Full Circle

THE INDEPENDENT MAGAZINE FOR THE UBUNTU LINUX COMMUNITY

LIBREOFFICE SERIES SPECIAL EDITION

LIBREOFFICE SERIES SPECIAL EDITION VOLUME 3

LIBREOFFICE

Volume Three Parts 17-26

About Full Circle

Full Circle is a free, independent, magazine dedicated to the Ubuntu family of Linux operating systems. Each month, it contains helpful how-to articles and reader-submitted stories.

Full Circle also features a companion podcast, the Full Circle Podcast which covers the magazine, along with other news of interest.

Please note: this Special Edition is provided with absolutely no warranty whatsoever; neither the contributors nor Full Circle Magazine accept any responsibility or liability for loss or damage resulting from readers choosing to apply this content to theirs or others computers and equipment.

Full Circle
THE INDEPENDENT MAGAZINE FOR THE UBUNTU LINUX COMMUNITY

The LibreOffice series continues...

We continue our assembly of Elmer Perry's LibreOffice series in this, Volume 3.

Here is a straight reprint of the series **'Libre Office', Parts 17-26** from issues #64 through #73, spanning Base, Macros, Math, Formulas and many other advanced features along the way.

Please bear in mind the original publication date; current versions of hardware and software may differ from those illustrated, so check your hardware and software versions before attempting to emulate the tutorials in these special editions. You may have later versions of software installed or available in your distributions' repositories.

Enjoy!

Find Us

Website:
http://www.fullcirclemagazine.org/

Forums:
http://ubuntuforums.org/forumdisplay.php?f=270

IRC: #fullcirclemagazine on chat.freenode.net

Editorial Team

Editor: Ronnie Tucker
(aka: RonnieTucker)
ronnie@fullcirclemagazine.org

Webmaster: Rob Kerfia
(aka: admin / linuxgeekery-
admin@fullcirclemagazine.org

Editing & Proofreading
Mike Kennedy, David Haas,
Gord Campbell, Robert Orsino

Our thanks go to Canonical and the many translation teams around the world.

LibreOffice Pt17: Macros

I recently got a request for a tutorial on LibreOffice macros, so we will take a short break from our work in Impress to briefly cover macros. Macros allow you to automate repetitious actions like typing a letterhead. This frees you from having to type or do the same task over and over again. In this how-to, we will cover how to record macros and use them. LibreOffice has a Basic scripting language which is beyond the scope of this particular how-to. Perhaps we will revisit macros at a later time to discuss the Basic scripting language.

NOTE: You can find information on LibreOffice Basic in the help, or download the documentation at http://wiki.documentfoundation.org/images/d/dd/BasicGuide_OOo3.2.0.odt.

Enabling Macro Recording

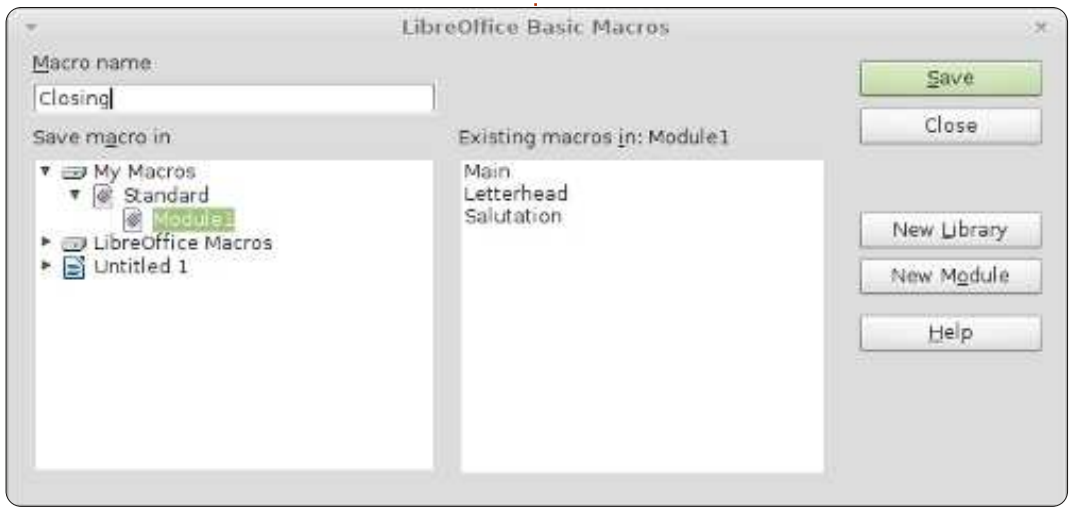
By default, macro recording is disabled. Apparently, macros are considered an "experimental (unstable) feature". To enable macro recording, Tools > Options. Select the General options under LibreOffice, and check "Enable experimental (unstable) features". This enables the "Record Macro" option under Tools > Macros.

Recording a Macro

When in macro record mode, the macro recorder tracks every action you make and all the text you type, and records it in the macro. Remember how we had to enable experimental features to get the Record Macro option? Well, that's because sometimes the macro recorder does fail. I haven't experienced this myself, but I thought I should point that out.

As a demonstration of recording a macro, we will create a macro called Closing. Every time you write a letter, you have to end with a closing, so why not make it into a macro.

Start with a new text document. Tools > Macros > Record Macro to start the macro recorder. The macro recorder toolbar will show. There is only one choice on this toolbar, Stop Recording. Press the Tab key on your keyboard three or four times (this should place the cursor close to the center of the page). Type Best wishes, love, or your favorite closing. Press Enter twice to leave room for a signature. Press Tab the same number of times you did before. Change the text to bold by clicking the Bold button on the formatting toolbar, and type in your name. Lastly, let's add a title under the name. Press Enter, Tab the same number of times as before, click the Bold button to turn off bold, and click the Italic button to change to italics. Type in your title. Press Enter. Click on Stop Recording.



The Basic Macros dialog box will show. Select the library where you want to save your macro, usually My Macros. Enter a name in the Macro name textbox, and click the Save button.

Testing Your Macro

You will want to test your macro to insure everything recorded correctly, Tools > Macros > Run Macro. The macro dialog box will show. Select the library where you saved your macro, select your

macro, and click on the Run button. The macro will run, repeating all the text you typed and the formatting you did.

If something didn't turn out just right, you can delete the macro and create a new one. To delete a macro, Tool > Macros > Organize Macros > LibreOffice Basic. Find your macro in the library, select it, and click the Delete button.

Create a Shortcut to your Macro

If you use a macro a lot, you don't want to go to Tools > Macros > Run Macro every time you need to use the macro. LibreOffice allows you to add your macros to menus, toolbars, keyboard shortcuts, and application events. You can add your macros through Tools > Customize.

As an example, let's add a menu named Macros in Writer and add our Closing macro to it. Tools > Customize. Select the Menus tab. Click the New button. Name the new menu Macros. Use the arrow buttons to move it from the bottom to the position before Help. Click OK. Your new menu is empty right now. Click on the Add button. Under category, find LibreOffice Macros and navigate through the tree to find your macro. Select the Closing macro and click the add button. The macro is added to the menu. Click on the Close button. Click OK on the Customize dialog box. You will now have a menu item named Macros, and, under it, the Closing macro. Now, you can select it from the menu when you need it, which is faster than having to navigate to Run Macro.

This has been a very short introduction to macros. Before deciding to use a macro, make sure there isn't a better way to accomplish what you are trying to do, but for often repeated action, macros might just be the solution you are looking for. There is a LibreOffice Basic scripting language, and perhaps we will cover it in the future. You can also download macros from the web that you can import and use in LibreOffice.

Next time, we are back to Impress and working with slides.

Elmer Perry's history of working, and programming, computers involves an Apple IIE, adding some Amiga, a generous helping of DOS and Windows, a dash of Unix, and blend well with Linux and Ubuntu.

When building a presentation, it is important to present the information in a pleasing and informative way. Using slide transitions provides a visual move from one topic to the next, and using animations helps to inform viewers or provide emphasis on the current point. Overuse of transitions and animations can cause your presentation to look less than professional. However, the appropriate use of these features will give your presentation a polished and professional appearance.

Slide Transitions

Transitions are the visual changes made when moving from one slide to the next. Transitions provide a visual clue to the audience that you are moving to a new topic. In general, you will use the same transition for all the slides, but, in some cases, you will want to use a different transition to show the viewers a change of topic.

With the slide you want to change displayed in the main view, select Slide Transition from the Tasks pane. The selection list provides you with a collection of different slide transitions. If you have Automatic preview checked at the bottom of the Slide Transition pane, you will see a preview of the transition when you select it or change its settings.

You can further modify the transition in the Modify transition section of the pane. Speed will change the rate at which the slide is displayed. Sound lets you play a sound with the transition. You can select a sound from the defaults provided, or select your own. Once you select a sound, you can select Loop until next sound. You will rarely have a use for this, but it is there should you need it.

In the Advance slide section, you set how and when you want the slide to advance. On click means the slide will display until you click the mouse or press the space-bar. Automatic after allows you to automatically advance the slide after a set number of seconds. When selected, you can adjust the number of seconds in the spinner box.

At the bottom of the pane, you have three buttons. Apply to All Slides does what it says; it applies the transition to all the slides in the presentation. Play causes the transition to run in the main view. Slide Show starts the presentation beginning with the current slide.

Animations

Animations are similar to transitions, but instead of acting on the slide, it acts on individual objects in the slide. Animations help create emphasis, flow, and visual interest as you present the objects on a slide. They keep the audience aware of the current subject, and act as a visual clue to the presenter.

To create animations, first select the slide which you want to create animations for. Select the object(s) you want to animate, and open the Custom Animations pane

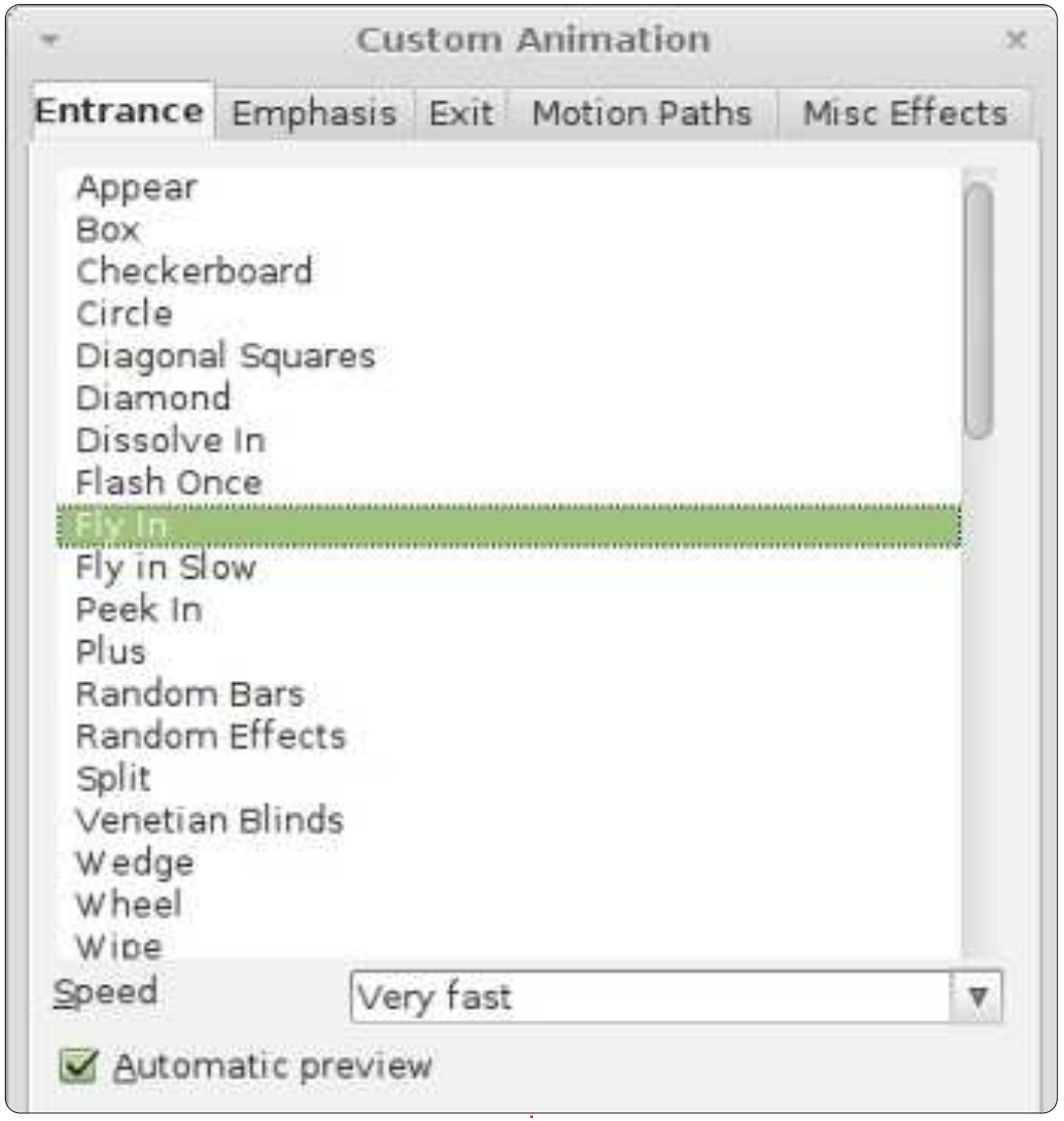
in the Tasks pane. Click on Add, which opens the animations dialog. Here you can select the animation you want for the object(s) selected.

Impress provides four different animation types:
Entrance: These animations play as the object appears on the page.
Emphasis: These animations are used to create emphasis such as changing colors, blinking, etc.
Exit: These animations play as the object leaves the page.
Motion Path: These animations cause the object to follow a defined path.

There is a fifth tab in the animations dialog related to media objects. They allow you to start, stop, and pause media objects.

Once you have selected your animation, click OK.

The Effect section of the animation pane give you the ability to fine-tune your animation. Start controls what event will trigger your animation. On Click will trigger the animation when the mouse button is clicked or you press the space-bar. With Previous triggers the animation when the animation before it plays. After Previous plays the animation after the previous animation. The second control relates directly to the type of animation you select. If it is a motion animation, it asks you for a direction. If the animation changes colors, it will asks you for a color. Finally, the Speed controls the speed at which the animation plays.



Animation Example

The real power of animations comes when you combine them to create interesting effects for your objects. In our example, the effect we will create will display items in a list one at a time. As the next items displays, the previous one will gray out. Finally, all the list items will fade out before the slide transition.

Create a new slide, and, in the text area, add four list items. Select all four list items, and click Add in the Custom Animation pane. On the Entrance tab, select Fly In and click OK. Select each of the animations in the animations pane and set the start to on click, direction to from bottom, and the speed to a speed that looks good on your machine.

For the color change effect, select the first three items in the text area of the slide, and click the Add button. On the Emphasis tab, select Change Font Color, and click OK. For each of these three new animations, change the start to with previous, the color to gray, and the speed to a speed that looks good on your machine. Move the color change animation for the first item up using the Change order arrows. Move it up under the entrance animation for the second item. Move the second change color animation up under the third entrance animation, and leave the third color change under the fourth entrance animation.

Finally, we will create the fade for all the items. Select all four list items in the slide's text area. Click on Add in the animation pane. On the Exit tab, select Dissolve and click OK. Set the first exit to start on click and the other three to after previous. Select a speed for the dissolve that works for your machine.

Test your animations by clicking on Slide Show in the animation pane. If you set everything correctly, each item should fly in from the bottom and gray out when you click the mouse. At the end, all four items should dissolve.

Transitions and animation are key to creating a professional looking presentation. If you are careful to not get carried away, you can create a polished and memorable presentation for your audience. Remember that the idea behind a presentation is to present your ideas to your audience, not to impress them with fancy, overdone transitions and animations.

Elmer Perry's history of working, and programming, computers involves an Apple IIE, adding some Amiga, a generous helping of DOS and Windows, a dash of Unix, and blend well with Linux and Ubuntu.

ubuntu uk podcast

The Ubuntu Podcast covers all the latest news and issues facing Ubuntu Linux users and Free Software fans in general. The show appeals to the newest user and the oldest coder. Our discussions cover the development of Ubuntu but aren't overly technical. We are lucky enough to have some great guests on the show, telling us first hand about the latest exciting developments they are working on, in a way that we can all understand! We also talk about the Ubuntu community and what it gets up to.

The show is presented by members of the UK's Ubuntu Linux community. Because it is covered by the Ubuntu Code of Conduct it is suitable for all.

The show is broadcast live every fortnight on a Tuesday evening (British time) and is available for download the following day.

podcast.ubuntu-uk.org

LibreOffice Pt19: Base

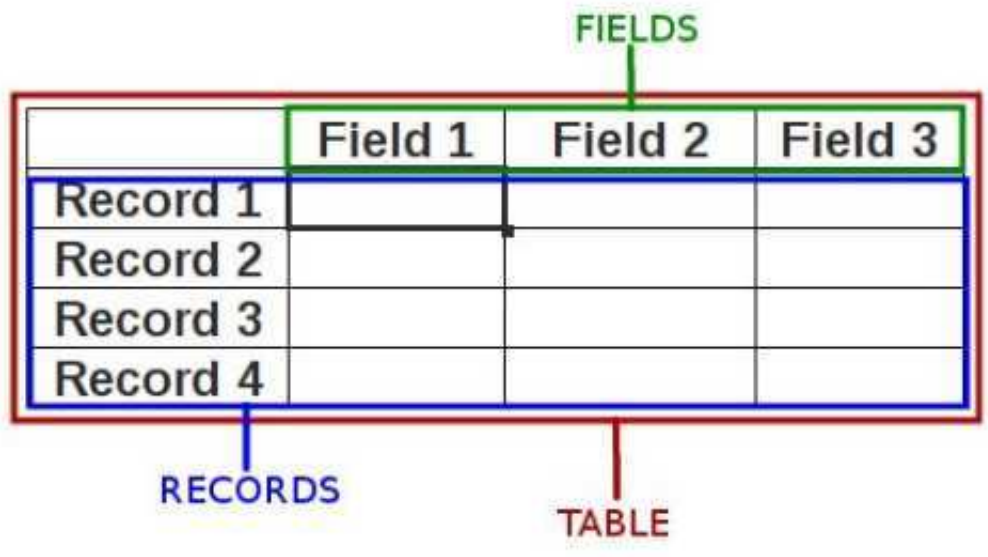
Many people collect things like sports cards, books, comic books, or butterflies. Sometimes, it is helpful to catalog these collections, so you create a catalog. You decide which characteristic about the items you want to track, you determine that some items share some of these characteristics in common, and you create a system for identifying each item uniquely. Finally, you begin to build your catalog. You can think of a database as a catalog of similar items. You have something you want to track, and what better way to track it than through your computer.

Base is the database module for LibreOffice. Base is not a database engine, but a front end for interacting with databases. By default, Base uses the HSQL database engine, which is an open source engine, but you can connect to other engines like MySQL or Oracle. You can even use a spreadsheet as the basis for a database, as we did in part 7 of this series (see Full Circle issue 52).

When creating a database, you get better results by sitting down and taking some time to plan out how your database will look and behave. You map out the characteristics you want to track, determine the common relationships, and create a unique way to identify each of the items in the collection. Taking the time to plan will save you time and effort later in the process when you begin to build reports and searches for your database.

What Makes a Database

Before we get into the actual planning of a database, let's talk about the parts making up a database. The smallest element of a database is the field. Think of a field as a single characteristic of the object we are defining in the database. A collection of fields is a record. A record defines all the characteristics of a single object we are collecting. We create tables to hold records. Tables define the fields for each record and contains the datum for each field in the records.



Think of a database table as a spreadsheet in Calc. Across the top, you have columns. The columns are the fields. Down the side, you have rows. The rows are records. The entire spreadsheet, containing all the data, is the table.

Planning Our Example Database

Through this series of articles on Base, we will use a database I created for tracking my book collection. I kept the database fairly simple, but including many elements to show the nature and aspects of relational databases, mostly the relational part. In this part, we will track the steps I took for planning the creation of the database. We will use the steps I have mentioned.

What Characteristics to Include

When I began planning my Books database, I knew I didn't want a big complicated thing with information I would never use. I

knew I needed the basics, title and author. However, I decided on including the year of publication, too. With all the different ways to "read" books today, I decided I needed to track the different types of media as well. So, in the end, I decided on these characteristics:

- Title
- Author
- Publication year
- Media type
- Relationships

Relationships put the "relational" in relational databases. When we first look at our characteristics list, we might think we just need a table with four fields. However, we would run into trouble when we have a book that has more than one author, or we own a book in more than one format. We could just stuff multiple authors in one field, but that would make searching for books by a single author difficult. We could create multiple fields for multiple authors, but how many do you create? If it is an anthology, the book could have many authors. The same is true for the media types.

The answer is relationships. Relationships help us link data in different tables to each other. There are three different relationships defined for relational databases.

One to One – For every one characteristic, you have one matching characteristic. As an example, for every one person, you can have one spouse.

One to Many – For every one characteristic, you have many matching characteristics. In our case, for every book, you can have multiple authors.

Many to Many – For many characteristics, you have many matching characteristics. As an example, in a school, you have many students who have many different teachers.

For our database, we have two one-to-many relationships. For every one book, we can have multiple authors, and for every one book, we can have multiple media types. We will take these relationships into consideration as we begin to map our database.

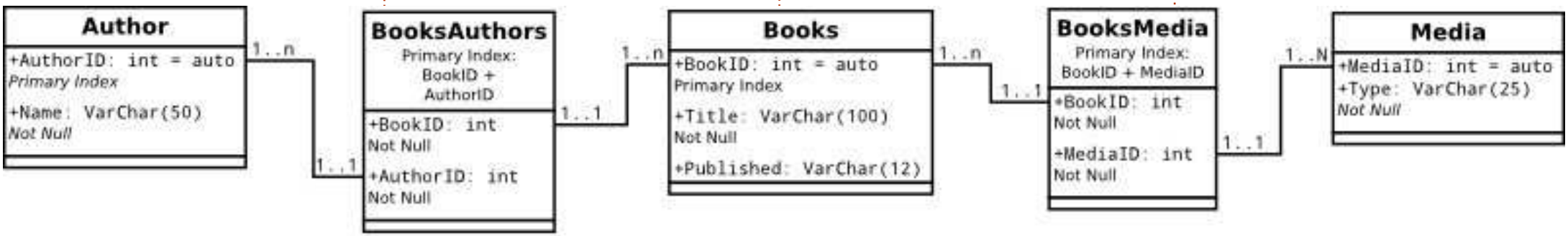
Mapping the Database

It is a good idea to lay out your database on paper, or using a diagram program, before you begin to work in Base. I used the open source program Dia, because it has a Unified Modeling Language (UML) module designed just for programming diagrams. Dia is available in the Ubuntu repositories. You don't need to know a lot about UML to lay out a database. I will walk you through the process in this section.

The main table for our database is the Books table. We know we need the fields Title and Published, but we also need a unique field to identify each record. Since two books could potentially have the same title, we will create an auto generated field named BookID.

For the Authors table, we need a field for the author's name (Name) and a unique auto generated field (AuthorID). Two fields for the Media table, too: MediaID and Type.

Now that we have our three



Relational database structure for Books project

tables, we need to link them together. Linking is done by what is known as foreign keys. A foreign key is a field used to create a relationship with a record in another table. Since both of our relationships are one-to-many, we can't just stick a field in the Books table to reference authors and media types. We will use intermediate tables to link the authors and media types together. These intermediate tables will contain foreign keys for the IDs to create the link.

We will need two intermediate tables. We will name them BooksAuthors and BooksMedia. BooksAuthors will have two fields named the BookID and the AuthorID, which link back the the ID fields in the Books and Authors tables. We do the same with the BooksMedia table. Two fields named BookID and MediaID, linking to the IDs in Books and Media.

I created a UML diagram showing the relationships between our five tables. Each box contains a table. The name of the table appears at the top of the box. Underneath, we list all the fields in the table and their types. We will discuss types in the next part of this tutorial. The lines between the boxes show the relationships from one table to the next. The notation 1..1 shows that field has a one-to-one relationship with the field in the other table. The notation 1..n shows that field has a one-to-many relationship with the field in the other table. For example, BookID in the Books table is connected to the BookID in the BooksAuthors tables. On the Books table BookID, the notation is 1..n, meaning this book can reference more than one record in the BooksAuthors table. On the booksAuthors, BookID has a notation of 1..1, meaning this is a reference to one specific record in Books.

With all this planning, we can easily create our database without having to make many changes. We know what tables we need and how they will relate to each other. While this may seem like a lot of work, it saves us a lot of time in the end, because we have actually put thought into how we will construct our database and how it will work.

Next time, we will build our tables and create the relationships in LibreOffice Base. Because of our planning, the process is quick and easy.

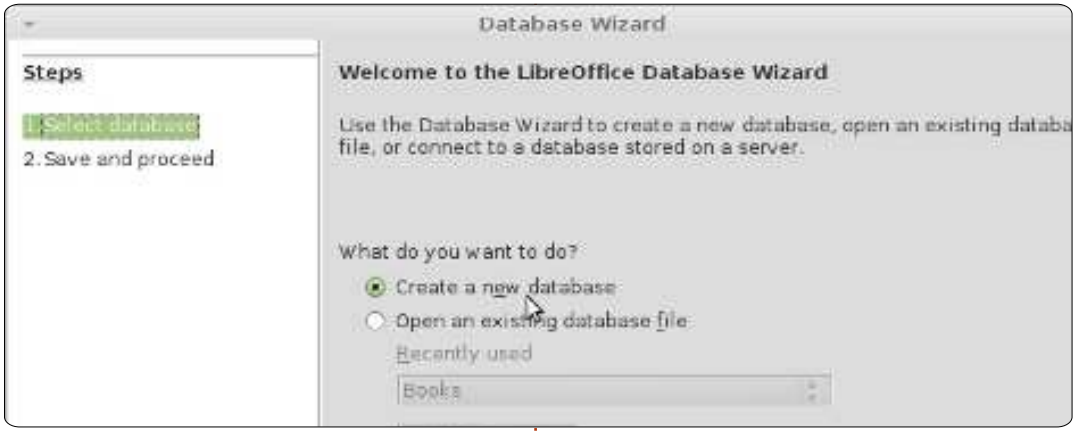
Elmer Perry's history of working, and programming, computers involves an Apple IIE, adding some Amiga, a generous helping of DOS and Windows, a dash of Unix, and blend well with Linux and Ubuntu.

LibreOffice Pt20: Base

Databases are used to store information about objects or data. In the previous tutorial, we mapped out how our books database would look. We designed tables for our data, and defined relationships between those tables. Now, we will put our planning into action by actually creating the database file, adding the tables, and creating the relationships.

Creating the Database File

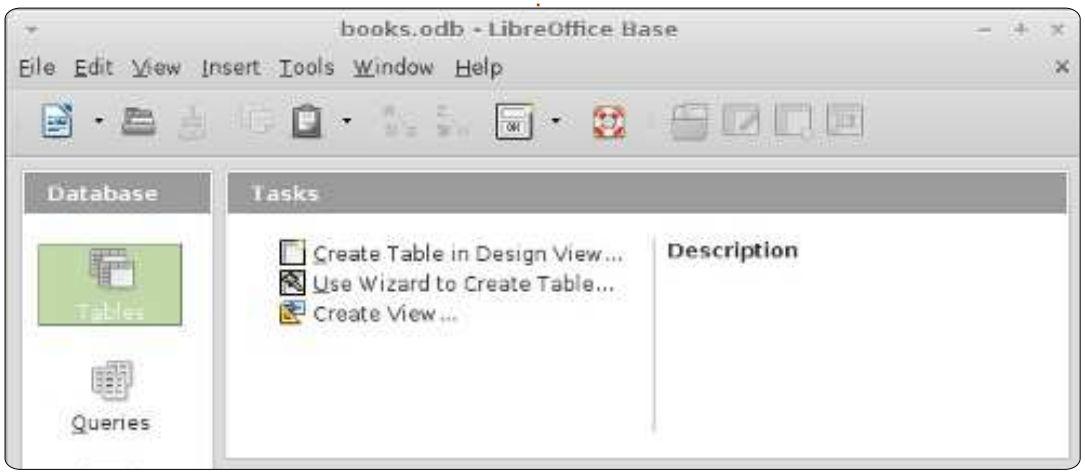
As I mentioned before, Base is not a database file but an interface for accessing and manipulating a database file. Although it is possible to connect to many different database types, we will use the default HSQL database for our books database.



To start the database wizard, select Database from the LibreOffice Home screen or File > New > Database. The first screen of the database wizard lets us choose whether we want to open an existing database or create a new one. Select Create a New Database, and click Next.

The second screen of the wizard asks us whether we want to register the database and what we want to do once the database has been created. Registering a database in LibreOffice makes it available in all our documents. We won't need this for our database, so select No – do not register the database. Check Open the Database for Editing, and click Finish. LibreOffice will open a file dialog to define a location and name for the database. I simply named the file: books

Once you have a name and location for the database file, the main Base screen opens. Down the left side, you have the different pieces which can make up a database file. The top right gives you access to the different actions you can take for each part, and the lower right shows the objects already created.



Field Types

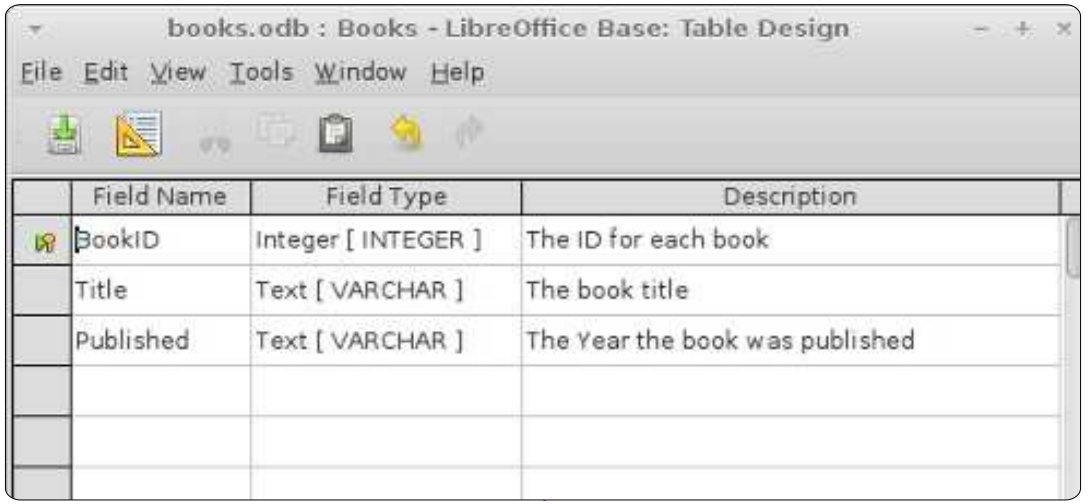
Before we create our first table, we need to discuss some of the common database field types. When you select a type for a field, you are presented with many options for the type. Many of the types are identical, and are there for compatibility reasons. The most common types are:

Integer – a whole number, eg. 123
VarChar – a variable length string of characters. You will define the maximum length for the VarChar.
Date – a date value, of course, eg. 10-15-2012 (the exact format is location specific)
Time – a time value, such as 09:15:25
decimal – a real number including the whole (integer) number and its fractional part, eg. 123.25 (the part separator is location specific)

For our purposes, we will use Integer and VarChar.

Creating the Tables

Base has three different ways to create tables: through a table wizard, through design view, and by SQL statements. The table wizard is good only to create specific types of tables by picking from a list of predefined field names. The SQL method requires you to know and understand the SQL language and is beyond the scope of this article. The design view is usually the best choice, and presents you with a list you fill in to create your table. We will use the design view to create our tables for this project.



We will start with the Books table. Select Tables from the Database pane on the left. In the Tasks pane, click on Create Table in Design View... to open the Design View dialog. Across the top you have labels for each of the elements of a field: Field Name, Field Type, and Description. The Description is optional but is useful for making notes about how a field is used. At the bottom, we see the Field Properties. This section will change according to the type of field we select.

In the first field, enter the name BookID. From the dropdown box in Field Type, select Integer. Adding a description is up to you. Under the field properties, change AutoValue to Yes. This will place a key icon in the box beside the field record showing it is the primary (or key) index. In the second row, type Title for the name. Give this one a type VarChar. Again, a description is up to you. In the field properties, leave the length at 100, the default for VarChar. The third field is Published with a type of VarChar. Change the length in the field properties to 12. I chose VarChar rather than date because we just want the year, and if the publishing date of a book is unknown, I can enter just "Unknown". Click on the save icon, and Base prompts you for a table name. Enter Books.

Our tables for Authors and Media are created in much the same way. For Authors, create two fields: AuthorID, integer (AutoValue Yes); and Name, VarChar (length 50). For Media, MediaID, integer (AutoValue Yes); and Type, VarChar (length 25).

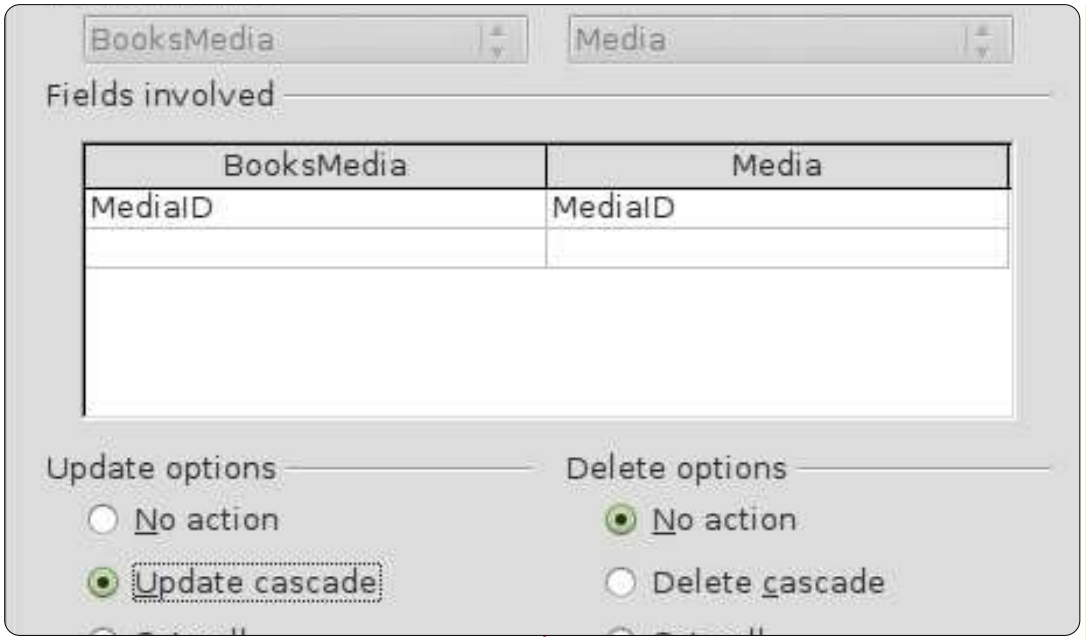
Our two foreign key tables need a little different treatment. In BooksAuthors, create two integer fields named BookID and AuthorID. Click on the icon box beside the first record. Holding down the Shift key, click in the icon box for the second. At this point, you should have both records selected. Right-click the icon box and select Primary Key from the context menu. This creates a combination key. The two values together create the primary key, which uniquely identifies each record in the table. For the BooksMedia table, create two integer fields named BookID and MediaID. Select both fields, right-click, and select Primary Key.

Create Relationships

Once we have all our tables defined, we can create the relationships that bind everything together. We will create relationships between our three main tables and our foreign key tables. The direction in which you drag the fields is important, so pay close attention to how you do it.

To start the Relation Design dialog, go to Tools > Relationships. You are presented with a list of tables. Select a table and click Add to add the table to the Relation Design. Add the tables in the following order to make it easy: Authors, BooksAuthors, Books, BooksMedia, Media. Once all the tables are added, select Close.

Drag the BookID field in Books to BookID in BooksAuthors. A Relation dialog pops up. Under Update option, pick Update cascade and OK. This will cause the field to update when the Books table updates. Drag the AuthorID in Authors to AuthorID in BooksAuthors. Select Update cascade in the Relation dialog. Next, drag the BookID in Books to BookID in BooksMedia. Select Update cascade. Finally, drag MediaID in Media to MediaID in BooksMedia. Select Update cascade. Your relation design should look something like the one pictured below.



With our tables and relationships created, we are ready to begin work on creating forms for data input. In our next How-To, we will create the forms for data entry. Everything will come together to create a usable data entry system.

Elmer Perry's history of working, and programming, computers involves an Apple IIE, adding some Amiga, a generous helping of DOS and Windows, a dash of Unix, and blend well with Linux and Ubuntu.

ubuntu uk podcast

The Ubuntu Podcast covers all the latest news and issues facing Ubuntu Linux users and Free Software fans in general. The show appeals to the newest user and the oldest coder. Our discussions cover the development of Ubuntu but aren't overly technical. We are lucky enough to have some great guests on the show, telling us first hand about the latest exciting developments they are working on, in a way that we can all understand! We also talk about the Ubuntu community and what it gets up to.

The show is presented by members of the UK's Ubuntu Linux community. Because it is covered by the Ubuntu Code of Conduct it is suitable for all.

The show is broadcast live every fortnight on a Tuesday evening (British time) and is available for download the following day.

podcast.ubuntu-uk.org

LibreOffice Pt21: Base Input Forms

Written by Elmer Perry

So far in our series on LibreOffice Base, we have planned the layout for our database, created the tables, and created the relationships between those tables. Now, we need to think about how we will input the data into our tables and link them all together. You can edit the tables manually in the tables section of the program, but the best way is through input forms. We will create three forms: Authors, Media, and Books. We will handle the forms for authors and media first as they are the easiest. The form for books will bring everything together in one place.

Create the Authors and Media Forms

Select forms from the Database pane on the left. You can create forms in two ways, through the wizard or through the Design View. We will use the wizard for Authors and Media. Click on Create from wizard... in the tasks pane. This starts the wizard.

On the first step of the wizard, select the Authors table from the dropdown, and move Name to the Fields to use. This is done by selecting the field and using the arrow keys. Click Next.

The second step deals with subforms, which we do not need for the Authors form. Click Next. You will notice that steps three and four are skipped. Those steps deal with setting up a subform. We will not use them in the our implementation of our database.

Step five asks us for a layout for our fields. Since we have only one field, we will choose the table layout. Select the table layout and click Next.

The sixth step is about how the form is used. We want the form to display all data, so select all data and click Next.

Step seven deals with the styles of the form. You can select different background layouts for your form. You can also select whether your controls are displayed with no border, 3D borders, or flat. Click Next.

On the eighth step, we give the form a name, Authors, and choose whether we want to edit it more or use it. After naming the form, click Finish. The new form will pop up on your screen. Close it and save your work.

The Media form is created in the same way, only using the Media table instead of the Author table.

Create the Books Form

Although we will use the wizard to start our Books form, we will need to edit it afterwards to add the connections to authors and media. Go through the wizard again with the Books table, adding the fields Title and Published. You will skip the subforms again. We will add our subforms manually. For the layout, use either Columnar – Labels on Top, or Columnar – Labels Left. I used Columnar – Labels on top. After naming the form Books in step eight, select Modify the form, and click Finish.

This time the form is opened for editing. We will add two subforms to make a connection to the Authors and Media tables.

On the toolbar at the bottom of the form, select the Form Navigator. This opens a small dialog containing the elements of our form. Right-click MainForm, then New > Form. A new form is added to the list. Since it is already selected, just start typing to rename it FormAuthors. We will use this subform to create the link to our Authors table through the BooksAuthors table. Right-click the FormAuthors and select Properties. On the Data tab, select table for the Content type, and BooksAuthors for the Content. Now, click on the ellipse button next to List Master Field. A link-fields dialog pops up. This is where we create our link between the Books table and the Authors table. Under BooksAuthors, select BookID and under Books select BookID. Click OK, and you can close the Form Properties dialog.

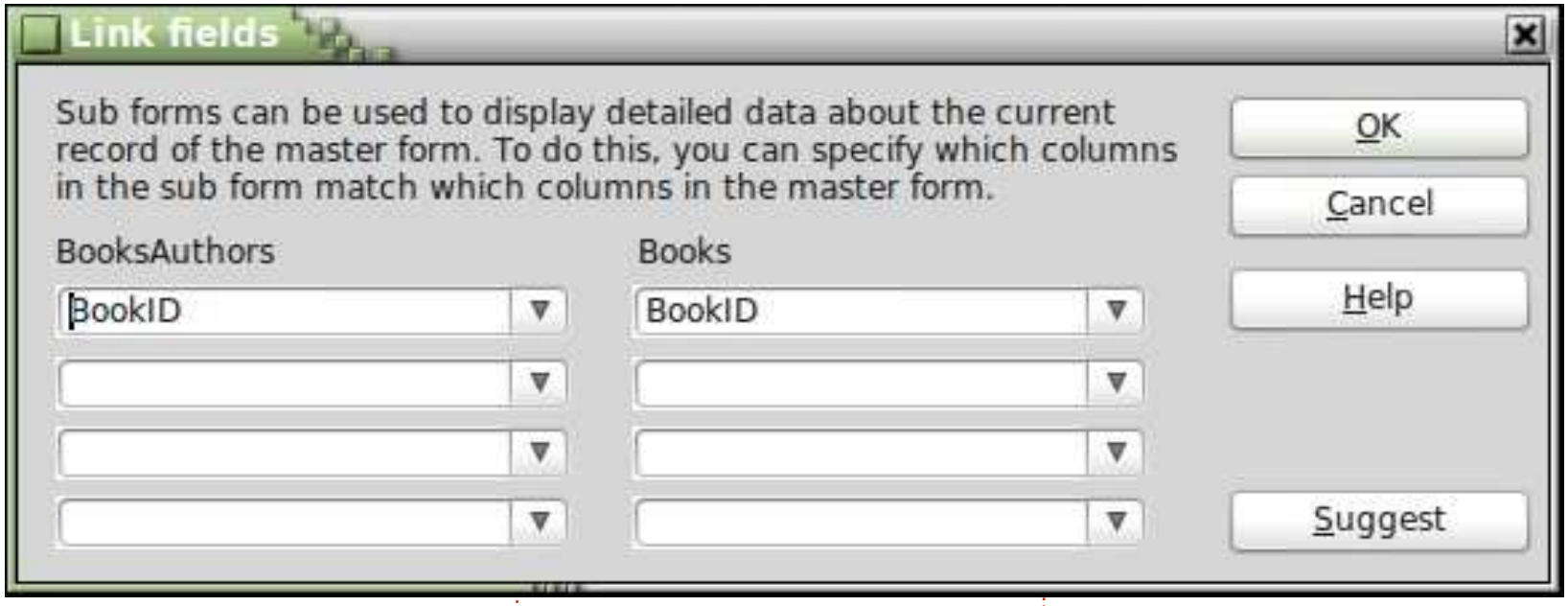
Create another subform under MainForm named FormMedia. Edit the properties, setting the Content Type to table, and the Content to BooksMedia. Click the ellipse next to List Master Field, and select BookID under both BooksMedia and Books. Click OK and close the Form Properties dialog.

Now, we need to create our controls that will contain our authors and media. Remember, we need the ability to select more than one, so a table is our best choice for the control. Click on the More Controls button on the toolbar on the left side. A More Controls toolbar will pop up.

Make sure you have the FormAuthors selected in the Form Navigator. Click on the Table button in the More Controls toolbar. Underneath the two input boxes for title and published, draw the table on the page. Right-click in the header section of the new table and select Insert Column > List Box. Right-click the newly created column and select Column. This displays the List Box properties. On the General tab, change the label to Authors. Switch to the Data tab. For the Data field, select AuhtorID. This tells Base we want to relate this to the AuthorID field in the BooksAuthors table. For Type of list content, select Sql. In List Content, we will write a short SQL statement to fill our list box with the names from the Authors table. You can click on the down-arrow to give yourself a small edit box to work in. The SQL statement is:

SELECT “Name”, “AuthorID” FROM “Authors”

This statement will select all the records from the Authors table. Set the Bound field to 1. The Bound field selects which field will fill the list box. Since we selected the Name field first, the bound field of 1 will fill the list box with the data from the Name fields. Close the list box properties and save your work.



We will create the Media control in the same way we did the Authors. Select the FormMedia in the Form Navigator. Draw the

contents

table control to the right of your title and published controls. Create a column in the control. Open the column properties and change the label to Media. Set the Data field to MediaID, and the Type of list content to Sql. The SQL statement for the List Content is:

SELECT “Type”, “MediaID” FROM “Media”

Set the Bound field to 1.

We are now finished with the form. Save and close it.

Using the Forms

The Authors and Media forms are simple to use. Just select an empty row and type in the name or type. You will want to add your authors and types before using the books form. The books form is easy to use as well. Fill in your Title and Published year (or Unknown if you don't know the year). The authors and media tables give you a list box from which you can select your authors and media types. Note that you can select more than one author and media type. To create a new record, click on the new record or next buttons in the form control toolbar.

We now have a working method for entering data into our database. While it works, there is one disadvantage to this setup. If you find that you need an author or media type that is not there, you have to close the books form and open the authors or media forms. In a later How-To, we will attempt to overcome this inconvenience.

In the next HowTo, we will create a query and a report for extracting information from our database.

Elmer Perry's history of working, and programming, computers involves an Apple IIE, adding some Amiga, a generous helping of DOS and Windows, a dash of Unix, and blend well with Linux and Ubuntu.

ubuntu uk podcast

The Ubuntu Podcast covers all the latest news and issues facing Ubuntu Linux users and Free Software fans in general. The show appeals to the newest user and the oldest coder. Our discussions cover the development of Ubuntu but aren’t overly technical. We are lucky enough to have some great guests on the show, telling us first hand about the latest exciting developments they are working on, in a way that we can all understand! We also talk about the Ubuntu community and what it gets up to.

The show is presented by members of the UK’s Ubuntu Linux community. Because it is covered by the Ubuntu Code of Conduct it is suitable for all.

The show is broadcast live every fortnight on a Tuesday evening (British time) and is available for download the following day.

podcast.ubuntu-uk.org

LibreOffice Pt22: Base Queries & Reports

Written by Elmer Perry

If you have been following along in this series on LibreOffice Base, you now have a database file with tables and relationships. You can enter information into your database through forms, but what about getting information out of the database? Queries and Reports are used for extracting data from your database - the Queries define what data is extracted, and the Reports define the appearance of the extracted data. We will create a query and a report to show how you can generate a report of your data.

Creating a Query

Queries poll the database for certain information in your database. You have three ways to create a query: a wizard, design view, and SQL. The wizard doesn't work with the type of relational database we have created and SQL is beyond the scope of this How-to, so we will use the design view to create our query. We will create a query that contains all the important fields in our tables: title, published year, author(s), and type(s).

Click on Queries in the Database pane, then click on Create Query in Design View in the Tasks pane. A Query design form will display with a Add Tables or Query pop-up dialog. Add all the tables to the Query Design form, and close the pop-up. You will end up with a form that looks a lot like the relationships design we created previously. Below the tables, you see a form that will contain the fields which we want to include in our query. From the Books table drag Title and Published into the form. Drag Name from Authors, and drag Type from Media. That is all we need for this query. Save it as AllFields. Close the Query Design form.

You now have a query to use in creating multiple reports for your database. If you're curious about the SQL used to create your query, you can right-click on the newly created query and select Edit in SQL View. This brings up the SQL View with the complete SQL statement for the query you just created. I wouldn't recommend changing this unless you are well versed in SQL, but creating multiple queries in Design View, and then viewing them in SQL View, could help you begin to learn SQL.

Creating A Report

A report runs a query and formats the query results into something you can use. You can create many different reports with the query we created – depending on how you group the data from the query. We will create a report for sorting our books by media type, and I will suggest how you might create other reports using

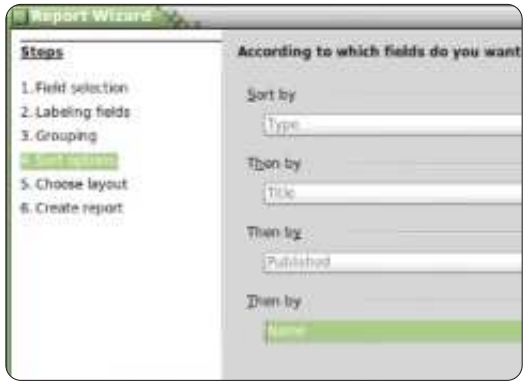
the same query.

There is only one way to create a report – use the wizard. Select Reports in the Database pane and click on Use Wizard to Create Report. A report template window will appear with the report wizard. You can actually watch your template fill in as you go through the steps of the wizard, giving you some ideas about how your final report will look.

In step 1, you will pick your query and the fields to use in the report. If not selected already, select the AllFields query we just created. Move all the fields into the Fields in report box by clicking on the >> button. If you wanted to create a report that uses just some of the fields, you would just select the required fields. For our report, we will use all the fields. Click Next.

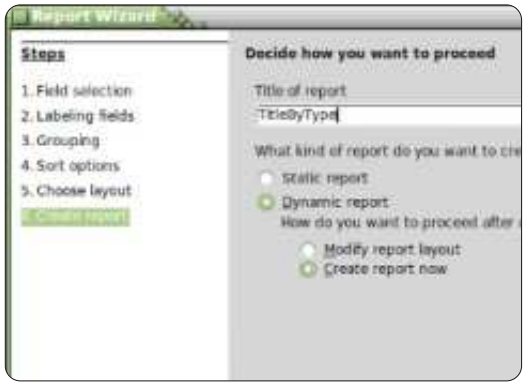
Step 2 is labeling our fields. Here we specify how the fields are labeled in our report. Change the label for Type to Media Type, and Name to Author(s). Click Next.

We create our groupings in step 3. Groupings control how the fields are consolidated and arranged in the report. Grouping is important, because we can use it to create a whole different report depending on how we group the fields. For example, if you wanted to create a report of titles by author, you would make Name your first grouping, then Title as a sub-group. For our report, we are creating a report of Titles by Media Type, so our grouping order is Type, Title, Published. We add published because, if there is more than one author, the published date will repeat in the report template, a byproduct we don't want. We exclude the Name field because if there is more than one author, we want them listed together. Click Next.



Sorting is done in step 4, but we don't have much use for it here. You will notice that you can change only whether the sorting is ascending or descending for our groupings. In the fourth box, select Name and leave on Ascending. Click Next.

In step 5, we can choose a layout for the data and the header. There are several for each, and they change the look of the report. I left mine at the defaults, but feel free to play around with these options. Under the Layout of data list box, you can select whether the report is landscape or portrait. For this report, I think portrait will work best. Click Next.



The final step is where we create the report. You can give it a title, indicate how the report is used, and what to do with the report. For this report, give it a title of TitleByType. Now, we need to answer the two questions. What type of report do you want to create? A static report is a one time report. It cannot change. Once it is created, the data is fixed. If, however, you want a report you can re-use, you want a dynamic report. A dynamic report is just a template you can use over again. For this report, we want a dynamic

report we can use again, so select Dynamic. How do you want to proceed after creating report? Modify report layout will allow you to edit the report as a writer document. Create report now is obvious; it will fill in the data and generate the report. We can always edit the layout later, so we will select Create report now. Click Finish.

Base generates our report and displays it in a Writer window. This generated report is read-only. If you want to edit the text or layout of the report, close it. In the Reports pane, right-click your report and edit. This opens the report template in Writer, where you can add text, graphics, etc, just like you would in any Writer document. Just take care when changing anything in the cells where the data is plugged in.

In this How-to, we created a query and a report. Play around with the grouping and sorting settings in the report wizard and see just how many different reports you can create from the one query we created.

Next time, we will use macros to create enhancements to our database and make it act more like an application.

Elmer Perry's history of working, and programming, computers involves an Apple IIE, adding some Amiga, a generous helping of DOS and Windows, a dash of Unix, and blend well with Linux and Ubuntu.

16x16 SUDOKU

Numbers 0 to 9 and letters A to F are to be filled into the 16x16 grid so that every row, every column, and every 4x4 box contains 0 to 9 and A - F.

	7	8		6		F		2	D	B	9		A	3	
6	F				E					3	A			C	2
2			1	D			A			8					5
			9	3	1			6			4	E	D		
C	1		3					8				D	E		4
B	A	4				E	3	9	5			F		0	
9					2	4		C	6	A					8
D			6	A	8	7				4			1		
		9			5				0	6	3	B			E
5					7	D	9		B	E					1
	D		B			3	C	5	9				8	6	F
0		3	2				4					A		9	7
		A	7	B			E			9	0	2			
8					F			3			B	5			D
1	B			9	0					F				8	3
	0	E		2	4	C	6		8		5		F	B	

Solutions are on the second last page.

LibreOffice Pt23: Base Forms & Macros

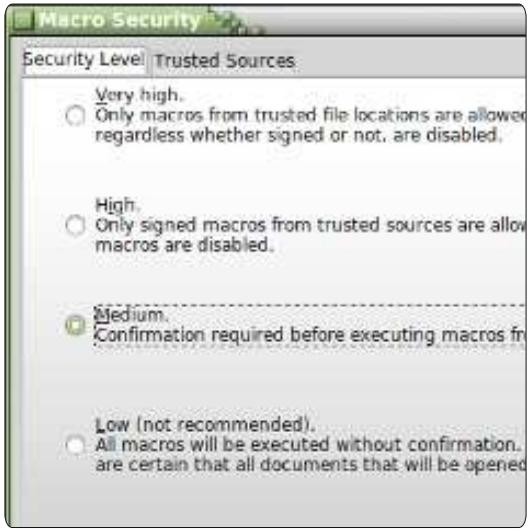
Written by Elmer Perry

For the previous four parts of this series, we have slowly built a database document using LibreOffice's Base module. We have a database with forms to enter our data, and queries and reports for extracting the data. We now have a usable document for recording our book library. However, our current design has one flaw we need to overcome. If we need to enter a new author or media type while we are in the books form, we have to close the book form and open one of the others. If we could enter new authors and media types directly from the books form, it would behave more like an application and make data entry even easier. We can accomplish this through a few short macros.

The LibreOffice Basic language is very similar to other Basic languages, such as Visual Basic for Applications. To manipulate the underlying LibreOffice document, we access the Uno framework controlling the document. The Uno framework is quite complex, but I will explain, as best I can, the properties and objects we will use. The goal is not to teach you how to write LibreOffice macros, but how you can use them.

Macro Security and Options

While macros allow us to do cool things in our documents, they can also cause problems. Some people use macros to compromise other people's systems, therefore, we need to take a few minutes to talk about macro security. Whether



you are running LibreOffice on Linux, Mac, or Windows, malicious code in a macro can compromise your data and possibly your entire system.

Macro security in LibreOffice is simple. Tools > Options opens the Options dialog for LibreOffice. Under LibreOffice, select Security. Click on the Macro Security button to pop up the macro security options. You have four options. Never use the Low security option – it will run macros without asking you. I recommend the Medium security level. With this level, you are prompted whether to run the macros when you open a document containing macros. The High and Very High options require a certificate or folder you designate as trusted. While this is great, I believe nothing trumps the instincts of the user. You usually know whether you were expecting a document to contain macros. When in doubt, click No. Click OK to save your choice and OK to close the options dialog.

Now, on to the fun stuff.

The Macros

We will write four macros for our database document. Three will deal with opening forms, and the last will update the list boxes for authors and media types. The general idea behind macros is to accomplish tasks that are not built into the program, or to simplify complex tasks. Our macros really accomplish both, as we will simplify the tasks of adding authors and media types and provide functionality not built into the program.

Before we can begin to write our macros, we need a container to hold them. Macros are contained in a module. Modules can live in the program itself or within a document. Since our macros are specific to our database file, we will embed them in the document. Macros embedded in a document are available only when the document is loaded. Macros contained in the program are available as long as the program is running.

Tools > Macros > Organize Macros > LibreOffice Basic. The LibreOffice Basic Macros dialog pops up. Select book.odb from the Macro from-list. Click the New button. A dialog pops up asking you for a name for the module. Name it FormCalls. Click OK. This brings up the LibreOffice macro editor. The macro comes with a default main subroutine. We will not use this subroutine. Highlight Sub main and End Sub and press the backspace key to delete them.

Our first macro is a generalized subroutine for opening a form. A generalized subroutine is written for reuse. We will call this routine twice from other routines we write. Type the subroutine shown above into the editor.

The first line of the subroutine is called the signature. The signature determines how the subroutine is called. A signature starts with the keyword Sub, which defines this call as a subroutine. Next, the name of the subroutine. In our case, OpenAForm is the name of the subroutine. Finally in the parenthesis, we have the arguments used when calling this subroutine. In our case, we have a variable named FormName which is a type String. In the second line of the subroutine, Dim is another keyword. Dim initializes a variable as a type, and, optionally, a value. We define a variable named GetForm as a type Object. The third line assigns a value to the variable GetForm through a chain of commands in the Uno framework.

```
Sub OpenAForm (FormName as String)
     Dim GetForm as Object
     GetForm = ThisDatabaseDocument.FormDocuments.GetByName(FormName)
     GetForm.Open
End Sub
```

ThisDatabaseDocument refers to the currently open database document. In our case, book.odb. FormDocuments is a collection of all the forms in the document. Finally, GetByName retrieves a specific form object from the collection. Notice, we pass the variable FormName from the signature to this method. Once the call is complete, the variable GetForm is the object of the form name passed to the subroutine. The fourth line calls the Open method of the form. On the fifth line, we tell Basic this is the end of the subroutine with the command End Sub.

We will call the OpenAform subroutine twice. Once to open the authors form, and once to open the media form. Add the two subroutines shown below to your editor.

The signature on these two subroutines are a little different. Since we will call them from a control within a form, we need to pass the object making the call as an argument, even though we do not use it. The argument oEv is a reference to the object making the call. We will use this to our advantage later, in the last subroutine, but here we do it because it is required. These two subroutines are pretty simple. We just make a call to OpenAForm passing the name of the form we want to open, Authors or Media.

```
Sub OpenAuthorsForm(oEv As Object)
     OpenAForm("Authors")
End Sub


Sub OpenMediaForm(oEv As Object)
     OpenAForm("Media")
End Sub
```

The final subroutine deals with our problem of refreshing the data in the list boxes for authors and media when we add authors or media using the two subroutines above.

```
Sub ListRefresh(oEv as
Object)

oEv.source.model.Refresh

End Sub
```

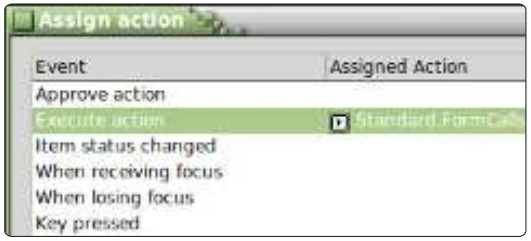
Once again, since we will call this subroutine (shown right) from a control, we need a reference to the control making the call. However, this time we will actually use the object. This subroutine makes a method call to the

underlying model of the list box and refreshes the data in the list, thus updating our list of authors or media types. Save your module and close the Basic editor.

Making Connections to Macros

At this point, our macros do nothing. We need to connect them to objects in our form to activate them when needed. First, we will connect the open form subroutines to buttons in our form, and then we will connect the ListRefresh to the list boxes.

In the database pane, click on Forms. Right-click the Books form and select edit. Add two push buttons to the form, one under the Authors table and another under the Media table. Right-click the button under the Authors table and select Control to bring up the buttons properties dialog. On the General tab, change the name to AddAuthors and the Label to Add Authors. On the Events tab, click the ellipses (…) button next to Execute Action – which brings up the Assign Action dialog. Click the Macro button to bring up the Macro Selector dialog. In the tree list under Library, select book.odb > Standard > FormCalls. Select OpenAuthorsForm from the Macro Name list and click OK. Click OK to close the Assign Action dialog. Close the buttons properties dialog.



Do the same with the button under the Media table, only name it AddMedia, make the label Add Media Type, and assign the macro OpenMediaForm to the Execute Action event.

Finally, we need to add the refresh subroutine to our list boxes. Right-click the Authors column in the authors table and select Column. On the Events tab, click the ellipse (…) button beside "When receiving focus". In the Assign Action button, use the Macro button to assign the ListRefresh macro to the action. This will cause the list to update data from the Authors table when you click on a list box in the column. Do the same for the Media column in the media table. Save your changes to the Books form and close it.

Testing Your Changes

Any time we make changes to our forms, we will want to test them and make sure we got everything right, especially in cases where we have used macros. One simple typo could cause things to not work. Double-click the Books form to open it. Add a new book with an author and media type you have not added already. Click the Add Authors button to make sure it opens the form. Add some authors. Close the Authors form. Click on the authors dropdown list box and verify that the authors you added are there. Do the same test with the Add Media Type button and listbox.

Final Thoughts and References

Again, I would like to emphasize that writing macros in LibreOffice Basic is complex. Documentation is pretty sparse, but it is out there. If you are interested in taking up the challenge, here are some references to get you started:

LibreOffice Basic Guide:
http://wiki.documentfoundation.org/images/d/dd/BasicGuide_OOo3.2.0.odt

Andrew Pitonyak's OpenOffice Macro Information:
http://www.pitonyak.org/oo.php

You can find the macros used in this How-To on pastebin.com at http://pastebin.com/MU2Ztizi

Next time, we will move on to another part of the LibreOffice suite and explore the Math module.

Elmer Perry's history of working, and programming, computers involves an Apple IIE, adding some Amiga, a generous helping of DOS and Windows, a dash of Unix, and blend well with Linux and Ubuntu.

LibreOffice Pt24: Intro To Math

Have you ever been working in a word processor and needed to insert a formula into the text? Perhaps you were writing a math or scientific paper for college, or even answering a question about statistics. If you need to enter anything beyond elementary math, you will quickly run into formatting issues. LibreOffice overcomes this problem by providing us with the Math or Formula module. You can use the module independently to create formulas, or use it directly in the other modules of LibreOffice. Today, we will learn how to enter formulas in the Math editor, and, in later articles, we will learn how to use formulas in Writer.

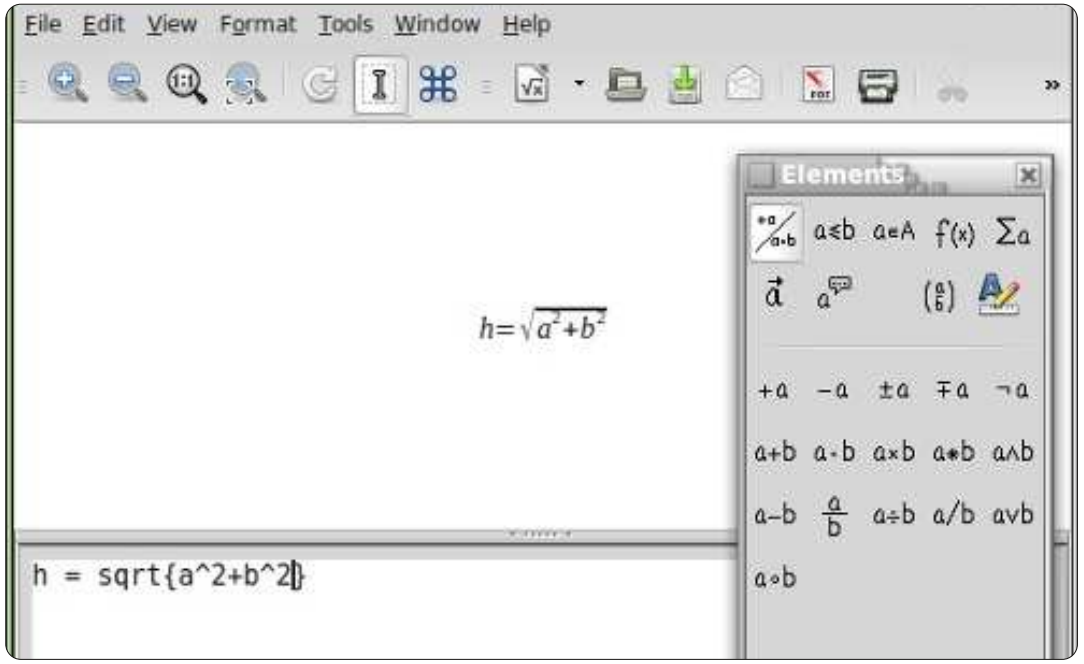
Open a new Math window by clicking on the Formula button on the LibreOffice Start Center, or through the menus with File > New > Formula.

The Formula Window

The formula window has three pieces: the preview pane, the formula editor, and the elements window. The preview pane at the top shows you your formula as it is created. The formula editor at the bottom is where you enter your formula. The floating Elements window provides you with shortcuts to different formula elements. Think of the elements as building blocks for creating your formula.

Three Ways to Enter Formulas

There are three ways to enter formulas into the formula editor: through the Elements window, through a context menu, or by direct entry.



The Elements Window

The Elements window is divided into two sections. The top section is the category section, and the lower section contains the elements in that category. If you select a category then click on one of the elements in that category, the program will enter the element into the editor with <?> as placeholders for the variables of the element. The first placeholder is highlighted. Use the F4 key to move to the next element. Shift-F4 will move backwards through the placeholders.

To get you familiar with the Elements window, I will walk you through the steps to write a formula using the Elements window. Starting with a new formula window, select the Relations category, then the equals element. <?> = <?> appears in the formula editor. The first <?> is highlighted. Enter the letter “h”. Press F4 to move to the other <?>. Select the Functions category, then the square root element. The <?> is replaced with sqrt{<?>} and the placeholder in the brackets is highlighted. Select the Unary/Binary category, then the addition element. The program inserts <?> + <?> into the square root's brackets. Select the Formats category, then the Superscript Right element. <?>^{<?>} replaces the highlighted placeholder. Enter the letter “a” and press F4 to move

to the next placeholder. Enter the the number “2”. Press F4 to move to the next placeholder. Select the Superscript Right from the Formats category. Enter the letter “b” and press F4 to move to the last placeholder. Enter the number “2”. The final result will look like this:

$$h=\sqrt{a^2+b^2}$$

and the text in the formula editor is:

```
h = sqrt{a^{2} + b^{2} }
```

The Context Menu

The context menu (shown below right) is much like the Elements window. Right-click in the formula editor, and you get a menu of all the categories. Each category has a submenu of the elements. Click on the element to insert it into the formula editor. Follow the example above again, but this time use the right-click context menu to create the formula. You should get the same results.

Direct Entry

As you work with Math and learn the elements, you can enter the formulas directly in the formula editor. By far, this is the quickest way to enter a formula. Now that you have created the formula twice, using the Elements window and context menu, see if you can enter it directly into the editor without using the element tools. If you need help, just reference the editor text shown above.

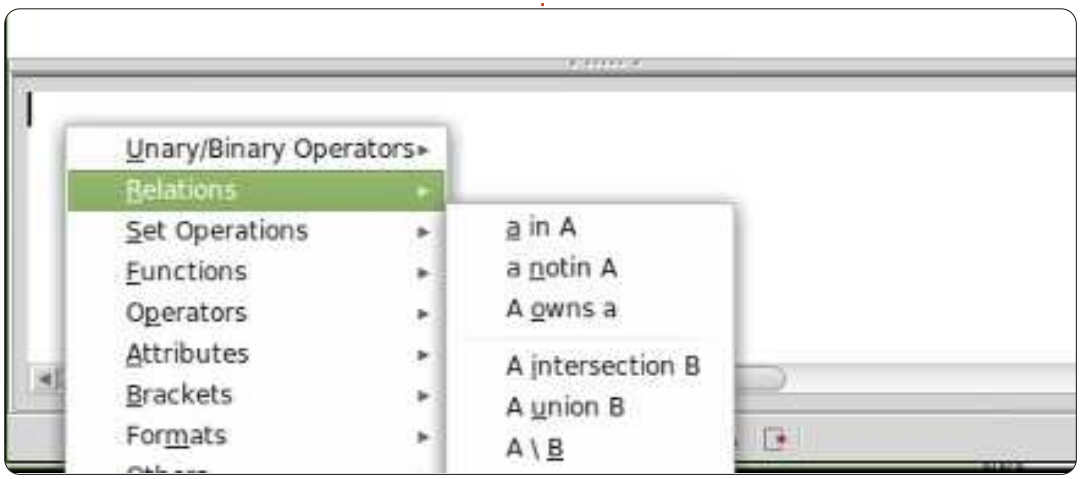
Special Characters

You won't find everything you need in the Elements window and context menu. Many equations use Greek characters and other symbols. LibreOffice Math allows you to enter special characters into your equation. If you find you need a special character not listed in the special characters, you can even add your own.

Adding Greek Characters

Through the menus Tools > Catalog, you can access the Greek letters through the character subsets Greek and iGreek. Greek is the letters in plain text and iGreek is the letters in italics. Just select the letter you want and click the Insert button. When finished, click the Close button.

For direct entry, type in % followed by the Greek letter name. For example, to get the Greek letter pi, enter %pi. To get the uppercase letter, make the name uppercase, %PI. To make the character italics, place a lowercase “i” before the letter's name, %ipi.



Other Special Characters

Other special characters are found in Tools > Catalog under the Special subset. Select the symbol you need and click the Insert button. As you use and learn the names of the symbols, you can enter them directly using the % and then the name of the symbol.

NOTE: The lowercase “i” for italics works with only the Greek letters. We will discuss inserting italics for other elements in the next How-To.

Adding Special Characters

If the catalog does not have the special character you need, you can add it to the catalog. One such character is the prime symbol. Let's add it to our special subset.

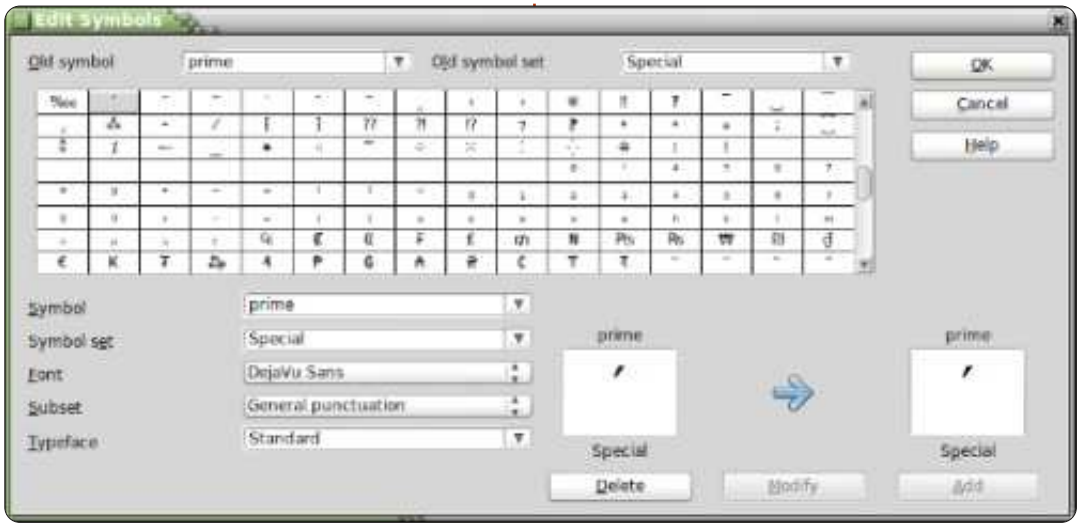
Tools > Catalog and select the Special symbol set. Click on the Edit button. This bring up the Edit Symbol dialog. For the font select DejaVu Sans, and for the subset select General Punctuation. The symbol you want is Ux2023. For the symbol name type in prime.

Click the Add and OK buttons. The prime symbol has now been added to the Special symbols list. You can use it by selecting it from the catalog, or enter it directly by typing %prime.

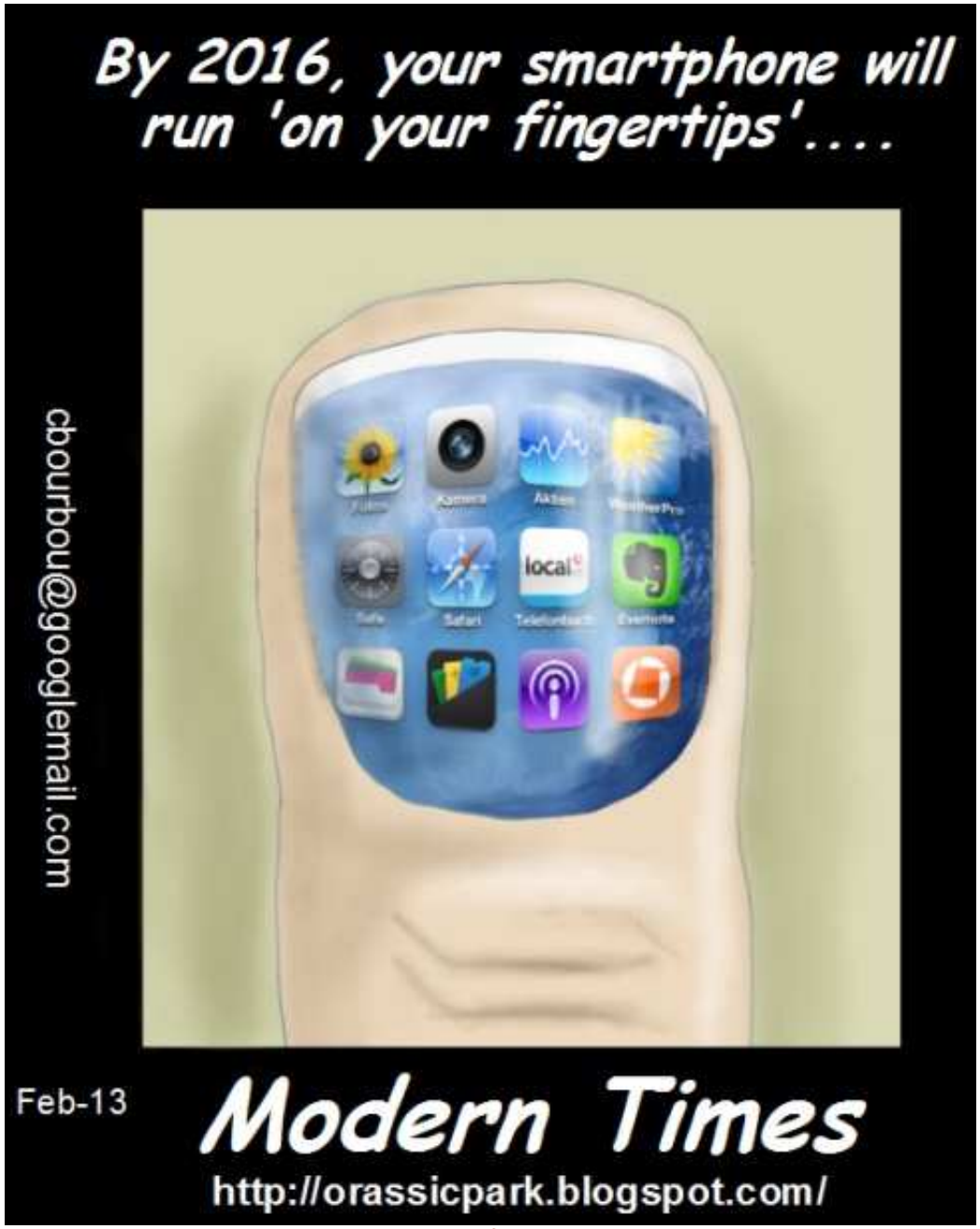
Conclusions

Math allows you to create formulas you can insert into your documents. You have three methods for entering formulas into the formula editor: through the Elements window, through the context menu, and by direct entry. The Elements window and the context menu help you to learn how to enter the different elements of a formula, but once you know how to enter an element, direct entry is the quickest way to create a formula.

In the next LibreOffice How-To, we will look at ways to format our formulas so they look the way you want them.



Elmer Perry's history of working, and programming, computers involves an Apple IIE, adding some Amiga, a generous helping of DOS and Windows, a dash of Unix, and blend well with Linux and Ubuntu.



LibreOffice Pt25: Formula Layout

As you create more complex formulas in Math, you will soon discover that things don't always display the way you were expecting. There are a few tricks to making formulas display the way we want them, especially complex formulas. Today, we will look at many of these tricks to make our formulas come out right.

Group Elements with Brackets

The curly brackets, {}, have a special use in formulas. They help you to group elements together. Without them you can get a different formula than the one you were expecting. Here are a couple of examples to show you what I mean.

Enter the following into the formula editor:

```
2 over x + 1
```

You will get the following result: $\frac{2}{x}+1$

But what if you actually wanted the x + 1 in the denominator of the fraction? You can use curly brackets to group the two elements together.

Enter the same formula in the editor, but with curly brackets grouping the addition:

```
2 over {x + 1}
```

You then get the result you wanted: $\frac{2}{x+1}$

Any time the formula doesn't flow the way you were expecting, you can use curly brackets to group items together to make things come out right. You will see more use of brackets as we work through other examples in this How-To.

Equations Spanning More Than One Line

Some equations make more sense if they are broken into multiple lines, or you need to show the progression of a formula through each step to its conclusion. Doing this all on one line would make the formula difficult to read. However, just pressing the Enter key in the editor does not result in a new line. In order to get a new line in the formula, you use the newline element.

Editor example:

```
x over 250 = 5 over 100
newline
 100x = 250(5) newline
 100x = 1250 newline
 100x over 100 = 1250 over
100 newline
 x = 12.5
```

Result:

$$\frac{x}{250}=\frac{5}{100}$$
$$100x=250(5)$$
$$100x=1250$$
$$\frac{100x}{100}=\frac{1250}{100}$$
$$x=12.5$$

Sum / Integral Limits

The sum and int commands can take optional parameters to signify the range of the sum or integral. The keywords 'from' and 'to' generate the lower and upper range of these commands. The following markup demonstrates:

```
 sum from x=0 to x=n f(n) "
or "
 int from x to n f(n+1)
```

Result: $\sum_{x=0}^{x=n} f(n)$ or $\int_{x}^{n} f(n+1)$

Scaled Brackets

Sometimes, you need a bracket to span more than one line. A good example of this is with a matrix. If you just use the bracket characters, you get an ugly looking matrix.

The markup:

```
(matrix {x#x+1##y#y+1})
```

The result: $\begin{pmatrix} x & x+1 \\ y & y+1 \end{pmatrix}$

To get brackets that scale to the size of our matrix we use the markup "left (" and "right)". This results in a much nicer looking matrix.

The markup:

```
left(matrix {x#x+1##y#y+1} right)
```

The result:

$$\begin{pmatrix} x & x+1 \\ y & y+1 \end{pmatrix}$$

To get scalable square brackets use “left [“ and “right]”. To get scalable curly brackets use “left lbrace” and “right rbrace”. You can find a full list of all the brackets in the LibreOffice help documentation.

Unpaired Brackets

Sometimes, you only need a bracket on one side, but not the other. If you don't put a closing bracket, you get an inverted question mark and the equation will look messy. To fix this problem, use the mark up “left none” or “right none”, depending on your needs, to indicate you have no opening or closing bracket.

A good example of this is the mathematical definition of the Lucas Numbers.

The markup:

```
L_n = left lbrace
matrix{2 # if n = 0; ##
1 # if n = 1; ##
 L_{n-1} + L_{n-2} # if n > 1.}
 right none
```

The result:

$$L_n = \begin{cases} 2 & \text{if } n=0; \\ 1 & \text{if } n=1; \\ L_{n-1}+L_{n-2} & \text{if } n>1. \end{cases}$$

Notice that I ended the definition with “right none” to get our definition to show correctly.

Aligning Elements Using Matrix

You will notice (in the Lucas numbers definition) I used a matrix to get everything to line up correctly. The matrix command is useful for this because Math doesn't have a command for aligning on a certain element. With the matrix command, we can use the columns and rows to get elements to align the way we want them. A good example of this is to get equations to align on the equals sign.

For example:

```
matrix {
 3x + 2x # `=` # 45 ##
alignr 6x # `=` # 45
}
```

The result:

$$\begin{aligned} 3x+2x &= 45 \\ 6x &= 45 \end{aligned}$$

You will notice the back tick or grave (` ) marks around the equals signs. This is necessary because the equals sign is a binary operator and requires an expression on both sides. The back tick ( `) is the small space mark in Math. You could accomplish the same thing with the Math symbol for a long space (~) or empty brackets ({}). Remember in a matrix that everything between the hashes is an independent expression.

You will also notice, in the second row, I use the command “alignr” to align the 6x to the right in its column. Use “alignl” to align to the left and “alignc” to align to the center. Center alignment is the default, except in a matrix, which defaults to left alignment.

Text in a Formula

Sometimes, you will need to add notes or text to your formula. You can add text by enclosing the text in quotation marks (“”).

Example:

```
c^2 = a^2 + b^2 newline
"The Pythagorean Theorem."
```

Result:

$$c^2 = a^2 + b^2$$

The Pythagorean Theorem

Chemical Formulas

Math was designed for mathematical equations, but you can make chemical formulas, too. Since variables are usually in italics, you will want to turn off the italics for variables (explained later).

Example:

```
matrix {
 "molecules" # H_2 SO_4 ##
 "Isotopes" # U lsub 92 lsup 238 ##
 "Ions" # SO_4^{2-{}}
 }
```

Result:

molecules	H_2SO_4
Isotopes	$^{238}_{92}U$
Ions	SO_4^{2-}

Note the “lsub” and “lsup” in the isotope formula. The “lsub” makes a left subscript, and the “lsup” makes a left superscript. You

will also need to add some special double arrows to your catalog for chemical formulas.

Color, Bold, and Italics

The color, bold, and ital commands allow you to emphasize certain parts of your formula. They affect only the elements which follow them. To affect more elements you need to group them together with brackets.

Example:

```
bold color blue c^2 = color
red {a^2 + b^2} newline
 ital color green "The
Pythagorean Theorem"
```

Result:

$$c^2 = a^2 + b^2$$

The Pythagorean Theorem

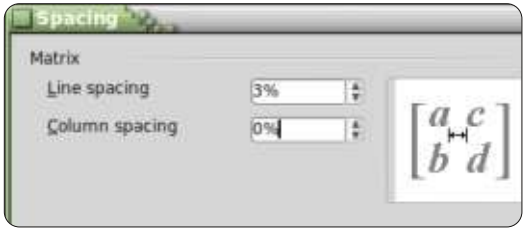
Notice that I put brackets around the elements in the square root to make them red. Also, for the c2, I combined color with the bold command. You can choose from eight colors: black, white, cyan, magenta, red, blue, green, and yellow.

Changing the Font and Font Size

Sometimes, you will want to use a font or font size other than the defaults. Format > Fonts brings up the dialog to change the fonts for variables, functions, numbers, and text. You can also set some custom fonts here as well. Format > Font Size brings up the dialog for font sizes. You set the base font, and then the element sizes are set as percentages of the base.

Changing Element Spacing and Alignment

Sometimes, it is necessary to change the spacing and alignment of the overall formula. Format > Spacing bring up the spacing dialog. In the Category dropdown, you select the type of element spacing you want to control. You set the different spacing for the elements as percentages of the elements width or height, depending on the spacing type.



For example, if we wanted to change the spacing for the alignment on the equals signs we did earlier, we would select matrix from the category. If we set the column spacing to 0%, the expressions will butt up against the equals sign.

Result:

$$\begin{aligned} 3x+2x &= 45 \\ 6x &= 45 \end{aligned}$$

Format > Alignment brings up the alignment dialog. Here we can change the default alignment for the formula, left, right, or center.

Conclusion

All the different options for formatting your formula can seem overwhelming, but you will get the hang of it with practice. The first and most important thing to remember is using curly brackets to group elements in a formula. Look in the Elements dialog or the context menu when in doubt about how to do something, and you might want to keep this article on hand as a reference.

Next month, we will discuss using formulas in LibreOffice Writer.

Elmer Perry's history of working, and programming, computers involves an Apple IIE, adding some Amiga, a generous helping of DOS and Windows, a dash of Unix, and blend well with Linux and Ubuntu.

LibreOffice Pt26: Formulas In Writer

Written by Elmer Perry

Being able to create formulas in the LibreOffice Math module is great, but what if you need to insert a formula into a text document? The integration between Writer and Math allows you to insert formulas into your text, to number equations, to reference numbered equations, and to insert equations created in the Math module.

Inserting A Formula into Writer

Insert a formula through the menus with Insert > Object > Formula. A formula editor window will open at the bottom of the Writer window, and the floating Elements window will open. A frame border will appear in the document where the formula will be displayed. You can use the formula editor just as you would in the Math module. When you are finished creating your formula, press the ESC key, or click anywhere in the document outside the formula frame.

In Writer, formulas are OLE objects, and, by default, are inserted as characters, meaning they stay in line with the text. You can change the way text flows around the formula by changing the anchor point. Right-click on the formula, Anchor > To Page. With the formula anchored to the page, you can move it to wherever you want in the document.

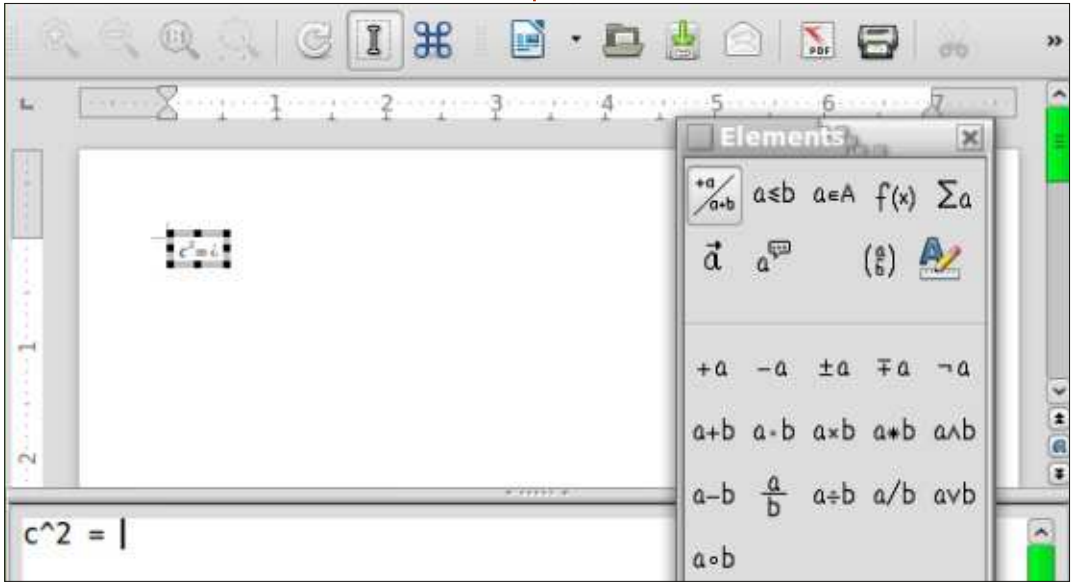
Formula Editor As A Floating Window

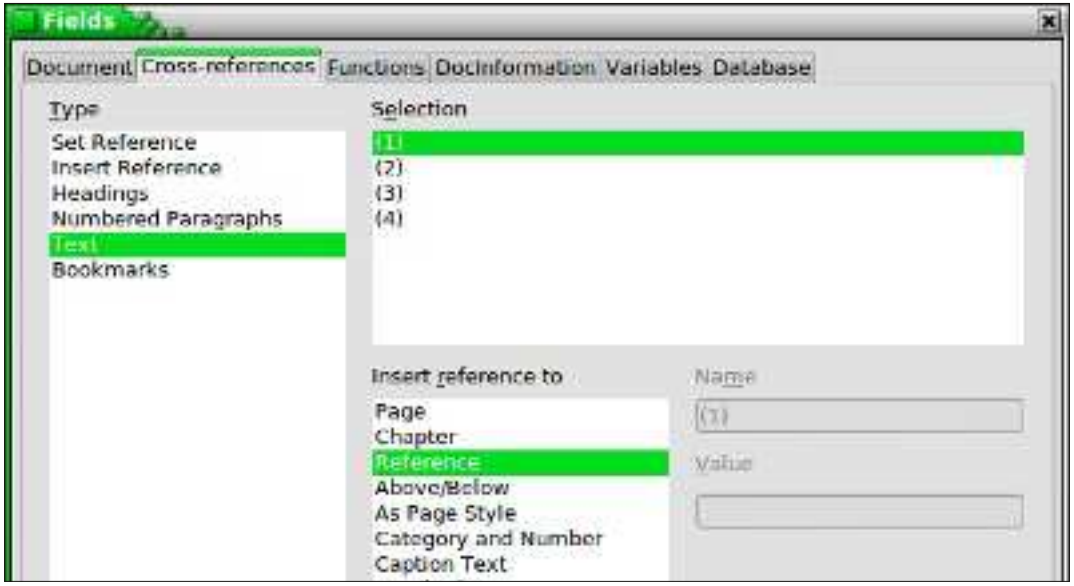
If the formula editor being at the bottom doesn't work for you, you can detach it as a floating window by CTRL double-clicking the border. You can also click and drag the border to detach the formula editor. Once it is detached, you can move it to wherever best suits you. CTRL double-click to reattach the formula editor back to the bottom of the screen.

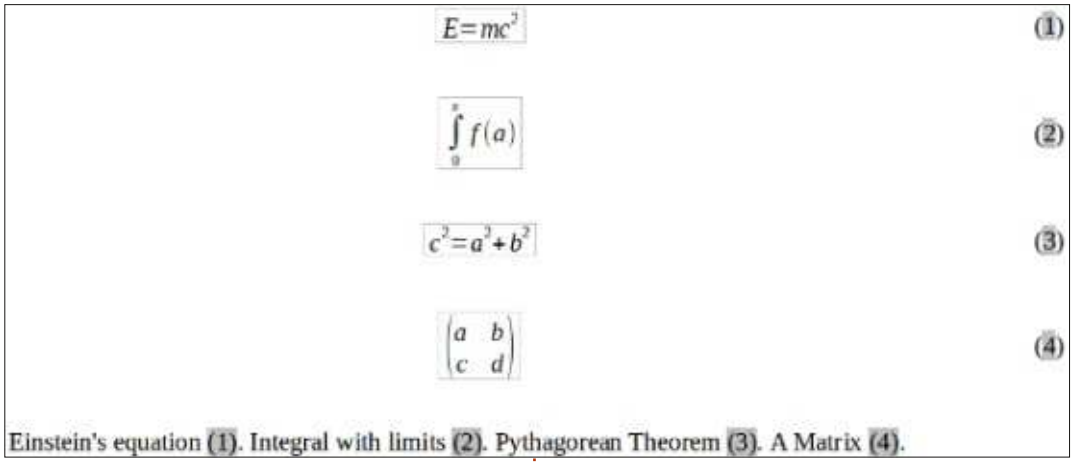
Numbering and Referencing Equations

Many times you will need to number equations in order to reference them in your text. Writer makes this easy for you by managing the references for you. Numbering equations is one of Writer's best hidden features.

On a new line in Writer, type







"fn" and press F3. The "fn" is replaced by a numbered equation. Double-click the equation to edit it and insert your own equation.

To create a reference to the a numbered equation, choose Insert > Cross-reference from the menu bar. Under type, select Text. In the Selection list, select the equation you want to reference. Under the Insert reference to list, select either Reference or Numbering. Reference will include the parenthesis but numbering will use just a number.

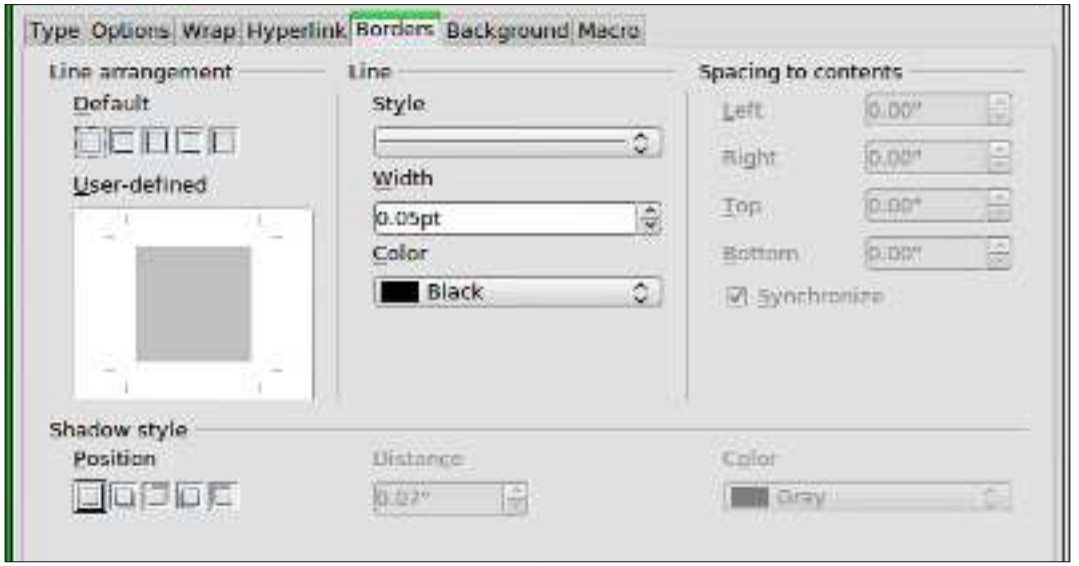
Text Mode

In most cases, you will number and reference your large formulas. Sometimes, you will need to use a larger formula in running text. When you do, use Format > Text Mode while editing the formula. Text Mode will attempt to make the formula fit the height of the text. Numerators and denominators are shrunk, and limits for sums and integrals are pushed to the right rather than top and bottom.

Samples in text mode $\sum_{i=2}^{5} i^2$ and $\frac{x+2-y}{z^2+y}$

Editing the Formula OLE Object

As I mentioned earlier, formulas are displayed in OLE object frames in Writer. This means you can add backgrounds, borders, word wrap,

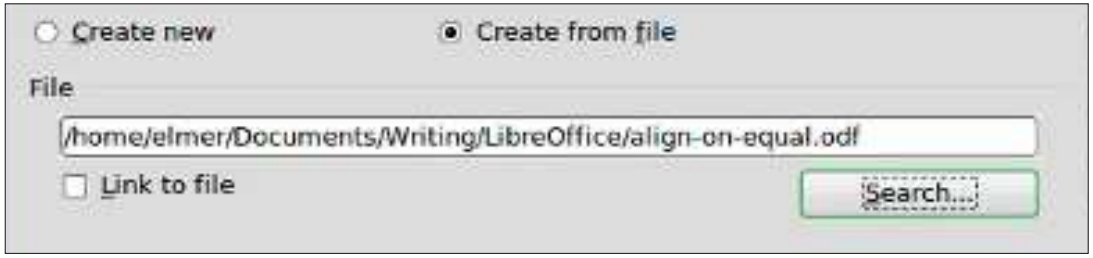


and margins to your formula. To change a formula object frame, select the formula by clicking it once, and Format > Object/Frame from the menus. The object dialog will open. You can also open the object dialog by right-clicking the formula, then click Object.

If you need to set defaults that apply to all formula objects, you can edit the formula frame style in the styles window. You will find the formula style under the frames category of the Styles and formatting window.

Create a Formula Library

If you use formulas often in your documents, you might want to create a formula library. When you save formulas created in the Math module, they save as ODF files. You can save formulas from your documents by right-clicking them and selecting "Save Copy as".



To insert a formula from your library, select Insert > Object > OLE Object from the menu bar. Select "Insert from file" and browse or type in the path for the file ODF file to insert.

Conclusions

Using formulas in Writer is actually very easy, making the creation of documents with advanced mathematics a fairly simple task. You can number formulas and reference them in your text. Using text mode, you can create formulas in your paragraph text which aligns as well as possible with the flow of the text. Change the appearance of your formula by editing the frame of the inserted OLE object. Import formulas you create in the Math module into your document by inserting an OLE object.

Elmer Perry's history of working, and programming, computers involves an Apple IIE, adding some Amiga, a generous helping of DOS and Windows, a dash of Unix, and blend well with Linux and Ubuntu.

QUICK REVIEW: Ubuntu 4.10

by Anas Alsaidy

I was reading that the magazine needs us to help, so I decided to write about my experience with this old distro usually the reviews are about new and modern stuff but I wanted this thing to be new to this magazine.

After downloading the .iso file I burned it to a CD, rebooted my computer and started the CD but I had problems and the CD didn't boot (it hangs on loading), so I had to try it VirtualBox. In VirtualBox I had no problems with booting at all and it worked exactly as I expected.

The first thing that I noticed was the GNOME desktop environment and I really liked it. The second thing was the old versions of modern apps such as GIMP, OpenOffice, Gedit, etc.

CONS:
1. I really didn't like the horrible wallpaper even for an OS released in 2004.
2. I also didn't like the ugly brown theme luckily though there were lots of theme that I could change between.
3. And I had a problem that it doesn't allow to save anything (I think the problem was with my computer).

PROS:
1. it's fast: really fast actually I didn't have any problems with multi-taksing (I will talk about the performance on the next section)
2. I really liked GNOME: I mean here the panels the settings manager NOT THE WALLPAPER AND THE THEME.

I said before that it was fast, because it really is I ran lots of applications and nothing went wrong, except for GIMP. It made the whole system lag and I had to restart the machine, but no problem as I'm using VirualBox. But everything else was very fast.

Ubuntu 4.10 was definitely a great OS but apparently it can't replace the newer versions of Ubuntu. But honestly I did like this OS and it really looked like the the newer versions would be good, and that really is what happened.