LibreOffice®

Version 3.5

Math Guide

Using the Equation Editor

Copyright

Contributors

Jean Hollis Weber
Hazel Russman
Laurent Balland-Poirier

Feedback

Please direct any comments or suggestions about this document to:
documentation@global.libreoffice.org

Acknowledgments

This guide is based on the *OpenOffice.org 3.3 Math Guide*. The contributors to that book are:

Daniel Carrera
Agnes Belzunce
TJ Frazier
Peter Kupfer
Ian Laurenson
Janet M. Swisher
Jean Hollis Weber
Michele Zarri

Additional material has been added from the German *Math Handbuch* for LibreOffice 3.4 (translated to English by Hazel Russman). Contributors to that book are:

Regina Henschel
Christian Kühl
Florian Reisinger
Gisbert Friege (Dmaths)
Jochen Schiffers

Additional material has been added from the French *How-To Math* for LibreOffice (translated to English by Laurent Balland-Poirier). Contributors to that book are:

Bernard Siaud
Frédéric Parrenin

Publication date and software version

Published 7 September 2012. Based on LibreOffice 3.5.5.

Note for Mac users

Some keystrokes and menu items are different on a Mac from those used in Windows and Linux. The table below gives some common substitutions for the instructions in this chapter. For a more detailed list, see the application Help.

Windows or Linux	*Mac equivalent*	*Effect*
Tools > Options menu selection	**LibreOffice > Preferences**	Access setup options
Right-click	*Control+click*	Opens a context menu
Ctrl (Control)	⌘ *(Command)*	Used with other keys
F5	*Shift+⌘+F5*	Opens the Navigator
F11	*⌘+T*	Opens the Styles and Formatting window

Contents

What is Math?

Math is LibreOffice's component for writing mathematical and chemical equations. It is most commonly used as an equation editor for text documents, but it can also be used with other types of documents or stand-alone. When used inside Writer, the equation is treated as an object inside the text document.

Note	The equation editor is for writing equations in symbolic form, as in equation 1. If you want to evaluate a numeric value, see the *Calc Guide*.

$$\frac{df(x)}{dx} = \ln(x) + \tan^{-1}(x^2) \qquad (1)$$

or

$$NH_3 + H_2O \rightleftharpoons NH_4^+ + OH^-$$

Getting started

You can create an equation (formula) as a separate document or insert it into a document in Writer or another component of LibreOffice.

Creating an equation as a separate document

To create an equation as a separate document, open the Math component of LibreOffice using one of these methods:

- On the menu bar, choose **File > New > Formula**.
- On the standard toolbar, click the triangle to the left of the **New** icon and choose **Formula**.
- From the Start Center, click **Formula**.

An empty formula document opens (see Figure 1).

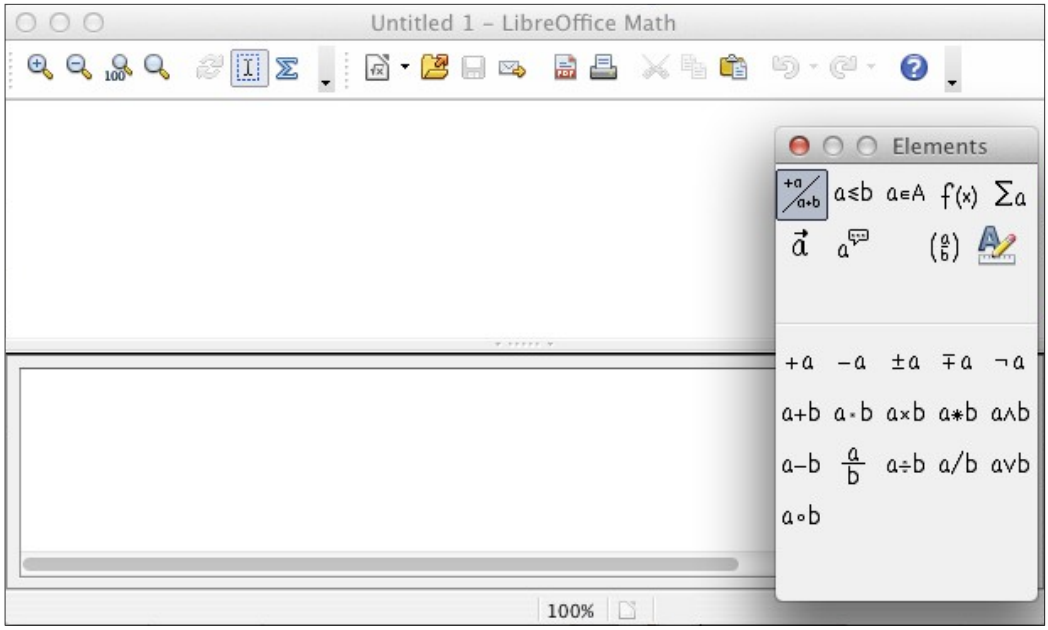


Figure 1: An empty formula document

The upper area is the preview window, where the equation will appear during and after input. The lower area is the equation editor, where the markup code for the equation is entered. The floating Elements window will also appear.

Inserting a formula into a Writer document

To insert a formula into a Writer document, open the document and then choose **Insert > Object > Formula** from the menu bar.

The formula editor opens at the bottom of the Writer window, and the floating Elements window appears. You will also see a small box with a gray border in your document, where the formula will be displayed, as shown in Figure 2.

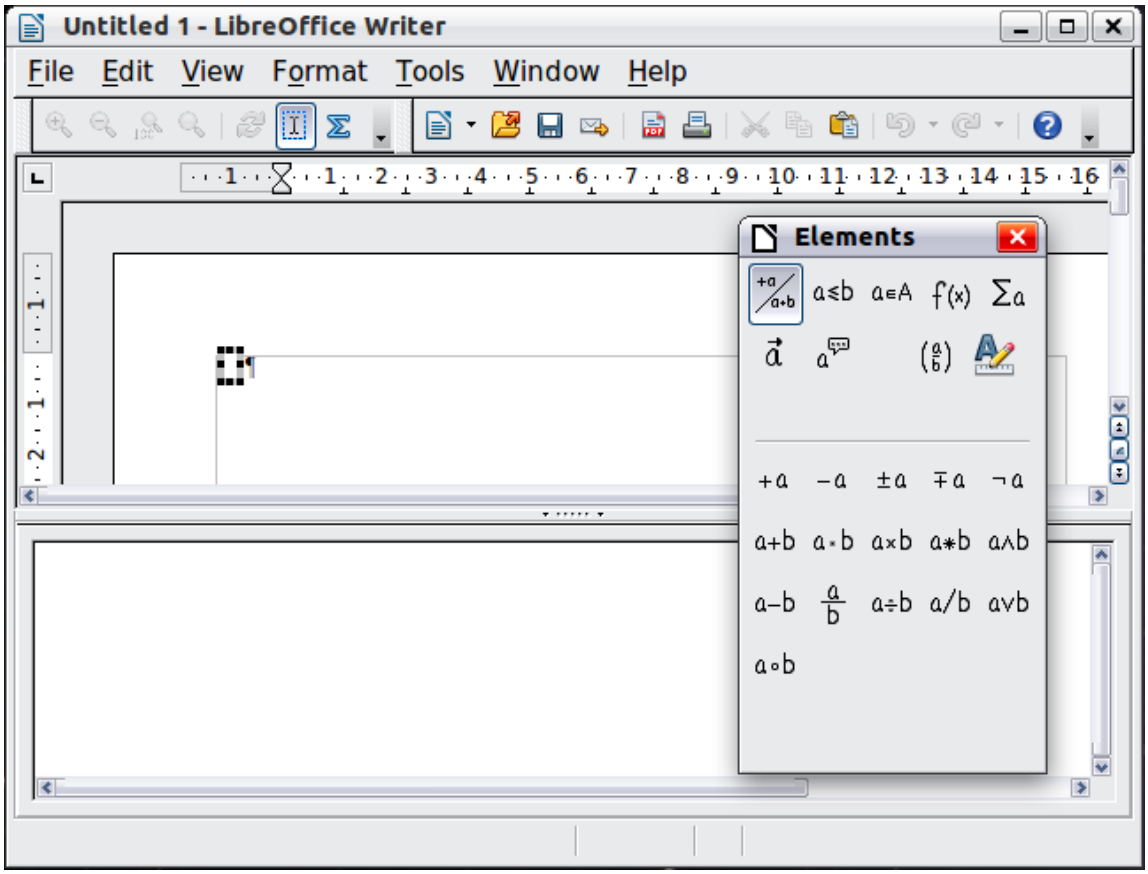


Figure 2: Equation Editor, Elements window, and location of resulting equation in Writer

Tip	If the Elements window is missing, show it with **View > Elements**.

When you are done entering the formula, you can close the editor by pressing the *Esc* key or by clicking an area outside the formula in the main document. A double-click will open the editor again, so you can edit the formula.

Formulas are inserted as OLE objects. In a Writer document, the formula is anchored as a character, so it is embedded in the continuous text. As with any other OLE object, you can change the anchor and make the formula floating. In Calc, Impress, and Draw documents, formulas are embedded as floating OLE objects.

If you frequently need to insert formulas, you may wish to add the **Formula** button to the Standard toolbar or create a keyboard shortcut. See “Add button to toolbar” or “Add keyboard shortcut” on page 28.

Entering a formula

The equation editor uses a markup language to represent formulas. For example, *%beta* creates the Greek character beta (β). This markup is designed to read similar to English whenever possible. For example, *a over b* produces a fraction: $\frac{a}{b}$.

You can enter a formula in three ways:

- Select a symbol from the Elements window.
- Right-click on the equation editor and select the symbol from the context menu.
- Type markup in the equation editor.

The context menu and the Elements window insert the markup corresponding to a symbol. This provides a convenient way to learn the LibreOffice Math markup.

Note	Click on the document body to exit the formula editor. Double-click on a formula to enter the formula editor again.

The Elements window

The simplest method for entering a formula is the Elements window.

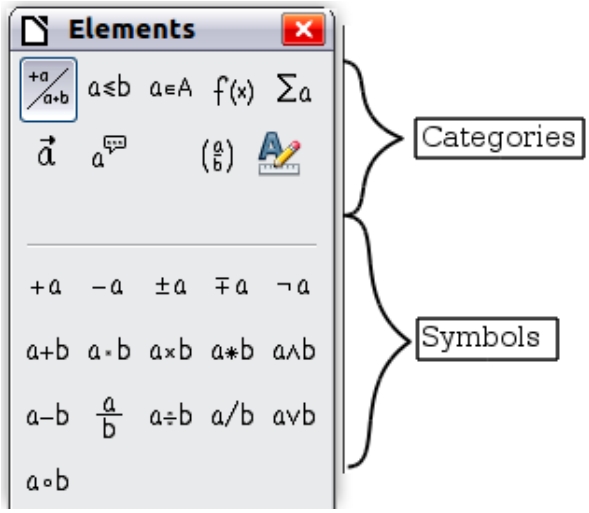


Figure 3: Symbols are divided into categories

The Elements window is divided into two main parts.

- The **top** shows the symbol categories. Click on these to change the list of symbols.
- The **bottom** shows the symbols available in the current category.

Tip	You can hide or show the Elements window with **View > Elements**.

Example 1: 5×4

For this example we will enter a simple formula: 5×4 . On the Elements window:

1) Select the top-left button of the categories (top) section.
2) Click on the multiplication symbol.

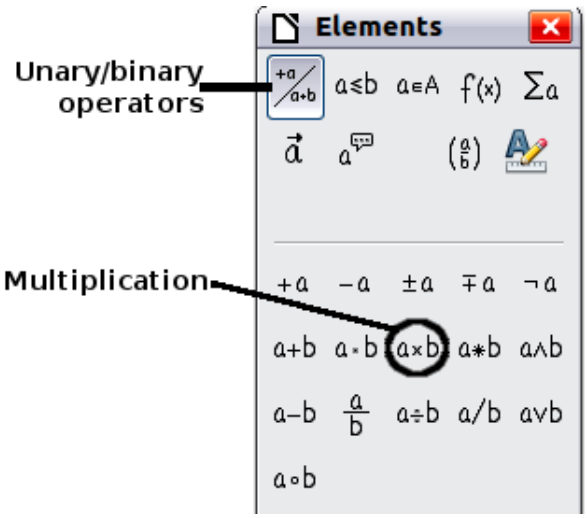


Figure 4: Selecting the multiplication symbol

When you select the multiplication symbol on the Elements window, two things happen:

- The equation editor shows the markup: <?> times <?>
- The body of the document shows a gray box like this: ❑×❑

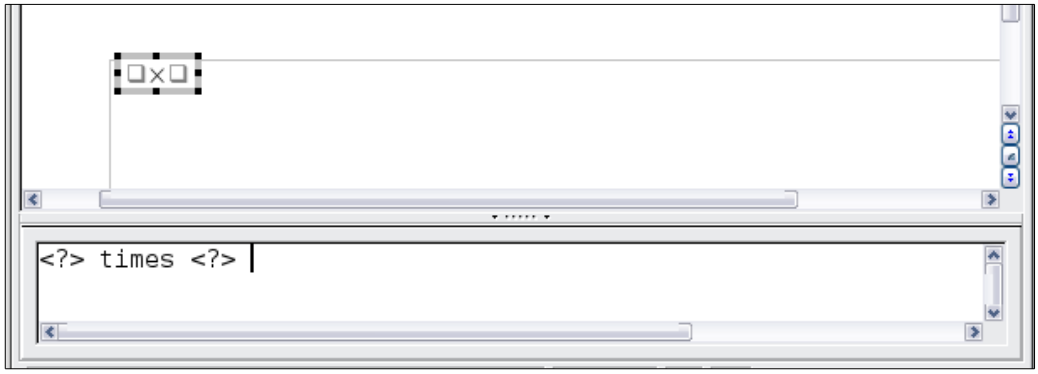


Figure 5: Result of selecting the multiplication symbol

The <?> symbols shown in Figure 5 are placeholders that you can replace by other text, for example **5** and **4**. The equation will update automatically, and the result should resemble Figure 6.

Tip	When you add a formula, reserved placeholders are indicated by squares in the formula and **<?>** in the command window. You can navigate through these placeholders using *F4* and *Shift+F4*.

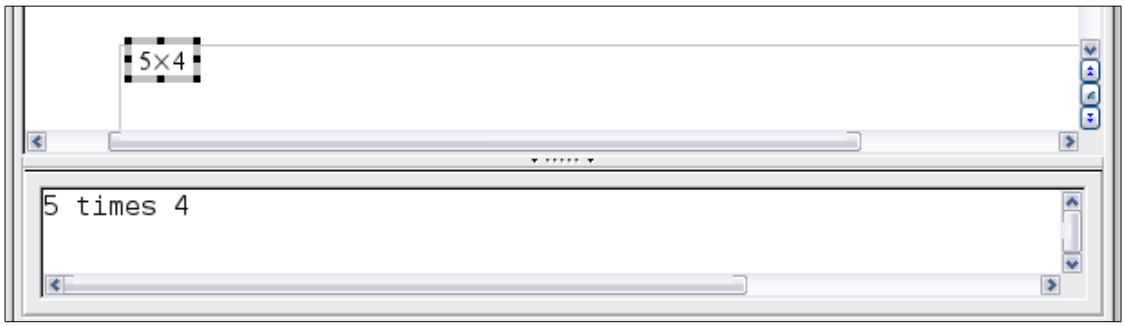


Figure 6: Result of entering 5 and 4 next to the times operator

Tip	To keep the equation from updating automatically, click **View > AutoUpdate display** to deselect it. To update a formula manually, press *F9* or select **View > Update**.

Right-click (context) menu

Another way to access mathematical symbols is to right-click on the equation editor. This pops up the menu shown in Figure 7. The items in this menu correspond to those in the Elements window, with some extra commands.

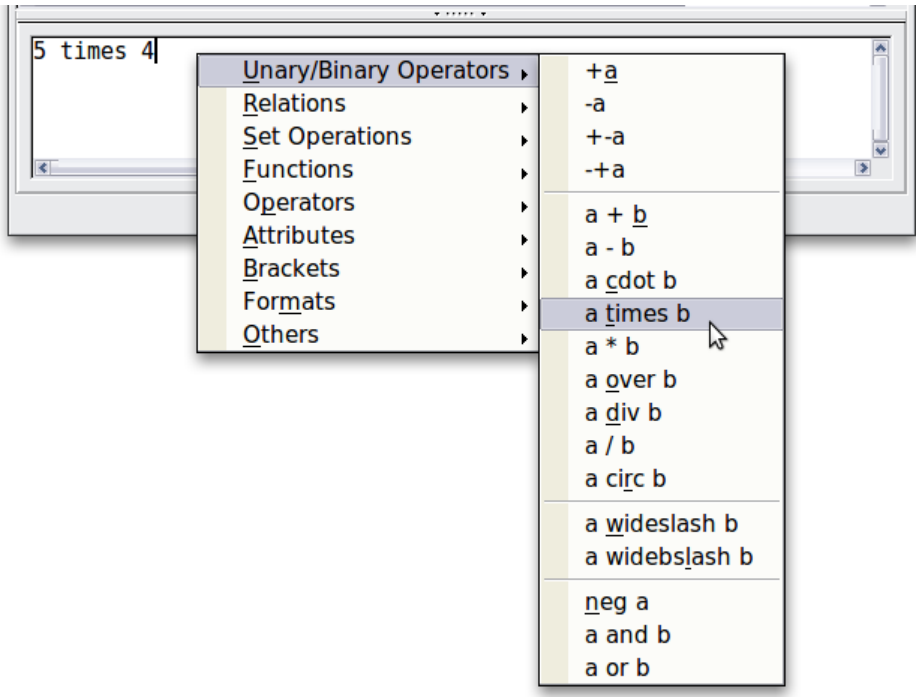


Figure 7: Right-click (context) menu

Note	Neither the window elements nor the context menu contains a complete list of commands. For some seldom-used commands, you must always enter the markup. A complete list of commands can be found in this book, starting on page 40.

Markup

You can type the markup directly into the equation editor. For example, you can type **`5 times 4`** to obtain 5×4 . If you know the markup, this can be the fastest way to enter a formula.

Tip	The formula markup resembles the way the formula reads in English.

Below is a short list of common equations and their corresponding markup.

Display	*Command*	*Display*	*Command*
$a=b$	a = b	$\sqrt{a}$	sqrt {a}
a^2	a^2	a_n	a_n
$\int f(x)dx$	int f(x) dx	$\sum a_n$	sum a_n

Display	Command	Display	Command
$a \leq b$	a <= b	∞	infinity
$a \times b$	a times b	$x \cdot y$	x cdot y

Greek characters

Greek characters ($\alpha, \beta, \gamma, \theta$, etc) are common in mathematical formulas. *These characters are not available in the Elements window or the right-click menu*. Fortunately, the markup for Greek characters is simple: Type a % sign followed by the name of the character, in English.

- To write a *lowercase* character, type the name of the character in lowercase.
- To write an *uppercase* character, type the name of the character in uppercase.
- To write in italic, type an `i` between % sign and the name of the character.

A complete table of Greek characters is provided on page 50. See the table below for some examples.

Lowercase	Uppercase	Italic lowercase	Italic uppercase
`%alpha` → α	`%ALPHA` → A	`%ialpha` → α	`%iALPHA` → A
`%beta` → β	`%BETA` → B	`%ibeta` → β	`%iBETA` → B
`%gamma` → γ	`%GAMMA` → Γ	`%igamma` → γ	`%iGAMMA` → Γ
`%psi` → ψ	`%PSI` → Ψ	`%ipsi` → ψ	`%iPSI` → Ψ
`%phi` → φ	`%PHI` → Φ	`%iphi` → φ	`%iPHI` → Φ
`%theta` → θ	`%THETA` → Θ	`%itheta` → θ	`%iTHETA` → Θ

Another way to enter Greek characters is by using the Symbols catalog window. Choose **Tools > Catalog**. This window is shown in Figure 8. Under *Symbol set*, select **Greek** and double-click on a Greek letter from the list. The markup name of the character is shown below the list window.

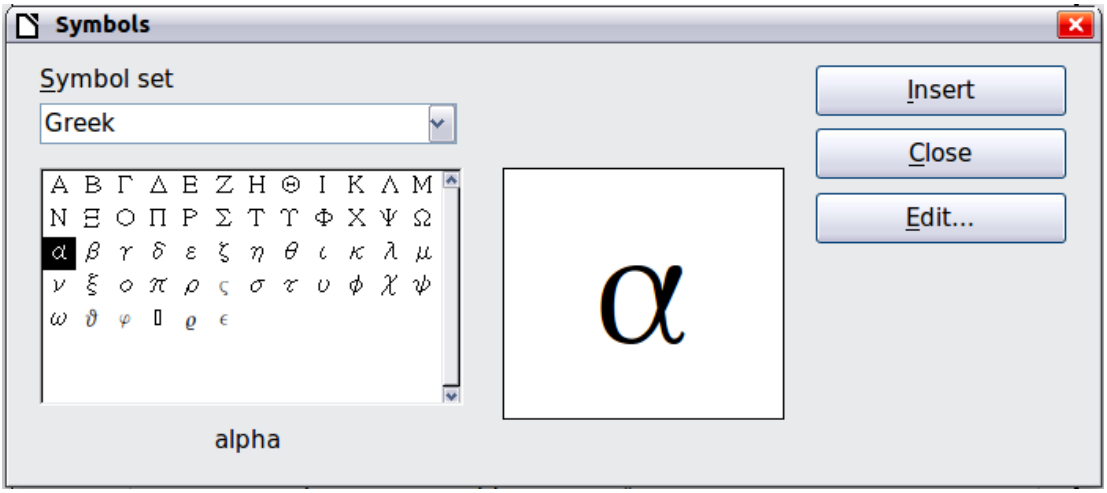


Figure 8: Symbols catalog, used for entering Greek characters and some special symbols

Example 2: $\pi \simeq 3.14159$

For this example we will suppose that:

- We want to enter the above formula (the value of pi rounded to 5 decimal places).
- We know the name of the Greek character (pi).
- But we do not know the markup associated with the $\simeq$ symbol.

Step 1: Type **%** followed by the text **`pi`**. This displays the Greek character π .

Step 2: Open the Elements window (**View > Elements**).

Step 3: The $\simeq$ symbol is a relation, so we click on the Relations button. If you hover the mouse over this button you see the tooltip *Relations* (Figure 9).

Figure 10 shows the Elements window after clicking the Relations button. The symbol we want is circled.

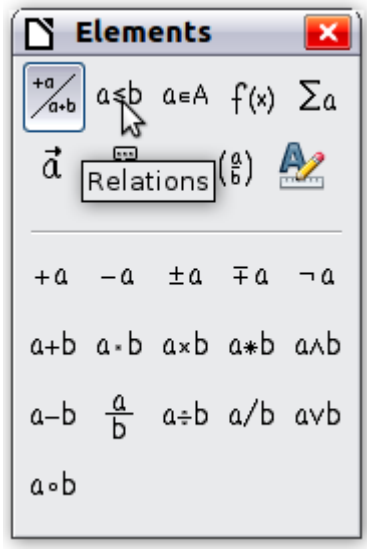


Figure 9: Tooltip indicates the Relations button

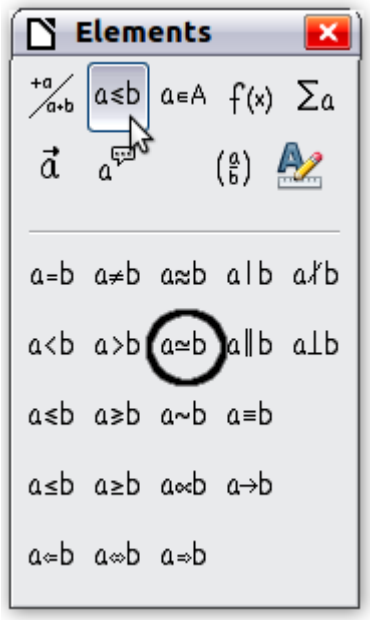


Figure 10: After selecting Relations

Step 4: Click on the a $\simeq$ b symbol. The equation editor now shows the markup **`%pi<?> simeq <?>`**.

Step 5: Delete the <?> text, press the *F4* key and type **`3.14159`** at the end of the equation. We end up with the markup **`%pi simeq 3.14159`**. The result is shown in Figure 11.

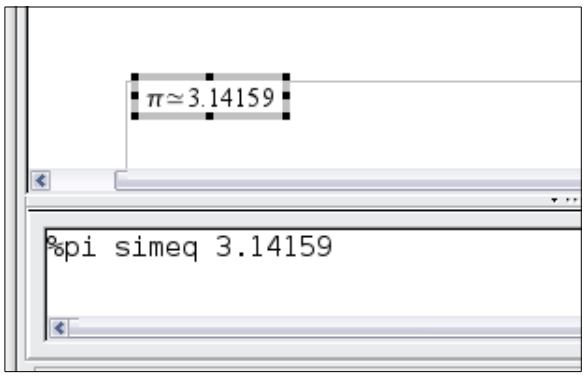


Figure 11. Final result

Changing a formula

You can change a formula at any time. To switch into edit mode, double-click on the formula.

Choosing the region to change

To get to the appropriate section in the markup code, do any one of the following:

- In the equation editor, click on the location.
- Select an area of the markup code that you wish to change.
- Click on an element in the preview area; the cursor will automatically move to the corresponding point in the equation editor.
- Double-click on an element in the preview area; the corresponding section in the equation editor will be selected.

To be able to work in the upper (preview) area in the stand-alone Math window (Figure 1), the formula cursor must be activated. Use the *Formula Cursor* button on the *Tools* toolbar.

Carrying out changes

You can change an equation by overwriting selected text or by inserting new markup code at the cursor position.

Tip	LibreOffice also provides a means of making changes directly in the preview area. This is still under development and therefore counts as “experimental”. In order to make use of it, you must make the following configuration change: go to **Tools > Options > LibreOffice > General** and check the box for **Enable experimental (unstable) features**. The activation of this option makes it impossible, by clicking on a formula element, to gain access to the corresponding position in the markup code.

Caution	The use of experimental features can lead to program crashes and/or loss of data. Only make the above change if you can accept this risk.

Formula layout

The most difficult part of using LibreOffice Math comes when writing complicated formulas. This section provides some advice.

Brackets are your friends

Math knows nothing about order of operation. You must use brackets to state the order of operations explicitly. Consider the following examples.

Markup	Result	Markup	Result
`2 over  x + 1`	$\frac{2}{x}+1$	`2 over {x + 1}`	$\frac{2}{x+1}$
`- 1 over 2`	$\frac{-1}{2}$	`- {1 over 2}`	$-\frac{1}{2}$

In the first example, Math has recognized that the **2** before and the **x** after the **over** belong to the fraction, and has represented them accordingly. If you want **x+1** rather than **x** to be the denominator, you must bracket them together so that both will be placed there.

In the second example, Math has recognized the minus sign as a prefix for the **1** and has therefore placed it in the numerator of the fraction. If you wish to show that the whole thing is negative, with the minus sign in front of the fraction, you must put the fraction in brackets in order to signify to Math that the characters belong together.

The braces belong solely to the layout of the markup code and are not printed. If you wish to use braces in the formula, use the commands **lbrace** and **rbrace.**

Compare the following examples:

Markup	Result	Markup	Result
`x over {-x + 1}`	$\frac{x}{-x+1}$	`x over lbrace -x + 1 rbrace`	$\frac{x}{\{-x+1\}}$

Brackets with matrices look ugly!

For background, we start with an overview of the matrix command.

Markup	Result
`matrix { a # b ## c # d }`	$\begin{matrix} a & b \\ c & d \end{matrix}$

Note Rows are separated by two #'s and entries within each row are separated by one #.

The first problem people have with matrices is that brackets do not scale with the matrix:

Markup	Result
`(  matrix { a # b ## c # d }  )`	$(\begin{matrix} a & b \\ c & d \end{matrix})$

Math provides scalable brackets. That is, the brackets grow in size to match the size of their contents. Use the commands *left(* and *right)* to make scalable brackets.

Markup	Result
`left(  matrix { a # b ## c # d }  right)`	$\left(\begin{matrix} a & b \\ c & d \end{matrix}\right)$

Tip	Use *left[* and *right]* to obtain square brackets. The list of all available brackets is on page 46.

Tip	If you want all brackets to be scalable, go to **Format > Spacing**. Then in the *Spacing* dialog, choose the category *Brackets* and check the option **Scale all brackets**.

These scalable brackets may also be used with any element, such as fractions, square root, etc.

Isolated and unpaired brackets

Math expects that for every opening bracket there will be a closing one. If you forget a bracket, Math places an inverted question mark by the corresponding bracket. This disappears when all brackets are matched. Sometimes forgetting a bracket causes the whole structure of the formula to fall apart.

However, an unpaired bracket is sometimes necessary. In such cases, you have two options:

- With non-scalable brackets, use a preceding backslash `\` to indicate that the following character should not be regarded as a bracket but as a literal character. So the half-open interval **`[a;b[`** is represented by **`\[a;b\[`** — try comparing this with **`[a;b[`**
- Scalable brackets can also be unpaired. The same half-open interval is represented by

```
left [ a; b right [
```

For scalable brackets, you can also use the command **none** to replace a non-existent paired bracket.

$$|x| = \begin{cases} x \text{ for } x \geq 0 \\ -x \text{ for } x < 0 \end{cases}$$

can be represented by

```
abs x = left lbrace stack {x "for" x >= 0 # -x "for" x < 0} right none
```

Recognizing functions in Math

In the basic installation, Math outputs variables in italics. If you enter a function, Math usually recognizes it and outputs it normally (a list of recognized functions begins on page 40). If Math fails to recognize a function, you can inform Math about it. Enter the markup code **func** before the function, and the following text will be recognized as a function.

Some functions recognized by Math need to be followed by numbers or variables. If these are missing, Math puts an inverted red question mark ¿ in their place, which you can only remove by correcting the formula: enter a variable or a number, or a pair of empty braces { } as a placeholder.

Tip	You can navigate through errors using *F3* and *Shift+F3*.

Equations over more than one line

Suppose you want to make an equation covering more than one line, for example: $x = 3$ / $y = 1$

Your first reaction would be to simply press the *Enter* key. However, if you press the *Enter* key, although the markup goes to a new line, the resulting equation does not. You must type the newline command explicitly. This is illustrated in the table below.

Markup	Result
`x = 3` `y = 1`	$x=3\,y=1$
`x = 3 newline` `y = 1`	$x=3$ $y=1$

Continuing the calculation on a new line without writing a complete new equation is not directly possible, because Math expects a term on the left hand side of an equals sign. You can substitute:

- Empty quotes "". This will automatically cause the line to be left-justified.
- Empty braces { }. The line will then be centered.
- Spaces characters ` or ~. The line will be centered with the spaces.

The alignment of equals signs under each other is described on page 17.

As well, spacing between elements in formulas are not set by space characters in the code. You need to use special markup to add spaces: ` (grave) for a small space, ~ for a large space. Another solution would be to add space characters between quotes, to be considered as text. Space markup at the end of a formula is ignored by default (see “Space at the end of a formula” on page 32).

How do I add limits to my sum/integral?

The sum and int commands (see complete list on page 44) can optionally take the parameters *from* and *to*. These are used for lower and upper limits respectively. These parameters can be used singly or together.

Markup	Result
`sum from k = 1 to n a_k`	$\sum_{k=1}^{n} a_k$

`int from 0 to x f(t) dt`
or
`int_0^x f(t) dt`
	$\int_{0}^{x} f(t)\,dt$ or $\int_0^x f(t)\,dt$

`int from Re f`	$\int_{\Re} f$
`sum to infinity 2^{-n}`	$\sum^{\infty} 2^{-n}$

Note For more details on integrals and sums, see page 44.

How do I write a derivative?

Writing derivatives essentially comes down to one trick: *Tell LibreOffice it’s a fraction*.

In other words, you have to use the *over* command. Combine this with either the letter *d* (for a total derivative) or the *partial* command (for a partial derivative) to achieve the effect of a derivative.

Note Notice that we have to use braces (squiggly brackets) to make the derivative.

Markup	Result
`{df} over {dx}`	$\frac{df}{dx}$
`{partial f} over {partial y}`	$\frac{\partial f}{\partial y}$
`{partial^2 f} over {partial t^2}`	$\frac{\partial^2 f}{\partial t^2}$

To write function names with primes, as is usual in school notation, you must first add the signs to the catalog. Using single and double quotes is typographically ugly. See “Customizing the catalog” on page 30.

Markup characters as regular characters

Characters that are used for controlling markup cannot be entered directly as normal characters. The characters concerned are: **%**, **{**, **}**, **&**, **|**, _, **^** and **"**. So, for example, you cannot write **2% = 0.02** or **1" = 2.56cm.** Two methods are available to overcome this limitation:

- Use double quotes to mark the character as text, for example **2"%"= 0.02.** Obviously this is not possible for the double-quote character itself.
- Add the character to the catalog. See “Customizing the catalog” on page 30.

In some cases you can use commands:

- **lbrace** and **rbrace** give you literal braces $\{\}$.
- **mline** gives you the vertical line, for example **2 mline 3** for $(2|3)$

Conversion into a character entity as in HTML or the use of an escape character is not possible in Math.

Text in a formula

To include text in a formula, enclose it in straight double-quotes:

```
abs x = left lbrace matrix {x # "for " x >= 0 ## -x # "for " x < 0} right none
```

$$|x| = \begin{cases} x & \text{for } x \geq 0 \\ -x & \text{for } x < 0 \end{cases}$$

All characters except double quotes are permissible in text. Unfortunately the Special Characters dialog is not available. If necessary, you can write the text in a text document and copy it into the equation editor using the clipboard. In this way you can insert smart quotes, as shown in Figure 12.

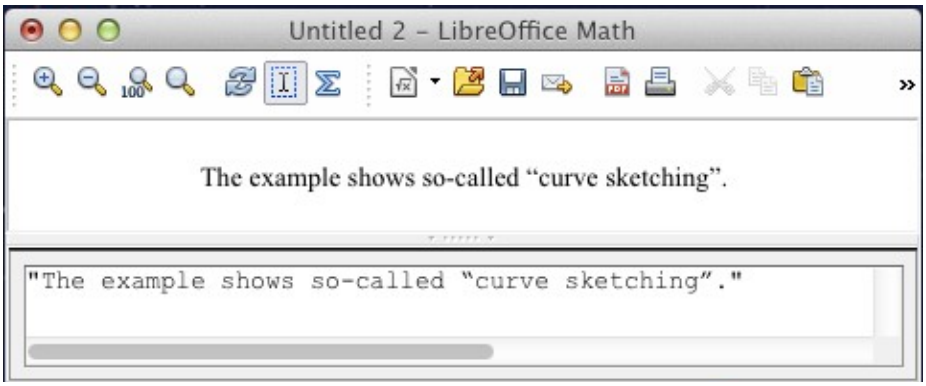


Figure 12: Smart quotes included by copy and paste from Writer

Text is shown in the font that was selected from the **Text** list in the **Fonts** dialog (compare with the section “Changing the font” on page 19). To use a font from the lower window of the equation editor, set the attribute **Serif**, **Sans** or **Fixed** before the text.

By default, text is left-justified. You can change the justification with **alignc** or **alignr** (see “Changing the alignment” on page 21).

Commands are not interpreted within text. Use quotes to break up the text if you wish to use special formatting commands.

```
"In " color blue bold "isosceles" "triangles, the base angles are equal"
```

In **isosceles** triangles, the base angles are equal

How do I align my equations at the equals sign?

Math does not have a command for aligning equations on a particular character, but you can use a matrix to do this, as shown below.

Markup	***Result***

`matrix{`
`  alignr x+y # {}={} # alignl 2 ##`
`  alignr x  # {}={} # alignl 2-y`
`}`
	$\begin{matrix} x+y & = & 2 \\ x & = & 2-y \end{matrix}$

The empty braces around = are necessary because = is a binary operator and thus needs an expression on each side. You may replace them with space characters (` or ~).

You can reduce the spacing around = if you change the inter-column spacing of the matrix:

1) With the equation editor open, choose **Format > Spacing** from the menu bar.
2) In the Spacing dialog (Figure 13), click the **Category** button and select **Matrices** in the drop-down menu.
3) Enter **0%** for Column spacing and click **OK**.

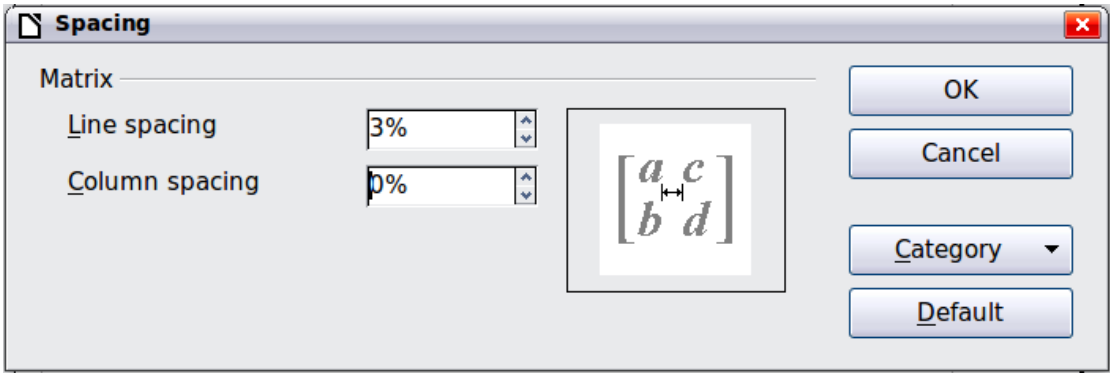


Figure 13: Changing spacing in a matrix formula

Changing the appearance of formulas

Changing the font size

This is one of the most common questions people ask about LibreOffice Math. The answer is simple, but not intuitive:

1) Start the formula editor and choose **Format > Font size**.
2) Select a larger font size under *Base size* (top-most entry).

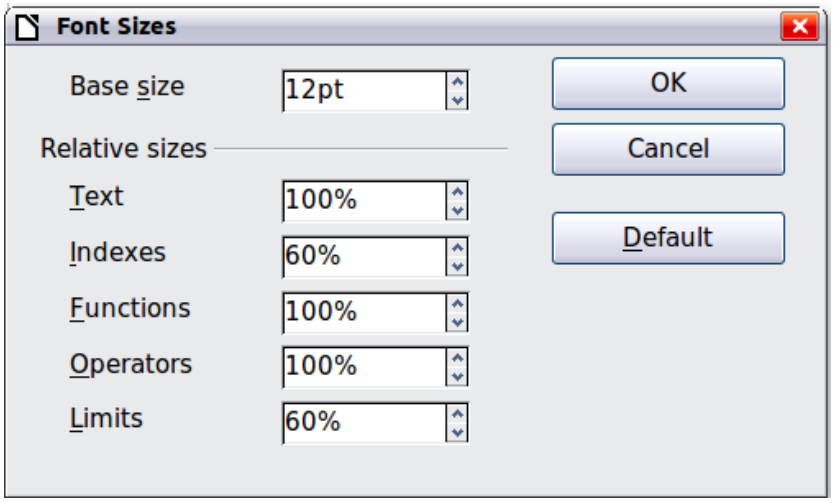


Figure 14. Edit Base size (top) to make a formula bigger

The result of this change is illustrated in Figure 15.

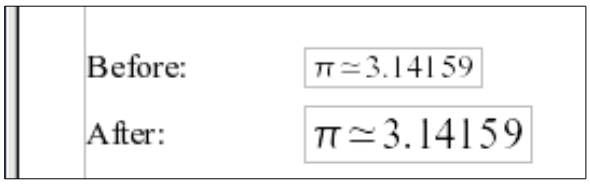


Figure 15. Result of changing the base font size

Note	The change in font size applies only to the current formula. To change the font size generally, click on the **Default** button and then **OK**. A general change in the font size might, for example, make your work easier when you are preparing a big presentation and want all the formulas in it to have a base size of 28pt—but do not forget to set the font size back to its original value when the work is finished. Warning: this method modifies only the current formula and future formulas you write. To modify all formulas already existing in the document, you need to use a macro (see page 38).

The size of a subset of characters in a formula may be modified using the **`size`** command. For example: **`b size 5{a}`** : b_a . In the Elements window, the icon  on the *Attributes* tab gives the **`size`** command. The value just after **`size`** may be absolute (numeric value) or relative to the context (base size by default); for example, **`+6`**, **`-3`**, **`/2`**, or **`*2`**.

Changing the font

The fonts used in formulas can be changed using **Format > Fonts.**

The *Formula fonts* section of the *Fonts* dialog (Figure 16) refers to the four specified formula elements. The font for operators, relationships and brackets is not affected, as these elements normally come from the OpenSymbol font. Similarly elements from the catalog (see “Customizing the catalog” page 30) continue to be displayed in the font specified there.

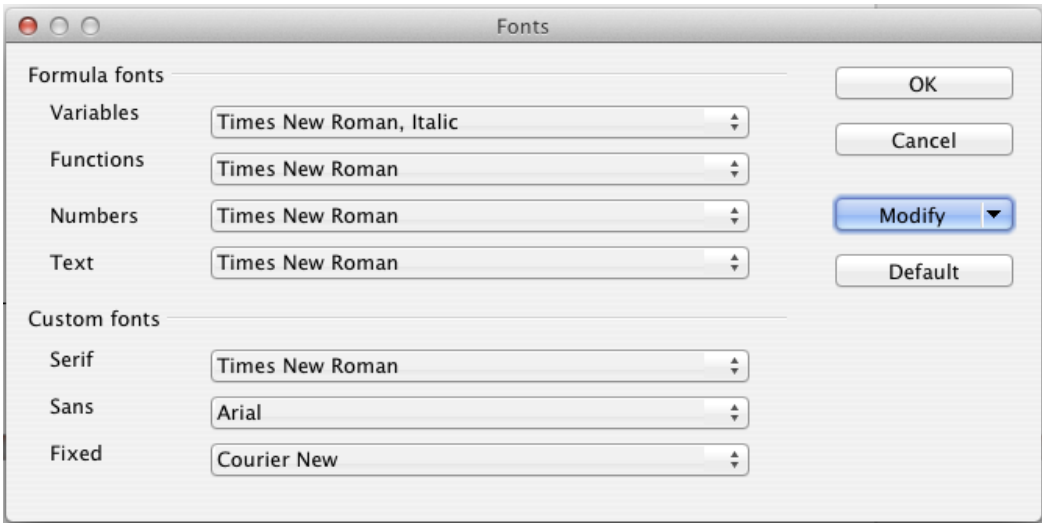


Figure 16: Fonts dialog

The *Custom fonts* section determines which font will be used when the attribute **font serif**, **font sans** or **font fixed** is specified.

To change a font, first click on **Modify** and choose the type of entry you wish to change. The Fonts dialog opens, showing all the fonts available on your system.

Figure 17: Fonts dialog

If you enter an initial character in the upper text box, the list will jump to that place. By entering a few additional characters, you can specify the exact font that you want. If you do not know its name, use the scrollbar to scroll through them. Click on any name and the box below will show a preview.

Variables should be written in italics, so make sure that the *Italic* box is checked. For all other elements, use the basic (Roman) form. The style can be easily altered in the formula itself by using the commands **italic** or **bold** to set these characteristics and **nitalic** or **nbold** to unset them.

When you have chosen a new font for a formula, the old font remains in the list alongside the new one and can be selected again. This applies only to the current session; the old font is not stored permanently.

You can choose whatever fonts you like, but if you are exchanging documents with someone else, you should choose fonts that are present on your colleague's computer.

Adjusting spacing in formulas

To increase or decrease spacing in formulas, do the following:

1) Go to **Format > Spacing.**
2) In the *Spacing* dialog, click the triangle next to **Category**, and choose an entry from the list.
3) Choose appropriate spacing values (the types of spacing that can be set depend on the category) and click **OK**.

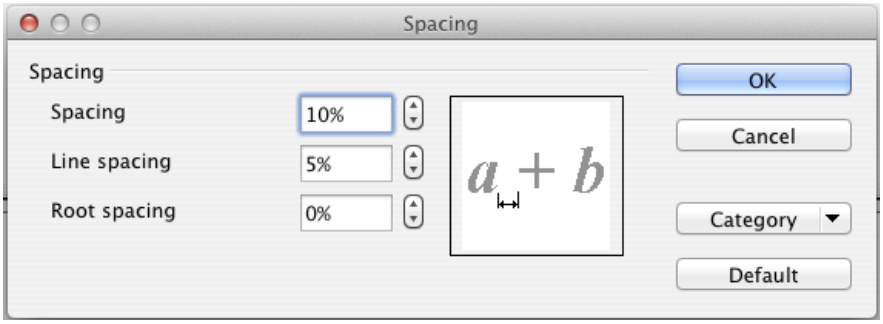
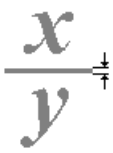
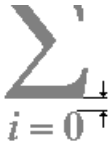
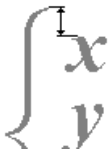
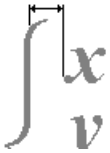
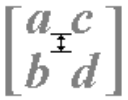
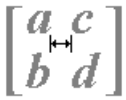
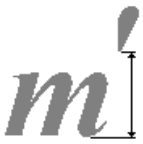
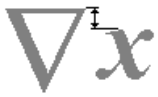
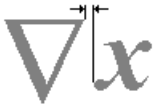
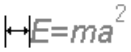
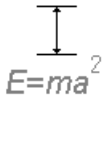
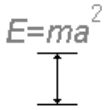


Figure 18: Spacing dialog

Note	Changes in spacing apply only to the current formula. To change spacing generally, click on the **Default** button and then **OK**.

Spacing changes are possible for the categories shown in the following table.

Category	***Possible adjustment***		
Spacing	$a + b$ Character spacing	$x = a$ $y = b$ Line spacing	$\sqrt{2}$ Root spacing
Indexes	m^2 Superscript	m_2 Subscript	
Fractions	$\frac{x}{y}$ Numerator	$\frac{x}{y}$ Denominator	

Fraction bars	 Excess length		 Line thickness	
Limits	 Upper limit		 Lower limit	
Brackets	 Excess size		 Spacing	
Matrices	 Line spacing		 Column spacing	
Symbols	 Primary height		 Minimum spacing	
Operators	 Excess size		 Spacing	
Borders	 Left	 Right	 Top	 Bottom

Changing the alignment

The alignment settings determine how formula elements located above one another are aligned horizontally relative to each other. Use **Format > Alignment** to choose between *Left*, *Centered or Right* on a global basis.

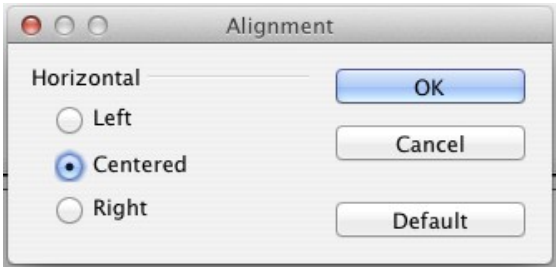


Figure 19: Alignment dialog

Here again you can use the **Default** button to make the change apply to all formulas and not just the current one. The following example shows the effect in different situations:

Alignment	*Examples*
left	$\frac{x^2-9}{x}$ $\left(\begin{matrix}100\\7\end{matrix}\right)$ $c^2=a^2+b^2-2ab\cos\gamma$ $\cos\gamma=\frac{c^2-a^2-b^2}{-2ab}$
centered (Default)	$\frac{x^2-9}{x}$ $\left(\begin{matrix}100\\7\end{matrix}\right)$ $c^2=a^2+b^2-2ab\cos\gamma$ $\cos\gamma=\frac{c^2-a^2-b^2}{-2ab}$
right	$\frac{x^2-9}{x}$ $\left(\begin{matrix}100\\7\end{matrix}\right)$ $c^2=a^2+b^2-2ab\cos\gamma$ $\cos\gamma=\frac{c^2-a^2-b^2}{-2ab}$

It is not possible to align formulas on a particular character.

Default settings of alignment do not apply to text elements; they are always aligned left. In the following examples, the default alignment is *right* but the first example begins the second line with text (albeit empty text) and is therefore aligned left.

Markup Code	*Result*

`{1+2+3+4} over 5 + 2 over {60+70+80+90}`
`newline`
`""=2+1 over 150`
	$\frac{1+2+3+4}{5}+\frac{2}{60+70+80+90}$ $=2+\frac{1}{150}$

`{1+2+3+4} over 5 + 2 over {60+70+80+90}`
`newline`
`{}=2+1 over 150`
	$\frac{1+2+3+4}{5}+\frac{2}{60+70+80+90}$ $=2+\frac{1}{150}$

Independent of the default alignment, it is possible to align formulas explicitly using the commands **alignl**, **alignc** and **alignr**. This attribute also works for text elements.

Changing the color

Use the command **color** to change the color of a subset of a formula: **color red ABC** gives *ABC* . Choose from 8 colors: **white**, **black**, **cyan**, **magenta**, **red**, **blue**, **green**, **yellow**.

You may give a color to a subset of a formula if it is gathered between **{ }** or other parentheses. For instance: **A B color green {C D} E** gives *A B C D E* .

If several colors are used, the one the more inside the formula is applied as with this example : **color blue {A B color yellow C D}** gives *A B C D* .

It is not possible to select a background color: it is always transparent in Math. The background color of the whole formula is then the same as the background of the document or frame (in a text document for instance). In Writer, you can use object properties (right-click > **Object**) to choose a background color for the whole formula (see “Background, borders, and size” on page 25).

Formulas in Writer documents

Numbering equations

Equation numbering is one of Writer's best hidden features. The steps are simple, but obscure:

1) Start a new line.
2) Type **fn** and then press *F3*.

The *fn* is replaced by a numbered formula:

$$E = mc^2 \tag{2}$$

Now you can double-click on the formula to edit it.

You can reference an equation ("as shown in Equation (2)") with these steps:

1) Choose **Insert > Cross-reference** from the menu bar.
2) On the *Cross-references* tab (Figure 20), under *Type*, select *Text*.
3) Under *Selection*, select the equation number.
4) Under *Insert reference to*, select *Reference*.
5) Click **Insert**.

If you later add more equations to the paper before the referenced equation, all the equations will automatically renumber and the cross-references will update.

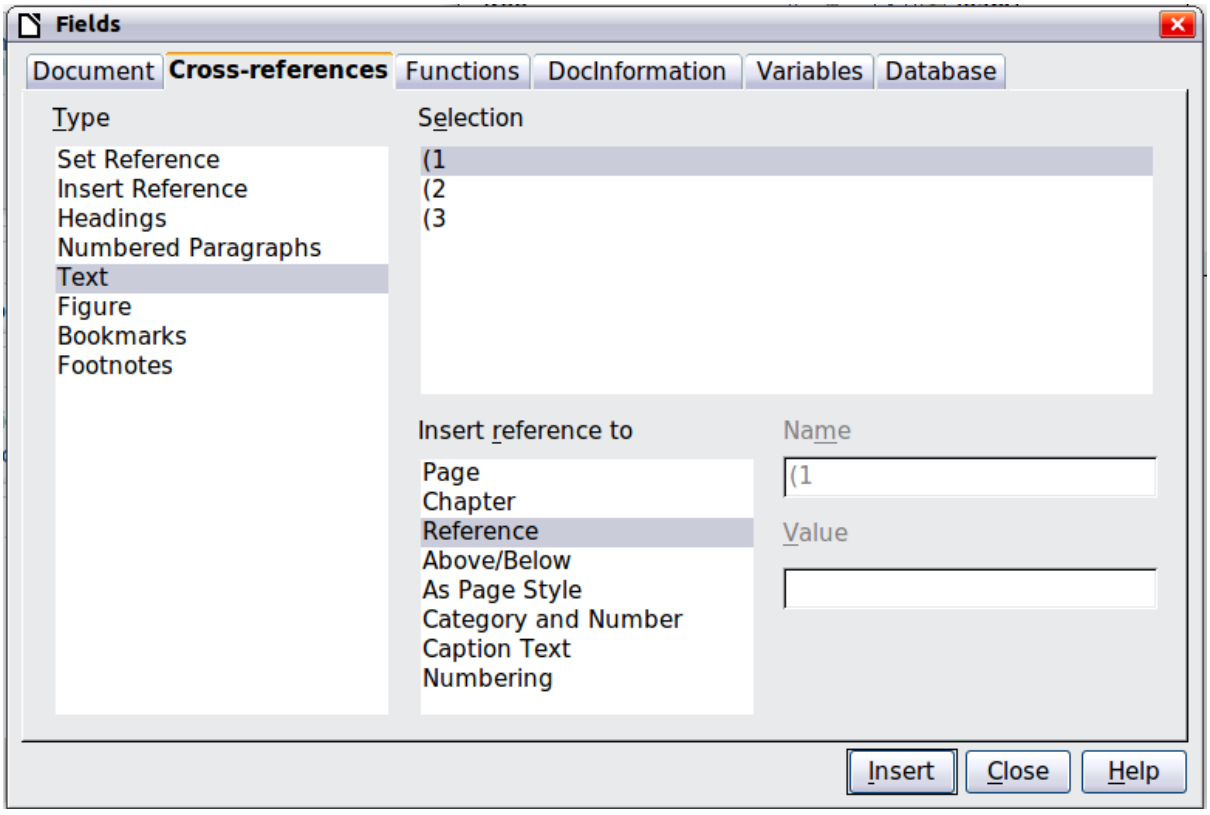


Figure 20: Inserting a cross-reference to an equation number

Tip	To insert the equation number without parentheses around it, choose *Numbering* instead of *Reference* under *Insert reference to.*

The AutoText inserted as a result of the above procedure consists of a 1x2 table. The left cell contains the formula and the right one the number. The number is an automatic counter named Text. You can edit the AutoText if, for example, you prefer square rather than round brackets, or if you want formula and number to be separated by tabs rather than formatted as a table. Read the section on “Using AutoText” in Chapter 3 (Working with Text) in the *Writer Guide*.

Position

Normally a formula is anchored *As character* in a Writer document. But as with any other OLE object, you can change the anchoring mode to position the formula where you want it. For more information, see Chapter 11 (Graphics, the Gallery, and Fontwork) of the *Getting Started g*uide.

By default, formula objects anchored *As character* are automatically aligned vertically to the baseline of the surrounding text. To align the formula manually, go to **Tools > Options > LibreOffice Writer > Formatting Aids** and uncheck the option **Math baseline alignment**. This setting is stored with the document and applies to all formulas within it. New documents use the current setting from this dialog.

Margins

An inserted Math object has margins to left and right of it, separating it from surrounding text. If you do not want this, it is best to alter the frame style for formulas, as this will apply simultaneously to all existing formulas and those still to be inserted in the document. Proceed as follows:

1) Press **F11**. The *Styles and Formatting* dialog opens.
2) Click on the *Frame Styles* tab.
3) Find the *Formula* frame style and right-click on it.
4) Choose *Modify* from the context menu. The *Frame Style* dialog opens.
5) Switch to the *Wrap* tab of the dialog. Change the values for *Left* and *Right* in the Spacing section to 0.00.
6) Click **OK** to close the dialog.

This changes the spacing for all formulas that have not had their spacing adjusted manually. You can find more information on using styles in “Default layout with style” on page 32 and in Chapter 3 (Using Styles and Templates) of the *Getting Started g*uide.

Text mode

Larger formulas should always be in a paragraph of their own, separated from the text. When you use formula elements in running text, they can often be higher than the letter height.

However, if it is necessary to place a formula within running text, switch into the equation editor and go to **Format > Text** mode. Math will try to shrink the formula to fit the letter height. The numerators and denominators of fractions are shrunk, and the limits of integrals and sums are placed beside the integral/sum sign.

Example:

A formula in a separate paragraph:

$$\sum_{i=2}^{5} i^2$$

and the same formula embedded in text mode: $\sum_{i=2}^{5} i^2$

Background, borders, and size

With regard to formatting, formulas are treated as objects of the *Frame Style* type with the Formula frame style. Background color and borders can be set using this style or directly with **Format > Frame/Object**, or by right-clicking the formula and choosing **Object** from the context menu. In the basic installation, formulas have a transparent background and no borders. The size of a formula cannot be adjusted; in a Writer document it depends directly on the way the formula is constructed (see “Changing the font size” on page 18).

Creating a formula library

When you use the Math component of LibreOffice directly with **File > New > Formula**, you create documents with the file suffix .odf, each containing a single formula. You can use these to build up a library of frequently-used formulas. Embedded formulas can also be stored as separate Math documents by right-clicking on the formula and choosing **Save copy as** from the context menu. To insert such a Math document into a Writer document, use **Insert > Object > OLE Object.** Select the option *Create from file* and enter the pathname of the file or browse for it using your system’s file manager by pressing the **Search** button.

Note	You cannot insert the document by dragging and dropping with the mouse, nor by using **Insert > File**.

Formulas cannot be stored in the gallery because they are not in graphical format. You can however store a formula as AutoText. Write the formula in a separate paragraph, select it and go to **Edit > AutoText**. For further information see “Using AutoText” in Chapter 3 (Working with Text) in the *Writer Guide*.

Fast insertion of formulas

If you already know the markup of your formula, here is a faster method to build your formula:

- Write the formula markup in Writer.
- Select the markup.
- Insert the formula using a toolbar button, a menu item, or a keyboard shortcut (see “Add keyboard shortcut” on page 29).

This method avoids the need to open and close the Math window and thus saves time.

Formulas in Calc, Impress, and Draw

Graphical properties

Formula objects have similar properties in Calc, Impress, and Draw. They are always inserted with a transparent background and without borders. In Draw and Impress, they are assigned the *Default* graphical style; there is no assigned style for them in Calc and properties must be specified directly. The properties shown in the Styles and Formatting window, the context menu, and the **Format > Graphic** menu are limited, when not available at all.

Line, Area, Shadow

You can set values, but the settings are completely ignored.

Text attributes

All text attributes, such as font and alignment, refer not to the formula text but to the text element present in all graphical objects. For a selected object, this can be accessed via the *F2* key. For further information, read the section “Working with text in Draw” in Chapter 10 (Advanced Draw Techniques) of the *Draw Guide*.

Position

A formula can be positioned anywhere you like. Use the mouse, the arrow keys, or the Position and Size dialog available from **Format > Position and Size**.

Size

In documents a formula is not represented directly but replaced by a graphic. The size of this graphic is initially protected. To change the size, open the Position and Size dialog and, on the first page, in the *Protect* section, uncheck the **Size** checkbox. This activates the *Size* section directly above. Enter your modifications. When the dialog is closed, the size will immediately be protected again.

Changing the size does not affect the formula itself but only its graphical representation. In particular the basic font size of the formula (see “Changing the font size” on page 18) does not change. To revert to the size determined by the formula content, use the *Original Size* option in the context menu. This allows sizing errors that are sometimes caused by inserting a formula from a different module to be corrected.

Rotation, shear, and flipping

Rotation, shearing, and flipping (creating mirror images) are not possible; the corresponding dialog options are inactivated. If you need to do this, convert the formula to a *GDI metafile* graphic. Once you have done this, it is not a formula any more but an image. Copy the formula onto the clipboard. Choose **Edit > Paste Special** or, on the Standard Toolbar, use the Paste menu (available from the small triangle to the right of the Paste icon) and choose the option **GDI Metafile.**

Combining formulas with text

Since a formula is an OLE object, you cannot insert it into the content of a spreadsheet cell, graphic or presentation object. Thus, unlike in Writer, you cannot integrate formulas with running text. Here are some alternatives.

Text elements within the formula

You can write the text directly into the formula. For example:

The illustration of a parallelogram suggests the relationship $\vec{AB}=\vec{CD} \Leftrightarrow \vec{AC}=\vec{BC}$.
But this does not constitute a proof.

The corresponding markup text is:

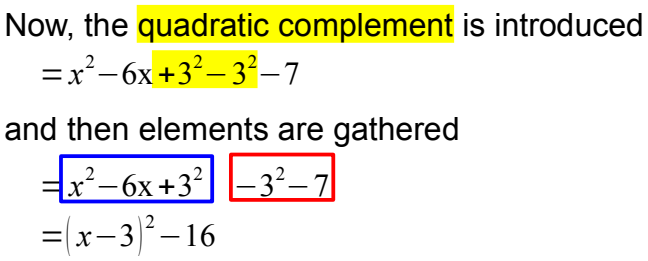
```
"The illustration of a parallelogram suggests the" newline
"relationship " widevec AB = widevec CD dlrarrow widevec AC = widevec BC "." newline
"But this does not constitute a proof."
```

Lines are not wrapped automatically; you must use the **newline** command.

You can use **Format > Fonts** to match the text to its surroundings. See “Changing the font” on page 19.

Grouping formulas with graphic objects

Formulas can easily be combined with other objects into a group, which is not possible in text documents. In the following example, text boxes are combined with formulas and rectangles are used to highlight parts of the formula.



Formulas in Writer OLE objects

To be able to use all the facilities possible in text documents, create the formula and its text in Writer. Choose a page size that roughly matches your eventual objective. Do not write more than one page because page selection causes problems later. Save the document.

In Impress (or Draw or Calc), use **Insert > Object > OLE Object** and then choose the option *Create from file*. The formula is embedded as an OLE object in a document that is itself an OLE object. As a result, the formula cannot be modified in Impress. For that reason, you should always activate the option *Link to file*. Then if you need to make a change, open the text document and edit text and formula there. In Impress, use **Edit > Links** and then the **Update** button to display the stored version of the text document. Do not try to modify the formula in Impress; it causes the representation to become faulty.

Formulas in charts

A chart is itself an OLE object. Therefore you cannot call up the equation editor within it. There is no freestanding module of LibreOffice that creates charts, so you cannot use anything similar to the above method using Writer either. However you can create a formula outside the charts editor, copy it to the clipboard, activate the chart for editing, and paste in the formula from the clipboard. This automatically converts the formula into a metafile graphic. If you wish to make further modifications, you must discard it and repeat the procedure.

Customizations

Here are a few ways to customize your use of Math.

Formula editor as a floating window

The formula editor can cover a large part of the Writer window. To turn the formula editor into a floating window, do this:

1) Hover the mouse over the editor frame, as shown in Figure 21.
2) Hold down the *Control* key and double-click.

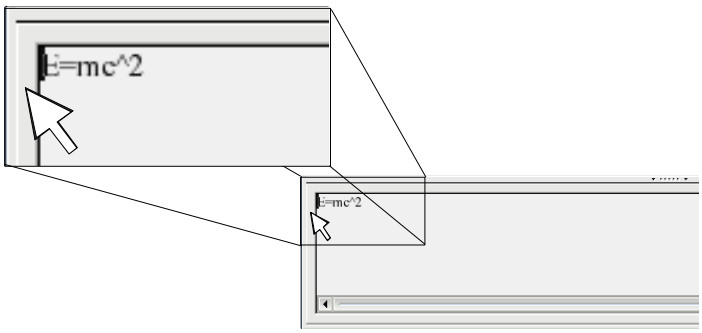


Figure 21: Turning the formula editor into a floating window

Figure 22 shows the result. You can dock the floating window again by using the same steps. Hold down the *Control* key and double-click the window frame.

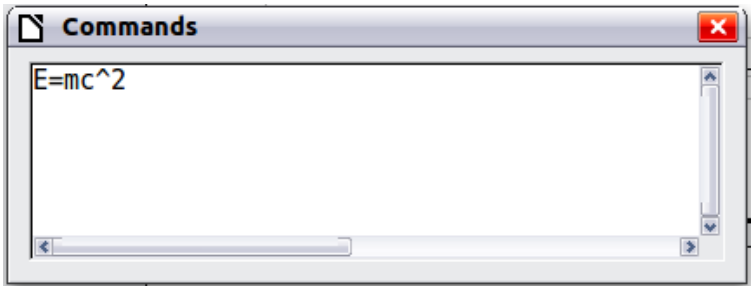


Figure 22: Formula editor as a floating window

Add button to toolbar

This button is by default present on the **Insert** toolbar in Writer and Impress. You can add it to other toolbars:

- **View > Toolbars > Customize** or right-click at the beginning of a toolbar **> Customize toolbar**. Choose the *Toolbars* tab and the toolbar you wish to modify.
- Select the position where you want to insert the button.
- Click **Add**.
- In the Add Commands dialog (Figure 23), select the **Insert** category, then the **Formula** command—the one corresponding to icon (command corresponds to an equivalent of a spreadsheet formula).

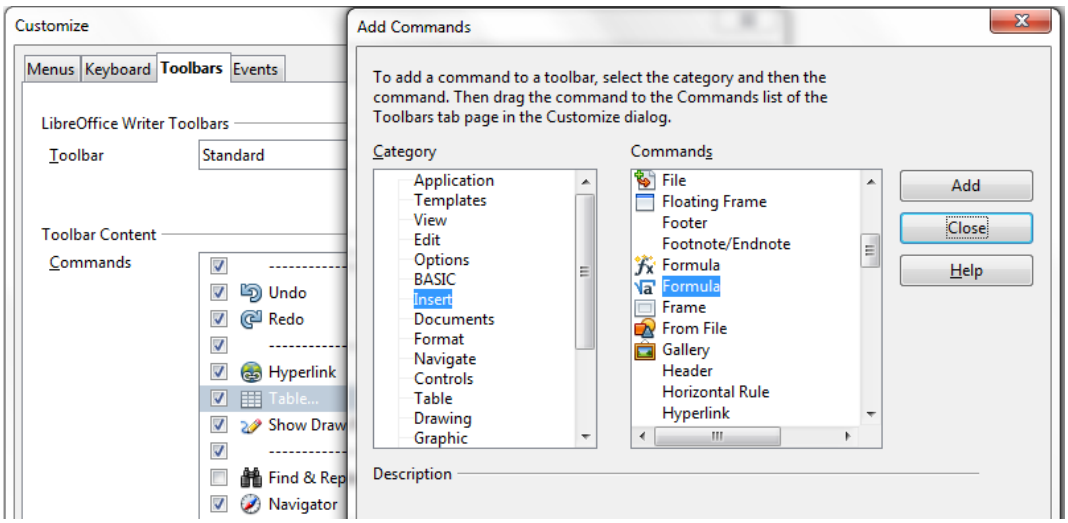


Figure 23: Add a button to insert a formula

- Click **Add**. Then you can continue to add buttons or click **Close**.
- On the Customize dialog, use the arrow buttons (not visible in) to move buttons up or down the list before clicking **OK** to save the changes.

Add keyboard shortcut

You can add a keyboard shortcut to insert a formula:

- **Tools > Customize**, *Keyboard* tab.
- Select the level of the shortcut: *LibreOffice* to benefit from it in all modules (if the shortcut is not used in this module) or only in the current module.
- Select the **Insert** category, then the **Formula** command—the one *not* corresponding to *F2* (*F2* corresponds to the equivalent of a spreadsheet formula).
- Select the shortcut to apply, then click **Modify**.
- Proceed the same way for other shortcuts with the same level, then click **OK** to save.

If your shortcut activates a formula bar (as for Calc), that means you chose the wrong *Formula* function.

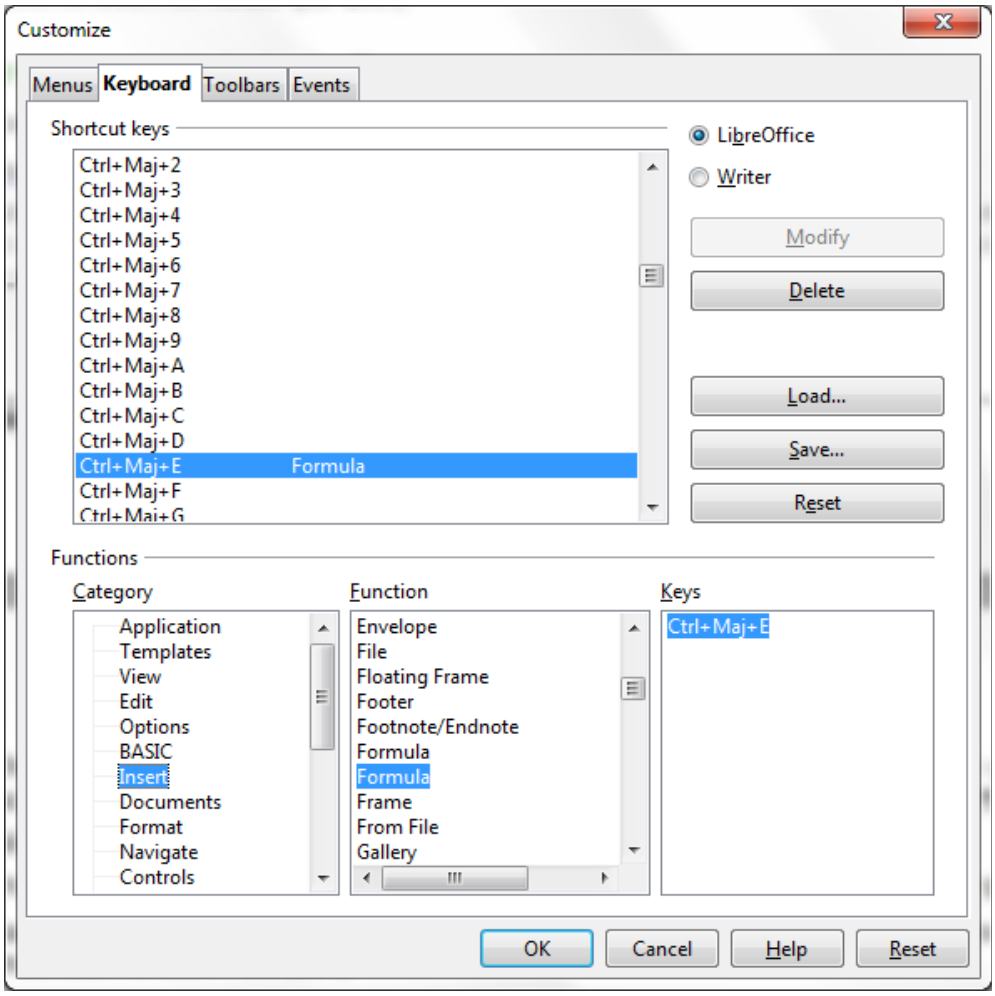


Figure 24: Add a keyboard shortcut

Customizing the catalog

If you need to use a symbol that is not available in Math, you can add it to the catalog yourself. The following example shows the procedure for symbols that are used in marking school work.

1) Go to **Tools > Catalog** or click on the catalog button to open the Symbols catalog (Figure 8).
2) Click the **Edit** button. The Edit symbols window (Figure 25) opens.
3) The symbols are arranged in so-called symbol sets. In the lower part of the window, choose an available set for your new symbol from the Symbol set list, for example the *Special* set. Or you can type the name of a new symbol set directly into the box.
4) From the Font list, choose a font that contains the desired symbol.

   When you have selected a font, its characters appear in the font summary window. You can scroll through it using the scrollbar at the side, or use the Subset list to go straight to the place you need.

   To follow this example, choose the font *DejaVu Sans* and the subset *General punctuation*.
5) Click on the desired symbol (here Ux2032). It appears enlarged in the right-hand preview box (see Figure 26). Make sure that the symbol set is set to **Special**.
6) In the Symbol field, enter a name for the symbol, for example *prime*.
7) If the name is not already in use, the **Add** button becomes active. Click on it.
8) You can immediately add more symbols. In the example, the “U+2033” symbol, named dblprime, and the “U+2034” symbol, named triprime, are added (see Figure 27).
9) Click **OK** to close the dialog.

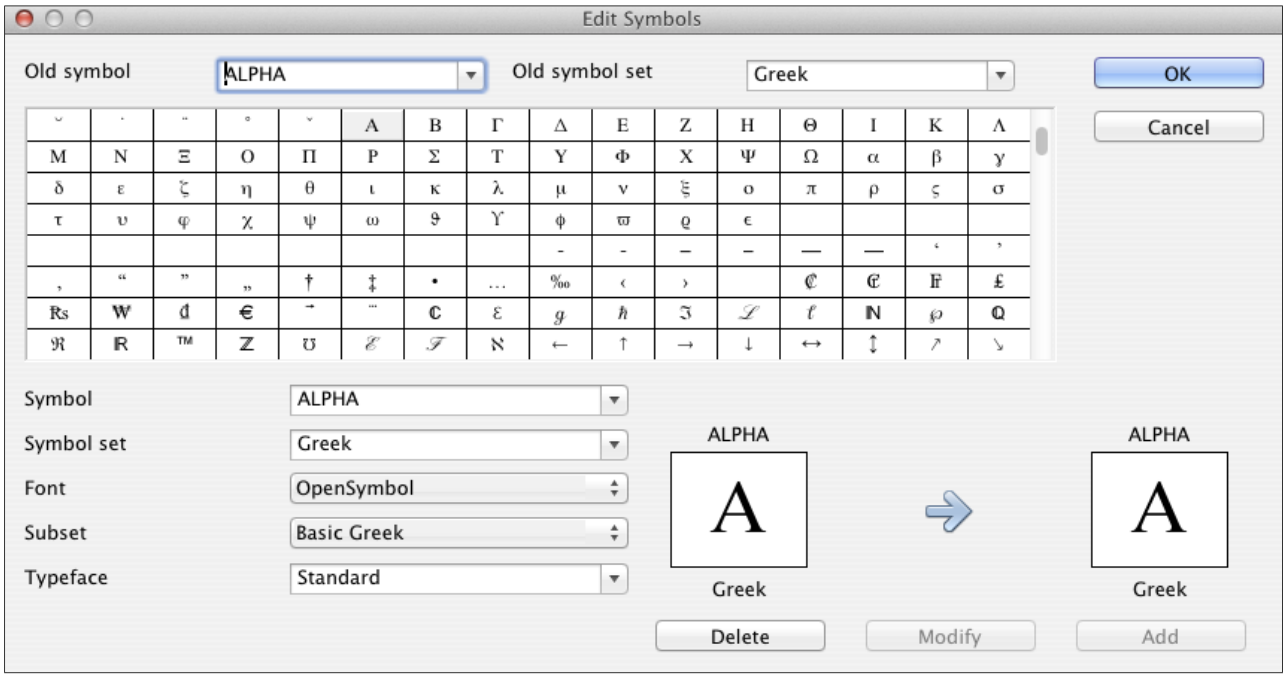


Figure 25: Edit Symbols dialog

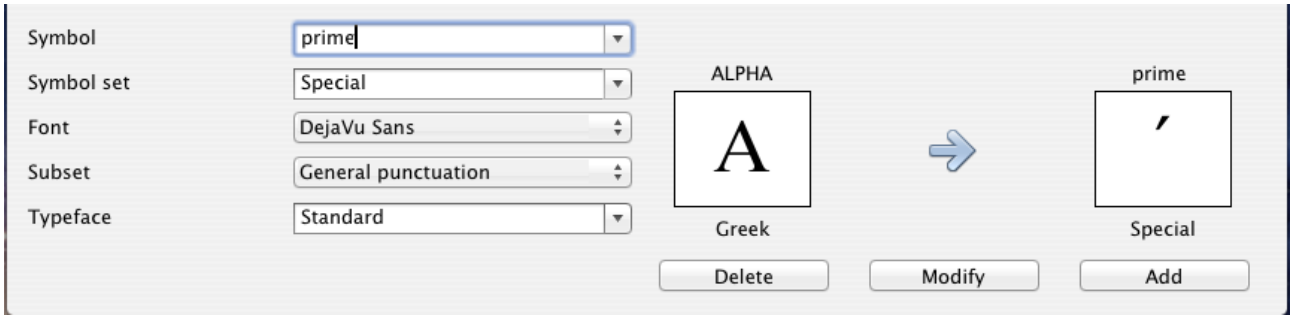


Figure 26: Edit Symbols dialog: characters selected

Now the Symbol set view area shows the new symbol. It can be selected just like the other symbols, either from here, or by writing them directly into the equation editor in the form %prime.

Caution

Symbols (Greek or Special), in contrast to commands, are case sensitive (lowercase/uppercase).

Numerous free fonts contain a great number of mathematical symbols. The "STIX[1]" font is worthy of special mention here. It was developed specially for writing mathematical/technical texts. The DejaVu[2] and Lucida[3] fonts also have a wide range of symbols.

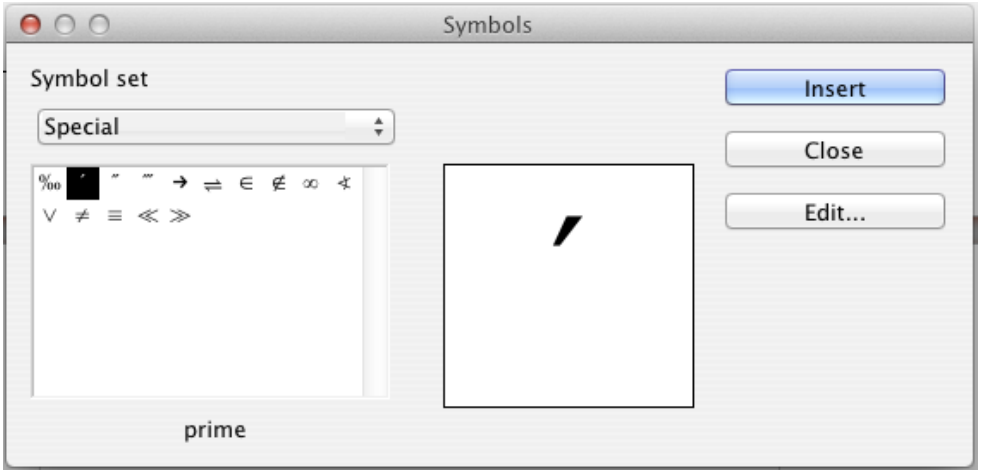


Figure 27: Catalog with new symbol

Note

Many symbols occur in more than one font. If you want to exchange documents with others, take care to use a font that is installed on their machine.

In the basic LibreOffice installation, only those user-defined symbols that actually occur in the document are stored with it. Sometimes it is useful to embed all the user-defined symbols, for example when the document is going to be further edited by another person. Go to **Tools > Options > LibreOffice Math** and on the **Settings** page, uncheck the option **Embed only used symbols (smaller file size)**.This setting is only available when you are working on a Math document.

1 Font files for STIX are available from http://www.stixfonts.org
2 Font files for DejaVu Sans are available from http://www.dejavu-fonts.org
3 Lucida Sans belongs to the JRE package, which is probably already on your computer.

Customizing operators

A name, to be recognized as a function (that is, to be written with regular font, not italic), must be preceded by the markup code **`func`**.

An operator, to be enlarged and to accept limits as integral or sum, must be preceded by the markup code **`oper`**. It is possible to use a letter, a name, or a personal symbol (see “Customizing the catalog” on page 30). $\mathop{F}_{i \in I} A_i$ is written **`oper F from {i in I} A_i`**.

You can modify the behavior of an operator or change one of your personal symbols in an operator, using the markup codes **`boper`**, to create a binary operator, and **`uoper`**, to create a unary operator.

Space at the end of a formula

The grave accent ` inserts a small additional space into the formula; the tilde ~ inserts a larger one. In the basic installation, these symbols are ignored when they occur at the end of a formula. However, when working with running text, it may be necessary to include spacing at the end of formulas as well. This customization is only available when you are working with a Math document, and not when you are embedding a Math object.

Open a Math document with **File > New > Formula**. Open the customization dialog with **Tools > Options** and choose **LibreOffice Math**. On the *Settings* page, uncheck the option **Ignore ~ and ` at the end of the line**.

Default layout with style

In Writer, formulas are formatted according to the Formula frame style. In the Style and Formatting window (displayed with *F11*), click on the third icon at the top: *Frame Styles*. Right-click on **Formula** and select **Modify**. By this means, you can directly modify all formulas in your document, regarding spacing (page 24) or background (page 25), unless you manually modify formula formatting.

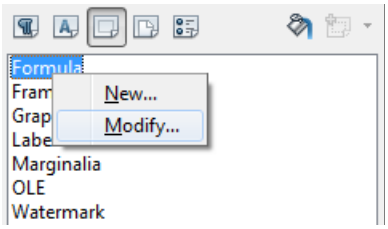


Figure 28 : Modify Formula Frame Style

To apply this style in all your new Writer documents, you must include the Formula style in your default template. To do so, create a new Writer document and modify the Formula frame style as you wish. Save the document using **File > Templates > Save**. To define this template as your default template, select **File > Templates > Organize** and double-click on **My Templates**. Right-click on the template you just created and choose **Set As Default Template**. If you wish to return to the default template given at installation, use the command **Reset Default Template**.

New text documents will be based on this template.

Application to chemical formulas

Math was designed to build mathematical formulas, but it can also be used to write chemical formulas. In chemistry, formulas look like H_2O: names are usually non-italic uppercase. To write chemical formulas with Math, you may begin with setting non italic for variables (see “Changing the font” on page 19).

Below are some examples of chemical formulas.

Construction	Example	Entry
Molecules	H_2SO_4	**H_2 SO_4** (please note the space!)
Isotopes	$^{238}_{92}U$	**U lsub 92 lsup 238**
Ions	SO_4^{2-}	**SO_4^{2-{}}** or **SO_4^{2"-"}**

Notes: **lsub** or **lsup** are abreviations for *left subscript* and *left superscript*. Empty braces after **2-** are necessary to avoid errors as there is no right member after the minus sign.

For reversible reactions, there is no satisfactory double arrow in *Math*. If you have a font with a correct symbol, you may use the method described in “Customizing the catalog” on page 30. For instance, DejaVu fonts have these double arrows ⇄ ⇆ ⇋ ⇌.

Otherwise, you may find a special character in a document and copy it, for instance in this formula: C+O⇌CO "⇌". You will find other double arrows here: http://dev.w3.org/html5/html-author/charref from character x021C4.

Export and import

Export as MathML

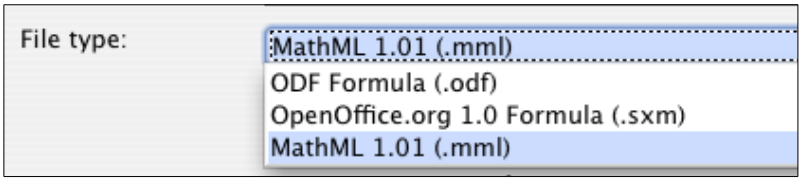
In addition to exporting documents as PDFs, as described in Chapter 10, Printing, Exporting, and Emailing, of the *Getting Started* guide, Math offers the possibility of exporting formulas as MathML. Unfortunately browsers do not follow this standard perfectly. If there are subsequent faults in display, the fault lies with your browser!

As an example, take this formula:

Markup	Result
`%zeta (z) = sum from {n = 1} to infinity {1 over {n^z}} newline left(matrix {a # b ## c # d}right) newline "Text" newline int_0^x f(t) dt newline x = 3 y = 1`	$\zeta(z)=\sum_{n=1}^{\infty}\frac{1}{n^z}$ $\begin{pmatrix} a & b \\ c & d \end{pmatrix}$ Text $\int_0^x f(t)dt$ $x=3\,y=1$

This formula combines various possibilities that Math offers, which makes it useful for testing.

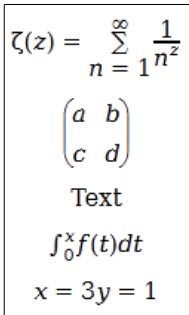
If you are working in a separate Math document, go to **File > Save as** to open the *Save as* dialog. Choose MathML from the list of available file formats, to save your formula as MathML.



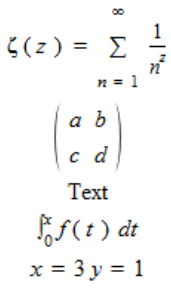
For an embedded formula, choose **Save copy as** from the context menu.

Result

This formula was subsequently tested in several browsers.



Firefox 14.0.1



Opera 12.00

```
<?xml version="1.0" encoding="UTF-8"?>

<math xmlns="http://www.w3.org/1998/Math/MathML">
 <semantics>
  <mtable>
   <mtr>
    <mtd>
     <mrow>
      <mo stretchy="false">Î¶</mo>
      <mrow>
```

Internet Explorer 9

This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
▼<math xmlns="http://www.w3.org/1998/Math/MathML">
  ▼<semantics>
    ▼<mtable>
      ▼<mtr>
        ▼<mtd>
          ▼<mrow>
             <mo stretchy="false">ζ</mo>
            ▼<mrow>
```

Chromium 13 and Google Chrome 20.0

Safari 5.1 only opens the file manager to save the data.

Microsoft file formats

The options in **Tools > Options > Load/Save > Microsoft Office** control how the import and export of Microsoft Office file formats is carried out in regard to formulas.

Loading

If [L] is checked, LibreOffice converts Microsoft formulas into native format when a document is loaded. This is possible if the formulas were created with MathType[4] (up to version 3.1) or with the Microsoft Equation Editor. The latter is a reduced, older version of MathType licensed by Microsoft and included in the Microsoft Office package.

Formulas created by newer versions of MathType or by the new Microsoft formula editor “OMML[5] Equation Editor” (Microsoft Office Suite 2010 and to some extent 2007) cannot be converted.

If a document created in Microsoft Office 2010 and containing an OMML formula is saved into a .doc file format, Microsoft Office converts the formula into a graphic. Only this graphic is then accessible to LibreOffice.

4 http://www.dessci.com/en/
5 Office Math Markup Language

If you load a .docx document that contains OMML formulas, these fail in conversion whether [L] is checked or not.

Inside Microsoft Office, formulas created with MathType or Microsoft Equation Editor are treated as OLE objects. If [L] is not checked, LibreOffice maintains this. Double-clicking on the object launches MathType and new formulas can be inserted using **Insert > Object > OLE Object**. This setup is to be recommended if you have MathType installed and wish to use it to create and edit formulas.

Saving

If [S] is checked, LibreOffice converts the formula into a form that can be read and modified by Microsoft Equation Editor and MathType. When [S] is not checked, the formula is treated as an OLE object on conversion into a .doc format, and remains linked to LibreOffice. A double-click on the object in Word will therefore attempt to launch LibreOffice.

If you save in the .docx format, formulas are not converted, whether [S] is checked or not.

Opening OpenDocument texts in Microsoft Word 2010

When you use Microsoft Office 2010, it is possible to open LibreOffice files in .odt format. Microsoft Office 2010 reports an error but nevertheless displays a “repaired” document. In this document, any original formulas are converted to OMML. If you save it again in .odt format, Microsoft Office 2010 converts the formulas into MathML, and these formulas can then be edited in LibreOffice. Take care that symbols that belong together are enclosed in grouping brackets, even if these are not necessary for display in LibreOffice itself. This enables LibreOffice to correctly translate the formula into MathML. For example, use **`sum from {i=1} to n {i^2}`** rather than **`sum from i=1 to n i^2`**.

In this procedure, the annotation element is lost. As a result, LibreOffice no longer shows the original Math markup code in the equation editor, but instead generates a new markup code from the internal MathML notation.

XHTML

The XHTML file format is available only as an export format. For this reason you will find the option under **File > Export** rather than **File > Save as**. If this option is missing in your setup, you can install the necessary filter (run LibreOffice Setup, select **Modify**, and then select **Optional Components > XSLT Sample Filters**). The replacement graphic for the formula is embedded in the file, but the original Math markup code, included as an annotation element in MathML, is absent in this case (compare with the section on “Export as MathML” on page 33).

Whether the document is correctly displayed depends on the browser. Firefox shows the document correctly including the formulas. Opera and Internet Explorer have some problems with the embedded graphic.

The older filter “HTML Document (OpenOffice.org Writer)(.html)“ does not save formulas at all but uses only a poorly resolved graphic in .gif file format.

Flat XML

This format can be both read and saved by LibreOffice. If the option is not present under **File > Save** or **File > Save as**, you can install the necessary filter. In this format, the whole content of the document, including the template, is saved within an XML document. In other words, it is not a compressed folder, as is usually the case for saved OpenDocument files. Any formulas are included as MathML elements, similar to the results of exporting as MathML. However individual formulas are not saved, only the whole document including the formulas. This format is suitable for direct entry into an external version control system.

Extensions

If you use *Math* frequently, you may wish to install one of the following extensions, which ease formula editing. The programs do not compete, but help one each other when it is possible. However, *Dmaths* and *CmathOOo* **cannot be simultaneously installed**. They are both complementary; you may install them on separate login to test them before choosing.

Dmaths – an extension for more than just faster formula input

Dmaths is a mathematical macro package for LibreOffice Writer, which can be installed as an extension. It can be downloaded from http://extensions.libreoffice.org/.

After installing Dmaths, close and reopen LibreOffice. Writer now shows a new menu (Figure 29), a new (customizable) toolbar with 36 buttons (Figure 30), a new toolbar with 26 buttons for use with Gdmath (Figure 31), a new toolbar with 10 buttons for use with AHmath3D (Figure 32), and two short toolbars to switch the longer ones on and off and to access a few other functions (Figure 33).

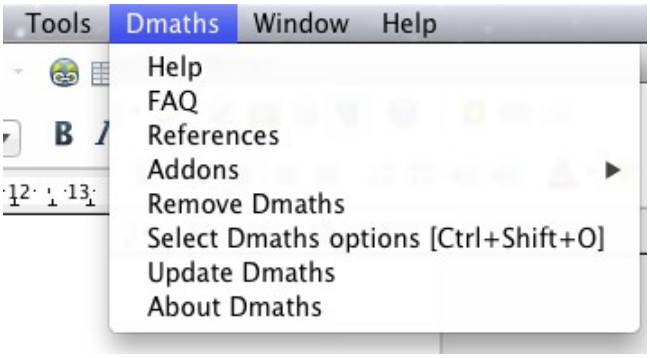


Figure 29: Dmaths menu

Figure 30: Customizable Dmaths toolbar

Figure 31: Gdmath toolbar

Figure 32: AHmath3D toolbar

Figure 33: Toolbars for switching on and off further functions

Dmaths offers numerous tools for creating mathematical documents:

- Single-click creation of Math formulas from text input (orange symbols)
- Creation of Math formulas using an input dialog (violet symbols)
- Tagging text (blue symbols)
- Changing the properties of existing formulas (green symbols)
- Drawing graphics, grids, statistical diagrams and geometrical figures (cyan symbols)
- Drawing and modification of geometric objects (gray Gdmath toolbar, also available in LibreOffice Draw)
- Drawing 3D objects (AHmath 3D toolbar with wire models)

The menu provides access to further add-ons, configuration, and a detailed *Dmaths Help Guide*.

Dmaths was initially created by the French mathematician, Didier Dorange, but is now available in German, English, and Spanish as well as French. The original website is www.dmaths.org.

Symbolic computation with CmathOOo and CmathOOoCAS

CmathOOo enables you to enter mathematical expressions into a word processing document using the same syntax as a high school calculator. *CmathOOoCAS* enables symbolic computation (as is done with Xcas[6]) directly from word processing; format and layout of the result is done with *CmathOOo*. These extensions can be downloaded from:
http://extensions.services.openoffice.org/fr/project/CmathOOo and
http://extensions.services.openoffice.org/fr/project/CmathOOoCAS or from Christophe Devalland's website http://cdeval.free.fr page http://cdeval.free.fr/spip.php?article85.

Technical details

OASIS Open Document Format for Office applications

The ODF specification does not lay down a completely new standard for Math formulas but takes over the Standard Mathematical Markup Language (MathML) developed by the World Wide Web Consortium (W3C) for representing formulas on websites (compare http://www.w3.org/TR/2003/REC-MathML2-20031021). Of the two types provided, Presentation markup and Content markup, ODF uses the former. It therefore describes the representation of a formula rather than its mathematical meaning.

The MathML markup language differs from the one used in the equation editor. So, for example, in the formula **`int from {i=1} to 3 {f(t) {nitalic d}t}`**, the parts **`from`**... **`to`**... are not translated but converted into MathML similarly to **`int_{i=1}^3`**. The original formula in the equation editor is a StarMath[7] 5.0 encoding. To prevent it from being lost, it is stored as an annotation element. Other applications do not need either to evaluate nor to save this element, nor do they themselves create it.

In LibreOffice a formula is always associated with a replacement graphic. In the container formats - .odf, .odt and so on – these graphics are stored in a separate folder and are internally linked to their document. However if you use the file format "Open Document (Flat XML)" or export to XHTML, these graphics must be embedded. For this purpose they are encoded in BASE64.

6 http://www-fourier.ujf-grenoble.fr/~parisse/giac.html
7 LibreOffice is based on the StarOffice suite, developed by StarDivision.

Handling of formulas in Basic

This section is not an introduction to Basic but simply describes some specific aspects of formulas. You can get more specific information about types and services at http://api.libreoffice.org/common/ref/com/sun/star/module-ix.html. If you look up the notations shown here in the index, you will quickly get to the relevant descriptions.

Formulas in a Writer document

In a Writer document, all OLE objects, including formulas, belong to a collection of type *SwXTextEmbeddedObjects*:

```
oCurrentController = ThisComponent.getCurrentController()
oTextDocument = oCurrentController.Model
oEmbeddedObjects = oTextDocument.EmbeddedObjects
```

This collection is numbered continuously from 0. You can find a given object by:

```
nEndIndex = oEmbeddedObjects.Count-1
for nIndex=0 to nEndIndex
  oMathObject = oEmbeddedObjects.getByIndex(nIndex)
```

To determine if this object is actually a formula, test whether it supports a service that is supported by formulas:

```
oModel = oMathObject.Model
if oModel.supportsService("com.sun.star.formula.FormulaProperties") then
```

Alternatively test if the CLSID, which every OLE object possesses, is that belonging to a Math object:

```
if oMathObject.CLSID = "078B7ABA-54FC-457F-8551-6147e776a997" then
```

Using the model, you can then alter the properties of the formula, for example the basic font size:

```
oModel.BaseFontHeight = 12
```

To make this change visible, the formula must be redrawn. The **`ExtendedControlOverEmbeddedObject`** property provides some methods that specifically apply to OLE objects:

```
oXCOEO = oMathObject.ExtendedControlOverEmbeddedObject
oXCOEO.update()
```

As an illustration, here is a macro modifying all formatting of all formulas already written in a document:

```
Sub ChangeFormatFormule
   oCurrentController = ThisComponent.getCurrentController()
   oTextDocument = oCurrentController.Model
   oEmbeddedObjects = oTextDocument.EmbeddedObjects
   nEndIndex = oEmbeddedObjects.Count-1
   for nIndex=0 to nEndIndex
      oMathObject = oEmbeddedObjects.getByIndex(nIndex)
      oModel = oMathObject.Model
      if (not isNull(oModel)) then
         if(not isEmpty(oModel)) then
            if oModel.supportsService("com.sun.star.formula.FormulaProperties") then
            ' or if oMathObject.CLSID =
            ' "078B7ABA-54FC-457F-8551-6147e776a997" then
               oModel.BaseFontHeight = 11
               policeCommune= "Liberation Serif"
            ' Variables
               oModel.FontNameVariables= policeCommune
```

```
                oModel.FontVariablesIsItalic=true
                oModel.FontVariablesIsBold=false
            ' Functions
                oModel.FontNameFunctions = policeCommune
                oModel.FontFunctionsIsItalic=false
                oModel.FontFunctionsIsBold=false
            ' Numbers
                oModel.FontNameNumbers= policeCommune
                oModel.FontNumbersIsItalic=false
                oModel.FontNumbersIsBold=false
            ' Text
                oModel.FontNameText= policeCommune
                oModel.FontTextIsItalic=false
                oModel.FontTextIsBold=false
            ' Update
                oXCOEO = oMathObject.ExtendedControlOverEmbeddedObject
                oXCOEO.update()
            endif ' if formula
        endif ' if not empty
    endif ' if not null
  next nIndex
  ThisComponent.reformat() ' Met à jour tous les éléments du document
End Sub
```

Check help page[8] on *FormulaProperties* to know all other properties you can modify.

Formulas in a Draw/Impress/Calc document

In a Draw, Impress or Calc document, OLE objects are similarly treated as graphics. After you have accessed a particular object, test if it is an OLE object, and then if it is a formula. The core of your macro is then something like:

```
if oShape.supportsService("com.sun.star.drawing.OLE2Shape") then
   if oShape.CLSID = "078B7ABA-54FC-457F-8551-6147e776a997" then
      oModelFormula = oShape.Model
      oModelFormula.BaseFontHeight = 12
```

In this case an explicit update is not necessary.

8 http://api.libreoffice.org/common/ref/com/sun/star/formula/FormulaProperties.html

Math commands - Reference

Unary / binary operators

Operation	*Command*	*Display*
+sign	`+1`	+1
–sign	`-1`	−1
+/– sign	`+-1` *or* `plusminus 1`	±1 *or* ±1
–/+ sign	`-+1` *or* `minusplus 1`	∓1 *or* ∓1
Addition +	`a + b`	*a*+*b*
Subtraction (–)	`a - b`	*a*−*b*
Dot product	`a cdot b`	*a*·*b*
Multiplication (X)	`a times b`	*a*×*b*
Multiplication (asterisk)	`a * b`	*a*∗*b*
Division (as a fraction)	`a over b`	$\frac{a}{b}$
Division (as an operator)	`a div b`	*a*÷*b*
Division (with a slash)	`a / b`	*a*/*b*
Concatenation	`a circ b`	*a*∘*b*
Division (with a wide slash)	`a wideslash b`	$a \mathbin{/} b$
Wide backslash	`a widebslash b`	$a \mathbin{\backslash} b$
Boolean not	`neg a`	¬*a*
Boolean and	`a and b` *or* `a & b`	*a*∧*b* *or* *a*∧*b*
Boolean or	`a or b` *or* `a | b`	*a*∨*b* *or* *a*∨*b*
Backslash	`a bslash b`	*a**b*
Direct sum	`a oplus b`	*a*⊕*b*
	`a ominus b`	*a*⊖*b*
Tensorial product	`a otimes b`	*a*⊗*b*
	`a odot b`	*a*⊙*b*
	`a odivide b`	*a*⊘*b*
Customizable unary operator	`uoper monOp b`	*monOp b*
Customizable binary operator	`a boper monOp b`	*a monOp b*

Relations

Operation	*Command*	*Display*
Is equal	`a = b`	$a=b$
Is not equal	`a <> b` *or* `a neq b`	$a \neq b$ *or* $a \neq b$
Less than	`a < b` *or* `a lt b`	$a<b$ *or* $a<b$
Less than or equal to	`a <= b` `a leslant b`	$a \leq b$ $a \leqslant b$
Very small	`a ll b` *or* `a << b`	$a \ll b$ *or* $a \ll b$
Greater than	`a > b` *or* `a gt b`	$a>b$ *or* $a>b$
Greater than or equal to	`a >= b` `a geslant b`	$a \geq b$ $a \geqslant b$
Very big	`a gg b` *or* `a >> b`	$a \gg b$ *or* $a \gg b$
Approximately	`a approx b`	$a \approx b$
Similar to	`a sim b`	$a \sim b$
Similar to or equal	`a simeq b`	$a \simeq b$
Congruent	`a equiv b`	$a \equiv b$
Proportional	`a prop b`	$a \propto b$
Parallel	`a parallel b`	$a \parallel b$
Orthogonal to	`a ortho b`	$a \perp b$
Divides	`a divides b`	$a \mid b$
Does not divide	`a ndivides b`	$a \nmid b$
Toward	`a toward b`	$a \rightarrow b$
Arrow left	`a dlarrow b`	$a \Leftarrow b$
Double arrow left and right	`a dlrarrow b`	$a \Leftrightarrow b$
Arrow right	`a drarrow b`	$a \Rightarrow b$
Precedes	`a prec b`	$a \prec b$
Succeeds	`a succ b`	$a \succ b$
Precedes or equal to	`a preccurlyeq b`	$a \preccurlyeq b$
Succeeds or equal to	`a succcurlyeq b`	$a \succcurlyeq b$
Precedes or equal to	`a precsim b`	$a \precsim b$
Succeeds or equal to	`a succsim b`	$a \succsim b$
Does not precede	`a nprec b`	$a \not\prec b$
Does not succeed	`a nsucc b`	$a \not\succ b$
Definition	`a def b`	$a \stackrel{\text{def}}{=} b$
Image from	`a transl b`	$a \multimap b$
Origin from	`a transr b`	$a \multimap b$

Set operations

Operation	*Command*	*Display*
Is in	`a in B`	$a \in B$
Is not in	`a notin B`	$a \notin B$
Owns	`A owns b` *or* `A ni b`	$A \ni b$ or $A \ni b$
Intersection	`A intersection B`	$A \cap B$
Union	`A union B`	$A \cup B$
Difference	`A setminus B`	$A \setminus B$
Quotient	`A slash B`	A / B
Subset	`A subset B`	$A \subset B$
Subset or equal to	`A subseteq B`	$A \subseteq B$
Superset	`A supset B`	$A \supset B$
Superset or equal to	`A supseteq B`	$A \supseteq B$
Not subset	`A nsubset B`	$A \not\subset B$
Not subset or equal	`A nsubseteq B`	$A \not\subseteq B$
Not superset	`A nsupset B`	$A \not\supset B$
Not superset or equal	`A nsupseteq B`	$A \not\supseteq B$
Empty set	`emptyset`	$\emptyset$
Aleph	`aleph`	$\aleph$
Set of natural numbers	`setN`	$\mathbb{N}$
Set of integers	`setZ`	$\mathbb{Z}$
Set of rational numbers	`setQ`	$\mathbb{Q}$
Set of real numbers	`setR`	$\mathbb{R}$
Set of complex numbers	`setC`	$\mathbb{C}$

Functions

Operation	*Command*	*Display*
Absolute value	`abs{a}`	$\|a\|$
Factorial	`fact{a}`	$a!$
Square root	`sqrt{a}`	$\sqrt{a}$
n^{th} root	`nroot{n}{a}`	$\sqrt[n]{a}$
Power	`a^{b}`	a^b
Exponential	`func e^{a}`	e^a
Natural logarithm	`ln(a)`	$\ln(a)$
Exponential function	`exp(a)`	$\exp(a)$
Logarithm	`log(a)`	$\log(a)$
Sine	`sin(a)`	$\sin(a)$
Cosine	`cos(a)`	$\cos(a)$
Tangent	`tan(a)`	$\tan(a)$
Cotangent	`cot(a)`	$\cot(a)$
Hyperbolic sine	`sinh(a)`	$\sinh(a)$
Hyperbolic cosine	`cosh(a)`	$\cosh(a)$
Hyperbolic tangent	`tanh(a)`	$\tanh(a)$
Hyperbolic otangent	`coth(a)`	$\coth(a)$
Arcsine	`arcsin(a)`	$\arcsin(a)$
Arccosine	`arccos(a)`	$\arccos(a)$
Arctangent	`arctan(a)`	$\arctan(a)$
Arccotangent	`arccot(a)`	$\operatorname{arccot}(a)$
Area hyperbolic sine	`arsinh(a)`	$\operatorname{arsinh}(a)$
Area hyperbolic cosine	`arcosh(a)`	$\operatorname{arcosh}(a)$
Area hyperbolic tangent	`artanh(a)`	$\operatorname{artanh}(a)$
Area hyperbolic cotangent	`arcoth(a)`	$\operatorname{arcoth}(a)$

Operators

All operators can be used with the limit functions (“from” and “to”).

Operation	*Command*	*Display*
Limit	`lim{a}`	$\lim a$
Lower limit	`liminf{a}`	$\liminf a$
Upper limit	`limsup{a}`	$\limsup a$
Sum	`sum{a}`	$\sum a$
Product	`prod{a}`	$\prod a$
Coproduc	`coprod{a}`	$\coprod a$
Integral	`int{a}`	$\int a$
Double integral	`iint{a}`	$\iint a$
Triple integral	`iiint{a}`	$\iiint a$
Contour integral	`lint a`	$\oint a$
Double curved integral	`llint a`	$\oint\oint a$
Triple curved integral	`lllint a`	$\oint\oint\oint a$
Lower bound shown with summation symbol	`sum from {3} b`	$\sum_{3} b$
Upper bound shown with product symbol	`prod to {3} r`	$\prod^{3} r$
Upper and lower bounds shown with integral	`int from {r_0} to {r_t} a`	$\int_{r_0}^{r_t} a$
Customized operator	`oper Op from 0 to 1 a`	$\mathit{Op}_{0}^{1} a$

Attributes

Operation	*Command*	*Display*
Acute accent	`acute a`	á
Grave accent	`grave a`	à
Reverse circumflex	`check a`	ǎ
Breve	`breve a`	ă
Circle	`circle a`	å
Dot	`dot a`	ȧ
Double dot	`ddot a`	ä
Triple dot	`dddot a`	a⃛
Line above	`bar a`	ā
Vector arrow	`vec a`	$\vec{a}$
Tilde	`tilde a`	ã
Circumflex	`hat a`	â
Wide vector arrow	`widevec abc`	$\overrightarrow{abc}$
Wide tilde	`widetilde abc`	$\widetilde{abc}$
Wide circumflex	`widehat abc`	$\widehat{abc}$
Line over	`overline abc`	$\overline{abc}$
Line under	`underline abc`	$\underline{abc}$
Line through	`overstrike abc`	~~acb~~
Transparent (useful to get a placeholder of a given size)	`phantom a`	
Bold font	`bold a`	***a***
Not bold font	`nbold a`	*a*
Italic font[9]	`ital "a"` *or* `italic "a"`	*a* *or* *a*
Not italic font	`nitalic a`	a
Font size	`size 16 qv`	*qv*
Font size	`size +12 qv`	*qv*
Font size	`size *1.5 qv`	*qv*
Following item in sans serif font[10]	`font sans qv`	*qv*
Following item in serif font	`font serif qv`	*qv*

9 Unquoted text that is not a command is considered to be a variable. Variables are, by default, italicized.

10 There are three custom fonts: sans serif (without kicks), serifs (with kicks), and fixed (non-proportional). To change the actual fonts used for custom fonts and the fonts used for variables (unquoted text), numbers and functions, use **Format >** Fonts (see page 19).

Operation	Command	Display
Following item in fixed font	`font fixed qv`	*qv*
Make color of following text cyan[11]	`color cyan qv`	*qv*
Make color of following text yellow	`color yellow qv`	*qv*
Make color of following text white	`color white qv`	*qv*
Make color of following text green	`color green qv`	*qv*
Make color of following text blue	`color blue qv`	*qv*
Make color of following text red	`color red qv`	*qv*
Make color of following text black	`color black qv`	*qv*
Make color of following text magenta	`color magenta qv`	*qv*
Make color green returns to default color black	`color green X qv`	*X qv*
Brace items to change color of more than one item	`color green {X qv}`	*X qv*

Brackets

Operation	Command	Display
Group brackets (used for program control)	`{a}`	a
Round Brackets	`(a)`	(a)
Square Brackets	`[b]`	$[b]$
Double Square Brackets	`ldbracket c rdbracket`	$[\![c]\!]$
Braces	`lbrace w rbrace`	$\{w\}$
Angle Brackets	`langle d rangle`	$\langle d \rangle$
Operator Brackets	`langle a mline b rangle`	$\langle a \mid b \rangle$
Upper half square brackets	`lceil a rceil`	$\lceil a \rceil$
Lower half square brackets	`lfloor a rfloor`	$\lfloor a \rfloor$
Single line	`lline a rline`	$\lvert a \rvert$
Double line	`ldline a rdline`	$\lVert a \rVert$
Scalable round brackets (add the word “left” before a left bracket and “right” before a right bracket)	`left ( stack{a # b # z} right )`	$\left(\begin{matrix} a \\ b \\ z \end{matrix} \right)$
Square brackets scalable (as above)	`left [ a over b right ]`	$\left[\frac{a}{b} \right]$

11 For all coloring, the color will apply only to the text immediately following the command until the next space is encountered. In order to have the color apply to more characters, place the text you want in color in curly brackets.

Operation	*Command*	*Display*
Double square brackets scalable	`left ldbracket a over b right rdbracket`	$\left[\!\left[\frac{a}{b} \right]\!\right]$
Brace scalable	`left lbrace a over b right rbrace`	$\left\{ \frac{a}{b} \right\}$
Angle bracket scalable	`left langle a over b right rangle`	$\left\langle \frac{a}{b} \right\rangle$
Operator brackets scalable	`left langle a over b mline c right rangle`	$\left\langle \frac{a}{b} \middle\vert c \right\rangle$
Upper half square brackets scalable	`left lceil a over b right rceil`	$\left\lceil \frac{a}{b} \right\rceil$
Lower half square brackets scalable	`left lfloor a over b right rfloor`	$\left\lfloor \frac{a}{b} \right\rfloor$
Line scalable	`left lline a over b right rline`	$\left\vert \frac{a}{b} \right\vert$
Double line scalable	`left ldline a over b right rdline`	$\left\Vert \frac{a}{b} \right\Vert$
Unpaired brackets (add `left` before left bracket and `right` before right bracket)	`left langle a over b right rfloor`	$\left\langle \frac{a}{b} \right\rfloor$
Isolated bracket	`left lbrace stack {a=2#b=3} right none`	$\left\{ \begin{matrix} a=2 \\ b=3 \end{matrix} \right.$
Over brace scalable	`{the brace is above} overbrace a`	$\overbrace{\textit{the brace is above}}^{a}$
Under brace scalable	`{the brace is below} underbrace {f}`	$\underbrace{\textit{the brace is below}}_{f}$

Formats

Operation	Command	Display
Right superscript	`a^{b}`	a^{b}
Right subscript	`a_{b}`	a_{b}
Left superscript	`a lsup{b}`	${}^{b}a$
Left subscript	`a lsub{b}`	${}_{b}a$
Center superscript	`a csup{b}`	$\overset{b}{a}$
Center subscript	`a csub{b}`	$\underset{b}{a}$
New line	`asldkfjo newline sadkfj`	*asldkfjo* *sadkfj*
Small gap (grave)	`stuff `stuff`	*stuff stuff*
Large gap (tilde)	`stuff~stuff`	*stuff stuff*
No gap	`nospace { x + y }`	*x+y*
Normal	`x+y`	*x+y*
Vertical stack of 2	`binom{a}{b}`	*a* *b*
Vertical stack, more than 2	`stack{a # b # z}`	*a* *b* *z*

Matrix	`matrix{`
	`a # b ##`
	`c # d`
	`}`
		a b
		c d

Align character to left (text is aligned center by default)	`stack{Hello world #` `alignl(a)}`	*Hello world* (*a*)
Align character to center	`stack{ Hello world #` `alignc(a)}`	*Hello world* (*a*)
Align character to right	`stack{ Hello world #` `alignr(a)}`	*Hello world* (*a*)

Equations aligned at '=' (using 'matrix')	`matrix{`
	`a # "=" # alignl{b} ##`
	`{} # "=" # alignl{c+1}`
	`}`
		a =*b*
		=*c*+1

Equations aligned at '=' (using 'phantom')	`stack{`
	`alignl{a} = b #`
	`alignl{phantom{a} = c+1}`
	`}`
		a =*b*
		=*c*+1

Others

Operation	*Command*	*Display*
Infinity	`infinity` *or* `infty`	∞ *or* ∞
Partial	`partial`	∂
Nabla	`nabla`	∇
There exists	`exists`	$\exists$
There does not exist	`notexists`	$\nexists$
For all	`forall`	$\forall$
H bar	`hbar`	$\hbar$
Lambda bar	`lambdabar`	λ
Real part	`re`	$\Re$
Imaginary part	`im`	$\Im$
Weierstrass p	`wp`	$\wp$
Reversed epsilon	backepsilon	$\ni$
Left arrow	`leftarrow`	$\leftarrow$
Right arrow	`rightarrow`	$\rightarrow$
Up arrow	`uparrow`	$\uparrow$
Down arrow	`downarrow`	$\downarrow$
Dots at bottom	`dotslow`	$\cdots$
Dots at middle	`dotsaxis`	$\cdots$
Dots vertical	`dotsvert`	$\vdots$
Dots diagonal upward	`dotsup` *or* `dotsdiag`	$\iddots$ *or* $\iddots$
Dots diagonal downward	`dotsdown`	$\ddots$

Characters – Greek

Caution

In localized versions of Writer, the markup names of Greek and special characters are localized. If this document is not localized to the same language, then the names below *may not work* for input. You may still use the Symbol catalog (Figure 8) to select the desired character by its glyph. This will also display the character's localized markup name.

Once entered, the characters will display properly in any language.

Contrary to commands, special characters are case sensitive.

Italic characters can be obtained by just adding an `i` after per cent character. For instance `%iPI` $\mathit{\Pi}$ instead of `%PI` Π .

%ALPHA	Α	%BETA	Β	%GAMMA	Γ	%DELTA	Δ	%EPSILON	Ε
%ZETA	Ζ	%ETA	Η	%THETA	Θ	%IOTA	Ι	%KAPPA	Κ
%LAMBDA	Λ	%MU	Μ	%NU	Ν	%XI	Ξ	%OMICRON	Ο
%PI	Π	%RHO	Ρ	%SIGMA	Σ	%TAU	Τ	%UPSILON	Υ
%PHI	Φ	%CHI	Χ	%PSI	Ψ	%OMEGA	Ω		
%alpha	α	%beta	β	%gamma	γ	%delta	δ	%epsilon	ϵ
%varepsilon	ε	%zeta	ζ	%eta	η	%theta	θ	%vartheta	ϑ
%iota	ι	%kappa	κ	%lambda	λ	%mu	μ	%nu	ν
%xi	ξ	%omicron	ο	%pi	π	%varpi	ϖ	%rho	ρ
%varrho	ϱ	%sigma	σ	%varsigma	ς	%tau	τ	%upsilon	υ
%phi	φ	%varphi	ϕ	%chi	χ	%psi	ψ	%omega	ω

Characters – Special

Some special characters correspond to same symbol as operator but there is no syntax verification (presence of left or right elements).

%perthousand	‰	%tendto	→	%element	∈
%noelement	∉	%infinite	∞	%angle	∢
%and	∧	%or	∨	%notequal	≠
%identical	≡	%strictlygreaterthan	≫	%strictlylessthan	≪

Reserved words in alphabetic order

A *reserved word* is an expression that is used in a particular way, which is controlled by LibreOffice.You cannot use it as a variable name. To be able to use these words without interference from Math, you need to surround them with double quotation marks (**"**).

` : small space. See ‘grave’ operator.

^ : superscript. See ‘hat’ and ‘widehat’ operators.

_ : subscript

- : subtraction

-+ : '-+' sign with '-' over '+'

. : point; need a character before

(: opening parenthesis

) : closing parenthesis

[: opening square bracket

] : closing square bracket

{ : opening bracket for grouping characters

} : closing bracket for grouping characters

* : product

/ : division on one line

\ : before a bracket (square bracket, parenthesis or brace), treat it as a normal character. See ‘bslash’ and ‘setminus’ operators.

& : Boolean operator 'and'

: separation between elements in a table

: separation between lines in a matrix

% : indicate special name (displayed if name not recognized)

%% : comment

+ : addition

+- : '+-' sign with '+' over '-'

< : operator less than

<?> : indicate placeholder around operators built with models. Displayed as a small square in the formula. *F4* and *Maj+F4* keys to navigate through them.

<< : operator much less than

<= : operator less or equal

<> : operator different

= : operator equality

> : operator greater than

>= : operator greater or equal

>> : operator much greater than

| : logical operator or

~ : large space. See operator ‘tilde’.

abs : function absolute value

acute : acute accent for one character

aleph : 1st letter of Hebrew alphabet (cardinal number).

alignb : (bottom vertical alignment) [12]

alignc : centered horizontal alignment

alignl : left alignment

alignm : (centered vertical alignment) [12]

alignr : right alignment

alignt : (top vertical alignment) [12]

and : logical operator and

approx : sign approximately, two ~ one over each

arccos : function arc cosine

arccot : function arc cotangent

arcosh : area hyperbolic cosine

arcoth : area hyperbolic cotangent

arcsin : function arc sine

arctan : function arc tangent

arsinh : function area hyperbolic sinus

artanh : function area hyperbolic tangent

backepsilon : upside down epsilon.

bar : add bar to next character.

binom : put 2 elements one over each

black : black color for 'color'

blue : blue color for 'color'

bold : bold font

boper : next character is treated as a binary operator (unchanged size)

breve : half circle turned up

bslash : operator \, as difference

cdot : operator '.' for multiplication

check : upside down circumflex accent

circ : 'round' operator for composition of functions

circle : add a circle over next character

color : set color: black, blue, cyan, green, magenta, red, white, yellow

coprod : coproduct (upside down Π) or direct sum

cos : function cosine

cosh : hyperbolic cosine

cot : cotangent

coth : hyperbolic cotangent

csub : centered subscript

csup : centered superscript

cyan : cyan color for 'color'

dddot : add three points over: triple derivative in physics

ddot : add two points over: double derivative in physics

def : equal sign with DEF overwritten

div : divide sign ':' with '-' in the middle

divides : operator | (Sheffer bar), same as logical sign ‘or’

dlarrow : left double arrow

dlrarrow : left and right double arrow (equivalent)

12 These obsolete shortcuts are for **align-bottom**, **align-mid** and **align-top** (vertical alignments). Math recognizes them, but nothing is changed.

`dot` : add a point over: derivative in physics
`dotsaxis` : align ... horizontally in the middle
`dotsdiag` : align three points with a +45° slop
`dotsdown` : align three points with a -45° slop
`dotslow` : align ... on the bottom
`dotsup` : align three points with a +45° slop
`dotsvert` : align vertically 3 points
`downarrow` : down arrow
`drarrow` : right double arrow (imply)
`emptyset` : empty set
`equiv` : equivalent (equal sign with three strikes)
`exists` : there exist (reversed E)
`exp` : exponential function
`fact` : factorial function (add '!' after)
`fixed` : font attribute
`font` : select a font
`forall` : whatever (upside down A)
`from` : goes with 'to' for limits of integrals, sums, etc.
`func` : transforms a variable in function
`ge` : greater or equal, horizontal ‘=’ sign
`geslant` : greater or equal, oblique ‘=’ sign
`gg` : much greater then '>>'
`grave` : add a grave accent
`green` : green color for 'color'
`gt` : operator plus grand que
`hat` : add a circumflex accent
`hbar` : strik h (ħ), reduced Planck’s constant (divided by π)
`iiint` : triple integral, three times integral sign
`iint` : double integral, twice integral sign
`im` : function imaginary part
`in` : is element of
`infinity` : infinity symbol
`infty` : infinity symbol
`int` : simple integral
`intersection` : intersection operator
`ital` : italicized font
`italic` : italicized font
`lambdabar` : strike lambda
`langle` : < to open '<...>' (angular bracket operator: langle ... mline ... rangle)
`lbrace` : visible left brace '{'
`lceil` : left square bracket without bottom
`ldbracket` : double left square bracket
`ldline` : left double line (norm) '||'
`le` : less or equal, horizontal ‘=’ sign
`left` : next character is treated as an opening bracket
`leftarrow` : left arrow
`leslant` : less or equal, oblique ‘=’ sign
`lfloor` : left square bracket without top
`lim` : limit operator
`liminf` : inferior limit operator
`limsup` : superior limit operator
`lint` : line integral (with a circle)
`ll` : much lower than operator
`lline` : left bar to open absolute value '|'
`llint` : double line integral (with a circle)
`lllint` : triple line integral (with a circle)
`ln` : function natural logarithm
`log` : function decimal logarithm
`lsub` : left subscript
`lsup` : left superscript
`lt` : operator less than '<'
`magenta` : magenta color for 'color'
`matrix` : defines a matrix
`minusplus` : '–+' sign, plus under '-'
`mline` : vertical line '|' (angular bracket operator: langle ... mline ... rangle)
`nabla` : nabla operator, upside down Δ.
`nbold` : not bold font.
`ndivides` : does not divide operator, / vertically stroked
`neg` : operator no
`neq` : operator different
`newline` : new line
`ni` : inverted sign is element of
`nitalic` : not italic font
`none` : combined with 'left' or 'right' to indicate invisible left or right bracket
`notin` : is not element of operator
`nroot` : n^{th} root
`nsubset` : is not strictly include in operator
`nsubseteq` : is not include or equal in operator
`nsupset` : reversed is not strictly include in operator
`nsupseteq` : reversed is not include or equal in operator
`odivide` : operator / in a circle
`odot` : operator '.' in a circle
`ominus` : operator '–' in a circle
`oper` : transforms next variable in a large operator with limits (like Σ)
`oplus` : direct sum operator, + in a circle
`or` : logical operator or, upside down ^
`ortho` : orthogonal operator, perpendicular symbol
`otimes` : tensor product operator, x in a circle
`over` : division operator, to write division with horizontal fraction bar
`overbrace` : put next element over previous one with an horizontal brace
`overline` : add an horizontal bar over next element
`overstrike` : strike-through characters
`owns` : reversed is element of
`parallel` : parallel operator '||'
`partial` : round d for partial derivative
`phantom` : invisible element, to let empty space

`plusminus` : operator '+-' with '+' over

`prod` : product operator, Π

`prop` : proportional operator

`rangle` : '>' to close '<...>' (angular bracket operator: langle ... mline ... rangle)

`rbrace` : visible right brace

`rceil` : right square bracket without bottom

`rdbracket` : double right square bracket

`rdline` : right double line (norm) '||'

`re` : real part function

`red` : red color for 'color'

`rfloor` : right square bracket without top

`right` : next character is treated as a closing bracket

`rightarrow` : right arrow

`rline` : right bar to close absolute value '|'

`rsub` : subscript

`rsup` : superscript

`sans` : font option

`serif` : font option

`setC` : set of complex numbers

`setminus` : operator \, subtraction of sets

`setN` : set of natural numbers

`setQ` : set of rational numbers

`setR` : set of real numbers

`setZ` : set of integers

`sim` : operator equivalent, write one ~

`simeq` : operator similarly equal, write a double ~

`sin` : function sine

`sinh` : function hyperbolic sine

`size` : change font size

`slash` : operator slash '/'

`sqrt` : operator square root

`stack` : define a stack of elements separated with '#'

`sub` : subscript

`subset` : strictly include operator

`subseteq` : include or equal operator

`sum` : sum operator, Σ

`sup` : superscript

`supset` : reversed strictly include operator

`supseteq` : reversed include or equal operator

`tan` : tangent function

`tanh` : hyperbolic tangent function

`tilde` : add a tilde '~' over next character

`times` : multiplication operator, 'X'

`to` : goes with 'from' for limits of integrals, sums, etc.

`toward` : arrow to the right

`transl` : 2 small joined circles, the one of the left is filled (sign of correspondence Image from)

`transr` : 2 small joined circles, the one of the right is filled (sign of correspondence Origin of)

`underbrace` : put next element under previous one with an horizontal brace

`underline` : add an horizontal bar under next element

`union` : union operator, U

`uoper` : next character is treated as a unary operator (unchanged size)

`uparrow` : up arrow

`vec` : add a small arrow over next character

`white` : white color for 'color'

`widebslash` : operator with a big '\'

`widehat` : add an hat over next element

`wideslash` : division operator with a big '/'

`widetilde` : add a tilde on next element

`widevec` : add an arrow over next element

`wp` : Weierstrass p function

`yellow` : yellow color for 'color'

Index

LibreOffice
The Document Foundation

Computer Software
LibreOffice

LibreOffice 3.5 Math Guide

What is LibreOffice?

LibreOffice is a freely-available, full-featured office suite. It runs on Windows, Linux, and Mac OS X computers. LibreOffice has been adopted by schools, governments, businesses, and individuals around the world.

What is LibreOffice Math?

Math is a simple equation editor that lets you lay out and display mathematical, chemical, electrical or scientific equations quickly in standard written notation. It is most commonly used in Writer for text documents, but it can also be used with other types of documents or stand-alone.

If you have never used LibreOffice before, or if you want an introduction to all of its components, you might like to read Getting Started with LibreOffice first.

What's in this book?

This book includes instructions for writing equations using Math and a reference list of the commands used in the program.

Who wrote this book?

This book was written by volunteers from the LibreOffice community. Profits from the sale of this book will be used to benefit the community. A PDF version of this book can be downloaded free from http://www.libreoffice.org/get-help/documentation/

Get LibreOffice

You can download LibreOffice free from http://www.libreoffice.org/