MathScript

Simon Ihmig

COLLABORATORS			
	TITLE : MathScript		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY	Simon Ihmig	July 19, 2024	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

MathScript

1.1 MathScript Help

MathScript © Copyright Simon Ihmig 1994 MathScript is SHAREWARE !

Guide for MathScript

1. Introduction

1.1 Description 1.2 System requirements

2. Using the program

2.1 Starting the program 2.2 Input of formulas 2.3 Operation of the program

3. Appendix

3.1 ARexx port 3.2 Control codes

Disclaimer Registration The future

My Adress:

Simon Ihmig Beim Rauhen Hause 30 22111 Hamburg Germany e-mail: Internet: Ihmig@tu-harburg.d400.de Z-Netz: GrafZahl@Online

1.2 Introduction

MathScript is a formula editor, which is used to integrate mathematical formulas into word processors. In principle this is nothing new but untill now it was a deficit for the Amiga. Although it is possible to create such formulas with TeX, it is not very convenient.

That is why MathScript was created.

With this program it is very easy to create complex formulas. To insert them into word processors or DTP programs they are saved as graphic files. To garantee the highest possible quality they are saved as vector graphics in the EPS format. They can be loaded into all wordprocessor and DTP programs which support this format (FinalWriter,PageStream,...). They can be printed out by PostScript printers and through software PostScript interpreters on all printers.

These are the features version 1.0:

- extensive amount of mathematical and physical symbols

- Control codes for formatting (super-, subscript, fractions, roots)

- easy input via Workbench GUI

- Export in the EPS format (Encapsuled PostScript)

- ARexx port

- support for the locale.library (beginning with OS 2.1). Available languages: english, german, french. You may also create your own languages (see Catalogs/readme).

1.3 System requirements

MathScript needs at least version 2.0 of the Amiga operating system. Beginning with OS 2.1 it supports localization.

To use the ARexx port of MathScript the ARexx interpreter RexxMast has to be active.

1.4 Starting the program

Starting MathScript ...

from the Workbench:

To start MathScript from the Workbench enter the directory where the program is installed and double click its icon. Alternativly you can click on an icon of a previously saved formula to start MathScript loading this formula automaticaly.

from the Shell:

To launch MathScript from the Shell type in "MathScript:MathScript". The following arguments are accepted: FILE,PUBSCREEN/K

FILE : name of a formula which shall be loaded immediately

PUBSCREEN : name of a public screen on which MathScript shall open its GUI

1.5 Input of formulas

For writing the formulas into MathScript the input gadget is used. The input is made with the keyboard or with the selection gadgets. Formatting functions are specified by special control codes, thus without WYSIWYG representation.

To enter standard ASCII letters the keyboard is used while the input gadget is active (to activate it just klick into the main window).

To insert control codes use the keyboard or the selection gadgets.

Special mathematical symbols have to be selected with the selection gadgets.

Principles of the use of control codes:

Control codes cause special formatting of the objects belonging to them. You can for example superscript a number to use it as an exponent (like x^2) or create an fraction. They cannot be seen in the final graphic file of the formula.

Lets's demonstrate an example:

To show a fraction, there is the control code '/'. Its Syntax is:

fraction: syntax: <Obj1>/<Obj2> function: creates a fraction. input: <Obj1> - numerator <Obj2> - denominator

The input of 1/2 results in the following display in the final graphic file:

1 - 2

How single characters are grouped into an object is explained in objects.

A complete overview of the control codes together with their syntax can be found under control codes.

Is your formula complete you can use the menus to save it as a text file to edit it later, or you can save (export) it as a graphic file, where it gets its final look.

1.6 objects

The control codes need objects for their formatting functions. Such a object is in the simplest case a single character. So 1/2x results in the following form: 1 - x 2 But if you wish the denominator to be 2x, thus using more than one character, you have to group it to a single object. This is done by surrounding the concerning characters by brackets. So 1/[2x] causes the following term: 1 -- 2x Of course it is possible to create a grouped object inside of another grouped object. To create a double fraction simply write [1/x]/3 which results in the following: 1 - x --- 3

1.7 Operation of the program

MathScript always opens the main window into which the Input of the formulas is made.

For that reason the Selection gadgets, a group of 13 gadgets, and below them the input gadget is used.

The other features of MathScript are available through the menus.

For experienced user there is also an ARexx port.

1.8 Selection gadgets

The selection gadgets consist of 13 gadgets. Each of them opens a selection window. In these selection windows there are several more gadgets, which contain different symbols. If you click on one of these gadget it will be inserted into the current position of the input gadget. Then the window will be closed again.

There are the following selection gadgets:

operators arrows logic brackets sum,prod,int greek chars GREEK CHARS

equations theory of sets symbols super-/subscript fraction,root derivatives

They are represented by some of their subgadgets.

Beside simple characters control codes can be inserted, too. They are shown in a symbolic representation and not in the way they are written. The objects belonging to them are symbolized in the following manner: stands for a grouped object, # for a single character.

Example: The symbol # means superscript. It inserts the following text into the input gadget: ^[] . Lets's suppose there is a single x standing behind the cursor. After clicking on the symbol the contents of the input gadget becomes like this: x^[]. The cursor is placed automatically behind the first grouped object, so behind the '['. By pressing the key '2' the text is "x^[2]", which leads to this "picture": x^{2}. But if the first object is a grouped one, you have to select its gadget first and type in its text afterwards in order to be placed between the brackets (see objects).

1.9 operators

Contains a collection of mathematical operators.

1.10 arrows

There are 5 single and 5 double arrows.

1.11 logic

Holds logical symbols.

1.12 brackets

Includes control codes for making brackets of variable size. There are for each bracket type ({,(,[,],|,<) left and right sided versions and brackets on both sides.

1.13 sum,product und integral

Three characters symbolizing a sum, product and an integral.

1.14 small greek characters

Contains all small greek characters.

1.15 big greek characters

Includes all greek characters.

1.16 equations

Holds the most important equation and inequality symbols.

1.17 symbols of the theory of sets

A collection of symbols of the theory of sets.

1.18 symbols

Includes symbols not belonging to other groups.

1.19 Super-/subscript

Contains all control codes for positioning objects.

1.20 Fraction and root

Holds control codes for making fractions and roots.

1.21 derivatives

Contains control codes for creating derivatives and vectors.

1.22 Input gadget

All formulas are typed into the input gadget. It is a standard string gadget and therefore accepts all key short cuts, which can be applied to other string gadgets. Moreover you can jump to the next grouped object with the TAB key, with SHIFT-TAB to the previous.

1.23 menus

Overview of all menu items:

Project ARexx Settings New Execute ARexx script ... Change settings... Load ... Save settings Save Save as ... Export EPS About ... Quit

1.24 new

Clears the input gadget.

Warning: This action cannot be made undone.

1.25 load

Opens a file requester in which a file can be chosen which should be loaded. Only files which have the suffix .fml are shown there. It is appended automatically by the Save function. If you wish to load an ASCII file without this suffix, just clear the gadget "Pattern".

1.26 save

Does the same as Save as ..., except that if a file has been loaded or saved before no file requester will pop up and the current file will be used for saving.

The name of the current file is shown on the title bar.

1.27 Save as

This menu item saves the current contents of the input gadget as a Formula file, in order to reuse or modify the formula later on. A file requester will be opened, in which a file can be chosen, which should be used for saving.

All formulas saved with MathScript have the suffix .fml, in order to distinguish them from regular text files.

1.28 Export

This menu item contains the main function of the program, that is the transformation of the text code into a graphic file.

In this version of Mathscript there is only one format available: EPS format

1.29 eps

This submenu item saves the current formula as a vector graphic file in the EPS format. It opens a file requester, where a file can be chosen, into which the formula should be saved.

This format is the most popular vector graphic format. In opposition to bitmap formats like the IFF-ILBM they have an excellent printing quality.

EPS means Encapsuled PostScript. PostScript is a interpretative page describing language, which is known by several laser printers. If you do not have such a printer, you are dependent on the software, in which the formulas should be used. For example the word processor "FinalWriter" is able to convert EPS files to Bitmaps so they can be printed on regular printers..

All formulas saved in MathScript as EPS file get the suffix .ps .

1.30 About

Opens an window, in which there are informations about the program and me.

1.31 quit

Quits the program.

1.32 Execute ARexx script

Opens a file requester in which an ARexx script can be chosen which should be executed. Detailled informations about the ARexx port of MathScript can be found under ARexx.

1.33 Change settings

This menu item opens a window in which MathScript's settings can be changed. There are the following options:

Screen With the cycle gadget you can chose between three kinds of screens which MathScript shall use for its windows. Workbench : Opens MathScript's windows on the Workbench. default public screen : Uses the default public screen. Usually it is the Workbench. public screen : Uses a public screen which name is entered in the gadget to the right. If a public screen name is stated during a start from the CLI these settings are ignored.

Path With the tiny file requester gadget or the string gadget to the left a path can be chosen which shall be used for loading and saving formulas by default.

Include PS-Font If this option is activated the postscript description of the math font is embedded into the EPS files. This is necessary if the postscript interpreter is unable to load fonts from your harddisk. This applies of course to all postscript printers. But if a application with a software interpreter (Post, PostView, FinalWriter...) is used, it will look (with a suitable init.ps file) in the directory PSFonts: after available fonts. So in this case it is better not to include the font, so the size of your EPS files will not grow too much.

PS-Font size This gadget allows you to determine the size of the PostScript font.

Create Icons If this option is activated MathScript will create suitable icons for all saved files.

To accept these settings click on "Ok" else on "Cancel"

1.34 Save settings

This will save your settings to be used next time.

1.35 arexx

With its ARexx port MathScript can be controlled from "outside" what means without intervention of the user. In order to control MathScript via the ARexx port the global interpreter RexxMast must be active (see also your system documentation). The port is named : "MathScript".

An ARexx script can be started by MathScript with the menu item Execute ARexx script.

Arguments in pointed brackets < ... > have to be specified, arguments in angular brackets [...] are optional.

Here is the list of all supported commands:

Clear GetContents

SetContents GetPosition

SetPosition Insert

GotonextObject GotoprevObject

Load Save

Export Info

Version ScreenToFront

LoadRequester Quit

GetPrefs SetPrefs

1.36 clear

Clear: syntax: Clear function: Clears the input gadget corresponding the menu item New. input: - result: -

1.37 getcontents

GetContents: syntax: GetContents function: Provides the current contents of the input gadget. input: - result: result - contents

1.38 setcontents

SetContents: syntax: SetContents <contents> function: Sets the new contents of the input gadget. input: <contents> - New contents of the input gadget result:

1.39 getposition

GetPosition: syntax: GetPosition function: Gets the position of the cursor in the input gadget input: result: result - position of the cursor

1.40 setposition

SetPosition: syntax: SetPosition <position> function: Sets the cursor on a new position. input: <Position> - new position of the cursor result:

1.41 insert

Insert: syntax: Insert <string> function: Inserts a string behind the current cursor position into the input gadget. input: <String> - string that should be inserted result: -

1.42 gotonextobject

GotonextObject: syntax: GotonextObject function: Places the cursor to the next grouped object like pressing the TAB key. input: - result: -

1.43 gotoprevobject

GotoprevObject: syntax: GotoprevObject function: Places the cursor to the previous grouped object like the SHIFT-TAB keys. input: - result: -

1.44 Load

Load: syntax: Load [file name] [FORCE] function: Loads a text file as a formula. If [file name] is not supplied a file requester will pop up. input: [file name] - name of the file which should be loaded [FORCE] - surpresses any warnings result: -

1.45 Save

Save: syntax: Save [file name] [FORCE] function: Saves the current formula as a text file. If [file name] is not supplied a file requester will pop up. input: [file name] - name of the file which should be saved [FORCE] - surpresses any warnings result: -

1.46 Export

Export: syntax: Export <format> [file name] [FORCE] function: Saves the current formula as a graphic file. If [file name] is not supplied a file requester will pop up. input: <format> - Format of the file. In this version of MathScript there is only "EPS" available [file name] - name of the file which should be saved [FORCE] - surpresses any warnings result: -

1.47 Info

Info: syntax: Info function: Opens the same info window as the menu item Info. input: - result: -

1.48 version

Version: syntax: Version function: Gets a string with the current version number of MathScript input: - result: result - version string in the form "MathScript xx.xx"

1.49 screentofront

ScreenToFront: syntax: ScreenToFront function: Moves the screen where MathScript has opened its windows to the front. input: - result: -

1.50 requester

Requester: syntax: Requester [title] [pattern] function: Opens a file requester for loading files. input: [title] - Titlebar of the requester [pattern] - Amiga-Dos pattern, which can be used for hiding files result: -

1.51 Quit

Quit: syntax: Quit [FORCE] function: Quits the program. input: [FORCE] - surpresses any warnings result: -

1.52 getprefs

GetPrefs: syntax: GetPrefs [PUBSCREEN] [PATH] [NOICONS] [INCLUDE_PS_FONT] function: Gets the current settings input: [PUBSCREEN] provides the Name of the public screen ('def' for the default public screen, 'Workbench' for the Workbench ;-) [PATH] gets the default path [NOICONS] provides the value YES or NO [INCLUDE_PS_FONT] provides the value YES or NO [PS_FONTSIZE] the PostScript font size result: from input requested value example: GetPrefs PUBSCREEN PATH NOICONS for example puts the following string into result: 'Workbench MathScript:Formulas NO'

1.53 setprefs

SetPrefs: syntax: SetPrefs [PUBSCREEN name] [PATH pfad] [NOICONS YES/NO] [INCLUDE_PS_FONT YES/NO] [PROMPT] function: Changes the settings input: [PUBSCREEN name] Places MathScript on a new screen. Possible values for name: 'def' for the default public screen, 'Workbench' for the Workbench or a name of an open public screen. [PATH path] Sets the default path [NOICONS YES/NO] determines whether icons should be created or not [INCLUDE_PS_FONT YES/NO] determines whether the math font should be integrated into EPS files [PROMPT] If this argument is provided, all other arguments are ignored and the settings window is opened. result: -

1.54 codes

Overview of all control codes

Positions:

Up: syntax: <Obj1>^<Obj2> function: Moves <Obj2> dependent on the size of <Obj1> up for example in order to show it as an exponent of a power. input: <Obj1> - base <Obj2> - exponent

Down: syntax:<Obj1>_<Obj2> function: Moves <Obj2> dependent on the size of <Obj1> down for example in order to show an subscript. input: <Obj1> - base <Obj2> - subscript

Up and down: syntax: <Obj1>^_<Obj2><Obj3> function: Moves <Obj2> dependent on the size of <Obj1> up and <Obj3> down. input: <Obj1> - base <Obj2> - exponent <Obj3> - subscript

Above: syntax: <Obj>\up<Obj2> function: Moves <Obj2> above <Obj1>. input: <Obj1> - base object <Obj2> - higher object

Example: -1 F\up[-1] -> F (inverse function)

Under: syntax: <Obj1>\dn<Obj2> function: Moves <Obj2> under <Obj1>. input: <Obj1> - base object <Obj2> - lower object Example: lim [lim]\dn[n->0] -> n->0

Above and under: syntax: <Obj1>\ud<Obj2><Obj3> function: Moves <Obj2> above and <Obj3> under <Obj1>. input: <Obj1> - base object <Obj2> - higher object <Obj3> - lower object

Brackets:

syntax: input output (<Obj1> -> (<Obj1> {<Obj1> -> {<Obj1> \[<Obj1> -> [<Obj1> \]<Obj1> ->]<Obj1> \|<Obj1> -> |<Obj1> <Obj1>) -> <Obj1>) <Obj1>} -> <Obj1>} <Obj1>\r] -> <Obj1>] <Obj1>\r[-> <Obj1>[<Obj1>\r| -> <Obj1>| Moreover there are special pointed brackets which are only available through the selection gadgets: <<Obj1> -> <<Obj1> <Obj1>> -> <Obj1>>

function: Scales the brackets dependent on the size of <Obj1> input: <Obj1> - specifies the size of the brackets

Example: / 1 | ([1/2x]\r| -> | -x | \ 2 |

fraction and root:

Fraction: syntax: <Obj1>/<Obj2> function: Creates a fraction. input: <Obj1> - nominator <Obj2> - denominator

Square root: syntax: \sqrt<Obj1> function: Creates a square root. input: <Obj1>

root: syntax: \rt<Obj1><Obj2> function: Creates a root. input: <Obj1> - root exponent <Obj2>

Because the characters '^','_' ,'\' and '/' are used as control codes you can use them as normal characters by writing a single backslash ('\') in front of them. Thus: \^ -> ^ _ -> _ \\ -> \ \/ -> /

1.55 Disclaimer

MathScript is shareware. It may be distributed and copied as long as the following conditions are fulfilled:

- The sales price must not be higher than the cost of an empty disk plus a nominal copying fee plus costs for shipping.

- All parts of the program and the documentation must be complete. The distribution of single parts is not allowed.

- MathScript or parts of it may not be sold together with commercial software without the written permission of the author.

- All parts of the software package must not be changed in any way.

- The Author is not responsible for misuse or damage caused by the program.

The program may be used for 14 days. After that period the registration is obligatory for further usage.

1.56 Registration

If you register for MathScript you will obtain the latest version of it and a keyfile. With this keyfile you are able and allowed to use future releases of MathScript. Moreover these annoying requesters will disappear.

The shareware fee is 15$ or 20DM.

To become a registered user print out the registration form (or write a similar form) and fill it out.

Send it to:

Simon Ihmig Beim Rauhen Hause 30 22111 Hamburg Germany

1.57 Registration form

I want to register for the shareware program MathScript. I will get a keyfile which allows me to use the program regularly. I guarantee that it is only for my personal use and that I do not distribute it.

Kind of payment:

o The fee of 15$ or 20DM is enclosed

o The fee of 15$ or 20DM is transfered to the following account: No: 1077/782033 BLZ: 200 505 50 (Germany) Hamburger Sparkasse Simon Ihmig

My address:

Name:_____

Christian name:_____

Street:_____

City:_____

Phone:_____

Country:_____

e-mail:_____

City,Date:_____ signature:_____

1.58 Future

For future versions of MathScript the following features are planed: (no guarantee for realization!)

- Export as IFF-ILBM bitmap file

- Embedded TIFF picture in EPS files for previewing them in word processors

- More control codes (matrix,underline...)

- More symbols

- Formulas with more than one line

- Online help ?

In order to be able to fulfill other wishes, to give suggestions or to indicate errors, PLEASE write !!! You can find my address below the list of contents.