se llaman métodos. Estas acciones se representan en forma de procedimientos y funciones en VBA, según puedan o no devolver valores. Algunas acciones pueden necesitar argumentos para funcionar (número de caracteres de un libro, con o sin espacios).

En programación con VBA, la sintaxis general de acceso a los métodos de un objeto es la siguiente:

■ `UnObjeto.SuMetodo [MiArgumento]`

En el siguiente ejemplo, se trata simplemente de refrescar el vínculo entre una tabla relacionada y la aplicación Access:

```
Sub RefrescarRelacionTabla()  
    Dim UnaVariableTemporal As TableDef  
    ...  
    UnaVariableTemporal.RefreshLink  
End Sub
```

Y, en caso de que el método sea una función, podemos almacenar el resultado en una variable, como en el siguiente ejemplo:

```
Sub Número_Campos()  
    Dim Nm_Campos As Integer  
    Dim MiTabla As TableDef  
    ...  
    Nm = MiTabla.Fields.Count  
End Sub
```

De la misma manera que en las llamadas a funciones o procedimientos, los argumentos se separan por comas en las llamadas a métodos del objeto.

1.3 Eventos

Para cada objeto, es posible detectar y tener en cuenta el resultado de acciones particulares externas, que llamamos eventos. Todos los eventos que tienen lugar durante la ejecución de un programa se detectan y gestionan por la máquina, pero, según la opción del desarrollador, solo algunas se pueden tratar de manera particular, como por ejemplo un clic en un botón, una casilla de selección que se marca, la apertura o cierre de una ventana, etc. Estos eventos se gestionan a través de procedimientos que algunas veces tienen argumentos y otras no.

En VBA, la sintaxis más frecuente para los eventos es la siguiente:

```
Sub MiObjeto_MiEventoDestacado()  
    'El código que se ejecutará cuando se detecte el evento  
End Sub
```

Por ejemplo, cuando se trata de un clic en un botón llamado MiBoton, tendrá el código siguiente:

```
Sub MiBoton_Click()  
    'código que se ejecutará  
End Sub
```

Los eventos en Access se tratarán más adelante en detalle, en el capítulo Los eventos de Access.

1.4 Las colecciones

Cuando varios objetos de una misma clase se agrupan, pueden pertenecer a una misma colección de objetos. Para referirse a un elemento en particular de una colección de objetos, hay varias sintaxis, entre las siguientes:

```
Nombre_Colección!Nombre_Objeto  
Nombre_Colección![nombreObjeto]  
Nombre_Colección("NombreObjeto")  
Nombre_Colección(variable_Nombre) 'variable_Nombre es una cadena de  
caracteres que contiene el nombre del objeto  
Nombre_Colección(variable_Numero) 'variable_Numero es un valor  
digital que contiene el número del objeto dentro de la colección.
```

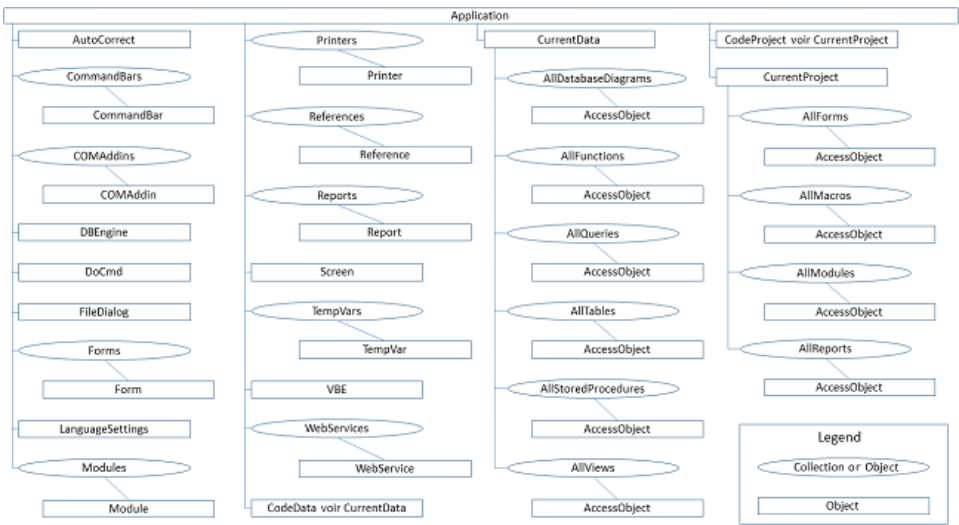
Las dos sintaxis que se utilizan más habitualmente son la tercera y la quinta, ya que permiten el uso de IntelliSense (ver capítulo VBE y seguridad en Access 2019).

■ Observación

Es habitual en las colecciones comenzar a contar en 0 y no en 1. Además, el número de un objeto dentro de una colección dependerá de la presencia de otros elementos antes o después de él, lo que no garantiza localizar el objeto correcto por este medio.

2. Modelo de objeto de Access

El objetivo de algunas de las secciones siguientes es mostrar el modelo jerárquico utilizado dentro de la aplicación Access, en forma de colecciones de objetos, que se van a explicar a continuación.



3. Colecciones en Access

A continuación se muestran las principales colecciones que podemos manipular con VBA en Access.

Colección	Contiene una colección de	Descripción
COMAddins	COMAddin	Colección de los complementos COM
CommandBars	CommandBar	Colección de las barras de comando
Forms	Form	Colección de los formularios abiertos . Ver también <code>CurrentProject.AllForms</code>

Colección	Contiene una colección de	Descripción
Modules	Module	Colección de los módulos
Printers	Printer	Colección de las impresoras disponibles
References	Reference	Colección de las referencias a librerías . Ver Menú: Herramientas\Referencias
Reports	Report	Colección de estados . Ver también <code>CurrentProject.AllReports</code>
TempVars	TempVar	Colección de las variables temporales
WebServices	WebService	Colección de las conexiones a servicios web

4. Objetos de Access

A continuación se muestran los principales objetos que es posible manipular en el modelo Access.

Objeto	Descripción
Application	Representa la aplicación Microsoft Access activa.
AutoCorrect	Representa las opciones de corrección automática de Access.
DBEngine	Representa el motor de base de datos Microsoft Jet. Este objeto permite controlar el resto de los objetos de acceso a los datos.
DoCmd	Permite convertir en VBA las acciones de Macros.
FileDialog	Permite acceder a las funcionalidades de los cuadros de diálogo (Abrir o Guardar, por ejemplo).