MSW Logo



A Simplified Reference

## (Version 2)

by J. P. Fuller



© 1998 J. P. Fuller

Contents

Introduction 3

A Brief Background to MSW Logo 3

Entering MSW Logo Commands 4

Summary of Selected MSW Logo Commands 5

Using Variables in MSW Logo 6

User Input 7

READLIST and READCHAR 8

Writing Procedures in MSW Logo 9

MSW Logo Exercises 11

Saving Your Work in MSW Logo 14

Writing Procedures in the Editor Window 15

APPENDIX

Further Examples 16

More on User Input and Numbers 17

Internet Resources 18

Logo Shapes and Patterns 19

Using MSW Logo in the Primary School 20

NOTES 21

Special thanks to Paul Dench for taking the time to read though preliminary drafts and making a number of valuable suggestions.

MSW Logo - A Simplified Reference (Version 2)

The Logo language was developed in the late 1970s and early1980s by a team led by Seymour Papert. Their aim was to produce a 'graphical' programming environment which was both powerful and simple to use.

The Logo 'metaphor' is a robotic turtle moving around under the control of a computer, drawing as it goes.



In most applications, the 'turtle' is reduced to an object in the centre of the screen, sometimes in the shape of a turtle, more often, simply a triangle.

The user enters English-like commands such as FORWARD, REVERSE, PENDOWN, PENUP, RIGHT, LEFT and so on. If the turtle's 'pen' is "down", it draws on the screen as it moves!

MSW Logo - A Brief Background.

The 'core' of MSW Logo (called: "Berkeley Logo") was developed at the University of California, Berkeley by Brian Harvey and his team (bh@anarres.cs.berkeley.edu). The original code was modified for UNIX, DOS and Macintosh systems. George Mills ([www.softronix.com](http://www.softronix.com)) went on to develop the Windows version and named it: "MSW Logo”

All versions of the software are FREE for non-profit use. It may be duplicated and distributed by educational institutions without restriction. You may pass on copies to anyone who wants one - including your students! MSW Logo will work on Windows 3.x and Windows 95/98. Minimum hardware is a 386SX.

The current version (October 1998) is V 6.3

*NOTE: MSW Logo was modified in 1994 to allow intput/output control via the Parallel port using functions called "INPORTB" and "OUTPORTB". Earlier versions (ie pre V 3.7) have NO output control facilities other than character output to a printer. See Version 1 of this manual for details on writing to and reading from the printer port.*

For up to date MSW Logo Internet resources see:

http://www.southwest.com.au/~jfuller/MSW Logo.html

Entering MSW Logo Commands

MSW Logo is an "interpreted" language. This means that commands issued by the user are interpreted by the computer and acted upon immediately, one statement at a time. "Compiled" programs, on the other hand, are first converted into machine language as an entire unit before any part of the program can be 'run'.

Commands are entered in the Input Box in the Commander window at the bottom of the screen.



Commands are executed, after being entered in the Input Box, by either pressing the Return key, or clicking on the Execute button. Each command is recorded in the Output/Command-Recall List Box above the Input Box.

 

 A Typical Logo Program Screen Dimensions

To exit MSW Logo enter **BYE** in the Input Box, or select Exit from the File menu.

Summary of Selected MSW Logo Commands

|  |  |  |
| --- | --- | --- |
| Command | Action | ShortVersion |
| FORWARD 100 | Turtle moves forward the specified number of units.  | FD 100 |
| BACK 50 | Turtle moves back the specified number of units. | BK 50 |
| RIGHT 90 | Turtle turns clockwise the number of degrees specified. | RT 90 |
| LEFT 45 | Turtle turns counterclockwise the angle specified. | LT 45 |
| PENUP | Turtle's pen is up. (Doesn't draw as it moves.) | PU |
| PENDOWN | Turtle's pen is in the down position. | PD |
| PENERASE | Turtle erases as it moves. | PE |
| HIDETURTLE | Removes the Turtle 'triangle' from the screen. | HT |
| SHOWTURTLE | Makes the Turtle visible again. | ST |
| CLEARSCREEN | Erases the screen and returns the Turtle to its "home" position in the centre of the drawing screen. | CS |
| HOME | Returns the Turtle to the "home" position without erasing the screen. (Issue a PU command before HOME to prevent it drawing its return path as it goes.) | HOME |
| LABEL [HELLO WORLD] | Displays text at the Turtle location. | - |
| SETPENCOLOR [..........] | Determines pen color according to:SETPENCOLOR [255 000 000] = RedSETPENCOLOR [000 255 000] = GreenSETPENCOLOR [000 000 255] = Blue | ---- |
| SOUND [1000 200] | SOUND [<frequency> <duration>] where “frequency” is in cps and “duration” is in 1000ths of a second. |  |
| SETPENSIZE [10 10] | Sets width and height of the drawing pen. MSW Logo uses only the second value. Set them both the same. | --- |
| BYE | Exits MSW Logo | - |

# Using Variables in MSW Logo

A variable can be thought of as a ‘container’ for different values. You can change the value of a variable at any time.

MSW Logo uses a statement in the form: **make “MyVariable 20** to assign a value to a variable.

The value of the variable is accessed in the form: **label :MyVariable** (NOTE: the colon in front of the variable name.)

Eg

**to multiply**

 **make “Number1 16**

 **make “Number2 3**

 **make “Answer :Number1 \* :Number2**

 **label :Answer**

**end**

IMPORTANT – **You need to understand the difference between using a variable and ordinary text. When accessing a variable’s value you must put a colon in front of the variable name.**

Eg.

**LABEL hello** will produce – **hello**

Whereas

###### Make “hello 79

**LABEL :hello** will produce – **79**

**LABEL hello \* 2** will produce **an error**

Whereas

**LABEL :hello \* 2** will produce **158**

# Passing Values into Procedures

Eg

**to box :size**

 Repeat 4 [fd :size rt 90]

**end**

The size of the box can be determined by the value typed in when running the procedure.

Eg **box 20**, **box 40** , etc.

NOTE: If you type in **box**, without the value, MSW Logo will report **an error**.

User Input

MSW Logo is a Windows programming language. It makes extensive use of the built-in Windows “Graphical Users Interface (“GUI”). If you want to write a programme which accepts user input and then acts on it you must use the standard Windows GUI for user input. There is no equivalent to Pascal’s “readln(variable)”

Eg. 1

**to multiply**

 **; A sample routine to demonstrate “questionbox”**

 **; and “first” to extract a number from user input.**

 **make “Number1 first questionbox [User Input] [Enter the first number]**

 **make “Number2 first questionbox [User Input] [Enter the second number]**

 **make “Answer :Number1 \* :Number2**

 **home**

 **cs**

 **ht**

 **label :Answer**

**end**

Produces …..



NOTE: The variable names can be just about anything. The word “first” before “questionbox” is a MSW Logo function that extracts the first value entered by the user as a number.

Eg.2 A simpler method …

**to multiply**

 **; A sample routine to demonstrate “readlist”**

 **; and “first” to extract a number from user input.**

 **; Based on suggestions made by Paul Dench**

 **make “Number1 first readlist**

 **make “Number2 first readlist**

 **; See the next page for an explanation of READLIST.**

 **make “Answer :Number1 \* :Number2**

 **home**

 **cs**

 **ht**

 **label :Answer**

**end**

READLIST and READCHAR (RL and RC)

READLIST

The function “readlist” displays the dialogue box (as below) and allows the user to enter text.



The contents of the input box may be entered as a variable and manipulated by functions such as LABEL.

Eg

**to writename**

 **make “TheirName RL**

 **label :TheirName**

**end**

READCHAR

Readchar performs a similar input function to readlist but only the first character (ie the “char”) is read into the variable. If the user enters a word (or sentence) only the first letter will be entered into the variable.

Converting Input to a Number

The function “FIRST” will extract the first item in a list. In the case of numerals it will convert the numeral to a number. NOTE: If you don’t use “first”, the numeral will be a character and you will not be able to carry out mathematical operations on it.

Eg.

**to GetFirstNumber**

 **; A sample routine to demonstrate “readchar”**

 **; and “first” to extract a number from user input.**

 **; based on suggestions made by Paul Dench**

 **make “Number1 readchar**

 **make “Answer :Number1 \* 10**

 **home**

 **cs**

 **ht**

 **label :Answer**

**end**

Writing Procedures in MSW Logo

Good programming technique dictates that large tasks should be broken down into small 'modules' where each module performs one small, simple task. This approach is called "Top-down Design". In MSW Logo this is achieved using structures called "Procedures".

If, for example, you wanted to draw a frame around each screen you use you could include the following lines each time you constructed the frame:

 LT 90

 PU

 FD 200

 RT 90

 PD

 FD 150

 RT 90

 FD 400

 RT 90

 FD 300

 RT 90

 FD 400

 RT 90

 FD 150

**OR ....**

Using the "Procedure" approach you could define "FRAME" at the beginning of the program and simply "call" the procedure each time it was needed.

Defining Procedures:

Start with the words: **TO FRAME**

End with the word: **END**

eg TO FRAME

 LT 90

PU

FD 200

RT 90

PD

FD150

RT 90

FD 400

 RT 90

FD 300

RT 90

FD 400

RT 90

FD 150

 END

Using Procedures:

Having defined what "Frame" means, you may call the procedure at any time during the program by simply writing the command: FRAME.

Sample Procedures:

TO SQUARE :SIZE

 FD :SIZE

 RT 90

 FD :SIZE

 RT 90

 FD :SIZE

 RT 90

 FD :SIZE

END

Called using - SQUARE 50

TO TRIANGLE :LENGTH

 REPEAT 3[FD :LENGTH RT 60]

END

Called using - TRIANGLE 100

TO TURN\_ON\_LINE :LINE

 OUTPORTB 888 :LINE

 ;NOTE: The value of "LINE" may be 1,2,4,8,16,32,64,128 (or combinations up to 255)

END

Called using - Turn\_on\_Line 4

to loadbitmap

 ; This procedure loads a bitmap file and places it at the turtle position

 home

 bitload "c:\\apw\\car.bmp

 ; NOTE: the double 'backslash' IS necessary

end

MSW Logo Exercises

Exercise 1

Drawing Shapes and Importing Pictures

1. Open MSW Logo.

2. Click in the Input Box. (See page 4 if your are not sure where the Input Box is.)

3. Type in the following:

 home <Press Enter>

 FD 100 <Press Enter>

 RT 90 <Press Enter>

 FD 100 <Press Enter>

 RT 90 <Press Enter>

 FD 100 <Press Enter>

 RT 90 <Press Enter>

 FD 100 <Press Enter>

NOTE: The "<" and ">" symbols contain instructions. Whenever they are used in these sheets, ignore them! Don't type them in!

4. Type in the following:

 home <Press Enter>

 repeat 4 [fd 100 rt 90] <Press Enter>

5. Type in the following:

 to square <Press Enter>

 cs <Press Enter>

 repeat 4 [fd 100 rt 90] <Press Enter>

 end <Press Enter>

6. Type:

 home <Press Enter>

 square <Press Enter>

The method used in steps 5 and 6 is called writing a "procedure". All Logo procedures begin with "to" and end with "end".

 \* Make sure you understand the differences between the techniques used above.

The Tasks –

**Task A.**

1. Construct a procedure to draw an equilateral (equal-sided) triangle of side-length 100 units.

2. Construct a shape consisting of three equilateral triangles and a square.

3. Save your work onto your data disk as a file called <\*\*EXER1>

 NOTE: Put in your initials instead of the asterisks!! eg <FBEXER1>

4. Use the "Label" function to add a title at the top of your screen.

 eg **label [This is my Drawing]**

**Task B.**

5. Use the process described below to place a picture on your screen ...

6. Go to the **Bitmap** menu and select **Load**

7. Locate a bitmap image on your disk and load it into MSW Logo

8. Close MSW Logo

9. Open a drawing package and create a drawing. Save the image to disk in **.BMP** format.

10. Create a New MSW Logo file and load your Bitmap image onto the screen

EXTRA

Create a scene (such as a golf course) and use the direction commands in MSW Logo to ‘play’ your way around the course.

Have a competition with someone else in your class to see who can complete the course in the lowest number of ‘stokes’.

MSW Logo Exercises

Exercise 2

Controlling Your Program with Buttons

MSW Logo allows you to create 'buttons' to call procedures. The process is quite complex, but you can use the example provided in the Help section and simply modify it to suit your own requirements.

• Open MSW Logo

• Go to the Help menu

• Select Index

• Type in "buttoncreate"

• Click on Display button

• Select the section of text (all five lines) from the top example starting with: **windowcreate** and ending with **windowdelete**.

• Select Edit/Copy and copy the text (or Ctrl/C)

• Put away the Help window

• Click in the Input Box

• Type in <edit "buttons>

• Put your cursor at the end of "to buttons" in the edit window.

• Press Enter

• Go to the Edit menu

• Click on Paste (or: Ctrl/V)

• Insert a semicolon (;) at the beginning of the line starting with <click left or right...>

• Insert a semicolon (;) at the beginning of the line <windowdelete "mywindow>

• Close the Help/Edit window

• Save the Edit window Contents

• Click in the Input Box

• Type in <buttons>

• Press Enter

Experiment with the program until you understand what it is doing.

Insert your own procedure unto the square brackets at the end of each line.

*NOTE: To close the window, type <windowdelete "mywindow> at the Input Box.*

Saving Your Work in MSW Logo:

The **File/Save** and **File/Save As…** menus can be used to save logo files in the usual way, BUT this will only save work that has been defined as procedures. MSW Logo will save ALL of the procedures you defined during the ‘session’ as a file with the extension **.LGO**. The programme does NOT save individual procedures as separate files. All the procedures you created will be saved. You will lose all of the commands typed directly into the Input Box that were not defined as part of a procedure. (Another great reason for using procedures!)

Loading Previously Saved Sessions:

The **File/Load** menu allows you to reload procedures you have previously defined. Only thosr files with a “.LGO” extension will listed in the dialogue box. If you want to load a file created with a text editor and saved without the .LGO extension select **All Files (\*.\*)** from the **Files of Type:** box.

WARNING: Loading a file will overwrite any procedures from the current session of the same name. ie If you have just defined a procedure called MyPicture and the file you load also has a procedure called MyPicture, the current one will be overwritten by the one contained in the file and you will lose all of the commands defined within the current version.

Copy and Paste from emails:

You can Copy and Paste sections of code from emails by selecting the section of code in the email and then pressing **Crtl/C** to copy. Go to the Editor window in MSW Logo and select **Ctrl/V** to paste. (See the next page for information about the Editor window.) Make sure that you remove any lines associated with the email, other than the MSW Logo code itself. (Lines with a semicolon (;) in front are ignored by MSW Logo.)

Writing Procedures in the Editor Window:

If you define a procedure via the **Input Box** by typing in: to box, the input dialogue window appears where you enter one line at a time followed by <Enter>. You are prompted to type in “End” to end the procedure.

A better approach for editing is to use the built-in Editor window…

1. Go to the **File/Edit…** menu



2. Type in a suitable procedure name and click the **OK** button. The Editor window appears…

You can type in commands, edit them, cut and paste, just like a conventional word processor. When you have finished defining the procedure go to **File/Exit** and click the **Yes** button to save your work..

Remember that nothing has been saved to disk until you do a **File/Save**.

APPENDIX

Further Examples:

to ButtonExample

 ; This procedure shows how to use buttons for control of your programs

 ; NOTE: Button and Window co-ordinates are: Horiz position, Vert position, width, height

 windowcreate "main "mywindow "mytitle 0 0 100 100 []

 buttoncreate "mywindow "myleft "left 25 25 25 25 [FD 50 LT 45]

 buttoncreate "mywindow "myright "right 50 25 25 25 [FD 50 RT 45]

 ; The window is 'closed' with : windowdelete "mywindow

end

Procedures within Procedures:

Drawing a house …

to base

 home

 lt 90

 fd 20

 lt 90

 fd 80

 lt 90

 fd 80

 lt 90

 fd 80

end

to roof

 home

 HT

 lt 90

 fd 50

 rt 135

 fd 100

 rt 90

 fd 100

 rt 135

 fd 100

end

to house

 cs

 base

 roof

end

More on User Input and Numbers …

to GetNumbers

 ; A sample routine to demonstrate "readword".

 ; Based on suggestions from Paul Dench

 make "Number1 readword

 make "Number2 readword

 make "Answer :Number1 \* :Number2

 home

 cs

 ht

 label :Answer

end

General …

to testRW

 ; Provided by Paul Dench

 make "first rw

 make "second rw

 show (word :first :second)

end

Internet Resources

The Logo Foundation - http://el.www.media.mit.edu/groups/logo-foundation/

Download MSW Logo from - http://www.softronix.com

ECAWA Logo SIG - http://www.cowan.edu.au/pa/ecawa/sig/logo/logo.htm

Jim Fuller’s MSW Logo Resource Page - <http://www.southwest.com.au/~jfuller/mswlogo/mswlogo.html>

Lego Mindstorms Home Page - http://www.legomindstorms.com/

Logo email ‘Lists’

ECAWA Logo SIG: logo@cleo.murdoch.edu.au

To subscribe, send an email to: majordomo@cleo.murdoch.edu.au

with the words **subscribe logo** in the body of the email (leave the “Subject” line blank).

LOGO-L: logo-l@gsn.org

To subscribe, send and email to: majordomo@gsn.org

with the words **subscribe logo-l** in the body of the email (leave the “Subject” line blank).

Logo-L Archive – http://archives.gsn.org/logo-l/ - A collection of emails from the Logo-L list from January 1995

Logo Shapes and Patterns

  

to triangle to square to circle1

 repeat 3[fd 100 lt 120] repeat 4[fd 100 rt 90] repeat 36[fd 10 lt 360/36]

end end end

 

to decagon to shape1

 repeat 10[fd 50 lt 360/10] repeat 25[circle1 triangle lt 90]

end end

  

 to shape2 to shape3

 repeat 50[square rt 10] repeat 60[triangle rt 10]

 end end

#### Using MSW Logo in the Primary School

Logo in general has a number of benefits over other programming languages:

Logo fosters group collaborative work.

Logo allows children to visualise mathematical and geometrical concepts.

Logo teaches children that the computer is a device to be controlled, rather than one that controls.

Logo introduces children to programming structures and concepts.

Logo promotes concepts associated with the Technology and Enterprise Learning Area.

Logo is immediate. There is no waiting while the programme is compiled.

Logo uses simple ‘English’ commands.

Logo is graphical in nature.

##### Some Ideas on using MSW Logo with Your Class …

Set up desks in a room (or outside) as a ‘maze’. Blindfold a student and have a group attempt to direct the blindfolded student through the maze using typical logo commands such as Forward ten, right turn 45 and so on. Make it a competition between groups in the class with the winning group the one that gets their subject though the maze quickest, with less collisions.

Have each group in the class design a simple pattern using logo commands. Fill a bucket with sand (or flour) and make a hole in the bottom of the bucket. Have a different group follow the instructions and walk around with the bucket on a basket ball/tennis court. ‘penup’ and ‘pendown’ would correspond to the subject blocking the hole, or allowing sand/flour to flow out.

Take the groups out to a sandy area where it is possible to draw with a stick. Follow the same sort of approach as above, with one group providing logo instructions for another group to follow.

Do the same sort of activity with chalk on a basketball/tennis court, or the parking area. (It’s not a good idea to use a roadway.)

Use a plan of the school as a basis for writing a series of logo commands to navigate from one point to another. Have students use a tape measure and protractor to use ‘real’ values where for example “FD 25” means go forward 25 metres. (Suggest that they use only simple angles such as 90 and 45.) As above, one group writes the instructions for another group to follow. *NOTE: The second group should not know their destination in advance.*

##### On a smaller scale …

One student writes logo commands for another to follow in using a pair of scissors and a sheet to paper to cut out a pattern.

As above, but the second student uses a pen and graph paper to follow the instructions (which might include a ‘message’ made up of letters constructed with logo commands).

**And after all this ….** Let them use MSW Logo on the computer. Encourage group work, pre-planning, evaluation, discussion, etc – all those great ‘Technology and Enterprise Learning Area’ concepts.

NOTES

|  |
| --- |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |