

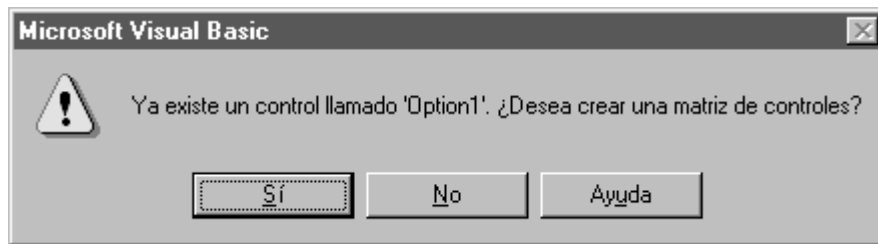
LECCIÓN 8

En esta lección hablaremos de unos objetos que ya hemos utilizado en la lección anterior, pero no vimos ni como funcionaban, ni como se utilizaban. Estamos hablando de los cuadros de mensajes.

¿Qué son los cuadros de mensajes?

En muchas ocasiones cuando realizamos acciones con cualquier programa de **Windows** nos aparecen pequeñas ventanas de información o de error.

Este por ejemplo, es un cuadro de diálogo con el que ya hemos trabajado en lecciones anteriores.



Podemos decir que tenemos dos tipos de cuadros de mensajes con los que podemos trabajar: los cuadros de mensajes propiamente dichos y los de entrada.

Estos cuadros los utilizaremos para mostrar algún tipo de mensaje al usuario de la aplicación, ya sea de error, aviso o de cualquier otro tipo.

Los cuadros de entrada en cambio son ventanas en las que se espera que el usuario escriba algún tipo de texto que nos servirá para continuar con la aplicación.

En ambos cuadros podremos modificar diferentes elementos como el *título*, el *icono*, los *mensajes de los botones*, la *cantidad de botones*, sus *funciones* y *otras características* que veremos a medida que vayamos hablando de cada uno de los tipos de cuadros.

Cuadros de mensajes (MsgBox)

Estos cuadros los utilizaremos para mostrar mensajes o para obtener por parte del usuario respuestas sobre determinadas acciones.

Vamos a enumerar las diferentes partes que podremos modificar en nuestros cuadros de mensajes.

Estos cuadros constan de un **título** en la parte superior de la pantalla. Estos cuadros carecen de menú de control y solo disponen del botón **cerrar** ya que no se puede modificar su tamaño. Suele aparecer un **icono** en la parte izquierda de la ventana. Este icono nos ayuda a identificar de que tipo es el mensaje. Suele aparecer un **mensaje** en el centro de la ventana. En la parte inferior aparecen los diferentes **botones**. Pueden aparecer 1, 2 o 3 botones con diferente texto en su interior.

Más adelante veremos como personalizar todas estas opciones.



Aquí vemos las partes de un mensaje de error de **Visual Basic**.

Sintaxis de un MsgBox

Nosotros mediante un **MsgBox** podemos saber que botón pulsa el usuario. Cada uno de los diferentes botones tiene un valor que se almacenará en una variable con la que después podremos trabajar.

Pongamos el caso del **MsgBox** anterior, si el usuario pulsa el botón **Aceptar** cerraremos dicho mensaje y detendremos la ejecución del programa, mientras que si pulsamos en **Ayuda** mostraremos una ayuda sobre este error. También hay **MsgBox** que no nos interesa saber que botón ha pulsado el usuario con lo que no hace falta almacenar el valor del botón en ninguna variable, este puede ser el caso de un mensaje de error en el que solo aparecerá un botón para cerrar el cuadro de mensaje.

Vamos a ver primero la sintaxis general de esta instrucción:

MsgBox Mensaje [, Botones e iconos][, Título]

Las partes entre corchetes indican parámetros opcionales.

Si no deseamos saber que botón ha pulsado el usuario de la aplicación deberemos poner la instrucción tal y como hemos indicado en la sintaxis anterior. En cambio si deseamos conocer que botón a pulsado y actuar en consecuencias deberemos almacenar en una variable el valor que se genera al pulsar dicho botón, entonces deberemos modificar la sintaxis de esta forma:

Valor = MsgBox (Mensaje [, Botones e iconos][, Título])

Observa que hemos añadido unos paréntesis que engloban a todas las opciones del **MsgBox**.

Valor: esta será la variable en la que se almacenará el valor del botón pulsado en el mensaje.

Observa que hemos insertado el signo igual ya que lo que estamos pasando el valor del botón pulsado a la variable **Valor**.

El **mensaje** es la única opción obligatoria que deberemos poner en un **MsgBox**.

Título: si indicamos algún título, este nos aparecerá en el título de la nueva ventana. Si por lo contrario no indicamos título, nos aparecerá el nombre de la aplicación actual.

Botones e iconos: aquí pondremos un valor que nos servirá para especificar que icono y que botones queremos que nos aparezcan.





Botones e iconos del mensaje

Como ya hemos dicho anteriormente en este lugar deberemos indicar un valor que nos indicará el "tipo" de nuestro mensaje. Este valor se obtendrá sumando 4 valores diferentes de 4 tablas que presentamos a continuación:

Botones

<i>Botones a mostrar</i>	<i>Valor</i>
Aceptar	0
Aceptar y cancelar	1
Anular, Reintentar e Ignorar	2
Sí, No y Cancelar	3
Sí y No	4
Reintentar y Cancelar	5

Iconos

<i>Iconos a mostrar</i>	<i>Valor</i>
	16
	32
	48
	64

Botón activado por defecto

<i>Botón por defecto</i>	<i>Valor</i>
Primero	0
Segundo	256
Tercero	512
Cuarto	768

Modalidad del mensaje

<i>Modalidad</i>	<i>Valor</i>
Aplicación modal	0
Sistema modal	4096

Para conseguir el valor que deberemos poner en el apartado **Botones e iconos** de nuestra sintaxis escogeremos un valor de cada uno de los diferentes grupos que hemos visto anteriormente y los sumaremos.

Antes de poner un ejemplo vamos a explicar cada uno de los diferentes grupos:

Botones: aquí tenemos una lista de las diferentes combinaciones de botones que podemos mostrar en nuestro mensaje.

Iconos: esta es una lista de los cuatro posibles iconos que podemos mostrar.

Botón activado por defecto: nosotros podremos indicar cual de los botones que tenemos en el mensaje se active en el momento de pulsar la tecla **Enter**.

Modalidad del mensaje: Vamos a definir las dos modalidades.

Aplicación modal: el usuario deberá "contestar" al cuadro de mensaje pulsando sobre alguno de los botones o cerrando dicho cuadro antes de proseguir con la aplicación actual. Con esta opción el usuario podrá seguir utilizando cualquier otra aplicación. Una vez contestada la pregunta el programa continuará según la respuesta.

Sistema modal: El usuario no podrá continuar el trabajo con ninguna aplicación hasta que se conteste el cuadro de mensaje actual. Esta no es una opción muy utilizada ya que se bloquean el resto de aplicaciones hasta que se responde el mensaje.

Vamos a ver como utilizar los objetos **MsgBox** en una aplicación. En este ejemplo veremos como diseñar nuestro mensaje.

Generar un MsgBox

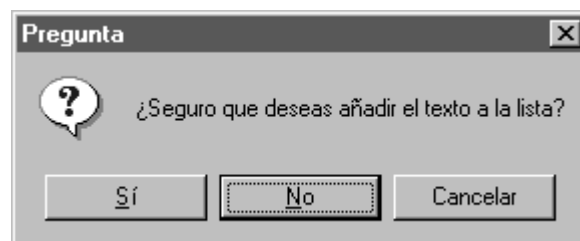
Vamos a crear una simple aplicación en la que tendremos tres objetos, un **TextBox**, un **ListBox** y un **CommandButton**. Esta aplicación nos permitirá escribir algo en el **TextBox** y al pulsar el **CommandButton** nos deberá aparecer un **MsgBox** con la pregunta: *¿Estás seguro que deseas añadir este elemento?*. Si el usuario responde afirmativamente el contenido del **TextBox** pasará al **ListBox**, si el usuario responde negativamente, opción que aparecerá marcada por defecto, no se añadirá el texto al **ListBox**, pero nos aparecerá un nuevo **MsgBox** indicando que no se añadirá ningún elemento a la lista.

. Práctica 1

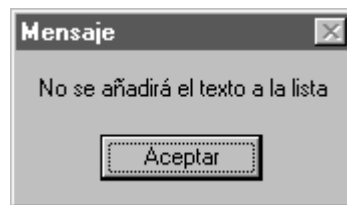
1. Crea un nuevo formulario.
2. Inserta un **TextBox**, borra el contenido y ponle como **(Nombre) Texto**.
3. Inserta un **ListBox**, ponle como **(Nombre) Lista**.
4. Inserta un **CommandButton** cámbiale el **(Nombre)** por **Insertar**. Cambia también la propiedad **caption** por **Insertar**.

Vamos a pasar a crear el código para que nos aparezca el mensaje deseado.

Nuestro **MsgBox** deberá mostrar como título: **Pregunta**. El mensaje interior deberá ser: **¿Seguro que deseas añadir el texto a la lista?**. Como icono nos aparecerá un signo de interrogación y nos deberán aparecer tres botones: **Sí**, **No** y **Cancelar**.



El primer mensaje deberá ser como este:



En cambio el segundo mensaje que mostrará esta aplicación tendrá este otro aspecto:

Vamos a ver como podemos generar el primero de los dos **MsgBox**.

En primer lugar vamos a calcular el valor para que nos aparezcan los **3 botones**, el **icono** y el **segundo botón como predeterminado**. Como ya hemos dicho anteriormente deberemos escoger un valor de cada uno de los cuatro grupos que hemos escrito anteriormente.

En el primer grupo tenemos los botones que deseamos aparezcan en el **MsgBox**, al mirar la tabla vemos que los botones **Sí**, **No** y **Cancelar** tienen como valor el **3**. Para que aparezca el icono de la interrogación deberemos mirar en el segundo grupo, este icono tiene como valor **32**. El tercer grupo nos determinará cual de los botones queremos que esté como predeterminado, en este caso será el **segundo**, mirando en la tabla veremos que tiene como valor **256**. Nosotros queremos que el mensaje sea **modal** a la aplicación, por lo tanto el valor del cuarto grupo es un **0**. Si sumamos los cuatro valores nos da: **3+32+256+0=291**

Ahora que ya sabemos el valor que debemos poner dentro de la definición de nuestro primer **MsgBox**.

Vamos a ver como quedaría definitivamente el código.

Recuerda que deseamos conocer la respuesta del usuario por lo que necesitamos almacenar el valor del botón pulsado.

5. Haz doble clic dentro de nuestro botón.

6. Escribe el siguiente código. (Por motivos de espacio en nuestro manual el código aparece en dos líneas, pero en Visual Basic se debería escribir en una sola).

Respuesta = MsgBox(«¿Seguro que deseas añadir el texto a la lista?», 291, «Pregunta»)

Fíjate que hemos creado una variable llamada **respuesta** para almacenar el valor del botón pulsado.

Vamos a ver como trabajar con estos valores.

Valores de retorno de los botones

Vamos a ver a continuación otra tabla con los valores que se devuelven al pulsar los diferentes botones.

Botón	Valor
Aceptar	1
Cancelar	2
Anular	3
Reintentar	4
Ignorar	5
Sí	6

No**7**

En nuestro ejemplo solo utilizaremos **2** valores el **6** para el **Sí** y el **7** para el **No**.

7. Modifica el código que tienes dentro del botón para que sea como este:

```
Private Sub Insertar_Click()
    Respuesta = MsgBox(«¿Seguro que deseas añadir el texto a la lista?», _ 291,
«Pregunta»)
    If Respuesta = 6 Then
        Lista.AddItem Texto.Text
    End If
End Sub
```

En el momento que escribimos el símbolo **_** al final de una línea, **Visual Basic** entiende que la siguiente línea de código va seguida. No la entiende como líneas separadas. Tú puedes escribir el código en una misma línea. Una cosa que deberás tener en cuenta es que antes de este símbolo deberá existir un espacio en blanco.

8. Inicia una ejecución de prueba.

9. Escribe algo en la casilla de texto.

10. Pulsa el botón **Insertar**.

11. Seguidamente aparecerá el **MsgBox** que hemos creado.

Observa como el botón **No** aparece remarcado. Si pulsamos **Intro** este será el botón que actuará.

12. Pulsa **Intro** y observa como no ocurre nada. (Más adelante insertaremos el código para que aparezca el otro cuadro de mensaje)

13. Vuelve a pulsar el botón **Insertar**.

14. Ahora pulsa en el botón **Sí**.

El **MsgBox** desaparecerá y el texto pasará a estar dentro de la **lista**.

Ahora vamos a insertar el código necesario para que nos aparezca el segundo mensaje.

15. Modifica el código del botón para que quede de esta forma:

```
Private Sub Insertar_Click()
    Respuesta = MsgBox(«¿Seguro que deseas añadir el texto a la lista?», _ 291,
«Pregunta»)
    If Respuesta = 6 Then
        Lista.AddItem Texto.Text
    End If
    If Respuesta = 7 Then MsgBox «No se añadirá el texto a la lista», 0, _ «Mensaje»
End Sub
```

16. Observa detenidamente las diferencias que existen entre los dos tipos de **MsgBox** que hemos colocado en nuestro código.

Como en el primer mensaje nos interesa conocer cual es la tecla que ha pulsado el usuario, ponemos todas las opciones del **MsgBox** entre paréntesis y además asignamos esta estructura a una variable.

En cambio en el segundo **MsgBox** no nos interesa saber el valor del botón pulsado con lo que no asignamos ninguna variable.

Observa también que hemos puesto como valor **0** ya que solo queremos que aparezca un botón y ningún icono.

17. Haz una ejecución de prueba.

18. Escribe algo en el **TextBox**, pulsa en **Insertar**.

19. Contesta afirmativamente.

20. Vuelve a pulsar en **Insertar**.

21. Ahora contesta **Negativamente** y observa como aparece un nuevo **MsgBox**.

22. Acepta el **MsgBox** actual y finaliza la ejecución.

Vamos a depurar un poco el código de esta aplicación. Para facilitar la lectura del código vamos a crear unas constantes que tengan como valor **6** y **7** para que así durante el código no tengamos que estar pensando a que botones pertenecen dichos valores.

23. Define dos constantes, a la primera le llamamos **Sí** y le asignamos como valor **6** y a la segunda le llamamos **No** y le asignamos un valor de **7**.

Recuerda donde deberás declarar dichas constantes.

Const Sí = 6
Const No = 7

Ahora vamos a pasar a cambiar el código de la aplicación para que quede un poco más entendible:

24. Accede al código del botón y realiza los cambios pertinentes para que quede como el siguiente código:

```
Private Sub Insertar_Click()
    Respuesta = MsgBox(«¿Seguro que deseas añadir el texto a la lista?», _ 291,
«Pregunta»)
    If Respuesta = Sí Then
        Lista.AddItem Texto.Text
    End If
    If Respuesta = No Then MsgBox «No se añadirá el texto a la lista», 0, _ «Mensaje»
End Sub
```

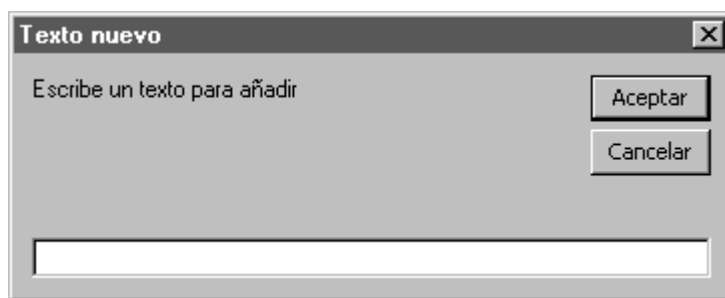
Ahora el código queda un poco más comprensible ya que no aparecen valores por medio.

Observa que en este código no hemos escrito nada para cuando el usuario pulsa el botón **Cancelar** ya que no deseamos que se realice ningún tipo de acción.

Una vez visto estos mensajes, vamos a ver como podemos introducir datos a través de otro tipo de mensajes.

Solicitud de datos (InputBox)

Vamos a ver una forma de pedir al usuario datos utilizando un nuevo tipo de ventana de mensajes.



En un **InputBox** sólo podremos modificar el texto que aparece en el **mensaje**, el **título** de la ventana de mensaje, si deseamos que aparezca algún tipo de **cadena** como predeterminada y la **posición** de la pantalla en la que deseamos que aparezca dicha ventana.

En la imagen anterior vemos un ejemplo de **InputBox** con las diferentes partes que lo componen.

Sintaxis de un InputBox

Al utilizar este tipo de ventana tenemos que asignar el contenido del cuadro de mensaje a una variable donde se almacenará lo que el usuario escriba dentro del **InputBox**.

Si el usuario pulsa **Aceptar**, el contenido del cuadro de texto pasará a la variable asignada para este efecto, mientras que si el usuario pulsa en **Cancelar** no se añade nada a variable.

Variable = InputBox (Mensaje, Título)

Vamos a ver como podemos trabajar con un **InputBox**.

Generar un InputBox

25. Borra el **TextBox** que teníamos en el formulario que hemos estado utilizando anteriormente.

26. Accede al código del botón **Insertar** y borra todo el código que habíamos escrito anteriormente.

Lo que vamos a pretender ahora es que al pulsar el botón **Insertar** nos aparezca un **InputBox** como el que hemos visto anteriormente. Dentro de este **InputBox** escribiremos el texto que deseamos añadir a la lista. Al pulsar **Aceptar** este texto pasará a la lista, mientras que si pulsamos en el botón **Cancelar** no ocurrirá nada.

Primero vamos a ver que debemos hacer para que nos aparezca el **InputBox** que hemos visto anteriormente.

27. Escribe el siguiente código dentro del botón **Insertar**.

```
Private Sub Insertar_Click()  
    Nuevo = InputBox(«Escribe un texto para añadir», «Texto nuevo»)  
End Sub
```

Observa que creamos una variable llamada **Nuevo** en la que almacenamos lo que escribe el usuario de la aplicación dentro del **InputBox**.

28. Realiza una ejecución de prueba y pulsa sobre **Insertar**.

29. Pulsa en **Aceptar**.

Podrás ver que no ocurre nada, ya que no hemos escrito el código para añadir el texto escrito en el **InputBox** dentro de la lista.

30. Detén la ejecución y accede al código del botón **Insertar**.

31. Modifica el código para que quede de la siguiente forma:

```
Private Sub Insertar_Click()
    Nuevo = InputBox(«Escribe un texto para añadir», «Texto nuevo»)
    Lista.AddItem Nuevo
End Sub
```

32. Haz una ejecución de prueba.

33. Pulsa en **Insertar**.

34. Escribe cualquier cosa dentro del **InputBox**.

35. Pulsa en **Cancelar**.

Observa como aparentemente no ha pasado nada.

36. Pulsa nuevamente en **Insertar**.

37. Vuelve a escribir algo dentro del **InputBox**.

38. Ahora pulsa en **Aceptar**.

Podemos ver como se ha añadido el texto en la lista, pero no en la primera posición. ¿Por qué ha ocurrido esto?

Esto ocurre porque en el momento que nosotros pulsamos el botón **Cancelar** del **InputBox** se le asigna un espacio en blanco a la variable y esto es lo que pasamos a añadir a la lista en la línea siguiente.

Depurando el código

Vamos a ver como podemos aprovechar la cualidad de asignar un espacio en blanco al pulsar el botón **Cancelar** para depurar la aplicación.

En el momento en el que cuando el usuario pulsa el botón **cancelar** no se debería añadir nada en la lista, esto lo podemos solucionar preguntando si la variable que se genera en el **InputBox** es diferente a "" con lo que se añadirá el texto a la lista. Vamos a ver como podemos hacer esto.

39. Detén la ejecución de la aplicación y accede al código del botón **Insertar**.

40. Modifícalo para que quede así:

```
Private Sub Insertar_Click()
    Nuevo = InputBox(«Escribe un texto para añadir», «Texto nuevo»)
    If Nuevo <> «» Then Lista.AddItem Nuevo
End Sub
```

41. Ejecuta la aplicación y observa como funciona.

Añadir sin parar

Imagina que deseas añadir utilizando este sistema varios elementos a la lista. Si tenemos el código como hasta este momento, para añadir una nueva entrada a la lista deberíamos ir pulsando consecutivamente a **Insertar** y después a **Aceptar** vamos a ver un sistema, utilizando un bucle, para que se repita la aparición de un **InputBox** y la inserción del elemento escrito a la lista hasta que el usuario no escriba nada dentro del **InputBox**.

42. Modifica el código para que quede de la siguiente forma:

```
Private Sub Insertar_Click()
    Do
        Nuevo = InputBox("Escribe un texto para añadir", "Texto nuevo")
        If Nuevo <> «» Then Lista.AddItem Nuevo
    Loop Until Nuevo = ""
End Sub
```

Este bucle nos repetirá la instrucción hasta que pulsamos la tecla **Cancelar** o **Aceptar** teniendo el cuadro de texto vacío.

43. Realiza una ejecución de prueba.

44. Accede al botón **Insertar**.

45. Escribe cualquier cosa, pulsa en **Aceptar**.

Seguidamente aparecerá otro **InputBox** con el cuadro de texto vacío. Si miras la lista podrás ver como el texto anterior se ha añadido. Si no ves la lista puedes mover el **InputBox** como si se tratase de cualquier otra ventana de **Windows**.

En el momento en el que no desees introducir más palabras a nuestra lista pulsa **Aceptar** sin haber escrito nada en el **InputBox**.

46. Detén la ejecución de prueba.

Texto por defecto

Si dentro del cuadro de texto de nuestro **InputBox** deseamos que aparezca algún tipo de texto por defecto lo podemos hacer de una forma muy sencilla.

Imagina que deseamos que en nuestra aplicación, siempre que aparece el **InputBox** aparezca la palabra **Texto** dentro del cuadro de texto.

47. Accede al código del botón **Insertar**.

48. Modifica el código del **InputBox** para que quede de la siguiente forma:

```
Nuevo = InputBox(«Escribe un texto para añadir», «Texto nuevo», «Texto»)
```

49. Ejecuta la aplicación y pulsa en **Insertar**.

Observa como aparece la palabra **Texto** seleccionada dentro del **InputBox**. Si pulsamos en **Aceptar** la palabra **Texto** pasará a la lista, si no nos interesa esta palabra la podemos sustituir por la que queramos. Si pulsamos en **Cancelar** no se añadirá nada a la lista.

Ahora vamos a ver como podemos colocar el **InputBox** en diferentes lugares de

la pantalla.

Cambiar la posición del InputBox

Si no indicamos en que posición deseamos situar el **InputBox** aparecerá en el centro de la pantalla. Pero puede ser que al estar situado en el centro nos oculte algún dato importante del formulario con el que estamos trabajando, con lo que podremos indicar en que lugar de la pantalla deseamos que aparezca.

El primer valor que introduciremos es la distancia entre el borde izquierdo del **InputBox** y el borde izquierdo de la pantalla.

El segundo valor es la distancia entre el borde superior del borde del **InputBox** con la parte superior de la pantalla.

50. Realiza los pasos necesarios para que el formulario aparezca centrado en la pantalla.

51. Ahora coloca los valores necesarios dentro del **InputBox** para que al aparecer este podamos ver con claridad la lista del formulario.

De esta manera podremos ver como se añaden los valores escritos dentro del **InputBox** en la lista sin necesidad de mover esta por la pantalla.

Por ejemplo:

```
Nuevo = InputBox(«Escribe un texto para añadir», «Texto nuevo», _ «Texto», 3000, 1500)
```

52. Realiza todas las ejecuciones de prueba que necesites hasta que consigas encontrar el lugar ideal.

Si te fijas en esta estructura estamos utilizando 5 parámetros diferentes, pero que pasaría por ejemplo si no deseamos que aparezca un texto predefinido.

Ausencia de elementos

Imagina que en el código anterior deseamos que el **InputBox** aparezca en un lugar determinado de la pantalla, pero no deseamos que aparezca un texto predeterminado.

Este problema se soluciona muy fácil. Solo deberás hacer como si estuviera el parámetro que quitamos respetando así la cantidad de comas que existen dentro del **InputBox** con todos los parámetros escritos.

Vamos a ver como quedaría el código, sin que aparezca **Texto** como palabra determinada.

```
Nuevo = InputBox(«Escribe un texto para añadir», «Texto nuevo», , 3000, _ 1500)
```

Observa como antes del valor de posición horizontal existen dos comas. Entre estas comas es donde estaba escrita la palabra que aparecía como predefinida en el **InputBox**.

En esta lección hemos aprendido como utilizar cuadros de diálogo de una forma fácil y rápida. Estos elementos se deben usar para hacer que el usuario encuentre la aplicación lo más fácil posible sin tener que estar intuyendo para que se utilizan los botones y los objetos que aparecen en ella.

Recomiendo utilizar los **MsgBox** para aclarar todo lo que se pueda, los errores y

las decisiones que debe tomar el usuario en determinados momentos. A partir de este momento espero que formen parte de las aplicaciones que realices y te familiarices con su funcionamiento.

Fin lección 8