Jose Ignacio González Gómez.
Departamento de Economía Financiera y Contabilidad
Universidad de La Laguna
www.jggomez.eu
www.ecofin.ull.es/users/jggomez.

Tema:

Apuntes Programación Excel VBA.

PARTE I

Indice

1 Activar la ficha del programador

Los comandos que se usan para editar y ejecutar macros en Excel se encuentran en el grupo **Código** de la ficha **Programador**, la cual de forma predeterminada se encuentra oculta. Por ello nuestro primer paso de este tutorial para proteger o desproteger una hoja Excel será abrir nuestro archivo y activar la ficha **Programador**. Recuerda que este paso solo aplicará en caso de que no tengas activa esta ficha; si ya la tienes activa puedes ir directamente al paso siguiente. En Excel 2010 ubica la cinta de opciones de la barra de herramientas e ingresarás en: **Botón Inicio**/**Opciones de Excel**/ **Más frecuentes**/ y deberás seleccionar la opción **Mostrar ficha programador en la cinta de opciones**. La imagen anexa te muestra cuál es la opción a seleccionar para activar esta ficha:

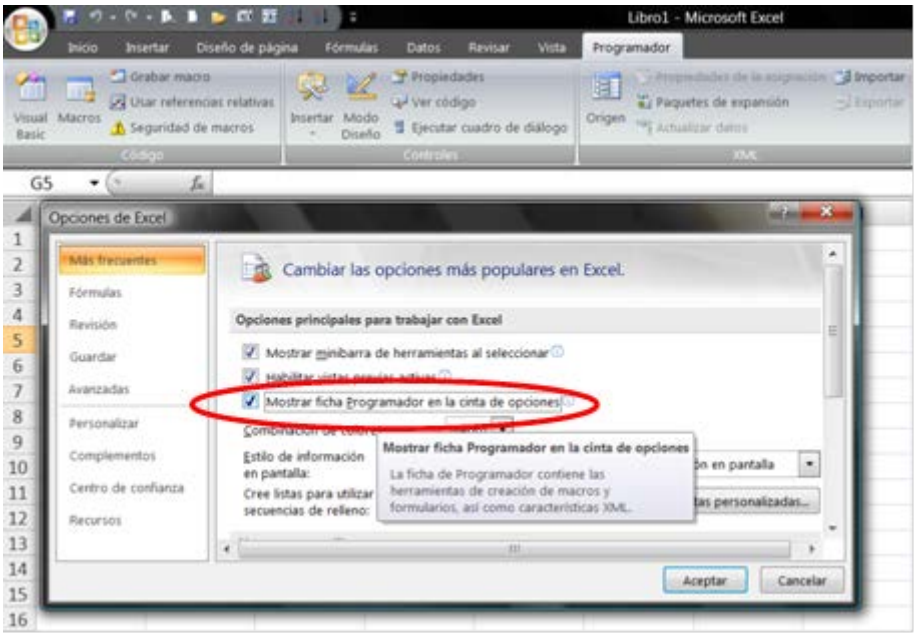


Ilustración 1

2 Aspectos Conceptuales Previos

2.1 Las Técnicas de Programación Orientada a Objetos.

La programación en Office se basa en las técnicas de programación orientadas a objetos, asi el lenguaje Visual Basic es un entorno de programación orientado a objetos.

Cada producto Excel, Word, etc tiene su propio modelo de objetos únicos. Mientras trabajamos con excel iremos comprendiendo gradualmente el modelo de objetos. Puede ser muy complicado de entender al principio, pero al final se comprenderá perfectamente.

En Visual Basic programamos todo lo que esta relacionado con lo que le pasa a un objeto: es decir si tenemos un botón podemos programar lo que pasa cuando hacemos click o doble clic o cuando pulsamos el botón derecho sobre el mismo.

Es decir, a cada objeto le corresponden unos eventos (cosas que le pasan al objeto).

2.2 Semejanzas y diferencias entre VBA y VB.

Visual Basic para Aplicaciones (VBA) es un subconjunto del lenguaje de programación Visual Basic (VB). El lenguaje VBA difiere de su hermano mayor VB básicamente en que VB se diseño para crear aplicaciones autosuficientes mientras que VBA se emplea para automatizar una aplicación existente tanto sea Word, Excel, Access, etc.

Sin embargo el lenguaje VB y VBA son muy similares en su estructura, lo que permite que si un usuario ya conoce VB el aprendizaje de VBA será muy sencillo. De igual forma si el usuario no conoce VB pero tiene cierta soltura con el VBA tendrá grandes bases para aprender VB.Por tanto un aspecto fundamental que nos motivará aprender VBA para Excel es que el conocimiento que podamos adquirir lo podemos aprovechar para el resto de las aplicaciones Word, Acces, etc.

2.3 VBA para Excel. Ventajas y posibilidades.

El objetivo fundamental por tanto del VBA Excel es un lenguaje de automatización que emplearemos para automatizar procedimientos y procesos comumente empleados, generar soluciones personalizadas, y si lo deseamos, implementar aplicaciones usando Excel como plataforma de desarrollo. Más concretamente podremos:

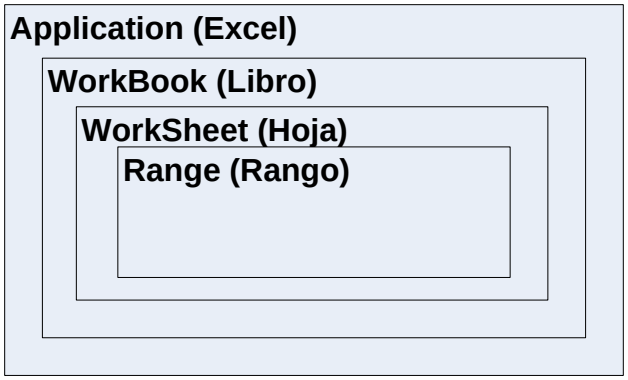
- Automatizar tareas repetitivas
- Personalizar la interfaz de Excel con barras de herramientas, menús y formularios.
- Simplificar el uso de plantillas
- Agregar funcionalidad al entorno de Excel.
- Crear informes.
- Ejecutar manipulación y análisis de datos refinados.

2.4 Introducción a los objetos Excel

Excel ofrece una variedad amplia de objetos que permiten automatizar y personalizar el trabajo diario. Así por ejemplo tenemos:

- Objeto ***Application*** es el objeto superior y representa en nuestro caso a la aplicación Excel, también podría ser el Word o PowerPoint, etc.. es decir cualquier elemento de la familia Office.
- Objeto ***Workbook*** se refiere a los distintos libros abiertos dentro de la aplicación Excel *depende* por tanto del objeto *Application*.
- Objeto ***WorkSheet*** es el conjunto de hojas de un libro, *depende* de un *Workbook*.
- Objeto ***Range*** se refiere a una celda o un rango de celdas. Normalmente *depende* de un objeto *WorkSheet*

En resumen un objeto Application puede contener varios libros (Workbooks), que contienen hojas (WorkSheets) que a su vez contienen otros objetos (por ejemplo Rangos).



2.5 Los objetos Excel. Clases y Jerarquía.

2.5.1 Introducción.

Retomando los visto hasta ahora podemos señalar que Excel ofrece más de 100 clases de objetos entre ellos tenemos como tipos:

- Libros
- Hojas
- Rangos
- Gráficos
- Un triangulo dibujado
- Un botón, etc..

Los objetos se manipulan con código VBA

2.5.2 Clasificación de los objetos Excel (Clases/Colecciones)

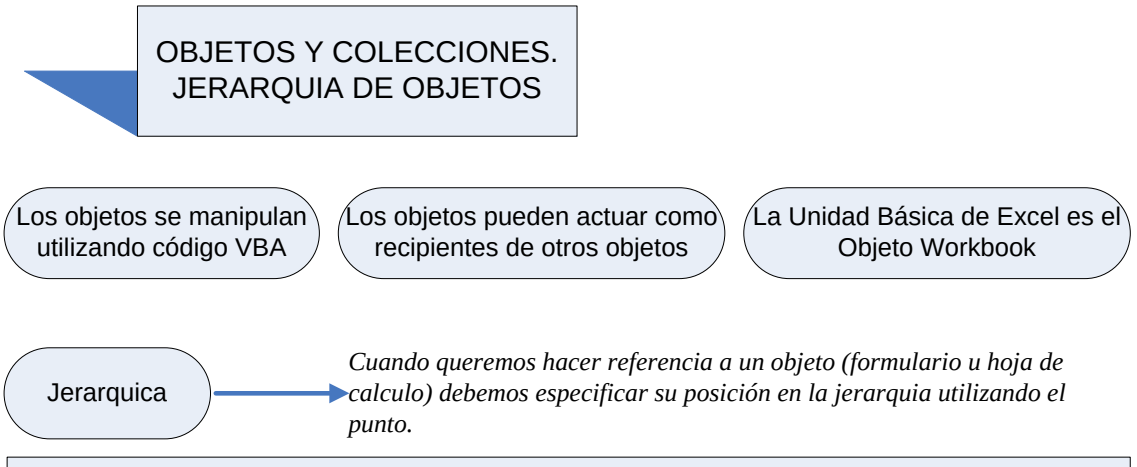
Los objetos Excel se clasifican en Clases y estas se ordenan en Jerarquias

Las Colecciones. → *Son el conjunto de objetos similares. Asi, una colección es un grupo de objetos de la misma clase y esta colección por si misma es un objeto.*

Por ejemplo:
Workbook es una colección de todos los objetos workbooks que en ese momento estan abiertos.
El objeto Worksheets (hoja de calculo) se compone de todas las hojas de calculo que forman el libro, así este conjunto de hojas de calculo forman una colección del objeto Worksheets.
*Para hacer **referencia a un único objeto de una colección**, colocamos el nombre del objeto o el número del indice entre parentesis despues del nombre de la colección, como en este caso:*
Worksheets ("hoja1")
Si Hoja1 es la primera hoja de la colección, tambien podemos utilizar la siguiente referencia
Worksheets (1) y a la segunda hoja seria Worksheets (2)
Tambien hay una colección especial llamada sheets que esta compuesta por todas las hojas de un libro, tanto si son graficos, hojas de calculo o no. Sheets (1)

2.5.3 Las Jerarquias.

Las objetos Excel como hemos visto se clasifican en Clases y estas se ordenan en Jerarquias.



Por ejemplo:

Podemos hacer referencia a un libro de excel llamado primas.xls de la siguiente forma:

Application. Workbooks ("primas.xls")

Esto hace referencia al libro primas.xls de la colección workbooks que se encuentra dentro del objeto Application de Excel.

Podemos también hacer referencia a la hoja 1 del lbro primas.xls de la siguiente forma:

Application. Workbooks ("primas.xls"). Worksheets ("hoja 1")

Avanzando podemos llevarlo a un nivel de desarrollo más detallado y referirnos a la celda A1 de la hoja anterior, de la siguiente forma:

Application. Workbooks ("primas.xls"). Worksheets ("hoja 1"). Range ("A1")

Si se omiten una referencia especifica a un objeto excel utiliza los objetos activos. Si el libro activo es primas.xls, la referencia anterior se puede hacer más simple como:

Worksheets ("hoja 1"). Range ("A1")

2.5.4 Como hacer referencia a los objetos. Primera aproximación.

Cuando se quiere hacer referencia a un objeto miembro o contenido, debemos especificar su posición en la jerarquía de objetos utilizando un punto para separar los contenedores de los miembros.

Por ejemplo, podemos hacer referencia a un libro llamado Libro1.xlsx de las siguientes formas:

* Application.Workbooks("Libro1.xlsx")
Esto hace referencia al libro 1.xlsx de la colección Workbooks. La colección Workbooks se encuentra dentro del objeto Application de Excel. Llevando esto a otro nivel, puedes hacer referencia a la Hoja1 del Libro1 así:
Application.Workbooks("Libro1.xlsx").Worksheets("Hoja1")
Todavía se puede llevar a otro nivel más y referirse a celdas específicas como se indica a continuación:
Application.Workbooks("Libro1.xlsx").Worksheets("Hoja1").Range("A1")
* Objetos activos: Si omites una referencia específica a un objeto, Excel utiliza los objetos activos. Si el libro activo es Libro1, la referencia anterior se puede hacer más simple:
Worksheets ("Hoja1").Range("A1")
Si sabes que la hoja active es Hoja1, puedes simplificar la referencia todavía más:
Range("A1")

Cuando queremos hacer referencia a un objeto (formulario u hoja de calculo) debemos especificar el tipo de objeto uniendolo al nombre del objeto con un punto. En el caso de

que tengamos dos libros abiertos a la vez y ambos tienen una hoja llamada hoja1, la solución seria añadir la referencia al nombre del objeto

Por ejemplo:

Workbooks ("libro1").Worksheets ("Hoja1")

Para hacer referencia a un rango determinado por ejemplo a A1seria de la siguiente forma

Workbooks ("libro1").Worksheets ("Hoja1"). Range (A1)

En el caso de que el libro 1 sea el unico abierto y la hoja activa sea la hoja 1, la referencia anterior se puede simplificar asi

Rage ("A1")

Hacer referencias a los objetos (como en estos ejemplos) no sirve de nada, para realizar algo con sentido, debemos leer o modificar las propiedades de los objetos o especificar un metodo para especificarlo con un objeto.

Puede haber muchas formas diferentes de hacer referencia al mismo objeto

Caso:

Supongamos que tenemos un libro llamado Ventas y es el único libro abierto. Este libro contiene una hoja llamada Resumen. Podemos hacer referencia a la hoja de cualquiera de las siguientes formas:

Workbooks ("Ventas.xls"). Worksheets ("Resumen")

Workbooks (1). Worksheets (1)

Workbooks (1). Sheets (1)

Application. ActiveWorkbooks.ActiveSheet

ActiveWorkbooks.ActiveSheet

ActiveSheet

El metodo que utilizaremos estara condicionado normalmente por nuestro conocimiento del espacio de trabajo. Por ejemplo si hay mas de un libro abierto, el segundo y tercer metodo no son fiables. Si queremos trabajar con la activa (la que sea), ninguno de los ultimos tres metodos funcionara.

Para estar absolutamente seguros de que estamos haciendo referencia a una hoja especifica de un determinado libro, el primer metodo es la mejor opcion.

2.5.5 Objetos y sus propiedades más comunes en Excel.

En Excel solo podemos tener activo a la vez un libro, una hoja, una celda o un rango y por tanto podemos hacer referencia a estos objetos activos de una forma simple, a traves de las propiedades del objeto application.

Principales Propiedades del objeto Application.

Activecell	Activechart	ActiveSheet	ActiveWorkbook	Selecction
Hace referencia a la celda activa	Hace referencia al gráfico activo	Hace referencia a la hoja activa	Hace referencia al libro activo	Hace referencia al objeto seleccionado

EJEMPLOS

ActiveCell.ClearContents

Esta instrucción elimina el contenido de la celda activa

ActiveCell.Value =1

Esta instrucción asigna el valor 1 a la celda activa

MsgBox ActiveSheet.Name

Esta instrucción muestra un mensaje con el nombre de la celda activa

MsgBox ActiveWorkbook.Name

Esta instrucción muestra un mensaje con el nombre del libro Activo

Selection.Value =12

Esta instrucción rellena el rango seleccionado con el valor 12

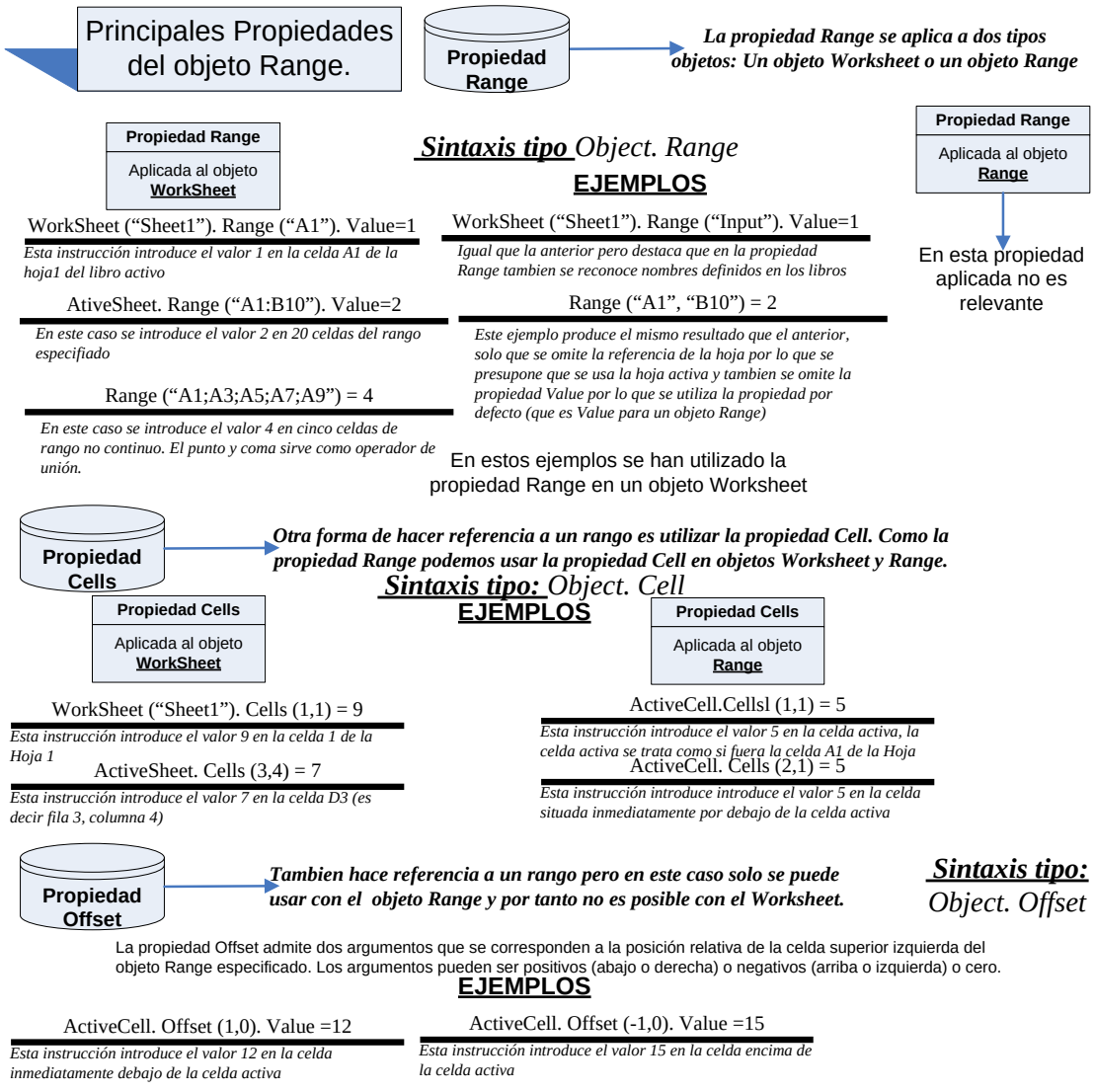
Observaciones:

La ventaja de utilizar estas propiedades es que no hace falta saber que celda, hoja o libro esta activo, ni proporcionar una referencia especifica para él. Debemos observar que no se aplico la referencia al objeto Application porque se presupone

2.5.6 Objeto Range. Celdas y Rangos.

Concepto:

Gran parte del trabajo de Excel se realiza sobre celdas y rangos de las hojas. Asi el objeto Range esta contenido dentro de un objeto más amplio Worksheet y consta de una única celda o serie de celdas de una hoja simple.



2.5.7 Resumen-Esquema

Objetos-Colecciones-Jerarquía

Objetos:
VBA manipula los objetos que se encuentran en su aplicación huésped (en este caso, Excel es la aplicación huésped). Excel te ofrece más de 100 clases de objetos para manipular, entre los que se encuentran los libros, las hojas, rangos de hojas, gráficos o un triángulo dibujado. Hay muchos más objetos a disposición del programador y puedes manipularlos utilizando el código VBA. Las clases de objetos están ordenadas por jerarquías.

Los objetos pueden actuar como recipientes de otros objetos. Por ejemplo, Excel es un objeto llamado Application y contiene otros objetos como Workbook y CommandBar. El objeto Workbook puede contener otros objetos como Worksheet y Chart. Un objeto Worksheet puede contener objetos como Range, PivotTable y así sucesivamente. La disposición de estos objetos recibe el nombre de modelo de objetos de Excel.

Colecciones:
Los objetos similares forman una colección. Por ejemplo, la colección Worksheet se compone de todas las hojas de un determinado libro. La colección CommandBars está formada por todos los objetos CommandBar. Las colecciones son objetos por sí mismos

Jerarquía de Objetos:
Cuando haces referencia a un objeto miembro o contenido, especificas su posición en la jerarquía de objetos utilizando un punto para separar los contenedores de los miembros. Por ejemplo, puedes hacer referencia a un libro llamado Libro1.xlsx así:

* Application.Workbooks("Libro1.xlsx")
Esto hace referencia al libro 1.xlsx de la colección Workbooks. La colección Workbooks se encuentra dentro del objeto Application de Excel. Llevando esto a otro nivel, puedes hacer referencia a la Hoja1 del Libro1 así:
Application.Workbooks("Libro1.xlsx").Worksheets("Hoja1")
Todavía se puede llevar a otro nivel más y referirse a celdas específicas como se indica a continuación:
Application.Workbooks("Libro1.xlsx").Worksheets("Hoja1").Range("A1")
* Objetos activos: Si omites una referencia específica a un objeto, Excel utiliza los objetos activos. Si el libro activo es Libro1, la referencia anterior se puede hacer más simple:
Worksheets ("Hoja1").Range("A1")
Si sabes que la hoja active es Hoja1, puedes simplificar la referencia todavía más:
Range("A1")

2.6 Propiedades, Métodos y Eventos de los Objetos.

3 Entender los Principales Objetos y Miembros (Métodos y Propiedades)

3.1.1 Introducción.

Hemos de tener en cuenta que los objetos tienen métodos o actividades y propiedades. A esta combinación de métodos y propiedades se le conoce en el entorno VB de Excel como Miembros del Objeto.
Así podemos preguntar ¿cuáles son los métodos y propiedades del objeto Hoja de cálculo?, sería igual que decir ¿Cuáles son los miembros del objeto Hoja de cálculo?.

Todos los objetos tienen tres componentes

Propiedades:
Caracteriza al objeto.
* Nombre Interno.
* Titulo
*Color
* Posición
* Tamaño

Métodos:
Son procedimientos que el usuario ordena al objeto.
* Mostrar en pantalla
* Cerrar en pantalla

Eventos:
Son ocurrencias sobre el objeto
* Mover ratón
* Pulsar Tecla

Se desarrollan en las subrutinas o gestores de eventos SUB

3.1.2 Propiedades.

Propiedades de Objetos:

Los objetos tienen propiedades.
Podemos imaginar una propiedad como un valor de un objeto. Por ejemplo, un objeto de rango tiene propiedades como Value y Name. Un objeto de gráfico tiene propiedades como HasTitle y Type. Puedes utilizar VBA para determinar las propiedades de un objeto y también para cambiarlas.
Hacemos referencia a las propiedades combinando el objeto con la propiedad, separados por un punto. Por ejemplo, podemos hacer referencia al valor de la celda A1 de la Hoja1 como se indica a continuación:
Worksheets ("Hoja1").Range("A1").Value

Propiedades de Objetos

Por ejemplo:
Para el objeto Range (Rango) tiene propiedades como Value y Name
Si quisieramos cambiar el valor de una celda tendriamos lo siguiente:
Worksheet ("Hoja1"). Range ("A1").Value =123

3.1.3 Métodos

Métodos Objetos:

Los objetos tienen metodos.
Un método es una acción que se realiza sobre el objeto. Por ejemplo, uno de los métodos para un objeto Range es ClearContents. Este método borra los contenidos del rango. Los métodos se especifican combinando el objeto con el método, separados por un punto. Por ejemplo, para borrar los contenidos de la celda A1 de la hoja activa utiliza: Range("A1").ClearContents

Métodos de Objetos

Los objetos también tienen metodos. Un metodo es una acción que se realiza sobre el objeto.

Por ejemplo:
Para el objeto Range (Rango) un metodo seria ClearContents, este metodo borra el contenido del rango. ***Los metodos se especifican combinando el objeto con el método separado por un punto****.*
Asi, para borrar los contenidos de la celda A1 de una hoja activa, tendriamos la siguiete instrucción:
Range ("A1"). ClearContents

3.1.4 Ejemplo de Propiedades y Métodos de un Objeto

Ejemplo Objeto Comment (Comentario) para entender las propiedades y métodos:
El objeto comment se crea a traves del menú Insertar-Comentario. Permite insertar un comentario en una celda.

Propiedades del Objeto Comment:
El objeto comment tiene 6 propiedades:
Application: Devuelve el objeto que creo la aplicación, en nuestro caso Excel
Autor: Devuelve el nombre de la persona que creo el comentario.
Creator: No relevante
Pater: Devuelve el objeto padre del cometario, siempre es un rago
Shape: Hace referencia a la forma del cometario
Visible: Si el comentario es visible

Métodos del Objeto Comment:
A continuación se muestran los metodos u operaciones que podemos realizar al objeto comment:
Delete: Elimina un comentario
Next: Representa un objeto comment que representa al siguiente comentario.
Previus: nos muestra el comentario anterior
Text: Devuelve o establece el texto de un comentario.

Ejemplo La Colección Comment (Comentario):
Recordemos que una colección es un grupo de objetos similares. Cada hoja contiene una colección de Comments Si una hoja no tiene comentarios, la colección esta vacia.
Worksheets ("Hoja1"). Comments (1). Este código hace referencia al primer comentario de la Hoja1.
Msgbox ActiveSheet. Comments. Count. Muestra el numero total de comentarios de la hoja activa.
Msgbox Activesheet.Comments (1). Parent. Address. Muestra en que celda esta el 1º comentario de la hoja.
Range ("A1"). Comment Hace referencia al comentario de la celda A1 de la hoja activa.
Comments (1). Se hace referencia al primer comentario de la hoja 1.

Resumen:
Los objetos tienen propiedades y metodos unicos, cada objeto tiene su propio conjunto de propiedades y métodos. Algunos objetos sin embargo comparten propiedades (por ejemplo, Name) y algunos métodos (como Delete).
La mayor parte del tiempo haremos referencia a un objeto de forma indirecta a traves de las colecciones en la que se encuentra. Por ejemplo para acceder a un objeto workbook llamado jose deberemos hacer referencia a la colección workbook: Workbooks ("jose.xls")

3.1.5 Argumentos de Propiedades y Metodos

Argumentos de Propiedades y Métodos

Es facil verse abrumado por las propiedades y metodos disponibles, hay cientos disponibles.

Algunos métodos y propiedades utilizan argumentos para clarificar más la acción que tienen que realizar.

3.1.6 Eventos de los Objetos (cosas que le pasa al objeto)

Como hemos dicho a cada objeto le corresponden unos eventos (cosas que le pasan al objeto).

Un evento es una acción que puede ser reconocida por un objeto. Cuando se produce un evento, se ejecuta el código asociado al mismo, si lo hay. Ejemplos de eventos podrían ser: al abrir un libro, al imprimir, al seleccionar una hoja, al cambiar la celda seleccionada, etc.

Eventos: Son ocurrencias sobre el objeto. Es una acción que puede ser reconocida por un objeto.

* Mover ratón * Pulsar Tecla

Se desarrollan en las subrutinas o gestores de eventos SUB. Cuando se produce un evento, se ejecuta el código asociado al mismo, si lo hay.

Ejemplos de Eventos:
* al abrir un libro,
* al imprimir
* al seleccionar una hoja
* al cambiar la celda seleccionada, etc.

Tipos de Eventos: En Excel hay muchos tipos de eventos.
* Los eventos de libros. (ThisWorkbook)
* Los eventos de hojas de cálculo
* Los eventos de hojas de gráfico, etc.

Sobre el código de los Eventos:

El código para un evento que queramos usar se tiene que situar en el módulo del objeto en el que se produce el evento. Si es el libro, en su módulo (ThisWorkbook), y si es una hoja de cálculo también en su módulo, cuyo nombre es el de la hoja.

Es conveniente dejar que sea Excel el que se encargue de crear el cuerpo del código del evento porque casi todos ellos tienen uno o varios parámetros que han de figurar necesariamente en su declaración. Por lo tanto, para crear un evento lo mejor es entrar en el editor de VBA (Alt + F11), hacer doble click en el objeto para el que se necesita crear el evento (ThisWorkbook o el módulo de una hoja) y cambiar la selección del cuadro de diálogo superior izquierdo de (General) a Workbook o Worksheet, según corresponda. Por defecto Excel creará el evento Workbook_Open si se trata de ThisWorkbook o el evento Worksheet_SelectionChange si se trata del módulo de una hoja, pero si el evento creado no es el que interesa no hay más que borrarlo y seleccionar en el desplegable superior derecho el que se necesite.

TIPOS DE EVENTOS.

En Excel hay muchos tipos de eventos. Los eventos de libros, hojas de cálculo y hojas de gráfico pueden usarse simplemente escribiendo el código necesario en su módulo correspondiente (ThisWorkbook para los libros). Los eventos para Application, gráficos incrustados y QueryTables requieren la creación de un objeto con eventos, lo que ha de hacerse desde un módulo de clase.

En www.excelsp.com/eventos01.htm hay algunos ejemplos de eventos.

¿Dónde se sitúa el código perteneciente a un evento?

El código para un evento que queramos usar se tiene que situar en el módulo del objeto en el que se produce el evento. Si es el libro, en su módulo (ThisWorkbook), y si es una hoja de cálculo también en su módulo, cuyo nombre es el de la hoja.

Es conveniente dejar que sea Excel el que se encargue de crear el cuerpo del código del evento porque casi todos ellos tienen uno o varios parámetros que han de figurar necesariamente en su declaración. Por lo tanto, para crear un evento lo mejor es entrar en el editor de VBA (Alt + F11), hacer doble click en el objeto para el que se necesita crear el evento (ThisWorkbook o el módulo de una hoja) y cambiar la selección del cuadro de diálogo superior izquierdo de (General) a Workbook o Worksheet, según corresponda. Por defecto Excel creará el evento Workbook_Open si se trata de ThisWorkbook o el evento Worksheet_SelectionChange si se trata del módulo de una hoja, pero si el evento creado no es el que interesa no hay más que borrarlo y seleccionar en el desplegable superior derecho el que se necesite.

3.2 Resumen. Puntos clave.

Aunque no lo creas, esto describe casi por completo VBA. A partir de ahora sólo aprenderemos los detalles para manejar todo esto.

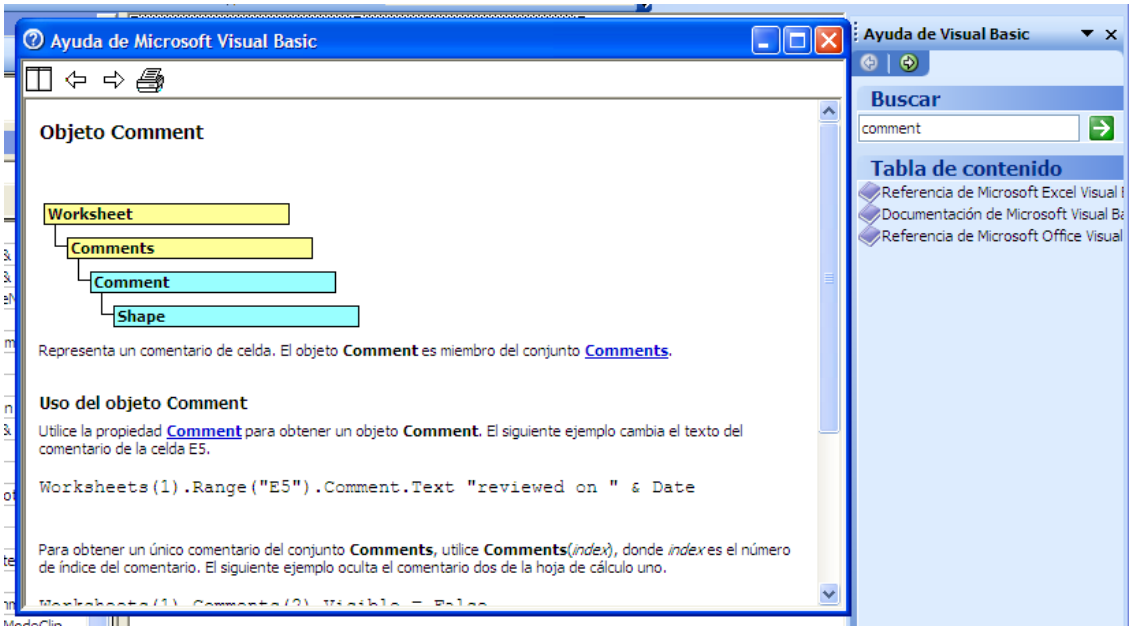
http://www.ayudaexcel.com/excel/es/excel-macros-vba/objetos-propiedades-metodos

4 Uso de la Ayuda

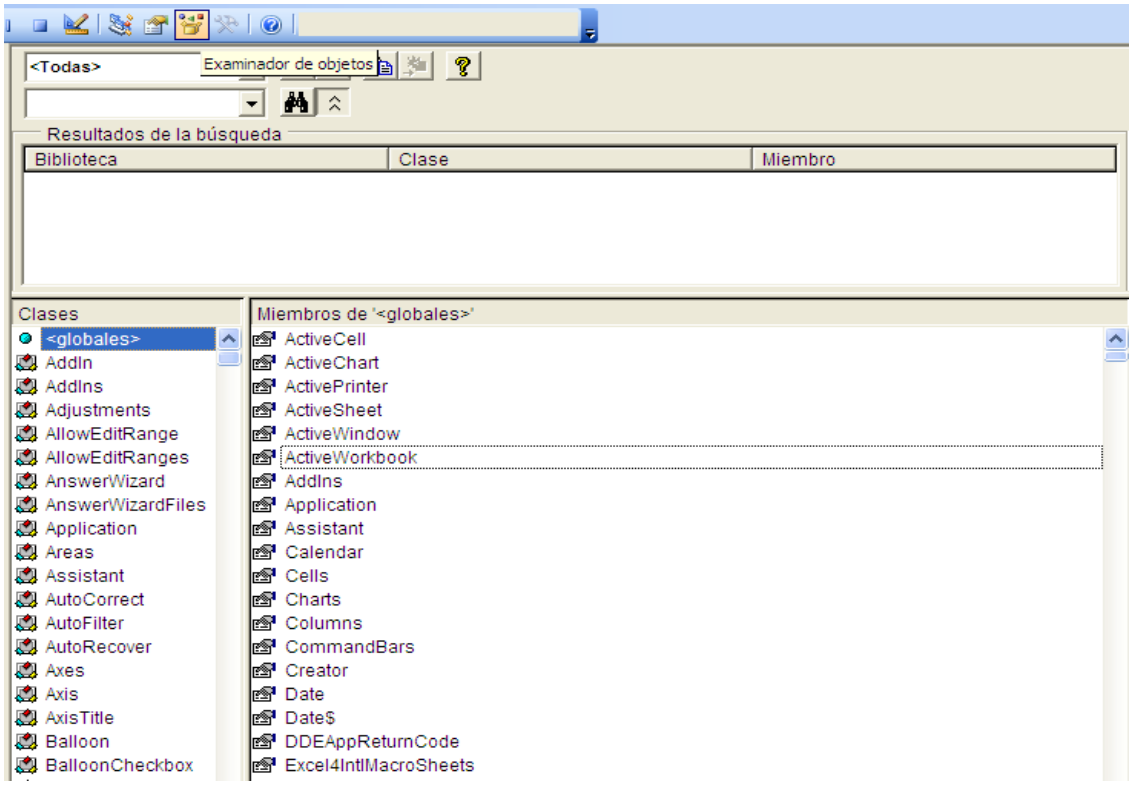
4.1 Introducción

Un modo de conocer un determinado objeto es a través de la ayuda. A continuación presentamos la ayuda del objeto Comment, escribiendo en la ayuda esta palabra, más fácil aun es escribir la palabra del método o del objeto el editor y pulsar F1

La principal fuente de información detallada sobre objetos, metodos y procedimientos de Excel es el sistema de ayuda.



4.2 El Examinador de Objetos



El Examinador de Objetos es una herramienta que enumera todas las propiedades y métodos de cada objeto disponible. La lista desplegable de la esquina superior izquierda del examinador de objetos incluye una lista de todas las bibliotecas de objetos a las que se tiene acceso:

* El propio Excel.
*Ms Forms utilizados para crear cuadros de dialogo personalizados
* Office Objetos comunes a todas las aplicaciones Microsoft Office
* VBA
* El proyecto actual

El Examinador de Objetos es una herramienta que enumera todas las propiedades y metodos de cada objeto disponible. La lista desplegable de la esquina superior izquierda del examinador de objetos incluye una lista de todas las bibliotecas de objetos a las que se tiene acceso:

* El propio Excel.
*Ms Forms utilizados para crear cuadros de dialogo personalizados
* Office Objetos comunes a todas las aplicaciones Microsoft Office
* VBA
* El proyecto actual

Lo que seleccionemos en la lista desplegable determinara lo que aparecerá en la ventana de clases y lo que seleccionemos en esta ventana determinara lo que será visible en la ventana miembros. Tras seleccionar una biblioteca podemos buscar una cadena de texto

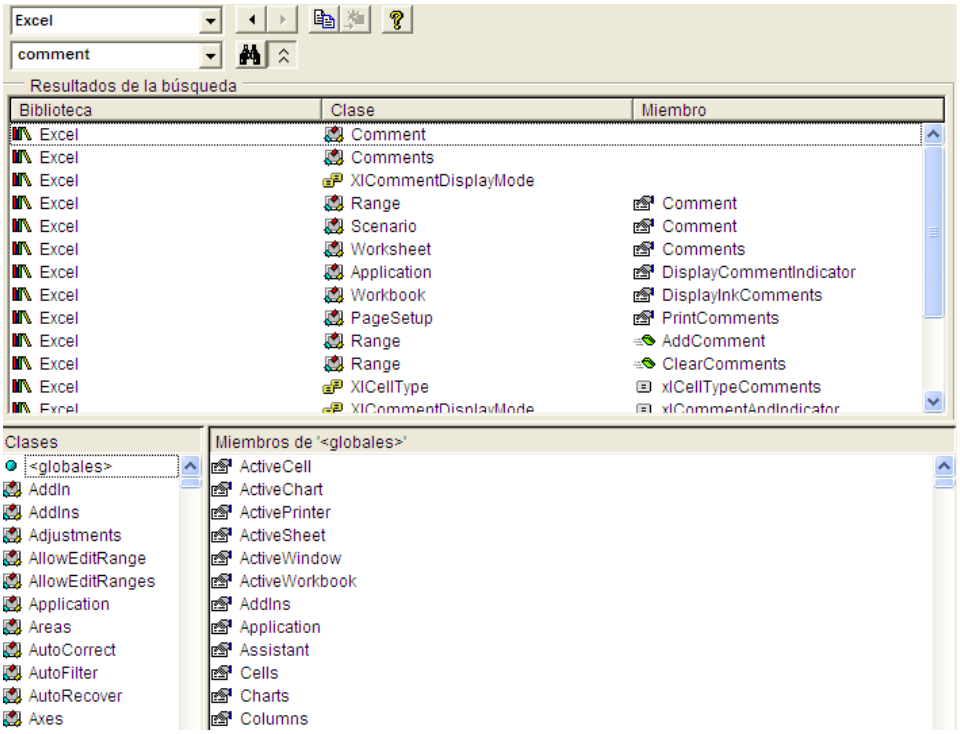
en particular para obtener una lista de las propiedades y metodos que contiene el texto. Esto se consigue introduciendo en la segunda lista desplegable y haciendo click en el icono de buscar. Por ejemplo, supongamos que estamos trabajando con un proyecto que manipula comentarios de celdas:

1) Seleccionamos la biblioteca que nos interesa, si no estamos seguros de que biblioteca seleccionamos Todas.

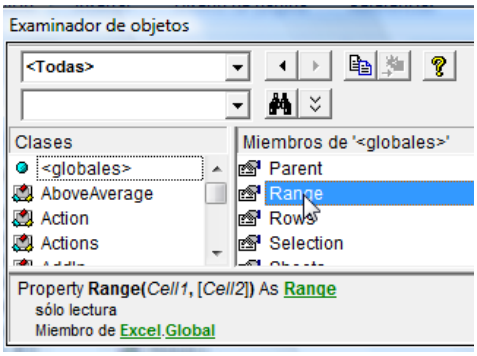
2) Escribiremos Comment en la lista desplegable

3) y pulsando los primaticos comienza la búsqueda

La ventana Resultados mostrara sus miembros (propiedades, métodos y constantes). En el panel inferior se muestra más información acerca del objeto. Podemos pulsa F1 para ir directamente al tema de ayuda adecuado.



En el examinador de objetos podemos ver básicamente dos listas: la de la izquierda con los nombres de las clases de objetos; y la de la derecha con los miembros (o métodos y propiedades) de la clase de objeto seleccionada. Al principio de la lista Clases aparece una clase de objeto muy especial, llamada <<globales>>.



En realidad esta no es una clase de objeto, pero dentro de su lista de miembros están todos los métodos y propiedades que se pueden utilizar sin especificar un objeto.
Así si seleccionamos en la lista de objetos la propiedad Range en el recuadro, al final del examinador aparece información sobre la propiedad Range. Esta propiedad tiene dos argumentos, y los corchetes del segundo indican que es opcional. La propiedad Range devuelve una referencia a un objeto Range.

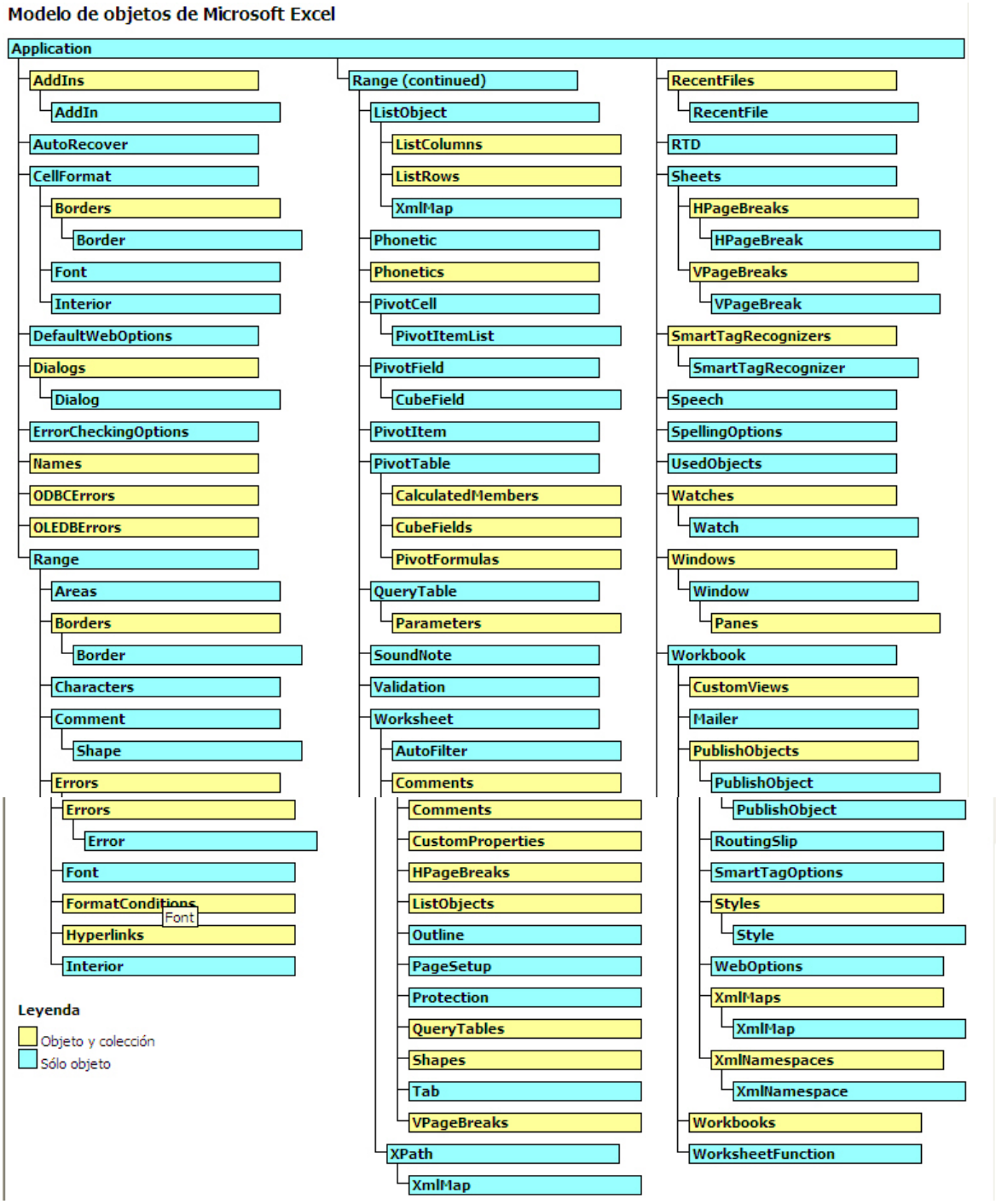
4.3 La Ventana Inmediato

La ventana Inmediato de Visual Basic es muy práctica para probar instrucciones y expresiones VBA. Generalmente se mantiene abierta para probar expresiones y depurar el código. **Ctrl+G**

También señalar que esta ventana de inmediato del editor VBA es una poderosa herramienta que Excel pone a disposición para explorar objetos propiedades y métodos. Con ella podemos ejecutar instrucciones VBA, teniendo en cuenta que todo lo que escribamos en la ventana Inmediato desaparecerá tan pronto cerremos la aplicación Excel.

La ventaja de escribir instrucciones en la ventana Inmediato es que vemos el resultado inmediatamente. Esta ventana es muy útil, sobre todo cuando queremos saber para qué sirve una instrucción. Por ejemplo, podemos usarla para crear una nueva macro o para depurar una ya existente.

4.4 Esquema del Modelo de Objetos de Microsoft Excel.



5 El Objeto WorkBook (Libro Excel). Características y Principales Miembros (Métodos y Propiedades).

5.1 Introducción. Aspectos Generales

Hemos de tener en cuenta que los objetos tienen métodos o actividades y propiedades. A esta combinación de métodos y propiedades se le conoce en el entorno VB de Excel como Miembros del Objeto.

Así podemos preguntar ¿cuáles son los métodos y propiedades del objeto Hoja de cálculo?, sería igual que decir ¿Cuáles son los miembros del objeto Hoja de cálculo?.

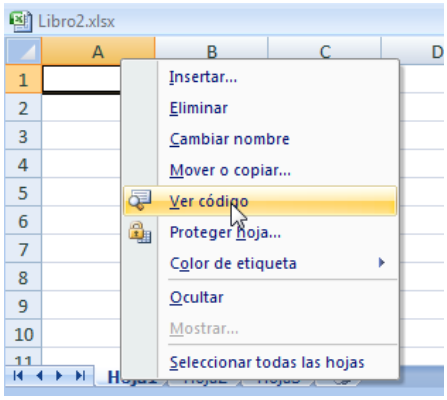
5.2 Objeto-Método. Entendiendo el objeto Libro Excel y sus métodos. Caso Método Add y Close

Vamos a continuación a analizar el Metodo Add de la colección de objetos Workbooks.

Recordemos, como hemos comentado anteriormente que la ventana de inmediato del editor VBA es una poderosa herramienta que Excel pone a disposición para explorar objetos propiedades y métodos. Con ella podemos ejecutar instrucciones VBA, teniendo en cuenta que todo lo que escribamos en la ventana Inmediato desaparecerá tan pronto cerremos la aplicación Excel.

Así vamos a analizar sintéticamente el objeto Libro Excel y sus miembros, los libros son el mayor nivel estructural de Excel como es evidente de entender para ello comenzaremos realizando los siguientes pasos:

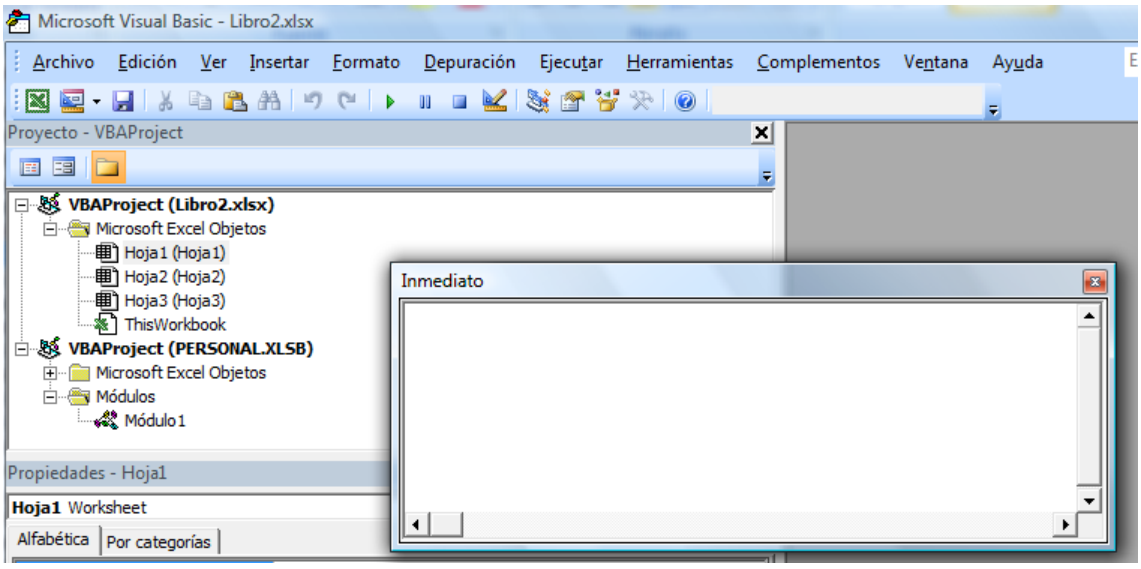
1. Abrimos la aplicación Excel y la ventana Inmediato.



Para ello en primer lugar abrimos una libro Excel y situados sobre cualquier hoja con el botón derecho seleccionamos la opción ver código lo que nos lleva al editor VBA de Excel.

Una vez ahí basta con abrir la ventana inmediato con Ctrl+G o con la opción Ver Ventana Inmediato.

De esta forma tenemos a disposición la ventana de inmediato dentro del editor VBA.

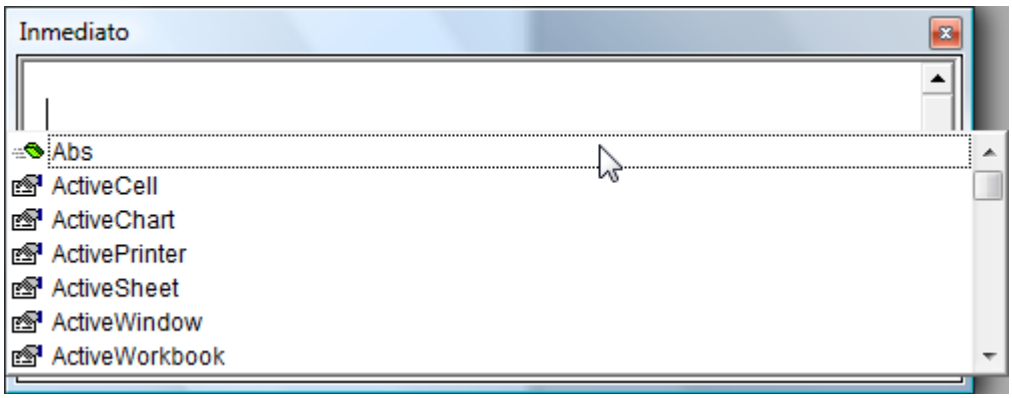


2. Activar Palabra Completa para Ventana Inmediato.

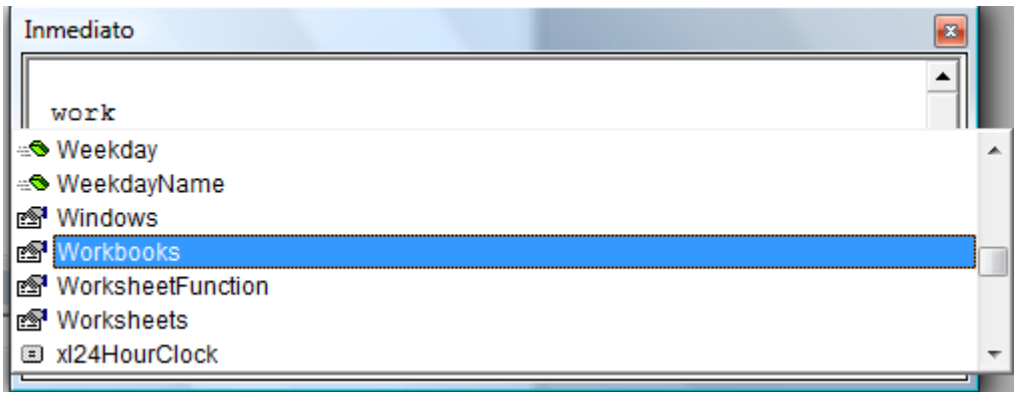
   Ya con la ventana de Inmediato abierta para facilitar nuestro trabajo, vamos a activar la opción Edición Palabra completa que nos permitirá seleccionar los

objetos según tecleemos las primeras letras de su identificación. Con ello aparece una lista de métodos y propiedades. Tambien podemos activar esta propiedad a través de Ctrl+espacio.

De esta forma obtendremos las siguiente opciones.

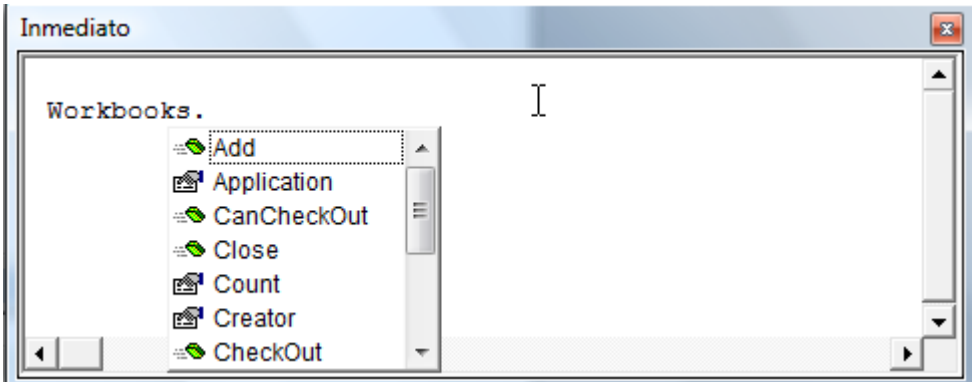


Ahora si vamos escribiendo la palabra de interés del objeto nos intentara localizarla tal y como vemos en la siguiente ilustración.



3. Seleccionar los miembros (Métodos o Propiedades) asociados al objeto.

Una vez seleccionado el objeto, en nuestro caso Workbooks toca determinar o asociar el miembro (Método o Actividad o Propiedad) deseado. Para ello ponemos al final del nombre del objeto un punto (.) y nos ofrecera todos los miembros disponibles para ese objeto, tal y como se muestra a continuación



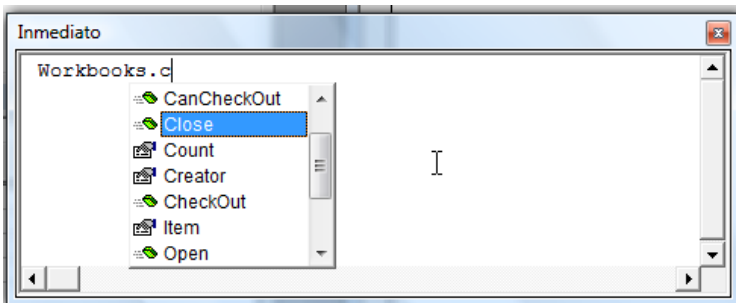
En nuestro caso vamos a seleccionar el método o la actividad Add (añadir) un nuevo libro.

Una vez escrita toda la instrucción en la ventana Inmediato y al pulsar la tecla intro se habrá añadido un nuevo libro.

La ventaja de escribir instrucciones en la ventana Inmediato es que vemos el resultado inmediatamente.

Ya vimos anteriormente que el método Add de la colección de objetos Workbooks añadia nuevos elementos a la colección.

Otro método similar asociado al objeto Workbooks es Close con el cual nos ayuda a cerrar todas las colecciones, tal y como vemos a continuación



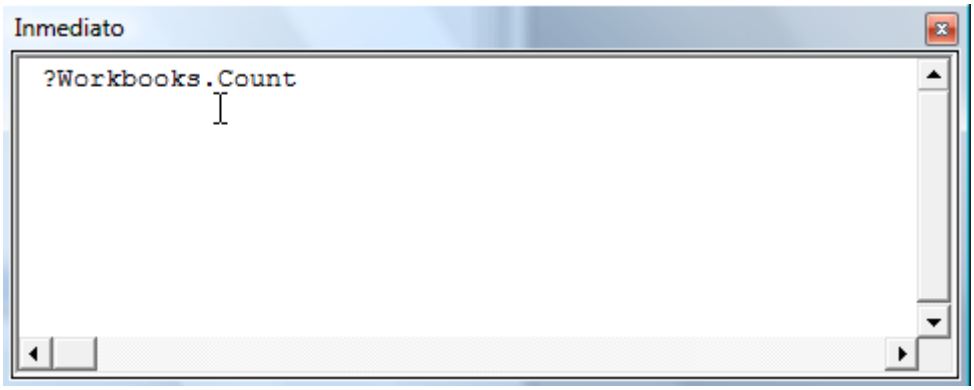
Es decir, de esta forma cerramos todos los libros (workbooks) abiertos

5.3 Objeto-Propiedad. Entendiendo el objeto Libro Excel y sus propiedades. Caso Propiedad Count.

En este segundo caso procederemos igual que anteriormente solo que vamos a trabajar con propiedades en concreto vamos a utilizar la propiedad Count (Contar) del objeto WorkBooks, de esta forma nos indicara la cantidad de elementos que hay en la colección.

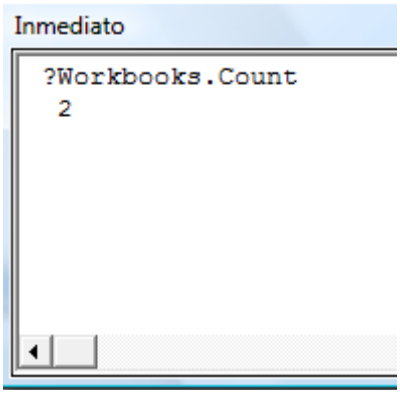
Para su uso procederemos de forma similar que como realizamos anteriormente, en primer lugar en la ventana de Inmediato, insertamos en primer lugar el símbolo interrogación, posteriormente utilizando el comando Palabra Completa y seleccionamos el objeto Workbooks asociándolo con el punto (.) y a través de la lista automática seleccionamos la propiedad Count tal y como se muestra en la siguiente ilustración.

Asi Count es una propiedad y para atarla al objeto que le corresponde debemos utilizar un punto igual que hacíamos para los métodos o acciones.



Cuando se ejecuta el método Add no necesitamos anteponer un signo de interrogación porque este es un método que no restituye valores. Por el contrario cuando utilizamos la propiedad Count (Contar) es porque queremos saber el valor de la propiedad

Al ejecutar esta instrucción nos dará el número de libros abiertos.



Señalar que en la ventana Inmediato cuando escribe un signo de interrogación acompañado de cualquier elemento que devuelve un valor, el valor aparece en la siguiente línea.

A través del signo de interrogación puede visualizarse el valor de la propiedad Count.

5.4 Objeto.Referencia. Miembros (método/propiedad). Hacer referencia a un solo libro (Item o "nombre"). ActiveWorkbook

Retomando el caso visto anteriormente sobre el método close del objeto WorkBook puede surgir la idea de cerrar todos los libros abiertos sino uno/s concreto/s.

Por tanto será necesario aplicar el método al conjunto de objetos deseados lo que implica hacer referencia al objeto/s deseado/s.

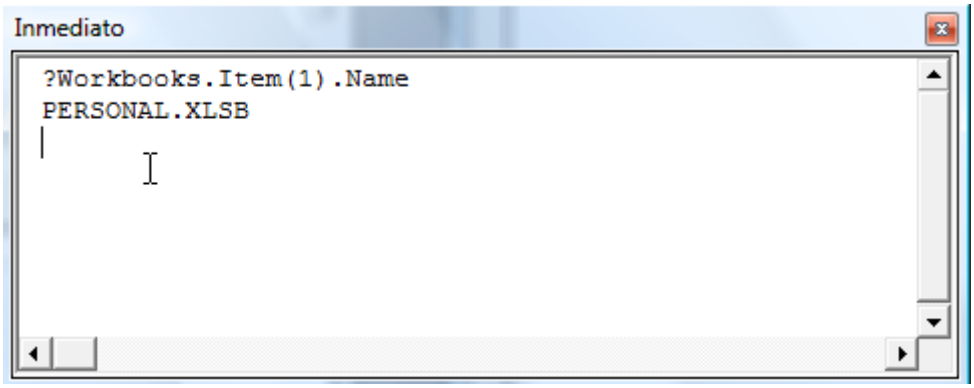
Así supongamos que tenemos tres libros abiertos, ejecutamos como hemos visto la ventana inmediato y establecemos la sentencia para que nos muestre el nombre del primer libro abierto, es decir queremos hacer referencia al primero de los libros abiertos, para ello ejecutaremos la siguiente sentencia:

?Workbooks.Item(1).Name

Si leemos esta instrucción de derecha a izquierda signifcaria algo así como ¿El nombre del primer elemento de la colección Workbooks es?. Analicemos la sintaxis de la instrucción:

- Name: Es una propiedad para un solo libro. Como name es una propiedad que puede dar un valor ponemos el signo de interrogación al de la instrucción. Esta propiedad no es válida para el objeto Workbooks, porque una colección de objetos no tiene nombre.
- Item. Es la propiedad de una colección y hace referencia a un único elemento de la colección.
- Workbooks cita a la colección de libros (el objeto workbooks). Una vez que se ha creado esta referencia, Item (1) la cambia para referirse a un elemento especifico de la colección (a un solo objeto libro). Despues de que este lista. Name nombra al objeto.

Finalmente si ejecutamos la instrucción en la ventana inmediato esto nos devolverá el resultado previsto



Esto nos devolverá el nombre del primer libro del nuevo grupo de libros abiertos.

Destacar que estas dos instrucciones son similares:

?Workbooks.Item(1).Name
?Workbooks (1).Name (esta es la mas usada)

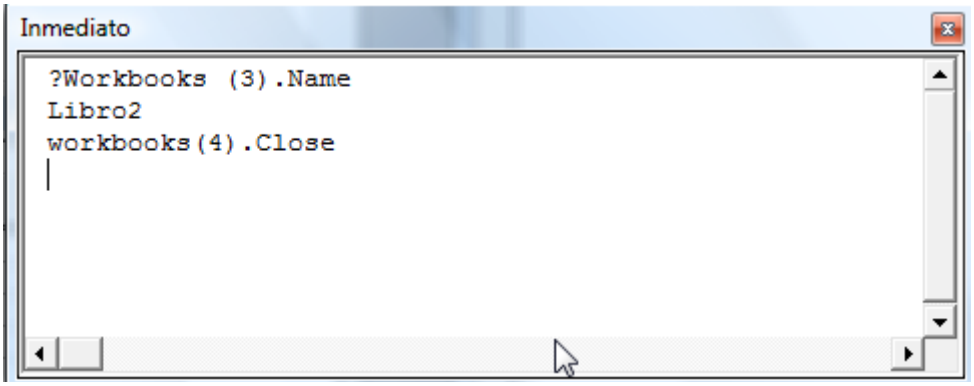
Así la propiedad Item especifica la posición que un libro ocupa en la colección Workbooks.

Siguiendo esta línea de estudio, si quisiéramos cerrar un libro concreto de los que tenemos abiertos el método a ejecutar, como hemos visto, es Close de la colección de objetos Workbooks, solo nos falta señalar el libro concreto sobre el que queremos aplicar la cita acción.

Para ello basta con ejecutar en la ventana inmediato la siguiente sentencia en su forma extendida o corta:

?Workbooks.Item(1).Close
?Workbooks (1).Close (esta es la más usada)

Analicemos las siguientes setencias:



En primer lugar queremos determinar el nombre del libro correspondiente al índice 3, como podemos observar esto nos ha devuelto Libro 2.

En la siguiente instrucción cerramos el libro que se corresponde al índice 4

En resumen, una vez que la propiedad Item ha hecho alusión a un solo objeto Workbook (libro), podemos comunicarnos con él a través de los métodos y propiedades que el entiende. Con la propiedad Name, podemos saber el nombre del libro y con el método Close podremos cerrarlo.

Otra forma de referirnos a un elemento concreto de una colección objeto como es workbooks es a través del nombre, en este caso cada vez que utilicemos el nombre

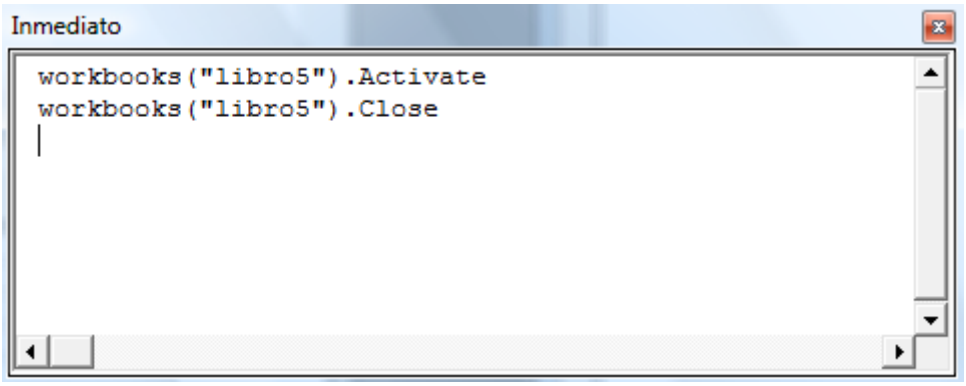
deberemos usarlo entre comillas, al contrario de lo que sucede cuando especificamos su posición.

Así por ejemplo si estamos en el libro 2 y quisiéramos poner como activo el libro 5 de la colección workbook bastaría con la siguiente setencia:

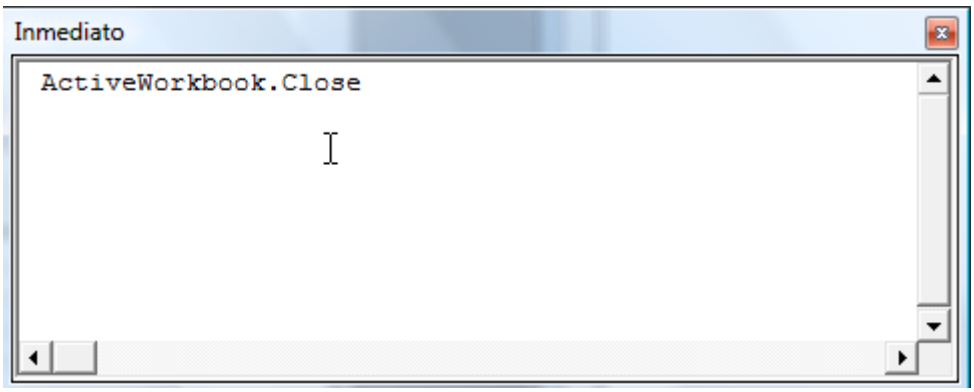
```
workbooks("libro5").Activate
```

Y si quisiéramos cerrar el libro 5 la instrucción sería:

```
workbooks("libro5").Close
```



Otra forma de hacer referencia a un libro es a través de la opción ActiveWorkbook, que significa libro activo, de esta forma si quisiéramos cerrar el libro activo simplemente tendríamos que poner la siguiente instrucción:



5.5 Ejemplos de Código relacionados la manipulación de los libros.

http://www.xltoday.net/vba_ejemplos_libros.asp

5.5.1 Cerrar libro Excel (guardar cambios)

ActiveWorkbook.CloseActiveWorkbook.Close
Savechanges:=TrueActiveWorkbook.Close(True)

5.5.2 Cerrar libro Excel (sin guardar cambios)

ActiveWorkbook.Close(False)ActiveWorkbook.Close Savechanges:=False

5.5.3 Cerrar libro Excel (variable, sin guardar cambios)

Application.DisplayAlerts = False
Windows(Libro_mayor).Close
Application.DisplayAlerts = True

5.5.4 Abrir libro Excel (ruta fija)

Workbooks.Open FileName:="C:\Trabajo\Informe.xls"

5.5.5 *Abrir un libro que está en el mismo directorio de trabajo*

```
Workbooks.Open (ThisWorkbook.Path & "\Notas.xlsm")
```

5.5.6 *Abrir libro Excel (diálogo)*

```
Msg = MsgBox("Elija archivo para abrir.", vbOKOnly, (""))
strArchivo = Application.GetOpenFilename
On Error GoTo 99
Workbooks.OpenText Filename:=strArchivo
If strArchivo = "" Then Exit Sub
strArchivo = ActiveWindow.Caption
99:
Exit sub
```

5.5.7 *Devolver nombre del libro Excel*

```
strNombre = ActiveSheet.Parent.FullNameMsgBox ActiveWorkbook.FullName
```

6 El Objeto WorkSheets (Hoja de Cálculo). Características y Principales Miembros (Métodos y Propiedades).

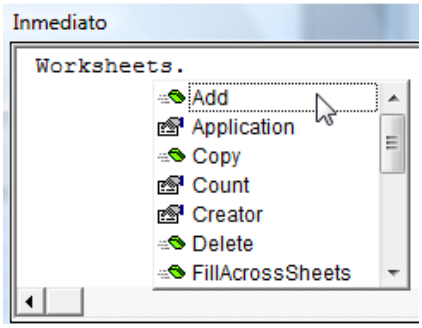
6.1 *Introducción. Aspectos Generales*

Una hoja de cálculo (worksheets) es un objeto con propiedades y métodos propios y puede ser utilizado para manipular hojas. A medida que trabajemos con las hojas de cálculo nos daremos cuenta que la lista automática de métodos y propiedades no siempre aparece después

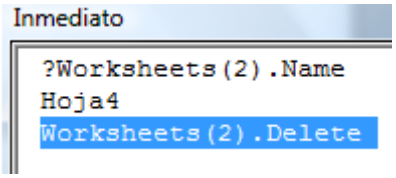
6.2 *Objeto-Método. Entendiendo el objeto hoja de cálculo y sus métodos. Caso Método Add , Delete, Copy y Select*

Al igual que los libros Excel, las hojas de cálculo presentan similitudes y también algunas diferencias respecto a los primeros.

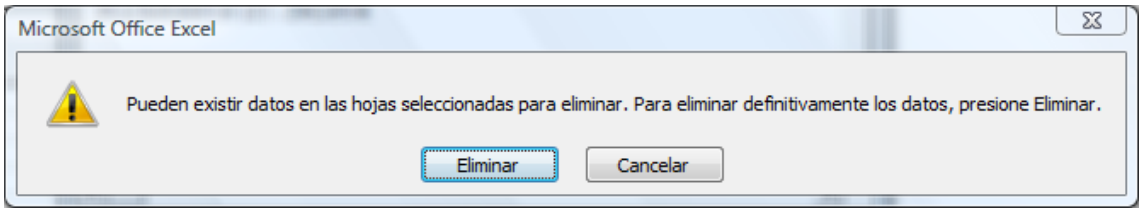
Así para añadir una nueva hoja de calculo al libro activo utilizaremos el método Add igual que hicimos con el estudio de los libros, es decir con la instrucción WorkSheets.Add.



Si quisiéramos borrar una determinada hoja tendremos que hacer referencia a la misma (tanto con el método corto o largo) y aplicar la acción Delete, tal y como mostramos en el siguiente ejemplo.



Esto borrara la Hoja 4 a través del Método Delete exponiendo previamente un aviso para evitar borrados accidentados.

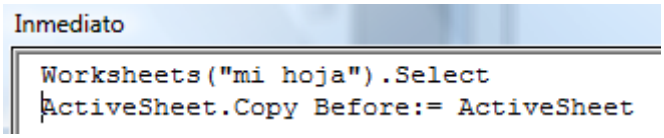


Copy es otro método muy útil para hacer una copia de una hoja para ello podemos utilizar la sentencia de la siguiente forma:

- Worksheet ("jggomez").copy
- ActiveSheet.Copy
- ActiveSheet.Copy Before:= ActiveSheet

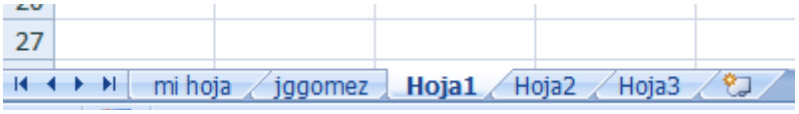
En el último caso copiamos la hoja active en el mismo libro y la posicionamos a continuación en el libro.

También debemos tener en cuenta la posibilidad de seleccionar una sola hoja, la sentencia seria:



Esto haría que la hoja activa sea la seleccionada y posteriormente hacemos una copia de la misma que situamos después de esta.

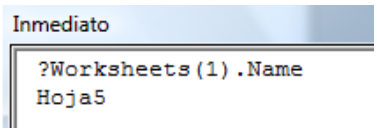
Así el estado del libro antes de ejecutar la sentencia anterior sería el que muestra la siguiente ilustración.



El resultado de ejecutar la sentencia anterior daría como resultado:

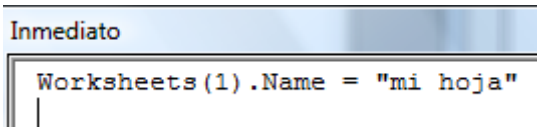
6.3 Objeto-Propiedad. Entendiendo el objeto Hoja de Cálculo y sus propiedades. Caso Propiedad Name, Activate

Si queremos ver el nombre de una hoja de calculo concreta, necesitaremos trabajar con la ventana inmediato y hacer referencia a la propiedad Name con el signo de interrogación previo, tal y como hemos visto anteriormente.



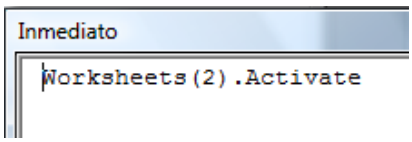
Podemos observar que de la misma manera que un objeto Workbook utiliza la propiedad Item (bien en su forma larga o corta) para referirse a un solo objeto Workbook (Libro), también la podemos utilizar en el objeto WorkSheet (Hoja). Solamente cuando hayamos establecido la referencia, podremos utilizar las propiedades del objeto Hoja con la propiedad Name, por ejemplo.

Debemos tener en cuenta que la propiedad Name de un libro (Workbook) es una propiedad de solo lectura y solo se puede cambiar al guardar un archivo como. Por el contrario, el nombre de una hoja de cálculo es una propiedad lectura-escritura que podemos cambiar.



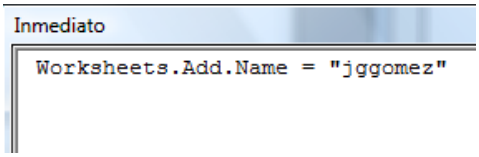
Tal y como vemos en la Ilustración anterior podemos cambiar el nombre de la hoja con índice referencia 1 a “mi hoja”.

Con la propiedad Activate, podemos activar o poner en activo una determinada hoja tal y como mostramos a continuación



6.4 Objeto-Método-Propiedad. Combinados.

Podemos aplicar una acción y una propiedad a la vez a un objeto, asi por ejemplo usando el método Add a un objeto Excel además podemos asignar la propiedad Name, de esta forma crearemos una nueva hoja en nuestro libro activo que llamaremos jggomez



Señalar que una vez asignado el método Add al objeto Worksheets la lista automática de métodos y propiedades no esta dispobible para la propiedad Name y por tanto tendremos que tecleerla manualmente al no poderla seleccionar de la lista automática.

Esto es debido a que cuando escribimos ActiveWorkbook en el editor de Visual Basic y después colocamos un punto, aparece la lista automática de métodos y propiedades apropiadas para el libro en cuestión. Pero cuando se escribie ActiveSheet y después ponemos un punto no aparece ninguna lista. Esto sucede porque un libro puede tener diferentes tipos de hojas: la hoja activa puede ser una hoja de calculo, una hoja de cuadro, veremos cómo solucionarlo en el apartado 5.6.

6.5 Objeto.Referencia. Miembros (método/propiedad).

6.5.1 Hacer referencia a una sola Hoja (Item o "nombre"). ActiveWorksheets

Al igual que sucede con los libros Excel, podemos hacer alusión a una sola hoja o varias hojas a la vez, como podremos ver a continuación.

Para refererirrnos a una hoja concreta podemos utilizar cualquiera de las siguientes expresiones, el resultado será similar, todas se refieren al mismo objeto Worksheet (hoja):

- Worksheets (1). Activate
- Worksheets ("mi hoja). Activate
- ActiveSheet

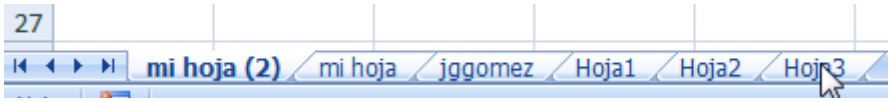
Asi también podremos emplear como hemos visto el método delete de la siguiente forma:

ActiveSheet. Delete

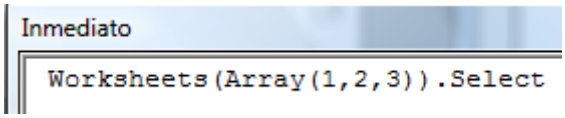
Esto hace que se elimine la hoja activa

6.5.2 Objeto.Referencia Multiples. Manipular varias hojas a la vez. FUNCION ARRAY.

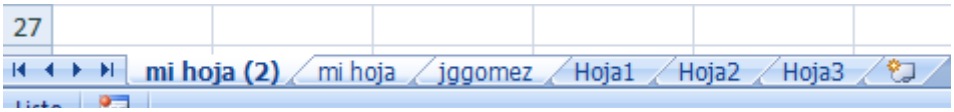
Señalar que Excel puede tener más de una hoja de cálculo activa, es como pulsar Ctl mientras hacemos clic en las fichas de Hojas tal y como mostramos en la siguiente Ilustración.



Para aplicar un método o propiedad a varias hojas a la vez hemos previamente de seleccionar varias hojas para lo cual utilizamos la función Array que nos permite hacer referencia a varios objetos de una colección, por ejemplo tal y como se muestra en la siguiente ilustración.



En este caso estamos seleccionando las hojas 1,2 y 3. El resultado de la aplicación de esta sentencia seria lo que se muestra a continuación.

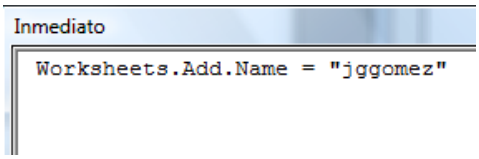


Array es una función que nos permite tratar muchos valores como si fueran uno solo.

No todas las colecciones permiten usar la función Array para crear selecciones. Así la colección Workbooks no admite esta función porque no se pueden seleccionar varios libros a la vez.

6.6 Declarar Variables para activar listas automáticas.

Tal y como vimos en el apartado 5.4 ocurre que al asignar l método Add al objeto Worksheets la lista automática de métodos y propiedades no esta dispobible para la propiedad Name y por tanto tendremos que tecleerla manualmente al no poderla seleccionar de la lista automática.



Esto es debido a que cuando escribimos ActiveWorkbook en el editor de Visual Basic y después colocamos un punto, aparece la lista automática de métodos y propiedades apropiadas para el libro en cuestión. Pero cuando se escribie ActiveSheet y después ponemos un punto no aparece ninguna lista. Esto sucede porque un libro puede tener diferentes tipos de hojas: la hoja activa puede ser una hoja de calculo, una hoja de cuadro.

6.7 Resumen Esquema: VBA y Hojas Excel

http://www.xltoday.net/vba_ejemplos_hojas.asp

6.7.1 Nombre de la hoja (variable)

```
'asigna nombre variable a la hoja a variable
strHoja = ActiveWindow.Caption
Windows(strHoja).Activate 'para activar el libro del nombre asignado
```

6.7.2 Insertar hoja nueva (elegir posición)

```
ActiveWorkbook.Sheets.Add Before:=Worksheets("Informe1")
```

6.7.3 Insertar hoja nueva (primera posición)

```
Sheets("Informe1").Copy After:=Worksheets(Worksheets.Count)
```

6.7.4 Mover hoja

```
Worksheets("informe5").Move After:=Worksheets("Informe4")
```

6.7.5 Ordenar hojas (orden alfabético)

```
intNumeroHojas = ActiveWorkbook.Worksheets.Count
For i = 1 To intNumeroHojas
  For j = i To intNumeroHojas
    If LCase(Worksheets(j).Name) < LCase(Worksheets(i).Name) Then
      Worksheets(j).Move Before:=Worksheets(i)
    End If
  Next j
Next i
```

6.7.6 Suprimir una hoja determinada

```
Application.DisplayAlerts = False

For i = 1 To Sheets.Count
  Sheets(i).Activate
    xxx = ActiveCell.Worksheet.Name
      If xxx = "Informe" Then
        ActiveWindow.SelectedSheets.Delete
```

```
      End If
Next

Application.DisplayAlerts = True
```

6.7.7 Seleccionar primera hoja

```
ActiveWindow.ScrollWorkbookTabs Position:=xlFirst
```

6.7.8 Seleccionar última hoja

```
ActiveWindow.ScrollWorkbookTabs Position:=xlLast
```

7 El Objeto Range (Rango). Características y Principales Miembros (Métodos y Propiedades).

7.1 Consideraciones Previas.

7.1.1 Referencias de celdas y rangos. Relativas, Absolutas, Fila absoluta y Columna absoluta.

Debemos tener en cuenta que en Excel las referencias a celdas pueden ser de cuatro tipos:

1. Relativas. La referencia es totalmente relativa. Cuando la formula se copia, la referencia de la celda se ajusta a su nueva localización. Ejemplo:A1
2. Absolutas. La referencia es totalmente absoluta. Cuando la formula se copia, la referencia de celda no cambia. Ejemplo: A1
3. Fila absoluta. La referencia es parcialmente absoluta. Cuando la fórmula se copia, la columna se ajusta pero la fila no cambia. Ejemplo: A$1
4. Columna absoluta. La referencia es parcialmente absoluta. Cuando la fórmula se copia, la fila se ajusta pero la coluna no cambia. Ejemplo: $A1

Por defecto, todas las referencias de celdas y rangos son relativas.

7.1.2 Notación F1C1

Normalmente Excel utiliza lo que se conoce como notación A1, cada dirección de celda consiste en letra de columna y un número de fila. Sin embargo Excel también contempla la notación F1C1, en este sistema la celda A1 hace referencia a la celda F1C1, la celda A2 hace referencia a la F2C2 y así sucesivamente.

7.1.3 Hacer referencias a otras celdas y rangos de otros libros.

Las referencias de celdas y rangos no tiene que estar en la misma hoja que la formula, para hacer referencia a una celda que está en otra hoja debemos anteponer el nombre de la hoja a la referencia de la celda seguida de un signo de exclamación, Ejemplo:

=Hoja1! A1

También podemos hacer referencia a una celda de hoja correspondiente a un libro distinto del que contiene la formula, para ello debemos anteponer el nombre del libro concreto que contiene los datos entre corchetes, tal y como se muestra a continuación:

= [mihoja.xlsx] Hoja1! A1

Si el nombre del libro al que hacemos referencia tiene espacios debemos poner comillas simples, tal y como se muestra en el siguiente ejemplo:

= '[mi hoja.xlsx] Hoja1!' A1

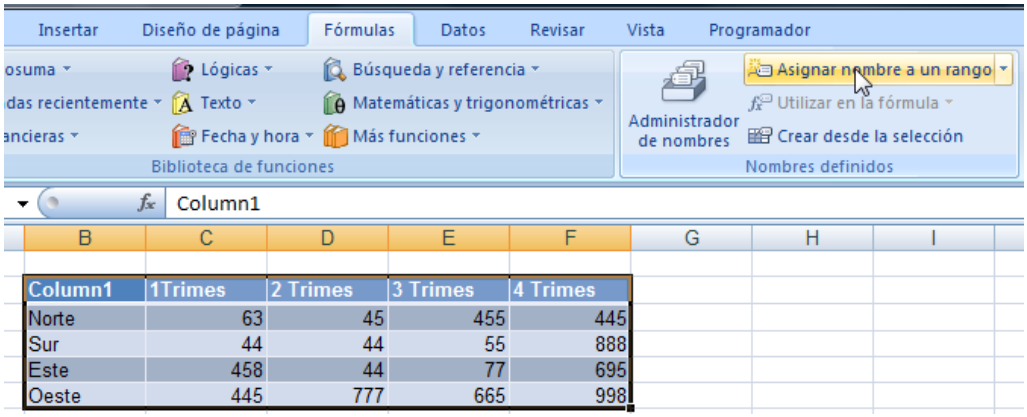
Si el libro vinculado está cerrado, debemos añadir la ruta completa hasta el libro de trabajo al que se hace referencia, como por ejemplo:

= 'c:/pruebas/Excel/ [mi hoja.xlsx] Hoja1!' A1

7.1.4 Usar nombres. Nombrar Celdas y Rangos. Nombre a Filas y Columnas

Podemos proporcionar nombres significativos a varios elementos, en concreto y para el caso que nos ocupa podemos asignar nombres a las celdas, rangos, filas, columnas, gráficos y otros objetos.

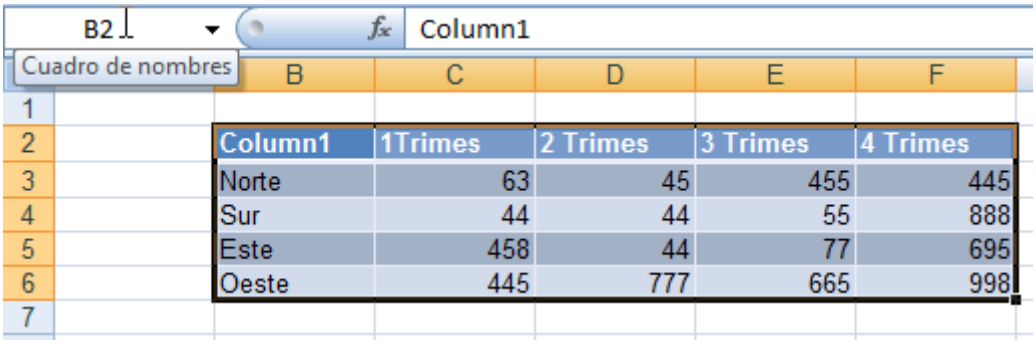
Existen varias maneras de asignar un nombre a una celda o a un rango presentamos los más habituales:



Column1	1Trimes	2 Trimes	3 Trimes	4 Trimes
Norte	63	45	455	445
Sur	44	44	55	888
Este	458	44	77	695
Oeste	445	777	665	998

Por ejemplo seleccionando el rango deseado y accediendo al menú formulas que nos llevara a definir nombre del rango deseado, tal y como se muestra en la Ilustración anterior.

Otra forma sería seleccionando las celdas o rangos y escribimos el nombre deseado en el cuadro nombre deseado.



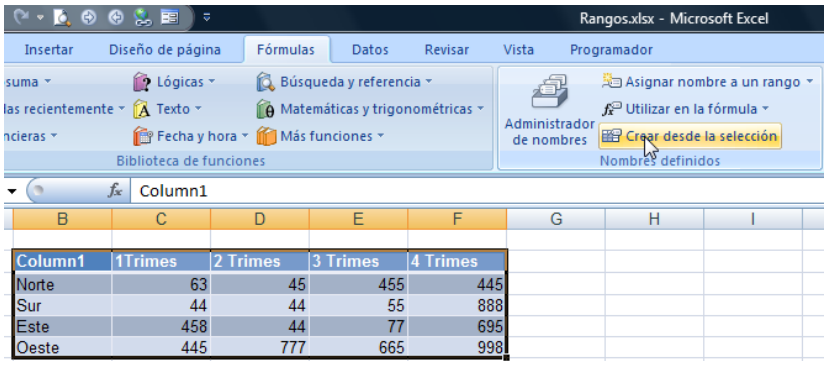
Column1	1Trimes	2 Trimes	3 Trimes	4 Trimes
Norte	63	45	455	445
Sur	44	44	55	888
Este	458	44	77	695
Oeste	445	777	665	998

También podemos dar nombre a Filas y Columnas Completas simplemente seleccionando la Fila o Columna y asignándole el nombre a través del cuadro de nombre tal y como vimos anteriormente.

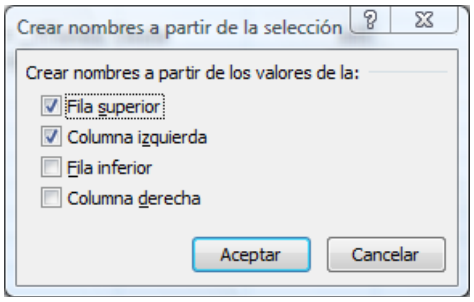
7.1.5 Cruzar Nombres.

Contamos con un operador especial llamado operador de intersección que entra en juego cuando estamos tratando con rangos. Este operador es un carácter espacio.

Primero seleccionamos el rango deseado y creamos los nombres deseados a través de la opción crear desde selección:



De esta forma tendremos el siguiente cuadro de dialogo que nos permitirá definir los nombres asociados al rango seleccionado.

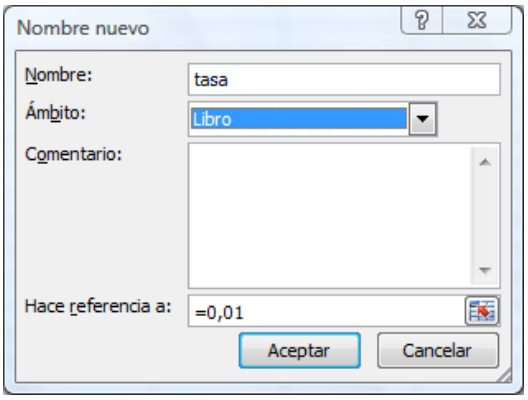


Column1	1Trimes	2 Trimes	3 Trimes	4 Trimes
Norte	63	45	455	445
Sur	44	44	55	888
Este	458	44	77	695
Oeste	445	777	665	998
	= _1Trimes Oeste		445	
	=SUMA(Oeste)		2885	

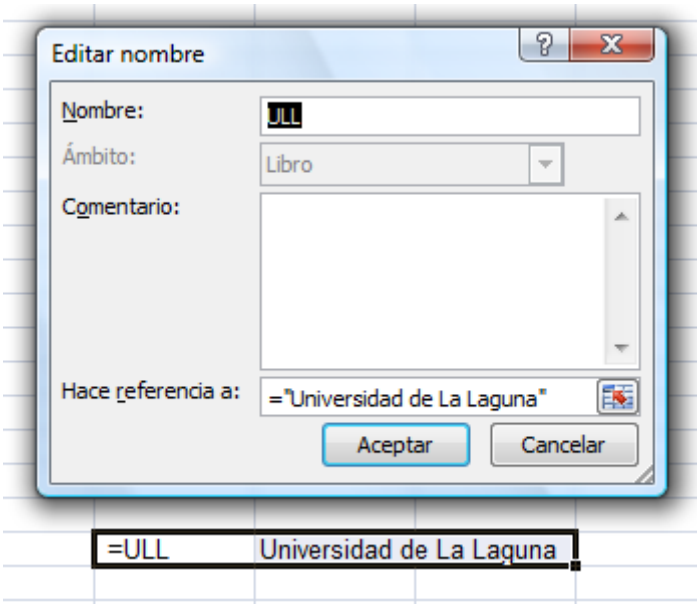
Así podremos cruzar la información tal y como se muestra en la Ilustración anterior y aplicarla a las formulas.

7.1.6 Asignar nombres a constantes y a fórmulas.

Es posible asignar un nombre a una constante sin necesidad de definirla en una celda, por ejemplo si quisiéramos que el nombre tasa tuviera un valor del 10% podríamos establecerlo como una constante de la siguiente forma:



Esto también vale para los nombres, por ejemplo podemos crear una abreviatura de ULL que una vez puesta en la fórmula nos escriba Universidad de La Laguna.



7.1.7 Ámbito de los nombres.

Los nombres de las celdas y rangos tienen un ámbito por defecto de libro, es decir podemos usarlo dentro de cualquier hoja del libro.

Otra opción es crear nombres con ámbito de hoja, para lo cual debemos anteponer el nombre de la hoja seguido de una exclamación, por ejemplo

Hoja1!Ventas

Si utilizamos el nombre en la hoja en que fue creada podemos prescindir del nombre de la hoja y solo utilizar Ventas, pero en caso que quisiéramos utilizarla desde otra hoja para hacer referencia a este nombre de ámbito hoja es necesario poner su ámbito.

7.1.8 Un tipo especial de Rango, Las Tablas

Excel 2007 admite un tipo especial de rangos que son las Tablas (Insertar-Tablas-Tabla). Esto presenta una importante ventaja, tal y como veremos a continuación.

	A	B	C	D	E	F
1						
2		Mes	Canarias	Galicia	Navarra	Total
3		Enero	63	45	455	563
4		Febrero	44	44	55	143
5		Marzo	458	44	77	579
6		Total	565	133	587	1285
7						

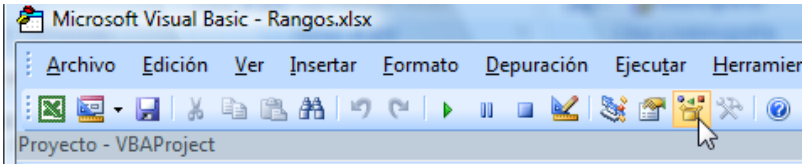
Supongamos el siguiente rango definido como Tabla1 en nuestra hoja de cálculo. Podemos crear formulas que hagan referencia celdas de la tabla si utilizamos los encabezados de columnas. Esto no solo puede hacer más fácil entender las fórmulas sino que además las formulas seguirán siendo válidas si se añaden o eliminan filas de la tabla. Por ejemplo todas estas formulas son válidas:

	A	B	C	D	E	F
1						
2		Mes	Canarias	Galicia	Navarra	Total
3		Enero	63	45	455	563
4		Febrero	44	44	55	143
5		Marzo	458	44	77	579
6		Total	565	133	587	1285
7						
8				=SUMA(Tabla1[Galicia])		133
9				=Tabla1[[#Totales];[Canarias]]		565
10			=Tabla1[[#Encabezados];[Navarra]]			Navarra

7.2 Aspectos Generales sobre el objeto Range.

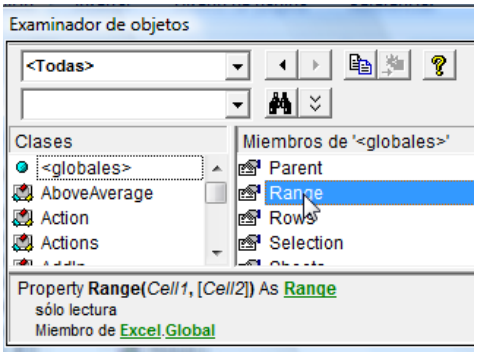
En este apartado pretendemos conocer las diferentes propiedades para referirse a los objetos Range. Destacar en primer lugar que Excel cuenta con una amplia variedad de métodos y propiedades para trabajar con los objetos Range.

Como primera aproximación vamos a analizar la información general relacionada con este objeto tal y como vimos en el apartado 3.2, para ello entramos en el programador VBA y más concretamente en el examinador de objetos.



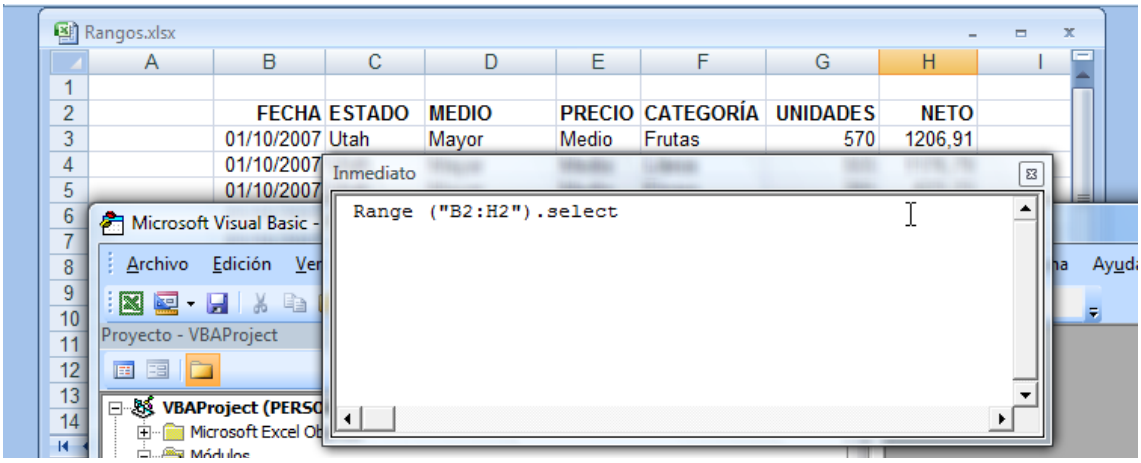
Recordemos que el Examinador de Objetos es una herramienta que enumera todas las propiedades y métodos de cada objeto disponible. La lista desplegable de la esquina superior izquierda del examinador de objetos incluye una lista de todas las bibliotecas de objetos a las que se tiene acceso el propio Excel.

En el examinador de objetos podemos ver básicamente dos listas: la de la izquierda con los nombres de las clases de objetos; y la de la derecha con los miembros (o métodos y propiedades) de la clase de objeto seleccionada. Al principio de la lista Clases aparece una clase de objeto muy especial, llamada <<globales>>.

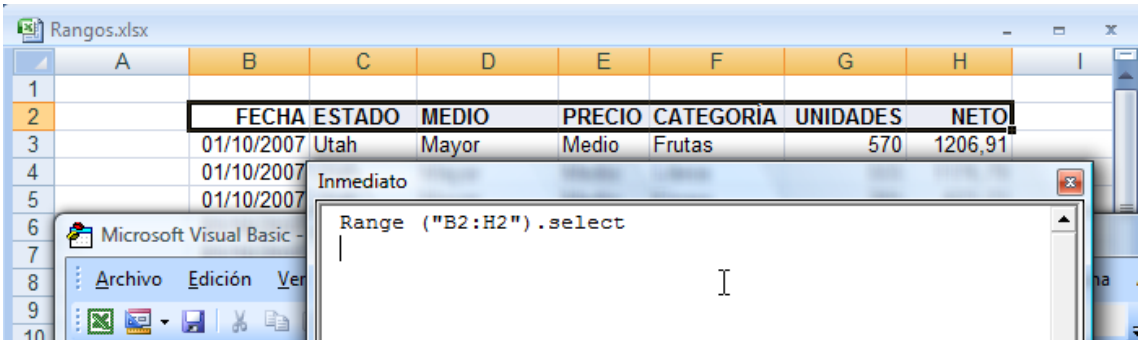


En realidad esta no es una clase de objeto, pero dentro de su lista de miembros están todos los métodos y propiedades que se pueden utilizar sin especificar un objeto.
Así si seleccionamos en la lista de objetos la propiedad Range en el recuadro, al final del examinador aparece información sobre la propiedad Range. Esta propiedad tiene dos argumentos, y los corchetes del segundo indican que es opcional. La propiedad Range devuelve una referencia a un objeto Range.

Ahora vamos a probar su uso en la ventana Inmediato de la siguiente forma:



Si pulsamos Intro, en la ventana Inmediato después de introducir la sentencia anterior, esto lo que hará sencillamente es seleccionar el rango establecido



Como podemos ver el primer argumento de Range puede ser un rango de varias celdas, de hecho puede ser cualquier cosa que Excel reconozca.

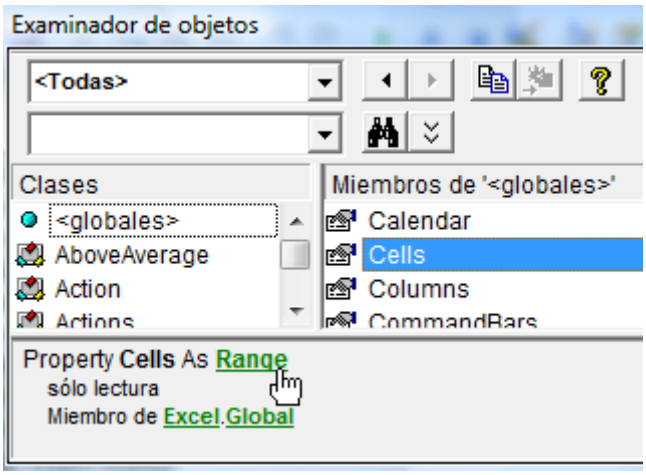
Tener en cuenta:

- ***Cells, Colum, Rows y Columns son propiedades del objeto Range y por tanto ninguna tiene su propia clase de objetos. Solo son maneras diferentes de representar el objeto Range.***

7.3 Hacer referencias a Rangos de las Hojas de Calculo

7.3.1 Referirse a un Rango como una colección de Celdas (Cells) y Columnas (Colum).

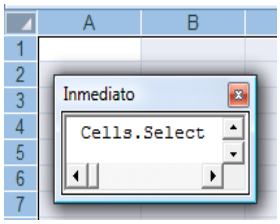
La colección de hojas de un libro (Worksheets) es una clase de objetos con métodos o propiedades diferentes al objeto Hoja de Cálculo (Worksheet).



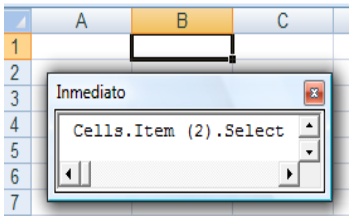
Una Hoja de Cálculo tiene muchas celdas pero no existe un objeto de la colección Cells. Una colección de celdas es un objeto rango y cell es una propiedad del objeto Range.

Así podemos ver con el Examinador de objetos como a través de la descripción final indica que la propiedad Cells devuelve un objeto Range.

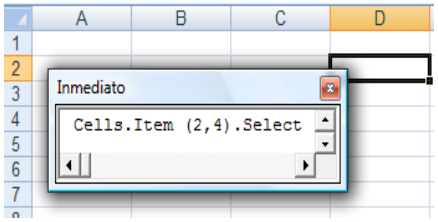
Presentamos los siguientes ejemplos desarrollados a través de la ventana inmediato.



Con esta instrucción seleccionamos todas las celdas de la hoja de cálculo.



Con esta instrucción seleccionamos la segunda celda de la primera fila.



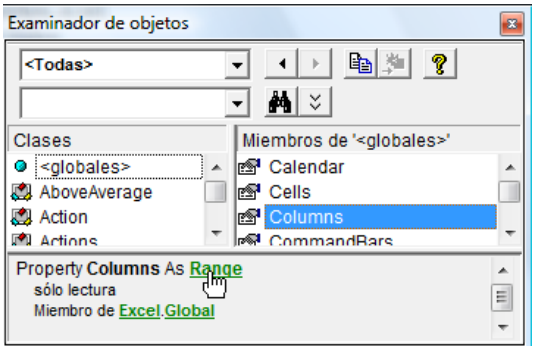
Aquí seleccionamos la celda correspondiente a la segunda fila y cuarta columna.

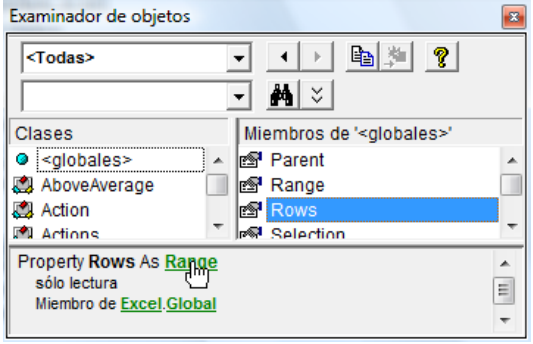
Tomando en consideración lo expuesto anteriormente, las siguientes instrucciones son equivalentes y darán todas ellas como resultado la selección de la celda A1.

Cells.Item (1). Select = Cells (1). Select = Range("A1").Select

7.3.2 Referirse a un Rango de Filas (Row) y Rango de Columnas (Columns)

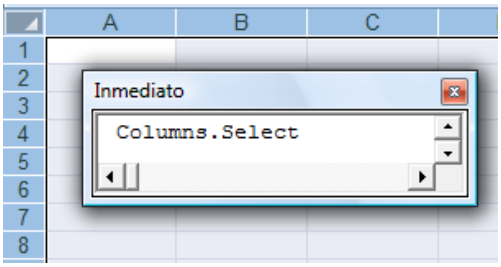
En este caso en vez de trabajar con una colección de celdas de la hoja de calculo como vimos anteriormente, trabajaremos con colecciones de filas y columnas a través de las propiedades Rows (que devuelve una colección de Filas) y Columns (que devuelve una colección de Columnas) respectivamente.



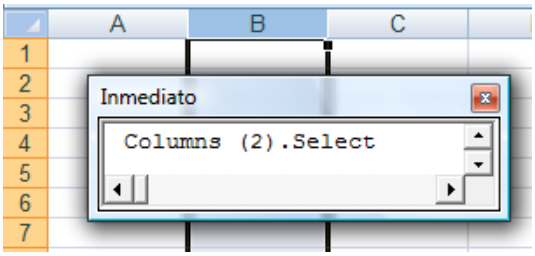


Igualmente señalar que tanto Rows como Columns son propiedades y no tienen su propia colección de objetos está asociado al objeto Range.

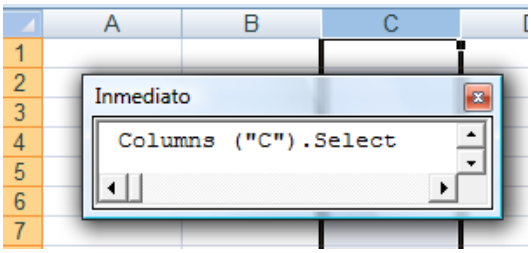
Presentamos los siguientes ejemplos desarrollados a través de la ventana inmediato correspondientes a la propiedad Columns.



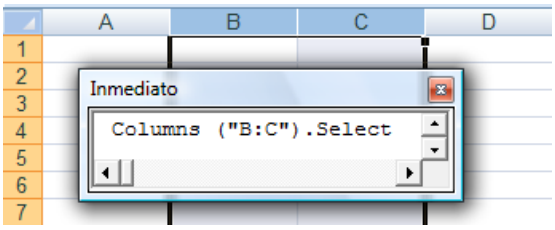
Con esta instrucción seleccionamos todas las celdas de la hoja de cálculo, igual que en el caso visto..



Con esta instrucción seleccionamos todas las celdas de la segunda columna.

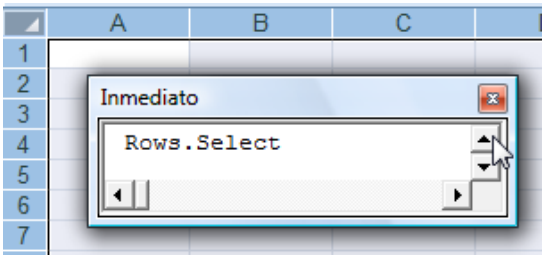


Con esta instrucción seleccionamos todas las celdas de la columna C.

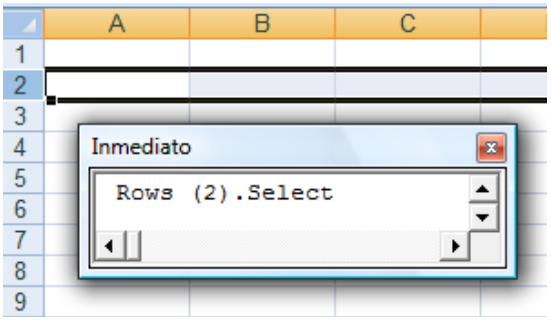


Con esta instrucción seleccionamos todas las celdas de las columnas B y C.

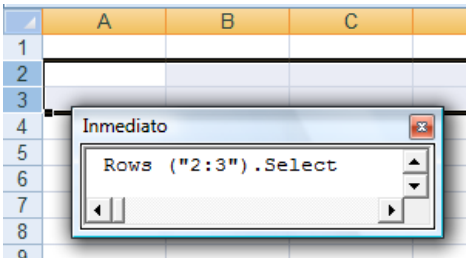
A continuación mostramos los siguientes ejemplos desarrollados a través de la ventana inmediato correspondientes a la propiedad Rows.



Con esta instrucción seleccionamos todas las celdas de la hoja de cálculo, igual que en el caso visto..



Con esta instrucción seleccionamos todas las celdas de la Fila 2.



Con esta instrucción seleccionamos todas las celdas de la Fila 2 y 3

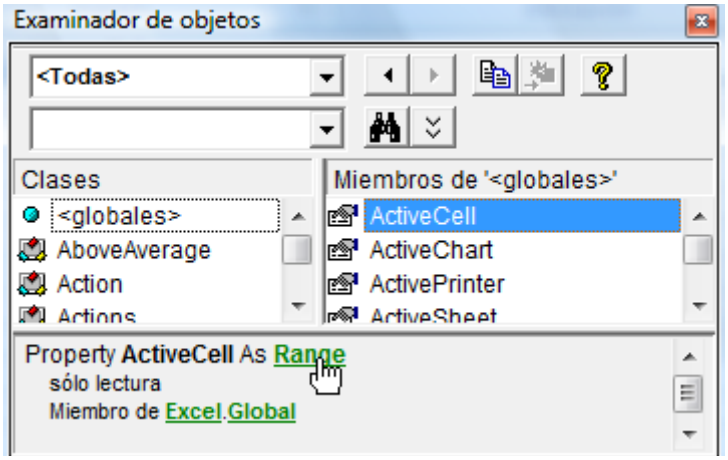
Debemos tener en cuenta como hemos comentado que Cells, Colum, Rows y Columns son propiedades del objeto Range y por tanto ninguna tiene su propia clase de objetos. Solo son maneras diferentes de representar el objeto Range.

7.3.3 Referirse a un Rango basado en la celda activa o celdas seleccionadas. ActiveCell-CurrentRegion-EntireColumn-EntireRow. Otras Propiedades del Objeto Range.

Para referirnos a un rango concreto que está relacionado con la celda activa o con la selección actual, contamos con una serie de propiedades Range como son ***ActiveCell*** y ***CurrentRegion***. Tambien analizaremos ***EntireColum*** y ***EntireRow***.

Pasamos a continuación a estudiar concretamente sus características más sobresalientes.

Veamos una vez más el examinador de objetos, en concreto <<globales>> la propiedad ActiveCell.



La descripción que aparece al final del objeto muestra que la propiedad devuelve un objeto Range.

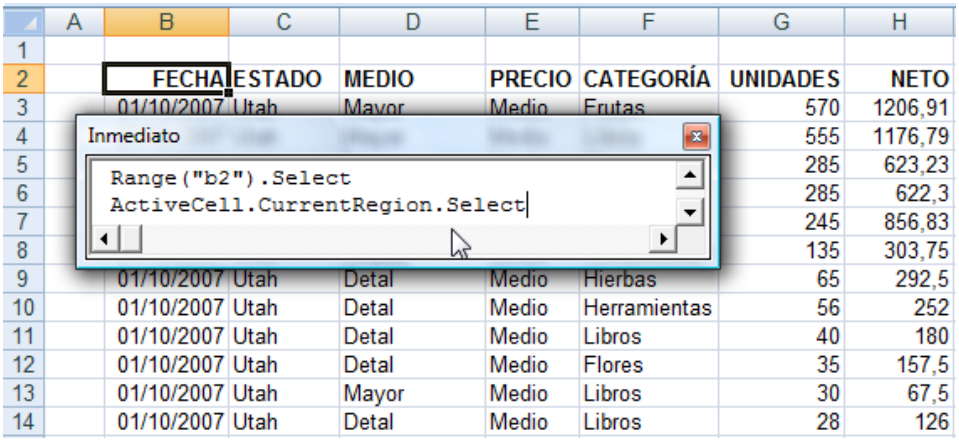
Veamos el siguiente ejemplo:

Partimos de la siguiente hoja de cálculo:

	A	B	C	D	E	F	G	H
1								
2		FECHA	ESTADO	MEDIO	PRECIO	CATEGORÍA	UNIDADES	NETO
3		01/10/2007	Utah	Mayor	Medio	Frutas	570	1206,91
4		01/10/2007	Utah	Mayor	Medio	Libros	555	1176,79
5		01/10/2007	Utah	Mayor	Medio	Flores	285	623,23
6		01/10/2007	Utah	Mayor	Medio	Hierbas	285	622,3
7		01/10/2007	Utah	Detal	Medio	Frutas	245	856,83
8		01/10/2007	Utah	Mayor	Medio	Herramientas	135	303,75
9		01/10/2007	Utah	Detal	Medio	Hierbas	65	292,5
10		01/10/2007	Utah	Detal	Medio	Herramientas	56	252
11		01/10/2007	Utah	Detal	Medio	Libros	40	180
12		01/10/2007	Utah	Detal	Medio	Flores	35	157,5
13		01/10/2007	Utah	Mayor	Medio	Libros	30	67,5
14		01/10/2007	Utah	Detal	Medio	Libros	28	126
15								

Ilustración 2

A través de la ventana Inmediato escribimos la siguiente instrucción:



	FECHA	ESTADO	MEDIO	PRECIO	CATEGORÍA	UNIDADES	NETO
3	01/10/2007	Utah	Mayor	Medio	Frutas	570	1206,91
4						555	1176,79
5						285	623,23
6						285	622,3
7						245	856,83
8						135	303,75
9	01/10/2007	Utah	Detal	Medio	Hierbas	65	292,5
10	01/10/2007	Utah	Detal	Medio	Herramientas	56	252
11	01/10/2007	Utah	Detal	Medio	Libros	40	180
12	01/10/2007	Utah	Detal	Medio	Flores	35	157,5
13	01/10/2007	Utah	Mayor	Medio	Libros	30	67,5
14	01/10/2007	Utah	Detal	Medio	Libros	28	126

Su significado es:

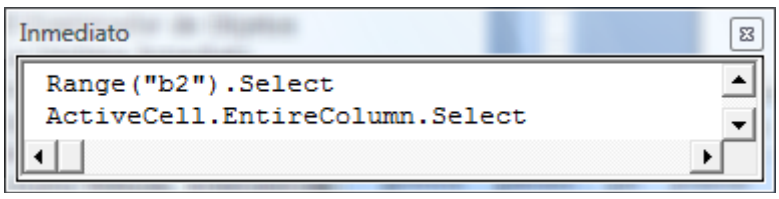
- Range (“b2”).Select. Nos situamos primero en el comienzo donde están los datos del rango de datos.
- ActiveCell.CurrentRegion.Select. Con la celda activa, determina la región o rango en la que esta pertenece y selecciona el mismo.

El resultado será:

	FECHA	ESTADO	MEDIO	PRECIO	CATEGORÍA	UNIDADES	NETO
3	01/10/2007	Utah	Mayor	Medio	Frutas	570	1206,91
4	01/10/2007	Utah	Mayor	Medio	Libros	555	1176,79
5	01/10/2007	Utah	Mayor	Medio	Flores	285	623,23
6	01/10/2007	Utah	Mayor	Medio	Hierbas	285	622,3
7	01/10/2007	Utah	Detal	Medio	Frutas	245	856,83
8	01/10/2007	Utah	Mayor	Medio	Herramientas	135	303,75
9	01/10/2007	Utah	Detal	Medio	Hierbas	65	292,5
10	01/10/2007	Utah	Detal	Medio	Herramientas	56	252
11	01/10/2007	Utah	Detal	Medio	Libros	40	180
12	01/10/2007	Utah	Detal	Medio	Flores	35	157,5
13	01/10/2007	Utah	Mayor	Medio	Libros	30	67,5
14	01/10/2007	Utah	Detal	Medio	Libros	28	126

Veamos ahora un ejemplo con las propiedades EntireColumn

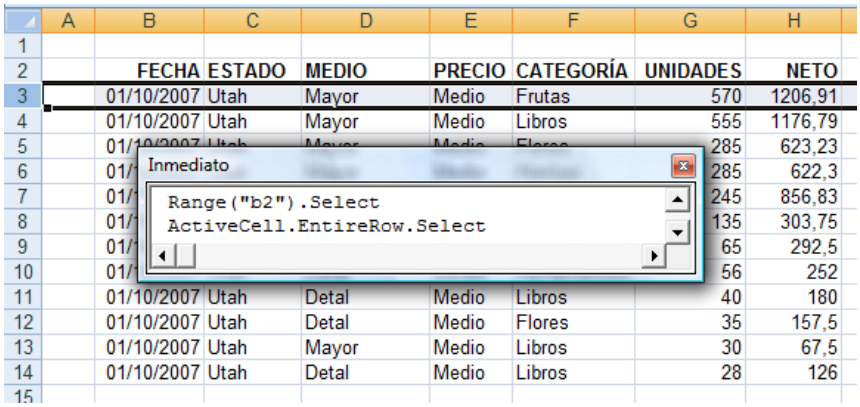
Retomando la Ilustración 1 vamos a introducir la siguiente sentencia a través de la ventana Inmediato.



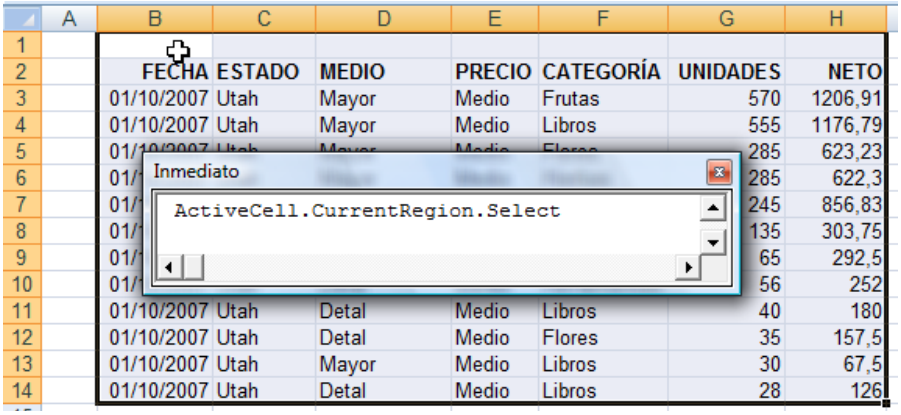
El resultado será la selección de la columna B:

	FECHA	ESTADO	MEDIO	PRECIO	CATEGORÍA	UNIDADES	NETO
3	01/10/2007	Utah	Mayor	Medio	Frutas	570	1206,91
4	01/10/2007	Utah	Mayor	Medio	Libros	555	1176,79
5	01/10/2007	Utah	Mayor	Medio	Flores	285	623,23
6	01/10/2007	Utah	Mayor	Medio	Hierbas	285	622,3
7	01/10/2007	Utah	Detal	Medio	Frutas	245	856,83
8	01/10/2007	Utah	Mayor	Medio	Herramientas	135	303,75
9	01/10/2007	Utah	Detal	Medio	Hierbas	65	292,5
10	01/10/2007	Utah	Detal	Medio	Herramientas	56	252
11	01/10/2007	Utah	Detal	Medio	Libros	40	180
12	01/10/2007	Utah	Detal	Medio	Flores	35	157,5
13	01/10/2007	Utah	Mayor	Medio	Libros	30	67,5
14	01/10/2007	Utah	Detal	Medio	Libros	28	126

Podríamos seleccionar la fila completa de la siguiente forma:



Por otro lado, situados sobre la celda B1 y con la siguiente instrucción podremos seleccionar toda la lista del ejemplo más una fila de la parte superior.



Destacar que cuando utilizamos la propiedad CurrentRegion con un rango de celdas múltiples como punto de partida, nos daremos cuenta que al calcular la región actual, se ignora todos los elementos de la celda superior del rango.

7.3.4 Hacer referencia a un rango relativo. Propiedad Offset, Riseze,CurrentRegion y el método Intersect.

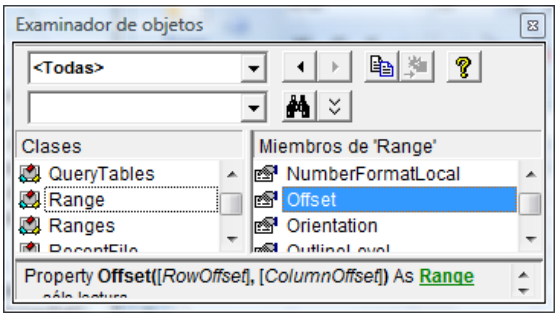
Dentro del objeto Range (no en global) encontramos tres propiedades interesantes como son Offset, Riseze e Intersect. Las dos primeras propiedades se encuentran dentro del objeto Range y la tercera es una función de <<globales>>.

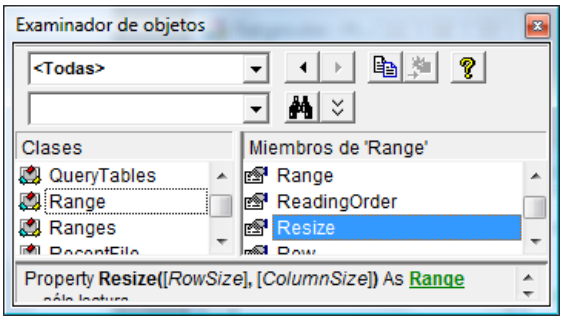
Las propiedades Offset, Resize, EntireRow, EntireColum, CurrentRegion y el método Intersect son herramientas flexibles que sirven para calcular nuevos objetos Range, basados en un Rango inicial determinado.

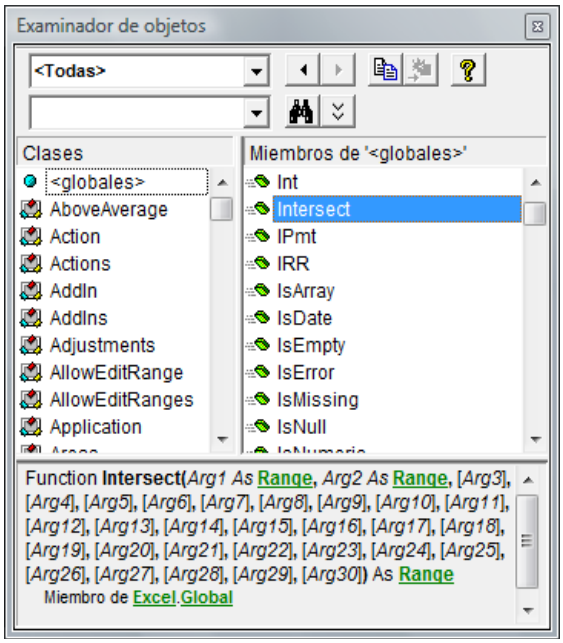
Por lo general la mejor manera de trabajar dentro de un rango es usar en primer lugar la propiedad CurrentRegion para definir el rango de base y enseguida utilizar la propiedad offset y el método Intersect para manipular el rango que acabamos de señalar.

La propiedad Offset alude a un rango que se ha movido hacia arriba, abajo, izquierda o derecha de un rango inicial. Resize hace referencia con más o menos filas o columnas, que el rango inicial.

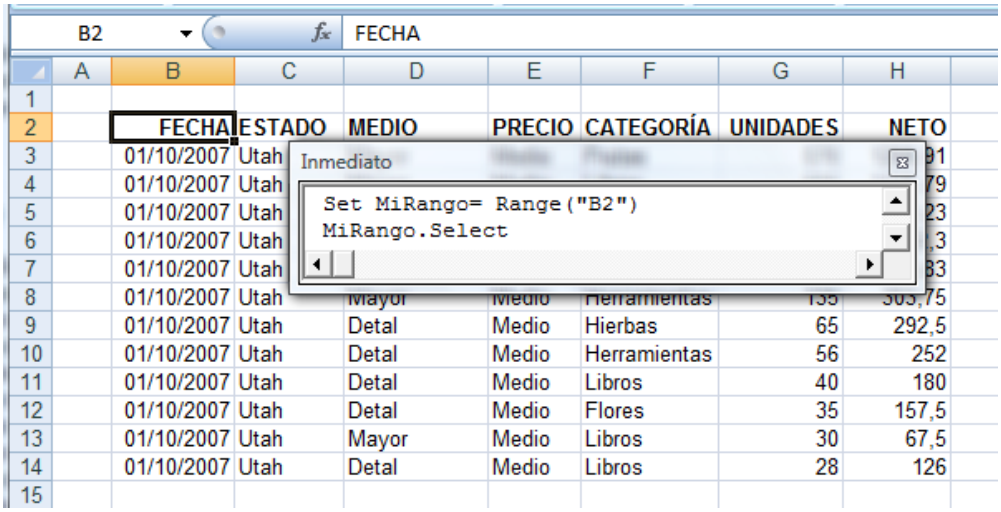
Veamos a continuación una serie de ejemplos relacionados con estas propiedades





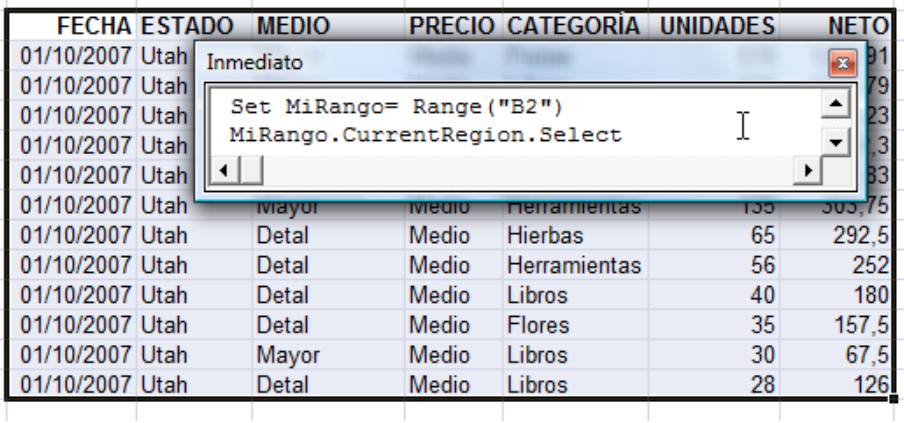


En este caso que presentamos a continuación hemos creado una variable MiRango que hace referencia a la celda B2, y hemos posteriormente expuesto la sentencia de que seleccione la misma (MiRango).



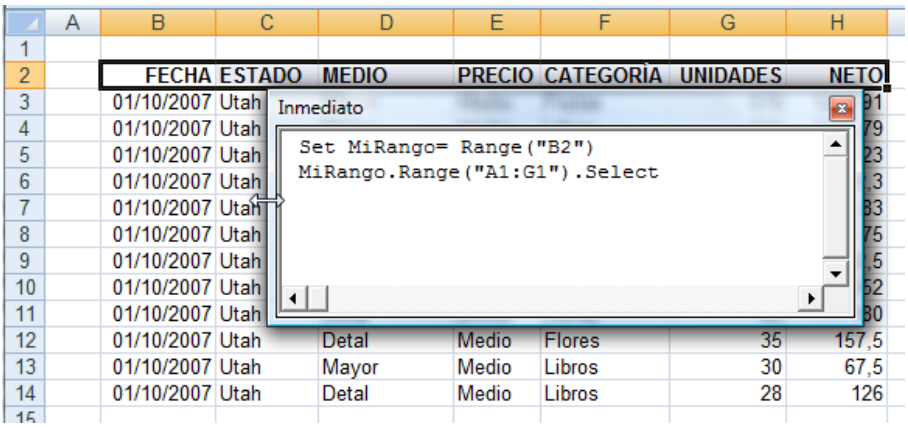
	A	B	C	D	E	F	G	H
1								
2		FECHA	ESTADO	MEDIO	PRECIO	CATEGORÍA	UNIDADES	NETO
3		01/10/2007	Utah					91
4		01/10/2007	Utah					79
5		01/10/2007	Utah					23
6		01/10/2007	Utah					,3
7		01/10/2007	Utah					33
8		01/10/2007	Utah	Mayor	Medio	Herramientas	135	303,75
9		01/10/2007	Utah	Detal	Medio	Hierbas	65	292,5
10		01/10/2007	Utah	Detal	Medio	Herramientas	56	252
11		01/10/2007	Utah	Detal	Medio	Libros	40	180
12		01/10/2007	Utah	Detal	Medio	Flores	35	157,5
13		01/10/2007	Utah	Mayor	Medio	Libros	30	67,5
14		01/10/2007	Utah	Detal	Medio	Libros	28	126
15								

Si quisiéramos seleccionar todo el rango al que pertenece la variable Mi Rango la instrucción seria como la siguiente:



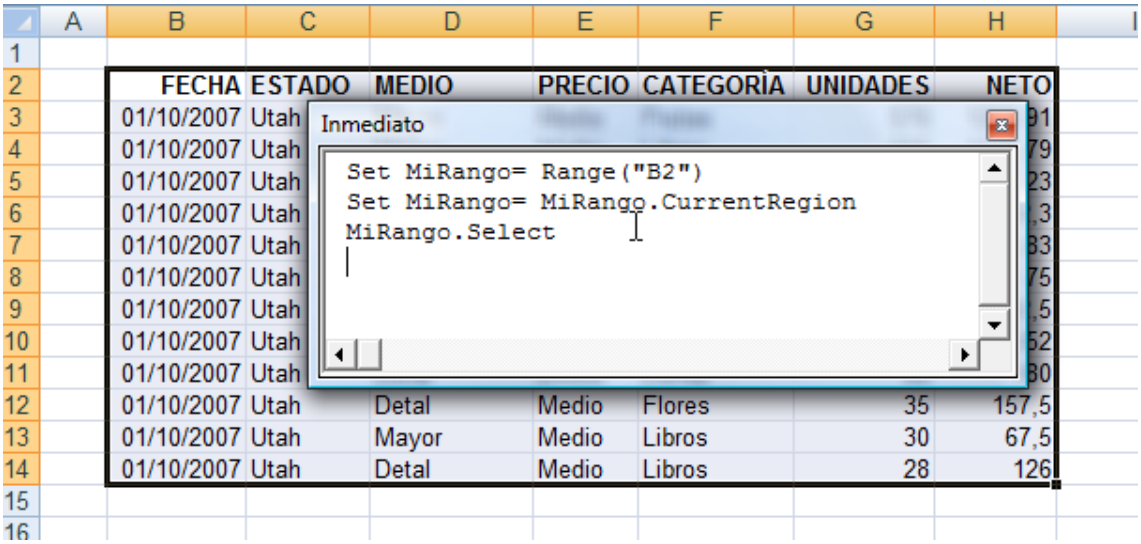
FECHA	ESTADO	MEDIO	PRECIO	CATEGORÌA	UNIDADES	NETO
01/10/2007	Utah					91
01/10/2007	Utah					79
01/10/2007	Utah					23
01/10/2007	Utah					,3
01/10/2007	Utah					33
01/10/2007	Utah	Mayor	Medio	Herramientas	135	303,75
01/10/2007	Utah	Detal	Medio	Hierbas	65	292,5
01/10/2007	Utah	Detal	Medio	Herramientas	56	252
01/10/2007	Utah	Detal	Medio	Libros	40	180
01/10/2007	Utah	Detal	Medio	Flores	35	157,5
01/10/2007	Utah	Mayor	Medio	Libros	30	67,5
01/10/2007	Utah	Detal	Medio	Libros	28	126

En este otro caso aunque la selección sea mas grande se limita al rango definido



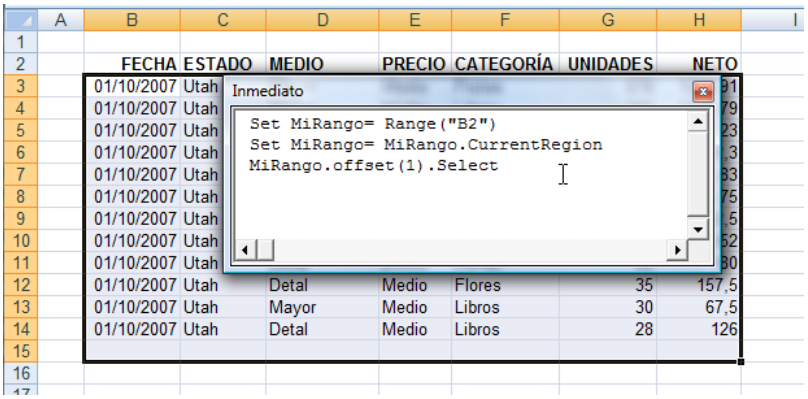
	A	B	C	D	E	F	G	H
1								
2		FECHA	ESTADO	MEDIO	PRECIO	CATEGORÌA	UNIDADES	NETO
3		01/10/2007	Utah					91
4		01/10/2007	Utah					79
5		01/10/2007	Utah					23
6		01/10/2007	Utah					,3
7		01/10/2007	Utah					33
8		01/10/2007	Utah					75
9		01/10/2007	Utah					,5
10		01/10/2007	Utah					52
11		01/10/2007	Utah					30
12		01/10/2007	Utah	Detal	Medio	Flores	35	157,5
13		01/10/2007	Utah	Mayor	Medio	Libros	30	67,5
14		01/10/2007	Utah	Detal	Medio	Libros	28	126
15								

En el caso siguiente vemos como al comienzo, la variable MiRango ya se refiere a la celda B2, que esta dentro de la lista del ejemplo, asi la segunda instrucción hace referencia a toda la lista y la segunda confirma que tiene una referencia hacia la variable correspondiente.



	A	B	C	D	E	F	G	H
1								
2		FECHA	ESTADO	MEDIO	PRECIO	CATEGORÌA	UNIDADES	NETO
3		01/10/2007	Utah					91
4		01/10/2007	Utah					79
5		01/10/2007	Utah					23
6		01/10/2007	Utah					,3
7		01/10/2007	Utah					33
8		01/10/2007	Utah					75
9		01/10/2007	Utah					,5
10		01/10/2007	Utah					52
11		01/10/2007	Utah					30
12		01/10/2007	Utah	Detal	Medio	Flores	35	157,5
13		01/10/2007	Utah	Mayor	Medio	Libros	30	67,5
14		01/10/2007	Utah	Detal	Medio	Libros	28	126
15								
16								

Ahora en nuestro caso, *aplicando la propiedad Offset* tendríamos:



Vemos en el caso anterior como el primer argumento de la propiedad offset indica el numero de filas que el rango debe moverse hacia abajo y el segundo indica cuantas columnas debe desplazarse hacia la derecha.. Si omitimos un argumento, es como si pusiéramos cero y por tanto no desplazaríamos el rango.

Vamos a *aplicar la propiedad offset y Resize*.

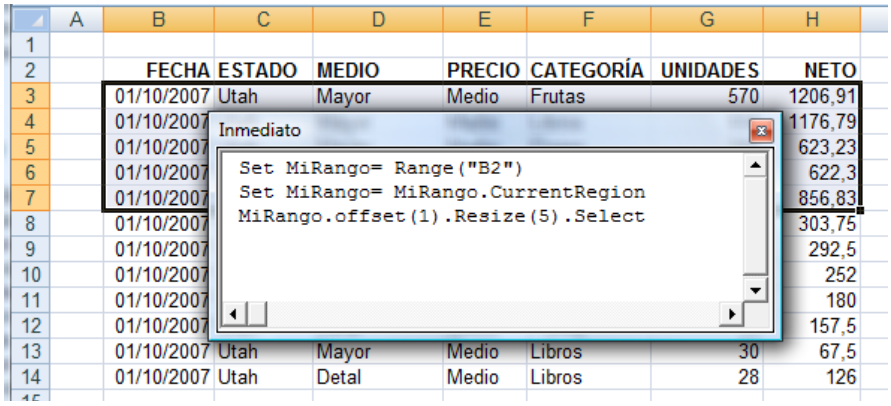
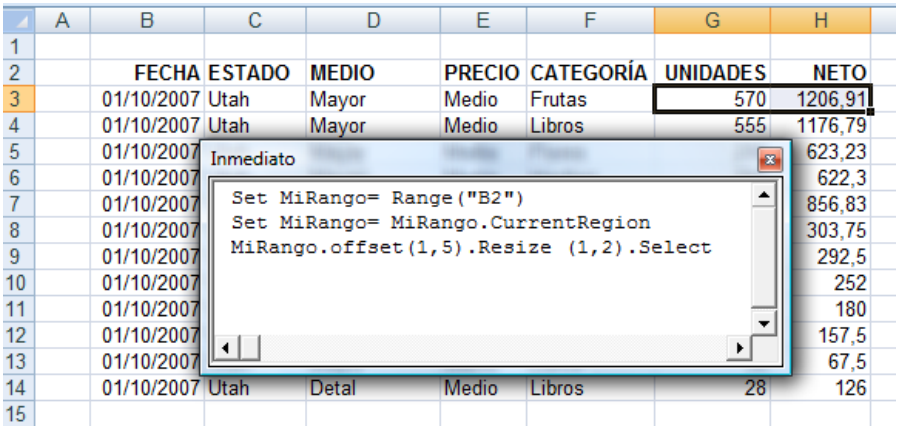


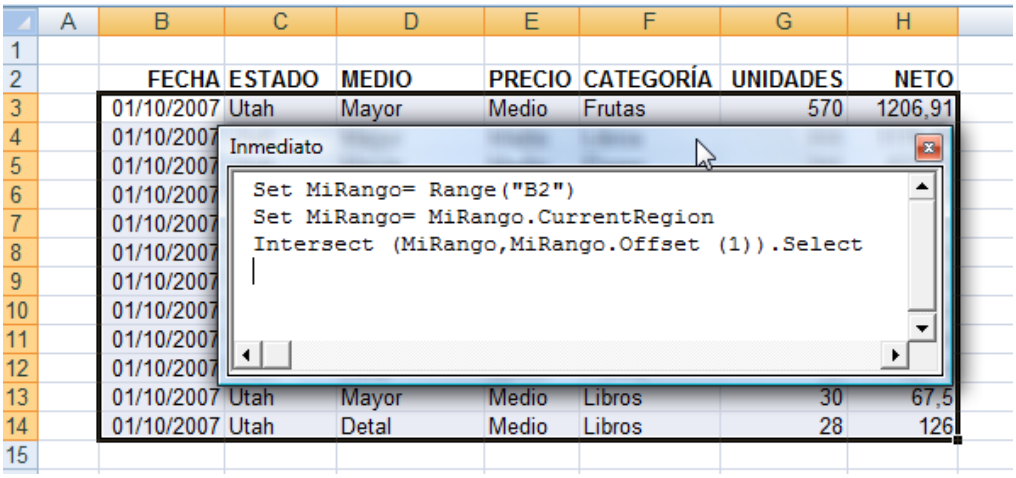
Ilustración 3

En este caso (Ilustración 2) hemos seleccionado las cinco primeras filas con datos. Por una parte , la propiedad Offset desplaza hacia abajoel rango para no incluir el encabezado, por otra parte el Riseze, redmensiona el rango final. El primer argumento de la propiedad Resize equivale al numero de filas y el segundo, a la cantidad de columnas del rango final. Cuando se omite este argumento es como si conservara el tamaño del rango original para esa dirección.

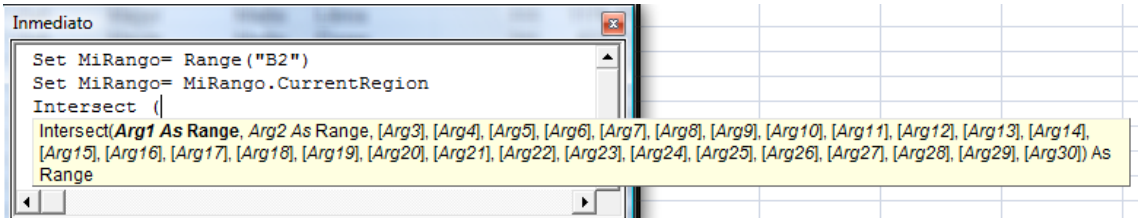


En el caso anterior muestra la selección del rango G3:H3, la cual resulta de ser el valor numérico de la primera fila del cuerpo de la lista.

Pasamos a continuación a analizar *la función Intersect.*



El método Intersect devuelve un objeto Range y que puede tomar hasta 30 argumentos.



Por lo general, solo se usan dos y los dos primeros son obligatorios. De igual forma se observa que los dos primeros deben ser objetos Range.

Para remover los encabezados podemos utilizar el método intersect y offset a la vez.

Recordemos que las propiedades Offset, Resize,EntireRow, EntireColum, CurrentRegion y el método Intersect son herramientas flexibles que sirven para calcular nuevos objetos Range, basados en un Rango inicial determinado.

Por lo general la mejor manera de trabajar dentro de un rango es usar en primer lugar la propiedad CurrentRegion para definir el rango de base y enseguida utilizar la propiedad offset y el método Intersect para manipular el rango que acabamos de señalar.

7.4 Formatear Rangos. Colección Borders

7.4.1 Añadir bodes a un rango. Ejemplo de Macros para añadir Bordes

Los bordes ayudan a delimitar una región dentro de un bloque de celdas, con las propiedades y métodos del objeto Range podemos manejar cualquier tipo de borde. Veremos a continuación diferentes ejemplos.

Ejemplo1: Establecer bordes a las diferentes celdas de la región.

En primer lugar estableceremos o definiremos en nuestra ventana de inmediato cual es el rango de la región, a través se la sentencia:

Set MiRango = Range (“B2”). CurrentRegion.

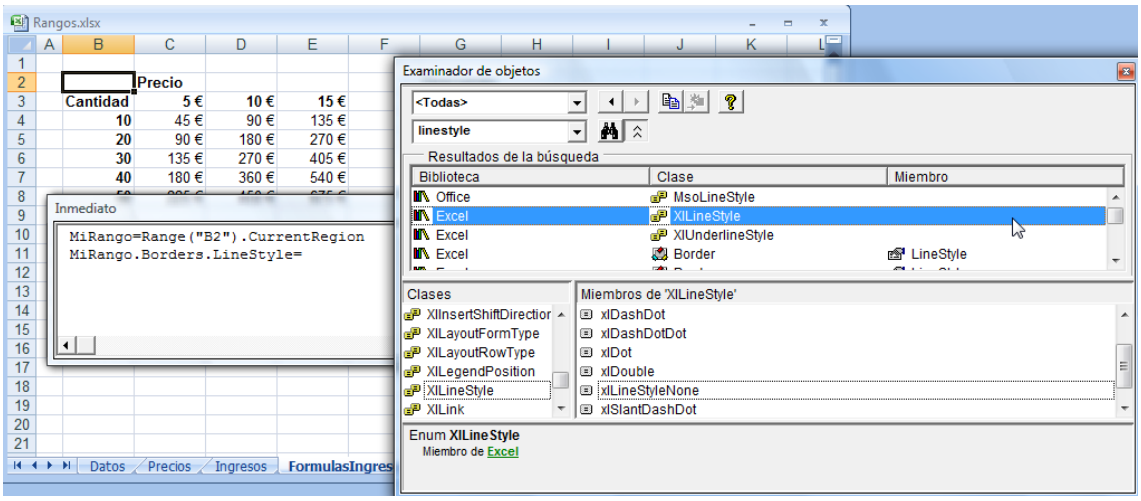
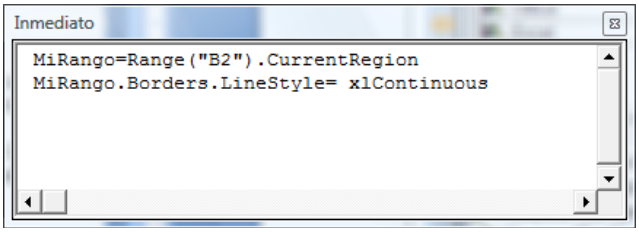


Ilustración 4

A continuación en la ventana de inmediato escribimos el formato a aplicar al rango con la siguiente sentencia:

MiRango.Borders.LineStyle= xlContinuous



Destaca que cuando ponemos un punto en una instrucción la lista automática nos muestra los miembros disponibles, pero no sucede lo mismo después del igual. En este caso debemos usar el Examinador de objetos para encontrar las opciones respectivas (ver Ilustración 3).

Una vez ejecutado las sentencias de la ventana de inmediato el resultado será el mostrado en la Ilustración 4, donde el rango aparece con todos los bordes de sus celdas.

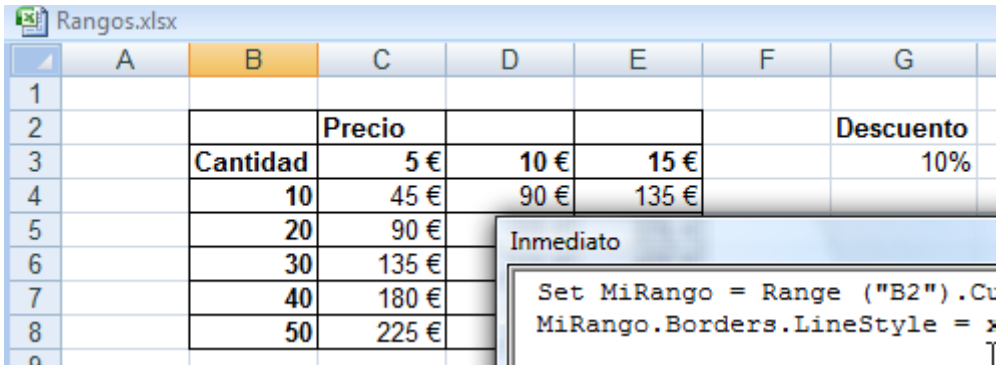
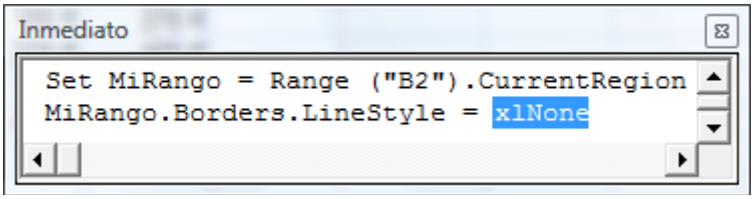


Ilustración 5

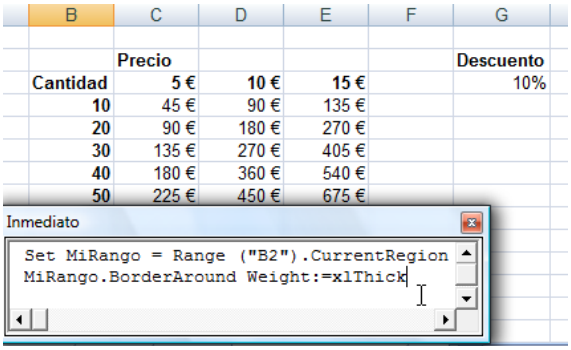
Ejemplo2: Quitar bordes a las diferentes celdas de la región.

Para ello lo que hacemos es utilizar la propiedad xlNone. Señalar que el valor xlNone no aparece en la lista de LineStyle porque es una constante global utilizada por distintos objetos Excel.



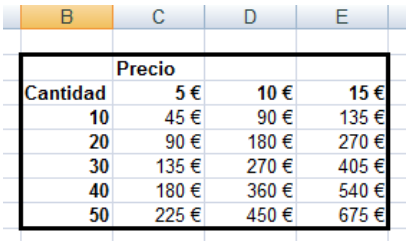
Ejemplo3: Borde Grueso solo al área del Rango. BorderAround.

Partiendo de la hoja original la instrucción para aplicar este formato sería la expuesta a continuación.




```
Set MiRango = Range ("B2").CurrentRegion
MiRango.BorderAround Weight:=xlThick
```

Estas instrucciones colocan un borde más grueso. Como weight quiere decir que la línea será continua. BorderAround es un método abreviado que sirve para asignar un bode exterior a cualquier rango de varias celdas.

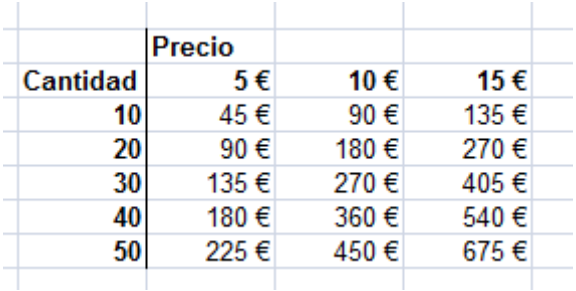


Ejemplo4: Borde aplicado a una sola columna su parte derecha.

Ahora vamos aplicar un borde al lado derecho de las cantidades. Para especificar que solo queremos un borde podemos usar un nombre enumerado, junto con la colección de Borders.

Así vamos a la lista de Inmediato y escribimos la siguiente sentencia:

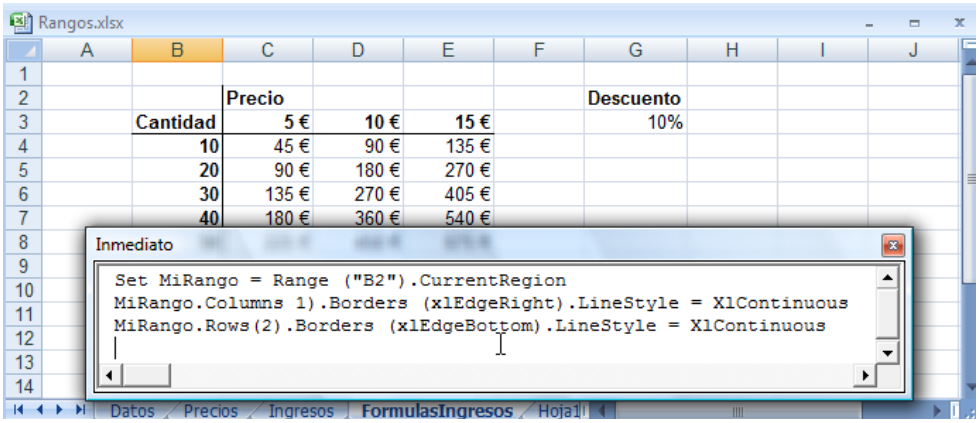
```
Set MiRango = Range ("B2").CurrentRegion
MiRango.Columns (1).Borders (xlEdgeRight).LineStyle = XlContinuous
```



Ejemplo5: Borde aplicado a una sola Fila parte inferior.

Retomando el ejemplo anterior vamos aplicar un borde bajo a la fila de precios, para ello las sintaxis será la que se muestra en la siguiente pantalla:

```
Set MiRango = Range ("B2").CurrentRegion
MiRango.Columns (1).Borders (xlEdgeRight).LineStyle = XlContinuous
MiRango.Rows(2).Borders (xlEdgeBottom).LineStyle = XlContinuous
```



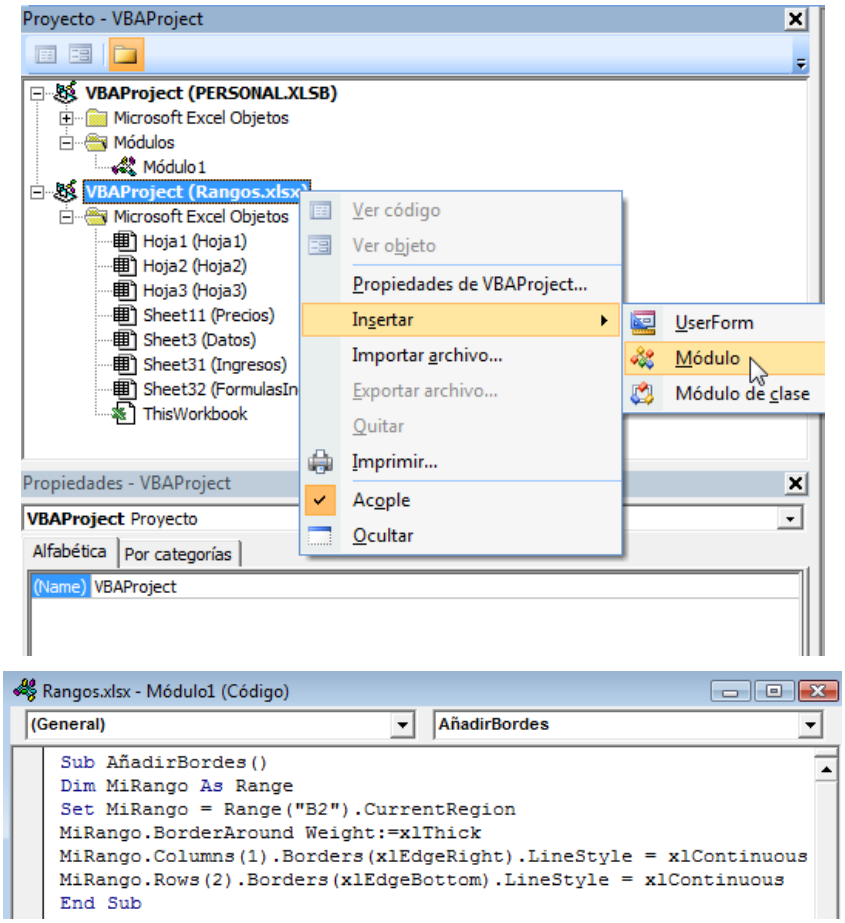
Ejemplo 6: Ejemplo de Macro para Añadir Bordes.

A continuación vamos a crear una macro que podemos asociar a cualquier evento para que se aplique de forma predeterminada un formato a un rango determinado. Para ello partimos del caso presentado anteriormente es decir contamos con un rango de valores sin formato:

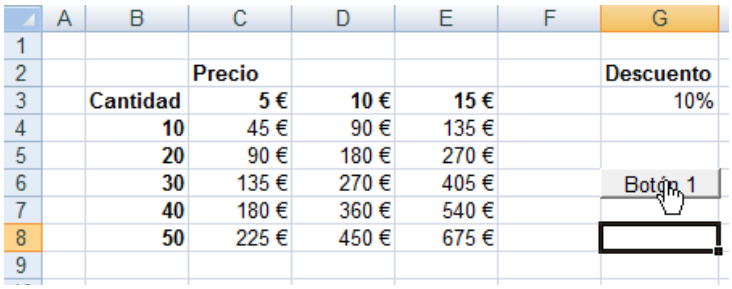
	A	B	C	D	E
1					
2			Precio		
3		Cantidad	5 €	10 €	15 €
4		10	45 €	90 €	135 €
5		20	90 €	180 €	270 €
6		30	135 €	270 €	405 €
7		40	180 €	360 €	540 €
8		50	225 €	450 €	675 €

Queremos crear una macro asociada a un evento determinado que nos permita dar formato a ese rango cada vez que este cambie o se modifique, para ello crearemos la siguiente macro que llamaremos “AñadirBordes”.

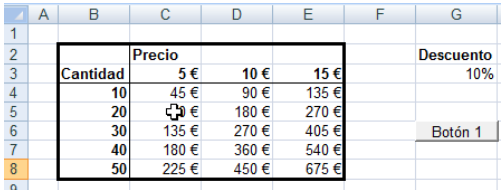
Comenzamos con nuestra hoja de trabajo “Rangos.xlsx y acudimos al programador donde vamos añadir un nuevo módulo tal y como se muestra en la siguiente ilustración



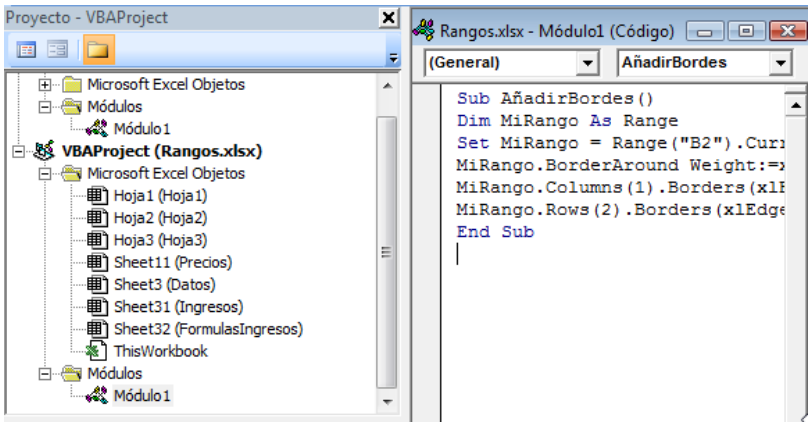
Una vez hecha la macro, para probarlo hemos puesto un botón que con el doble clic se ejecutara.



El resultado de la macro anterior será:



Mostramos a continuación como quedará nuestro proyecto.



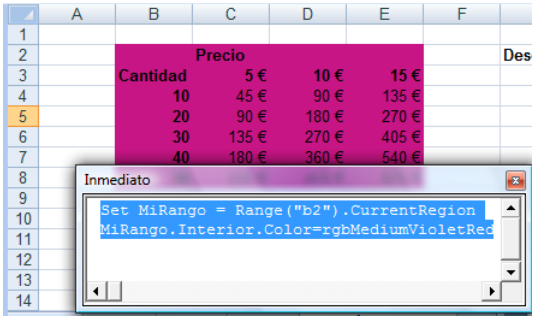
7.4.2 Formatear un Rango Interior. Creación de Macro formato de rango para celdas de valores y para formulas.

Para mejorar la legibilidad podemos aplicar diferentes colores de fondo en diferentes partes del informe, por ejemplo para la celdas que se pueden llenar con valores y para aquellas que son formulas. Veamos varios ejemplos

Ejemplo 1: Colorear el fondo de un rango. Opción 1

La sentencia básica seria:

```
Set MiRango = Range("b2").CurrentRegion
MiRango.Interior.Color=rgbMediumVioletRed
```



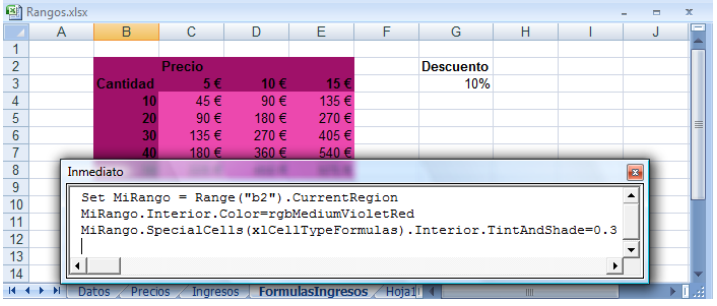
Podemos ver como el color del fondo cambia a un médium violet red.

Ejemplo 2: Colorear el fondo de un rango. Opción 2

Las instrucciones en este caso serian:

```
Set MiRango = Range("b2").CurrentRegion
MiRango.Interior.TintAndShade = -0.2
```
(permite aplicar un degradado)
```
MiRango.SpecialCells(xlCellTypeFormulas).Interior.TintAndShade =0.3
```

El resultado en este caso sería:



El tono del bloque de formulas se vuelve más claro. En este rango las formulas forman un bloque contiguo, pero SpecialCells devuelve un rango discontinuo de celdas.

Ejemplo 3: Colorear y formatear el interior de un rango para datos y formulas

Set MiRango = Range("b2").CurrentRegion
(Aquí seleccionamos el rango con el que queremos trabajar)
MiRango.Interior.Color=rgbMediumVioletRed
(Le damos color a todo el rango interior seleccionado)
MiRango.SpecialCells(xlCellTypeFormulas).Interior.TintAndShade=0.3
(A las celdas con formulas le cambiamos la intensidad)
MiRango.SpecialCells(xlCellTypeConstants,xlTextValues).Font.Color=rgbWhite
(ahora cambiamos el color de la fuente a blanco, pero estaria merjor si ademas fuera en negrita)
MiRango.SpecialCells(xlCellTypeConstants).Font.Bold=True
(Cambiamos el color de la fuente a negrita)

Así mostramos finalmente el resultado final.

	Precio			Descuento
Cantidad	5 €	10 €	15 €	10%
10	45 €	90 €	135 €	
20	90 €	180 €	270 €	
30	135 €	270 €	405 €	
40	180 €	360 €	540 €	
50	225 €	450 €	675 €	

7.5 Resumen Esquema: Formatear celdas en Excel (VBA).

http://www.xltoday.net/vba_ejemplos_formatos.asp

Excel pone a disposición un montón de formatos. Abajo presentamos como modificar algunos de ellos a través de macros Excel VBA.

7.5.1 Redondear celdas

```
Set Area = Selection
For Each Cell In Area
  z = Round(Cell, 2)
    Cell.Value = z
  Cell.NumberFormat = "#,##0.00"
Next Cell
```

7.5.2 Formatear fuente

```
Cells.Select
With Selection.Font
  .Name = "MS Sans Serif"
  .Size = 10
End With
```

7.5.3 Líneas de división

```
ActiveWindow.DisplayGridlines = False
```

7.5.4 Indice de colores

```
ActiveWorkbook.Colors(44) = RGB(236, 235, 194) 'verde
```

7.5.5 Colorear rango

```
Range("A1:B10").Interior.ColorIndex = 44
```

7.5.6 Cambiar entre estilos A1 / RC

```
Application.ReferenceStyle = xlA1Application.ReferenceStyle = xlR1C1
```

8 Resumen. Puntos Clave.

8.1 Sobre Objetos. Métodos

Los objetos preceden a los métodos y están separados por un punto (**objeto.método**). Gran parte de las colecciones de Excel utilizan el método Add para agregar un nuevo elemento a la colección.

8.2 Sobre Objetos. Propiedades

Igualmente los objetos preceden a las propiedades y están separados por un punto (**objeto.propiedad**). En el caso de trabajar con la ventana Inmediato insertamos en primer lugar el símbolo interrogación (**?**) porque queremos saber el valor de la propiedad.

Señalar que en la ventana Inmediato cuando escribe un signo de interrogación acompañado de cualquier elemento que devuelve un valor, el valor aparece en la siguiente línea.

Entendemos que cuando se ejecuta el método Add no necesitamos anteponer un signo de interrogación porque este es un método que no restituye valores.

8.3 Símbolos a tener en cuenta. Como diferenciar métodos y propiedades

 Este símbolo representa los métodos o acciones disponibles asociadas al objeto.

 Este símbolo representa las propiedades disponibles asociadas al objeto.

8.4 Referencias específicas a objetos (Item "nombre")

A través del Item o posición que ocupa dentro del objeto.

Para hacer referencia a un elemento concreto de un objeto podemos hacerlo a través de la propiedad Item como mostramos a continuación:

Workbooks.Item(1)

Esta instrucción hace referencia al primer libro abierto de la colección Workbook

Así Item es la propiedad de una colección y hace referencia a un único elemento de la colección. Por tano la propiedad Item especifica la posición que un libro ocupa en la colección Workbook.

Destacar que estas dos instrucciones son similares:

?Workbooks.Item(1).Name
?Workbooks (1).Name (esta es la más usada)

Igualmente:

?Workbooks.Item(1).Close
?Workbooks (1).Close (esta es la mas usada)

En resumen, una vez que la propiedad Item ha hecho alusión a un solo objeto Workbook (libro), podemos comunicarnos con él a través de los métodos y propiedades que el entiende. Con la propiedad Name, podemos saber el nombre del libro y con el método Close podremos cerrarlo.

A través del Nombre del objeto.

Otra forma de referirnos a un elemento concreto de una colección objeto como es workbooks es a través del nombre, en este caso cada vez que utilicemos el nombre deberemos usarlo entre comillas, al contrario de lo que sucede cuando especificamos su posición. workbooks("libro5").Activate o workbooks("libro5").Close

A través de ActiveWorkbook.

Otra forma de hacer referencia a un libro es a través de la opción ActiveWorkbook, que significa libro activo, de esta forma si quisiéramos cerrar el libro activo simplemente tendríamos que poner la siguiente instrucción: ActiveWorkbook.Close

8.5 La función Array

Array es una función que nos permite tratar muchos valores como si fueran uno solo. No todas las colecciones permiten usar la función Array para crear selecciones. Así la colección Workbooks no admite esta función porque no se pueden seleccionar varios libros a la vez.

8.6 Sobre el Objeto Range. Cells, Columns y Rows.

Cells, Colum, Rows y Columns son propiedades del objeto Range y por tanto ninguna tiene su propia clase de objetos. Solo son maneras diferentes de representar el objeto Range.

8.7 Manipulando Rangos. Propiedades Offset, Resize,EntireRow, EntireColum, CurrentRegion y el método Intersect

Las propiedades Offset, Resize,EntireRow, EntireColum, CurrentRegion y el método Intersect son herramientas flexibles que sirven para calcular nuevos objetos Range, basados en un Rango inicial determinado.

Por lo general la mejor manera de trabajar dentro de un rango es usar en primer lugar la propiedad CurrentRegion para definir el rango de base y enseguida utilizar la propiedad offset y el método Intersect para manipular el rango que acabamos de señalar.

8.8 Otras consideraciones

- Debemos utilizar la ventana Inmediato para probar instrucciones antes de introducirlas en las macros.
- Es aconsejable aprovechar al máximo las macros que asignan valores a las propiedades de los objetos.

9 Bibliografia.

http://www.fermu.com/content/view/397/2/lang.es/

http://www.observatorio.cnice.mec.es Programación Excel Observatorio Tecnologico

Aprendiendo Programación con Microsofot Excel 2000 en 24 Horas. Sharon Podlin. Editorial Pearson Educación 2001.