1 MICROSOFT WINDOWS LOGO 6.5a → GUÍA PRÁCTICA

1.1.- Introducción

El lenguaje Logo fue creado en la década de los 60 por Seymour Papert para que los niños aprendieran ideas matemáticas programando con este lenguaje. En las siguientes décadas él y su equipo continuaron con su desarrollo y tratando de crear un programa con un entorno gráfico que fuera a la vez potente y fácil de usar.

En 1994 MSWLogo fue modificado para permitir el control por ordenador a través de los puertos serie y paralelo.

Todas las versiones del software Logo para los sistemas operativos Unix, MS-Dos, Macintosh y Windows desarrolladas inicialmente por la Universidad de California (Berkeley), son gratuitas y pueden ser copiadas sin ningún tipo de restricción por instituciones educativas.

El símbolo de Logo es una tortuga robotizada que se mueve bajo el control de un ordenador dibujando a medida que se desplaza por la pantalla. En algunas versiones de este lenguaje la tortuga ha evolucionado hasta convertirse en otro tipo de objetos. Por ejemplo, en MSWLogo se ha convertido en un triángulo.

MSWLogo se puede utilizar para múltiples utilidades: para dibujar, para crear sonidos, para manipular una controladora a través del ordenador, etc.

La versión que se intentará explicar en esta documentación es l*a versión que presenta el Centro Nacional de Información y Comunicación Educativa (CNICE)*, que *es una versión del MSWLogo 6.5a traducida al castellano.*

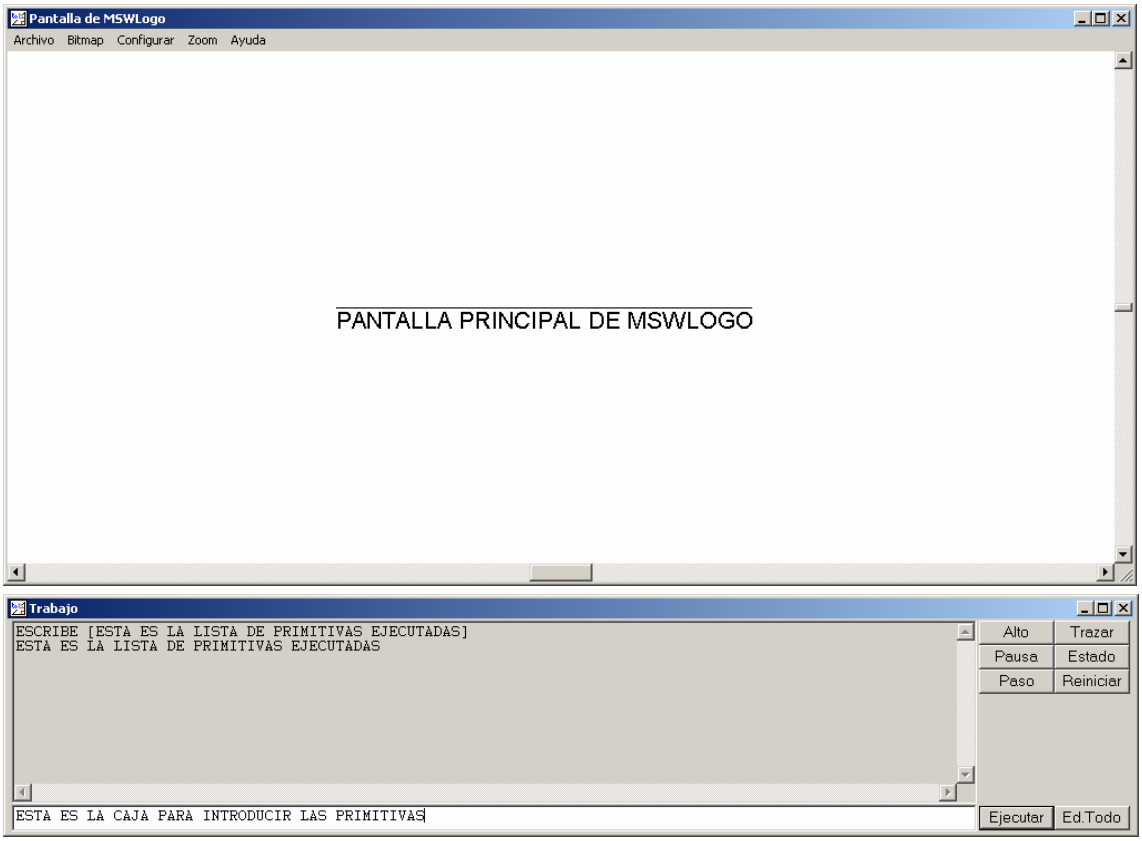
1.2.- Introducción y uso de Primitivas en MSWLogo

MswLogo es un lenguaje interpretado, es decir, las órdenes introducidas por el usuario son interpretadas por el ordenador y ejecutadas inmediatamente por orden. En cambio, los programas "compilados" son convertidos primero a código máquina antes de que cualquier parte del programa pueda empezar a funcionar.

La pantalla de MSWLogo se divide en dos partes:

1. La pantalla principal. En esta ventana es donde se dibuja, se cargan imágenes, etc.

2. La ventana de trabajo o de comandos. La ventana de comandos se divide a su vez en dos partes:
    2.a) La caja de entrada de datos o primitivas. Se pueden usar minúsculas o mayúsculas para las primitivas puesto que Logo no hace distinciones entre unas y otras. Las primitivas se ejecutan después de ser introducidas en la ventana de trabajo y de pulsar la tecla *ENTER* o de hacer clic en el botón *Ejecutar*.
    2.b) Lista de comandos o primitivas ejecutadas. Cada primitiva se graba en una lista de órdenes y comandos de la ventana superior a la de entrada de datos.



Algunas **primitivas** tienen abreviaturas y otras no, por ejemplo, la primitiva *borratexto* tiene la abreviatura *bt*. Se puede usar una u otra indistintamente y el resultado será el mismo.

Para salir de MSWLogo se pueden seguir tres caminos:

1. Usar la primitiva *adios*.
2. Seleccionar la opción *Salir* del menú *Archivo*.
3. Pinchar en el aspa de la ventana.

1.3.- Listado de primitivas seleccionadas de MSWLogo

1.3.1 Primitivas para dibujar

PRIMITIVA	MÉTODO ABREVIADO	DESCRIPCIÓN
AVANZA 200	*AV 200*	La tortuga avanza el número especificado de unidades.
RETROCEDE 150	*RE 150*	La tortuga retrocede el número especificado de unidades.
GIRADERECHA 90	*GD 90* *VIRA 90*	La tortuga gira en el sentido de las agujas del reloj el número de grados especificado.
GIRAIZQUIERDA 45	*GI 45*	La tortuga gira en sentido contrario a las agujas del reloj el ángulo especificado.
PONLAPIZ	*PLA*	Pone el lápiz BAJADO y el modo PINTA (PINTA es el modo normal de la tortuga para dibujar).
GOMA	*GO*	Pone el lápiz BAJADO y el modo a BORRA (la tortuga va borrando por donde se mueve).
SUBELAPIZ	*SL*	Pone el lápiz LEVANTADO, sin cambiar su modo.
BAJALAPIZ	*BL*	Pone el lápiz BAJADO, sin cambiar su modo.
CENTRO		Devuelve a la tortuga al centro de la pantalla sin borrarla. Hay que utilizar la primitiva SUBELAPIZ para evitar que dibuje mientras va hacia el centro.
PONCOLORLAPIZ [... ... ...]	*PONCL [... ... ...]*	Pone el lápiz del color determinado de acuerdo con: PONCOLORLAPIZ [255 000 000] = Rojo PONCOLORLAPIZ [000 255 000] = Verde PONCOLORLAPIZ [000 000 255] = Azul
OCULTATORTUGA	*OT*	Oculta la tortuga (el triángulo) en la pantalla.
MUESTRATORTUGA	*MT*	Muestra la tortuga (el triángulo) en la pantalla

BORRAPANTALLA	*BP*	Borra la pantalla y coloca a la tortuga en el centro.
ROTULA [HOLA]	*RO [HOLA]*	Escribe un texto en la dirección en que se encuentra la tortuga.
PONGROSOR [10 10]	*PONG [10 10]*	Coloca el ancho del trazado del lápiz y la altura especificados. MSWLogo utiliza solamente el valor de la anchura.

1.3.2 Primitivas para escribir, editar y trabajar con ficheros

PRIMITIVA	MÉTODO ABREVIADO	DESCRIPCIÓN
MUESTRA objeto		Comando que escribe la entrada o entradas en la salida de escritura que actualmente esté (inicialmente es el terminal).
ESCRIBE objeto	*ES objeto*	Comando que escribe la entrada o entradas en la salida de escritura que actualmente esté (inicialmente es el terminal). Si la entrada es una lista, los corchetes más externos no se muestran, pero sí los de los de las sublistas.
EDITA CONTENIDO	*EDITATODO*	Comando que abre la ventana del editor y permite modifica las definiciones de los elementos (procedimientos, variables, etc.). Hace lo mismo que el botón *Ed. Todo*.
EDITA PROCEDIMIENTOS	*EDITAPROCEDIMIENTOS*	Comando que abre la ventana del editor y permite modifica las definiciones de los procedimientos.
CARGA "nombfichero		Comando que lee las instrucciones del fichero dado y las ejecuta.
GUARDA "nombfichero		Comando que guarda en el fichero dado las definiciones de todos los procedimientos, variables y listas de propiedades sin tapar.

1.3.3 Primitivas para trabajar con controladoras

Estas primitivas sirven para trabajar con las controladoras ENCONOR y CNICE.

Hay que tener en cuenta que estas controladoras no tienen el mismo número de entradas analógicas, por ello hay algunas primitivas que no se pueden utilizar. Además la controladora CNICE no tiene salidas analógicas y por tanto las primitivas que para este tipo de salidas no se podrán usar con ella.

PRIMITIVAS PARA TRABAJAR CON LAS SALIDAS DIGITALES	
PRIMITIVA	**DESCRIPCIÓN**
M1 "D	Hace girar en un determinado sentido un motor conectado a dos de las salidas digitales (las salidas elegidas dependen del tipo de controladora). Si el actuador es una bombilla, relé o electroimán, simplemente lo activa.
M1 "I	Hace girar en sentido contrario a la primitiva anterior un motor conectado a dos de las salidas digitales. No se diferencia de la primitiva anterior si están conectados otros actuadotes (bombilla, relé, etc.)
M1 "P	Desactiva el actuador que esté conectado a dos de las salidas digitales (para conseguirlo se desactivan ambas salidas).
M2 "D, M2 "I, M2 "P ***M3 "D, M3 "I, M3 "P*** ***M4 "D, M4 "I, M2 "P***	Lo mismo que las primitivas anteriores, aunque los actuadotes se conectan a los restantes pares de salidas digitales.
M?	Devuelve una lista con el estado de los cuatro motores. Ej.: MUESTRA M? *[P D D I]*
M :L	Permite activar/desactivar todos los motores simultáneamente. El parámetro *:L* tiene que ser una lista. Ej.: M [P P I D]→ Se desactivan los motores M1 y M2, M3 se hace girar a la Izquierda y M4 se hace girar a la derecha.
CONECTA NUM	Activa la salida digital indicada, siendo *:NUM* 1, 2, 3, 4, 5, 6, 7 o 8.
DESCONECTA NUM	Desactiva la salida digital indicada, siendo *:NUM* 1, 2, 3, 4, 5, 6, 7 o 8.
CONECTAR	Conecta todas las salidas digitales de la controladora.
DESCONECTAR	Desconecta todas las salidas digitales de la controladora.
CONECTADO?	Devuelve una lista de 4 elementos (uno por cada motor) cuyos valores son 0 ó 1 según estén activos o no.
SALIDA NUM	Controla individualmente o en conjunto cada una de las ocho salidas digitales. Envía a las salidas digitales el número binario equivalente al decimal que pongamos en *:NUM*. Para activar, por ejemplo, las salidas 1 y 8 se escribirá el comando: *SALIDA 129*
EnviaOcteto dato	Procedimiento para la compatibilidad con las primitivas

	proporcionadas por ENCONOR (tiene la misma función que la primitiva *SALIDA*). Controla individualmente o en conjunto cada una de las ocho salidas digitales. Envía a las salidas digitales el número binario equivalente al decimal que pongamos en *:dato*.
APAGA n	Procedimiento para la compatibilidad con las primitivas proporcionadas por ENCONOR. Desconecta la salida digital que se le indique en *:n*, dejando el resto de salidas digitales como estén.
SALIDA?	Devuelve una lista de ocho elementos con el estado de las ocho salidas digitales (1 si la salida está activada o 0 si está desactivada).
VS?	Devuelve un número decimal que indica qué salidas digitales están activadas. Por ejemplo, si devuelve 3 indica que las salidas digitales 1 y 3 están activadas.
segundos :tiempo	Produce un retardo o espera de tantos segundos como le indiquemos en *:tiempo*. Se pueden utilizar fracciones de segundo (Ej.: 0.2).

PRIMITIVAS PARA TRABAJAR CON LAS ENTRADAS DIGITALES	
PRIMITIVA	**DESCRIPCIÓN**
entrada n	Procedimiento para la compatibilidad con las primitivas proporcionadas por ENCONOR. Devuelve VERDADERO si la entrada digital indicada en :n está conectado o FALSO en caso contrario.
VE?	Devuelve un número decimal que indica qué entradas digitales están activadas. Por ejemplo, si ejecutamos la primitiva y esta devuelve 129, las entradas digitales 1 y 8 tienen valor 1.
SD entrada	Devuelve el valor del sensor digital expresado en *:entrada* (1, 2, 3, ... , 8), 1 si la entrada está activa o 0 en caso contrario.
SD?	Devuelve el estado de los ocho sensores digitales en forma de lista (0s ó 1s).
EsperaOn entrada	Deja el programa parado a la espera de que se active la entrada digital indicada en *:entrada*.
EsperaOff entrada	Deja el programa parado a la espera de que se desactive la entrada digital indicada en *:entrada*.

PRIMITIVAS PARA TRABAJAR CON LAS ENTRADAS ANALÓGICAS	
PRIMITIVA	**DESCRIPCIÓN**
leeanalogica dato	Procedimiento para la compatibilidad con las primitivas proporcionadas por ENCONOR. Lee y escribe en el puerto los datos necesarios para obtener el valor decimal (valor entre 0 y 255) que proporciona la entrada analógica indicada en *:dato* (que puede ser 1, 2, 3 ó 4. Si consigue leerlas, devuelve su valor. Para obtener el valor en voltios correspondiente al valor decimal devuelto por la primitiva se debe aplicar la siguiente fórmula: **Voltaje Entrada (V)= (Nº decimal * 5) / 256**
SAV	Devuelve directamente el valor en voltios del sensor analógico conectado a la entrada analógica 1.
SAW	Devuelve directamente el valor en voltios del sensor analógico conectado a la entrada analógica 2.
SAX	Devuelve directamente el valor en voltios del sensor analógico conectado a la entrada analógica 3.
SAY	Devuelve directamente el valor en voltios del sensor analógico conectado a la entrada analógica 4.
SAZ	Devuelve directamente el valor en voltios del sensor analógico conectado a la entrada analógica 5.
SA?	Devuelve una lista con el valor en voltios de los todos sensores analógicos.

PRIMITIVAS PARA TRABAJAR CON LAS SALIDAS ANALÓGICAS	
PRIMITIVA	**DESCRIPCIÓN**
salidaanalogica :NUM :VALOR	Escribe en el puerto los datos necesarios para poner en la salida analógica que se le indica en *:*NUM el valor en tensión indicado en *:VALOR* (varía entre 0 y 255). Para conocer el valor en tensión en la salida se utiliza la siguiente fórmula: **Voltaje Salida (Voltios) = :*VALOR* / 23.8**
VOLTAJE :NUM :VALOR	Fija en la salida analógica indicada en *:NUM* el valor en tensión especificado en :VALOR (varía entre 0 y 10.5).
VOLTAJE?	Devuelve una lista con el valor en voltios de todas las salidas analógicas.

PRIMITIVAS PARA TRABAJAR EN EL ENTORNO GRÁFICO	
PRIMITIVA	**DESCRIPCIÓN**

graficos	Crea las ventanas necesarias para manejar las entradas y salidas analógicas (primitiva *graficosan*) y digitales y la ventana para informar del estado de la comunicación entre el ordenador y la controladora (primitiva *informa*).
graficosan	*Crea una ventana para controlar las entradas y salidas analógicas.*
informa	Crea una ventana que informa de datos referentes a la comunicación con el puerto.

1.3.4 Primitivas de sonido

PRIMITIVA	DESCRIPCIÓN
TONO flujosonido	Emite un sonido con la frecuencia (hertzios) y duración (milésimas de segundo) indicados. *Flujosonido* tiene que ser una lista de pares [<frecuencia> <duración>].
SUENATONO frecuencia duracion	Emite un sonido con la frecuencia y duración indicadas.
ENCIENDESONIDO frecuencia	Emite un sonido cuya frecuencia coincide con el parámetro que se le pasa. El sonido continuará hasta que invoque un comando *APAGASONIDO*.
APAGASONIDO	Quitará un sonido que se haya puesto con el comando *ENCIENDESONIDO*.
SUENAWAVE ficherowave opciones	Reproduce un fichero de tipo .WAV según las opciones que se indiquen.

1.3.5 Primitiva para crear condicionales y bucles

PRIMITIVA	DESCRIPCIÓN
SI condicion instrucciones	Permite ejecutar una o varias instrucciones dependiendo si se cumple la condición
SISINO condicion instrucciones1 instrucciones2	Dependiendo si se cumple una condición ejecuta la primera lista de instrucciones, y si no se cumple ejecuta la siguiente lista de instrucciones
HAZ.HASTA instrucciones condicion	Evalúa las instrucciones repetidamente tantas veces hasta que se cumpla la condición. De esta manera se asegura que las instrucciones se ejecutan al menos una vez, antes de comprobar la condición. EQUIVALENTE a HASTA
HASTA condicion instrucciones	Repite las instrucciones tantas veces hasta que la condición expresada se cumpla.

HAZ.MIENTRAS instrucciones condicion	Evalúa las instrucciones repetidamente tantas veces como se cumpla la condición. De esta manera las instrucciones se pueden o no ejecutar , dependiendo si se cumple la condición. EQUIVALENTE A MIENTRAS
MIENTRAS condicion instrucciones	Repite las instrucciones tantas veces como la condición expresada se cumpla.
SIEMPRE instrucciones	Ejecuta las instrucciones de forma repetida.
REPITE numero instrucciones	Ejecuta las instrucciones tantas veces como se le indica en número

1.3.6 Primitiva para salir de MSWLogo

PRIMITIVA	DESCRIPCIÓN
ADIOS	Comando que sale de Logo, devolviendo el control al sistema operativo.

1.4.- Utilización de variables

Para asignar un valor a una variable en MSWLogo se utiliza la primitiva *HAZ* seguida del nombre de la variable y del valor que se le quiera dar a la variable. El nombre de la variable debe ir precedido de comillas dobles (“). La sintaxis es la siguiente:

HAZ “nombVariable valor

Por ejemplo:

HAZ “x 20
HAZ “listacolores [ROJO AMARILLO AZUL]

Para acceder al valor de una variable se debe anteponer al nombre de la variable dos puntos (:). Por ejemplo:

MUESTRA :x
ROTULA :listacolores

1.5.- Definición y utilización de procedimientos

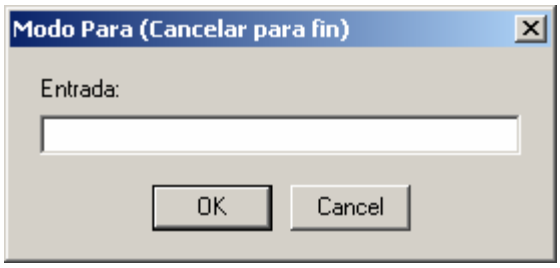
Para crear un procedimiento en MSWLogo hay que utilizar la palabra reservada *PARA* seguida del nombre del procedimiento y de los parámetros del mismo (si es que los tiene). Después se pondrá la definición del procedimiento y esta se terminará con la palabra reservada *FIN*.

```
PARA nombprocedimiento [lista de parámetros]
  ;Definición del procedimiento
  .
  .

FIN
```

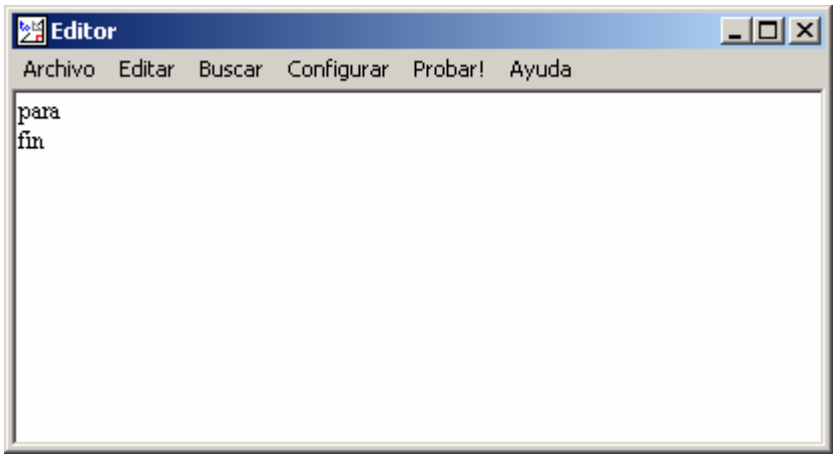
Para crear los procedimientos existen dos opciones:

1. Utilizar la Ventana de Trabajo: para que aparezca esta ventana se debe escribir en la caja de entrada de datos o primitivas *PARA nombre [lista de parámetros]* y luego pulsar la tecla *ENTER* o el botón *Ejecutar*.



   Luego se irán introduciendo en esta ventana una a una las sentencias de las que consta el procedimiento y se irá pulsando el botón *OK*. Para terminar con la definición del procedimiento se introducirá la primitiva *FIN*.

2. Utilizar la Ventana de Edición: para acceder a la Ventana de Edición se usará la primitiva *EDITATODO* o en el menú *Archivo → Editar*. Además de definir procedimientos, en la ventana de edición se pueden definir variables, propiedades, comentarios, etc. Cuando se ha acabado de definir el procedimiento se guardará en la opción del menú *Archivo → Guardar*.



A continuación se verán un por de ejemplos de definición de procedimientos. Supongamos que queremos dibujar un cuadrado en el centro de la pantalla de lado 100 y otro de lado variable. Para ello se deben definir dos procedimientos: uno no necesitará parámetros y el otro sí.

El procedimiento necesario para dibujar un cuadrado de lado 100 sería el siguiente:

```
PARA cuadrado1
  centro
  borrapantalla
  avanza 100
  giraizquierda 90
  avanza 100
  giraizquierda 90
  avanza 100
  giraizquierda 90
  avanza 100
  giraizquierda 90
FIN
```

El procedimiento necesario para dibujar un cuadrado de lado variable sería el siguiente:

```
PARA cuadrado2 :tamaño
  centro
  borrapantalla
  repite 4 [avanza :tamaño giraizquierda 90]
FIN
```

donde el parámetro *:tamaño* indica la longitud del lado del cuadrado que se quiere dibujar.

Una vez definidos los procedimientos, estos se puede llamar en cualquier momento en un programa o desde la ventana de ejecución de primitivas escribiendo su nombre seguido de los parámetros necesarios (si es que el procedimiento los tiene). En este caso sería, por ejemplo:

```
 cuadrado1
cuadrado2 100
cuadrado2 150
```

1.6.- Comentarios

Los comentarios en MSWLogo se crean poniendo delante punto y coma (;). Por ejemplo:

```
;ESTO ES UN COMENTARIO EN MSWLOGO
```

1.7.- Guardar el trabajo en MSWLogo

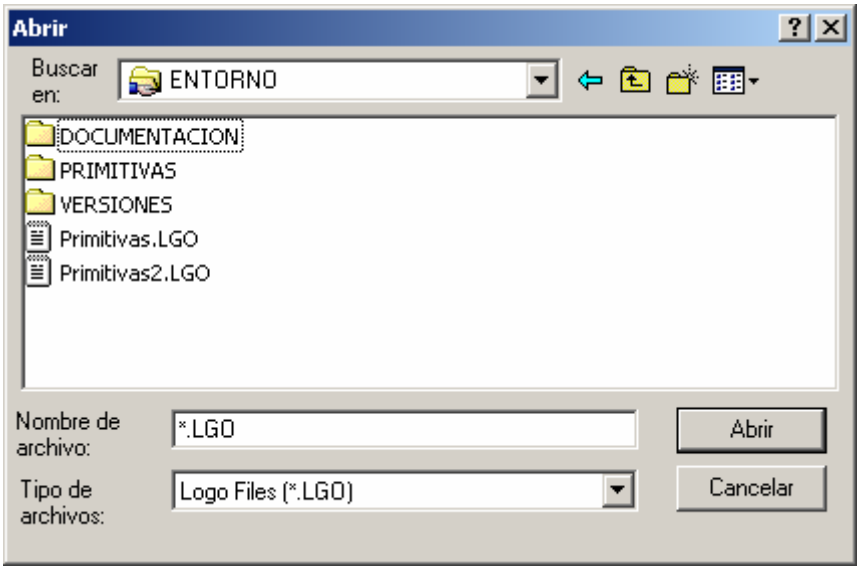
Las opciones del menú *Archivo→ Guardar* o *Archivo→ Guardar como* y la primitiva *Guarda "nombfichero* permiten guardar en un archivo de extensión *.LGO* las definiciones de todos los procedimientos, variables y listas de propiedades sin tapar.

Destacar que no se guarda cada procedimiento en un archivo, sino que todo el trabajo definido durante una sesión de MSWLogo se guarda junto en el mismo archivo.

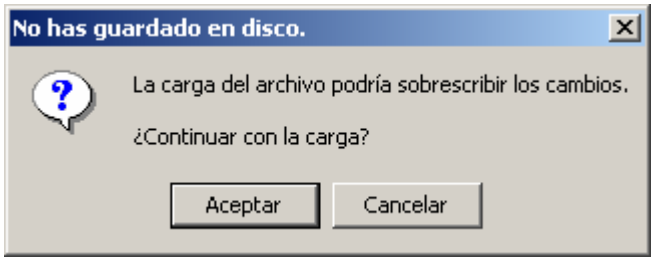
1.8.- Cargar un fichero

La opción del menú *Archivo → Cargar* y la primitiva *Carga "nombfichero* permiten la carga de las instrucciones del fichero dado. Es importante destacar que cuando se carga un fichero se leen las instrucciones que contiene y se ejecutan.

En la ventana de selección del archivo a cargar solo aparecen los archivos con extensión .LGO. Para que aparezcan todos los archivos se debe seleccionar en *Tipo de archivos:* todos los archivos (*.*).



Si no se han guardado el entorno de trabajo de la sesión actual en disco o si en el archivo existen procedimientos ya definidos en la sesión actual, se sobrescribirá. Pero antes de eso MSWLogo emite una ventana de aviso:



1.9.- Trabajar con imágenes

El menú *Bitmap* de MSWLogo ofrece la posibilidad de trabajar con imágenes o dibujos: cargarlos, guardarlos e imprimirlos.

MSWLogo permite la posibilidad de cargar imágenes con formato *.bmp*. Para poner cargar una imagen de este tipo en la pantalla principal se debe utilizar la función *Bitmap → Cargar*.

También permite guardar imágenes en formato *.bmp*. Para ello se usa el menú *Bitmap → Guardar* o *Bitmap → Guardar como*.

1.10.- Usar ventanas y botones en MSWLogo

Con MSWLogo se pueden crear aplicaciones y procedimientos con ventanas y botones. Para ello se utilizan primitivas como *creaventana*, *creaboton*,…

A continuación se muestra un ejemplo de una aplicación que consta de varios procedimientos que crean una ventana que consta de varios botones y que permite escribir y borrar un texto:

```
PARA  VentanaEjemplo
;Procedimiento que crea la ventana del ejemplo
  BT     ;borra pantalla de texto
  BP     ;borra pantalla de gráficos
  ocultatortuga
  creaventana "principal "V_EJEMP [Ejemplo de aplicación con ventanas] 120 150 287 100 []
  creagroupbox "V_EJEMP "E_Ventana 5 0 274 85
  creagroupbox "V_EJEMP "E_Escribir 20 2 120 50
  creaboton "V_EJEMP "Escribir [ESCRIBIR] 25 10 110 35 [actualizaestatico "texto [Ejemplo de
ventanas]]
  creagroupbox "V_EJEMP "E_Borrar 145 2 120 50
  creaboton "V_EJEMP "Borrar [BORRAR] 150 10 110 35 [actualizaestatico "texto []]
  creagroupbox "V_EJEMP "E_texto 20 50 120 30
  creaestatico "V_EJEMP "texto [Pulse un botón] 25 60 80 15
  creaboton "V_EJEMP "Salir [Salir] 150 60 110 20 [BT BP muestratortuga borraventana
"V_EJEMP]
FIN
```

```
PARA MostrarEjemplo
; Ventana inicial del ejemplo
; Permite ver el ejemplo o abandonarlo
  sisino sinobox [Visualizar ejemplo] [¿Quiere ver el ejemplo?.] [VentanaEjemplo] [BT BP]
FIN
PARA BorrarVentanaEjemplo
  borraventana "V_EJEMP
FIN
```

2 BIBLIOGRAFÍA

- Guía práctica de MSWLogo en inglés:
  http://www.southwest.com.au/~jfuller/logotut/logo1.htm

- Manuales de Logo en español:
  http://www.southwest.com.au/~jfuller/mswlogo/spanish.zip
  http://www.angeltowns.com/members/mondragon/paradiso/ - sect2