

Home	Art&co	GIMP	Inkscape	Blender	Scribus	Galerie	Liens	Contact	Compte
------	--------	------	----------	---------	---------	---------	-------	---------	--------

News Externes
Inkscape 0.43
Libération de Synfig
Scribus 1.3.1
Scribus 1.2.3
Dernière Stable de Gimp : 2.2.7
Scribus 1.2.1
Gimp 2.2.1
Blender 2.36
Gimp 2.2
Inkscape 0.40
Inkscape 0.39
Screem 0.11.
Inkscape 0.38
Gimp 2.0

DADVSI

Signez [la pétition demandant le retrait de l'ordre du jour parlementaire](#) du projet de loi [DADVSI](#) !
Déjà plus de 109104 signatures depuis le 2 décembre 2005 !



Gimp 2 Efficace

de Cédric GEMY, publié aux [Editions Eyrolles](#). Le potentiel de Gimp en un livre (c'est ce qu'on dit). Sortie début septembre, commander auprès de votre libraire ou sur le site des [éditions Eyrolles](#).

[Vos commentaires sur le livre](#)

[Cédric GEMY](#) est graphiste et formateur PAO et multimedia indépendant.

[couverture du GIMP](#)

la] SOS GIMP [A
de/byCédric GEMY

[Accéder au contenu](#)
[Télécharger les images](#)
[Télécharger les exercices](#)
[Télécharger le cours \(mai en cours\)](#)
[Version pdf](#)

[Cédric GEMY](#) est graphiste et formateur PAO et multimedia indépendant.

A Scribus UI analysis

Analyse utilisateur de l'interface de Scribus

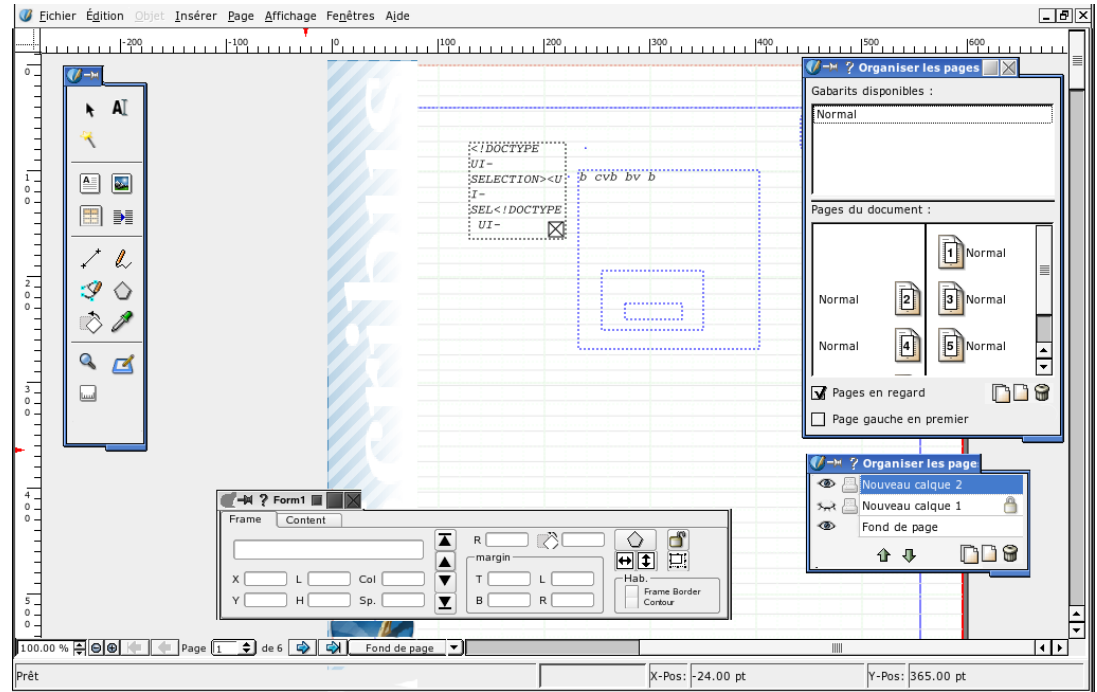


Table of Contents

A Scribus UI analysis 1

Preface 4

Organizing 8

Introduction 8

The Scribus GUI 8

Generalities 9

Menus 17

Place item in the right (sub)menu 18

Grouping related commands 19

Be logical between different UI element 19

When splitting into subitems ? 20

Unused menus 20

The right choice between Menu and Window option 21

Windows 22

Advantages in using windows 22

Transient or permanent 23

Controlling windows 26

Follow the same design 27

Window orientation	28
Tab orientation	29
Contracting windows	31
Find the right control	32
Editable fields content	35
Toolbox design	36
Icons and graphics	39
General consideration	40
Consistency again	40
self-consistency	43
Finding the good metaphor	43
Keep the metaphor consistent	44
Improving integrity	46
Icon colors	46
Icon contrast	49
Icon size	50
3d effect	54
General Cross-Application consistency	56
Icon File Format and management	57
Tools	57
Windows	58
Layer window	58
Text properties	59
Conclusion	60

Preface

Why such an analysis ?

Scribus is soon ready to be considered as professional software. It now has many functionalities and tools that allow the user/graphical artist to create nice and worthy layouts.

But since Scribus is very young software, it still has to convince new users in the same time it is being developed. Most of those users may have habits taken from "concurrent" software (QuarkXpress or Adobe InDesign).

These habits have to be considered while Scribus has to reaffirm its difference. Most companies spend lots of money for that, because developers are involved in computer abstraction, not necessarily the user.

More than that, the Scribus user is not a common user. In general, graphics software is used by people involved in design so that they are often harder to convince. Graphics software also has a particular use of the mouse: not only for moving, but for putting information in documents. This makes Scribus different to Firefox or Ethereal, and that is why general user interface guidelines have to be reinterpreted.

The little story

If you are using free software for a long time, you may be considered as a "E.T." in your neighborhood. Sometimes, you decide to show that you are not so crazy, and that free software really does work well.

But you have to forget you are used to all this, and that your friends are not always. In this context, each little detail will be important. Of course, I also lived through that several times. As a Gimp teacher I have to fight everyday against Photoshoper's habits.

One day, French publisher Eyrolles invites me to do a little conference on graphics with free software. Most of people here were not usual Scribus users, but they all complained about the Scribus interface, and nearly nothing left. Most of them were InDesign users on Mac OS X.

Of course, everybody here knows that the Scribus UI can be "skinned" via preferences but the trouble is clearly somewhere else.

Think of the user

A piece of software, whatever it does, is made for users. Users will give existence to the software's capabilities. Therefore users should easily be able to find what they need most. As Free Software is contributive, a part of this effort can be demanded from users. However,

- since the user has not always the knowledge to help (that's what most of them think)
- since they have to learn a bit of the software before contributing
- since new users come to free software just because they are gratis and not because they are free

Therefore, the user interface has to be attractive. This is more true if Scribus wants to be considered as a professional tool.

For the developer

Thinking of the interface should be as important as implementing new functionality. In fact, what should a new command be made for, if people can't find it or understand what it is made for ?

Thinking this way, developers will find this constraint has a big chance: it will enhance their user point of view and can give information about how to implement their functionality.

The developer should consider these periods to think of the GUI:

- When a new user-visible feature is added
- The application gets larger
- Inconsistencies are found, particularly when new version of the UI libraries (for us, Qt) are made, or when a related software is evolving (for example, Inkscape or Gimp)

After that the developer will have to consider how to choose the UI type for his feature:

- Can the user act on it ? So it has to be visible.
- Is it related, in use, to an existing feature ? If yes, then is it a new property of it ?
- Is there already a quick access for a related feature ?
- Will this feature change many things compared to the existing ones in terms of quantity (need a new group in the interface such as window or sub-menu) or type ?
- Are there any other applications with a feature with the same functionality? In this case, how is it implemented there, and has Scribus to follow it ?
- Is that a useful feature, or just a minor one ? What % of users will use this feature: if more than 80% how do they want to access to it ?

I could go on. I'm sure most of Scribus developers already have these things in mind, they could not have done such a good software if they did not.

A most complete list of question to have in mind are available at:
http://developer.gnome.org/projects/gup/templates/fui_templates.html.

So how did I do this study?

Very simple. There is no theory here. I've taken 3 ways to analyze the Scribus UI and try to understand its structure and detect what could be changed:

- Put some of my trainees on Scribus and observe how they worked, especially, the little things that made manipulations heavy (some of these have used Quark Xpress for years);
- Ask some very simple questions. Something like: what you do in a software if you wanted to do ...? This reflects the way they are thinking with a typical GUI;
- Compare Scribus to other software, alike (InDesign or Xpress) or not.
- Think of a FreeDTP integration with Inkscape and Gimp.

Please...

These are only observations, and never a denial of Scribus quality. If you are not used to Scribus or don't know it, take these observations with care, this is a working document, an inner draft to help improving the software.

So what will you find here ?

1. Observations, based on real Scribus use, especially with non Scribus users (they have a virgin glance)
2. Explanations that I made as clear as possible
3. and sometimes proposals.

I just hope it will be clear and useful. If you have some remarks on this document please contact me at cedric@le-radar.com or [pygmee](#) on #scribus at irc.freenode.net.

Cedric

Organizing

Introduction

When a developer adds a new function, he may have his own idea on the subject. He often has a good knowledge of what this command is made for. He wants it or may have listened to a user's requests. That way, new menu items or new icons are added to the interface.

And the more items are added, the less the GUI is understandable by a beginner or a new user.

The more items the GUI has, the more complex it is.

The more items the GUI has, the heavier it will be to organize future functionalities.

The Scribus GUI

The Scribus GUI, as many other, is made of several parts: menus, toolbars, windows. These parts are made to give fast, easy and comprehensive way to change parts of a Scribus document. But some difficulties can appear.

Xpress users will recognize very easily that the Scribus menus have many things to compare with the master application of Quark. This is a good way to help those users to get rid of Scribus but these also have faults:

- The Xpress UI was very efficient for a simple application, made for professionals (most of them typographers): is it so efficient for beginners or new graphical artists?
- The Xpress UI doesn't follow new UI guidelines: for example, many things are hidden to the user and many commands cannot be used if they are not known, just because they don't appear in the UI. But that's also because it is so fast for accustomed users.
- Many people will say Xpress UI is too austere. This may not be important but is enough to reject.

Scribus will inherit from Xpress advantages and also its defaults. But the worst may be that Scribus is also including parts of Adobe's UI system, which is different. This is introducing some inner inconsistencies that have to be resolved.

Trouble is that Adobe InDesign is also growing as a source inspiration. This can lead to inner inconsistencies that we will try to point at, since Xpress and InDesign are not designed the same way. My purpose is to help in the process of finding a Scribus way.

Generalities

We will deal here with general behavior we have observed. These notes can be considered as basis for further ones.

See&Point

Users can work in 2 ways when they want to do something. For example, they want to have bolded text in a page (a title for example):

- Accustomed user will think to have bold, i need a text, to have text i need a frame, to have a frame, i need a page.
- But other may not: they would like to click in the middle of the page and write directly.

A guy I met one day, a journalist knowing DTP very well, told me that he needed few seconds to understand how to activate grayed icons in Scribus.



Context is certainly the best thing to use but it also has to be handled with care. In most cases, people will select them and then search for related options in a menu, or a window. These options have in this case to be easily reachable and understandable.

Consistency

Many users won't know exactly what they will be looking for. It is in fact impossible to have all the Scribus features in mind :). So the features will have to help the user memorize where he can find related options or commands. This can be done in 2 directions:

- Use general habits, for example Ctrl+C often copies, but not in bash! Following this method makes you keep the project in mind and the context in which the function will be used.
- Organizing the inner functions in groups (windows, menus...). Most of specific functions have to be considered like this.

Generally, consistency should be very strong when using toolbars and shortcuts.

Follow design habits

Usual elements of a UI should be considered in their traditional ways. If you have the idea to change the use of an element, the user may be frustrated. Just try to know why you need this modification and look at other possibilities.

For example, Scribus has a so-called Toolbox. In all software and especially in graphic ones, toolboxes give access to elements that allow direct action on the screen, without an intermediate dialog. Tools are often completed with a property window when they need more parameters to be used.

The toolbox, should not be placed near the toolbar. The toolbar give access to general functionalities and often not software specific.

The toolbox should be floating and be dockable.

The toolbox should not contain options that make dialog appear. If the dialog is tool related, use the property window or a modifier to display it (often, Alt+click).

Toolbox and toolbar controls should be placed in the same category and order as there are in the menus.



Actual Scribus Toolbox.

Vertical but resizable and dockable toolbox that could be placed as a left bar at launch: this could provide at better integration with other apps: OpenOffice, Inkscape...

If Scribus wishes to be considered as professional software, toolbar could be deletable by default.

Productivity

Accustomed users will like quick access. There are several way to do that:

- shortcut (Xpress for Ctrl+Alt+Shift+F to constraint a picture in a frame with no other way to do it)
- modifiers (InDesign: Duplicating a frame with Alt+drag): we could have for example Ctrl+letter+UP to increase character size.
- permanent windows (InDesign)

Nothing is best. With Xpress, you have only what you need in screen. The most important thing is the document. It is designed for professionals with years of practice. With InDesign, everything is easily reachable but it can overload the screen.

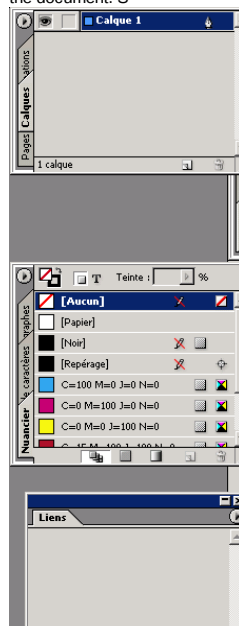
Each of these ways have to be considered when adding a new feature. The choice will depend on Scribus aims (professional or not) and there should be at least 2 ways to reach a command or option. Only in this case, Scribus will be able to be seen as adaptive.

Helpers

As Scribus users will have to do precise documents they will need helpers for that. Scribus already gives rulers, guides and such related items.

Others could be implemented such:

- have a frame magnetism or block frame superposition
- have a fullscreen working mode to allow to work without menus and windows, only the document. S





ome palettes available in InDesign: they are many, maybe too many.

There are many things like that in feature requests and the development team can already be thanked for being attentive to them.

Usage and progressive disclosure

Why should the user go again where he has already been ? When there are too many options (example menu items), some that are not often used can be masked. It's up to the user to display them.

Below: Two screenshots of Gimp's New Image Dialog

In this case, and in general, already used options and commands should be more reachable. Especially if they are user-defined such as custom colors.

Progressive disclosure is very useful for beginner or non-regular user to help them considering most important command before working in details. Accustomed will most often have constraint the software to their habits.

Progressive disclosure is recommended but my experience shows me that people like it in dialogs but really not in menus.

Add explicit items

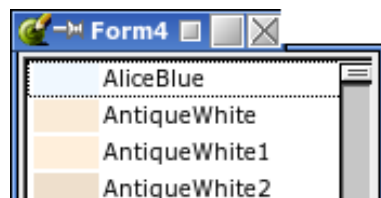
UI elements should be explicit and immediately recognizable and understandable.

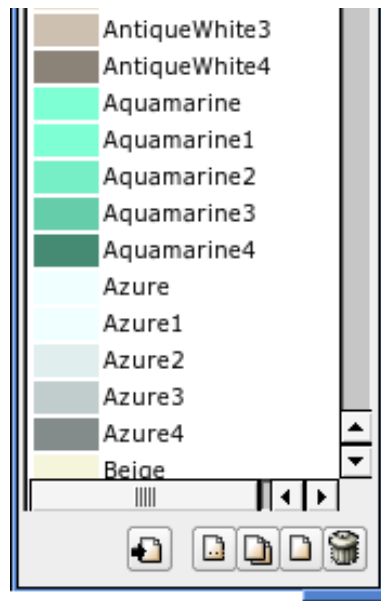
This implies that icons should be used as often as possible, even on buttons. Icons are much faster interpreted than text items if they are sufficiently different and well designed.

This also implies that labels have to be well chosen. For example, The Color Window has "Append" and "Add" buttons. These labels could be:

- changed into icons
- at least have more explicit name: "Append" may not be very comprehensive for non programmers. They will often prefer "Import" regardless to the preciseness.

III-color-window-new





A new color window. In this model, Edit button could be replaced by a pen picture. Modifier on Import button could give options.

Menus

Place item in the right (sub)menu

Where can i find the windows to use colors ?

Choosing the right place for a new control is not easy. A new function can be related to several existing ones.

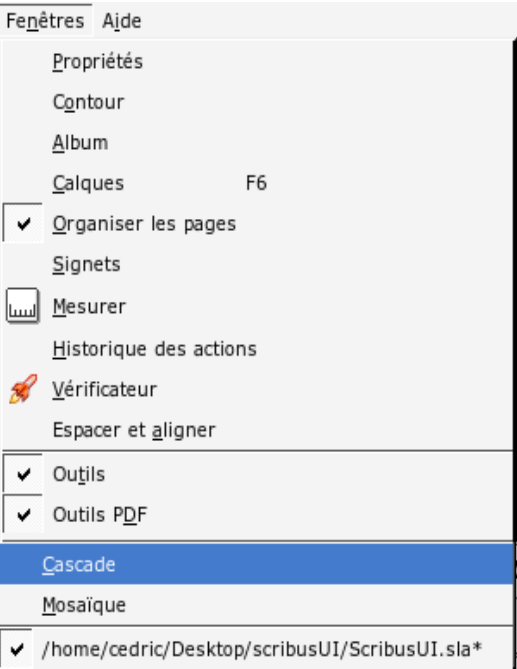
If it is possible to have different kind of access to a function (menu, icon, ...) never use several times the same type for the same command. For example, avoid displaying a single item in different menus.

Especially when the developer have to chosen between software functionality and standard menu.

For example:

Standard UI have a Window menu entry in common. In Scribus (as in OpenOffice), this entry contains document windows display options. In other application (Adobe's, Gimp), it also give access to several tool-related palettes. Other entries are placed in *Edit (Color...)* or *Tools (Property...)*.

As it is a standard in DTP softwares to have a window menu giving access to the great parts of software palettes, Scribus should follow this behavior and move in the same menu entry, i.e. *Windows*.



Proposal for a general window menu containing all non transient windows.

Grouping related commands

Is there no way to add bookmarks to a PDF ?☐

As a matter of fact, more has to be done to organize controls. if it is possible to have different type of access for a command, related command should not be put in different part of the user

interface but at least be grouped in one.

For example:

Scribus has several advanced PDF tools. Some are available in *PDFTools* bar and other in the *Object>PDFOptions* menu.

All of these controls should be put in one place, whatever it is.

For usability, the PDF Toolbar would be a better place to put PDF controls. It will be easier and faster to reach these controls.

As another example, we also think Paragraph styles should be grouped into categories. It is not rare to have more than one thousand styles for a magazine. In this case a drop-down will be too long. Each category could have sub-element, to organize style access.

Be logical between different UI elements

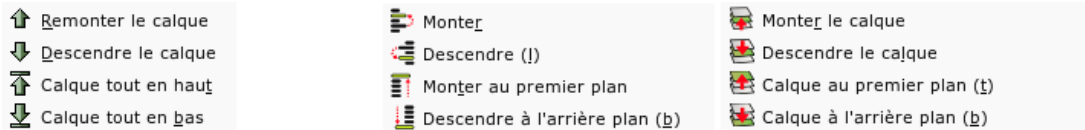
When a command is reachable via different types of control, it is necessary to the increase their identity by using same practices or elements.

For example:

Scribus gives access to frame superposition in the *Item>Level* menu and also in the *Property>XYZ* window. Icons used in those places are not similar even if they do the same action.

Inner inconsistency for Object stack order

Inner consistency can be expressed in different ways. It is necessary to keep things logical and as simple as possible.



Cross application inconsistencies.

A nomenclature of icons and related commands and menus should be made for user purpose. This could help to improve UI if possible.

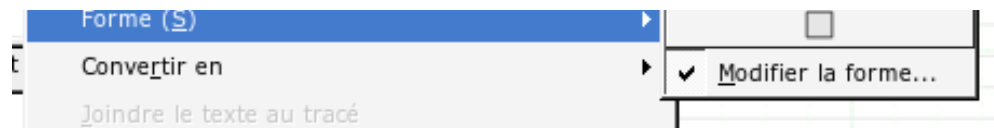
When splitting into subitems ?

A user interface needs to be understandable, but also easy to memorize and fast in use. In this case, grouping in submenus can be considered as a way to organize and enhance usability. But groups should not be too small, and should rarely contain less than 3 items. In other cases, commands will be put in the main menu, separator above and below.

For example:

The Menu *Object>Shape* as only two items: one to choose a preselect shape, another to edit the shape. More than that, the shape selector also has submenu.





Shape menu contains too few options to be kept. The option should be available another easy way: double-click, Ctrl+click and it is already available in Property window.

Every menu should contain a text label. Users are not accustomed to iconified menu items. Traditionally, icon use in menus is just a complement of the label, to make it more visible and recognizable.

Unused menus

â€œI would like to define style for text but i can't click on itâ€.

Scribus already deactivates menus and generally options that cannot be used with actual selected object. It should be much better if those element were completely put off. User don't have to be tempted by unavailable items.

The right choice between Menu and Window option

Generally put items in windows that:

- needs to be contextual
- is part of a whole (for example typographic options)
- give preciseness to an item.

Put in menus:

- actions that are not object oriented
- what is general on the document or Scribus
- what is important, because it should be in several places
- needs to be permanently available

Windows

Windows are very important in interfaces. They are a very common way to give interactivity in the software. If menus are a way to command Scribus, windows can be considered as the royal way to talk with it. Windows can have very different kinds of controls that make it more usable in many cases. We will discuss about these in this section.

Advantages in using windows

Windows or palettes are excellent ways to display several options at a time. The user can have an overview of what is available, and of what he can act on.

In most cases, windows are more usable than menus:

- they are closed only when needed by the user, so that
 - he can change things several times (a test is called: user forgiveness)
 - he can change several things in the same time before applying
- they are easier to point with the mouse
- and so on

So, what can be put in a windows has to be.

Alert windows are very common and useful for feedback. But we are not here talking about them, only production windows.

There are already two ways to implement windows use:

- Xpress, is very light. Only few permanent with a preference to *Property*. Other windows can be called menus or shortcut. They are of 3 types:
 - stay-here window, such as *Page*, which is opened until it is closed
 - open-external, such as *Album*, which is open until it is closed but displays foreign data
 - option-window, such as *Paragraph*, that needs to be closed to allow to continue (modal)
 - info-window, such as Usage for Image and Type control.
- InDesign make every window persistent. Everything can stay on screen and available. But this is sometimes too much for many users. The screen (almost if it is small) can be overloaded. In this case, the user spend time moving them. To improve that, Adobe has added edge-dockable-window. We'll come back to this further.

Risks are where important things stay. The *mpalette.cpp* condense the most important options for the user. It should be treated with a particular care.

Transient or permanent

I would like to add many PDF links. Should I display a window each time and close it or can I keep on screen ?

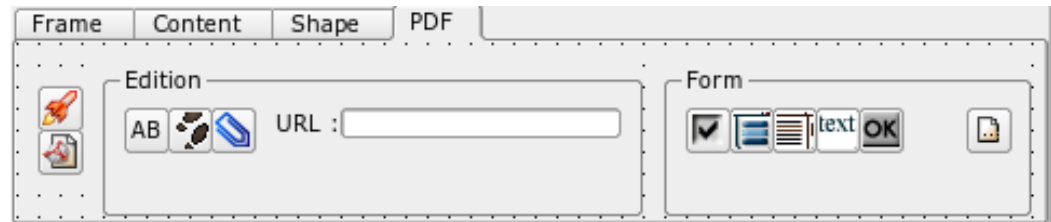
In this context, a palette is opposed to a transient dialog. A transient dialog can be defined as a window that only appears in certain cases, often when a button or an option is clicked. A transient dialog's purpose is to display more precise options. Transient dialogs can be used when punctual information is needed for example in Scribus when a table is being created.

For Example:

Scribus PDF objects can be customized with a dialog. This dialog is blocking (modal), so that nothing else can be done during the editing period. Other actions should be allowed and access to options keep permanent.



Actual PDF window state displaying link options.



Adding a tab to property with PDF options such as Export, Preflight, Objects and objects properties.

In our example, Form controls have too many options to be displayed in a Tab. Transient window should be kept for them. This is not a problem because forms are often one or two pages, not as complete PDFBooks that have to be done in a longer period.

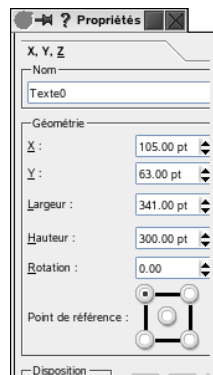
Controlling windows

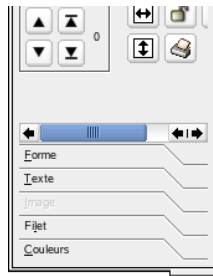
Windows can have different property. Most of them, except document window in an MDI, should not be maximizable. This is a way to prevent unwanted actions, doing something like this:

Going on this, the windows should not be resizable in both directions:

- vertical should be resizable vertically
- horizontal should be resizable horizontally.

It is in fact hard for the user to find the right size. If too big, empty spaces appear. If too small, a scrollbar that is not welcomed in windows full of controls. This loses space, and add some noise to the window. The user has to resize the windows to be able to have an complete overview of the content.






Horizontal scrolling is always a bad thing.

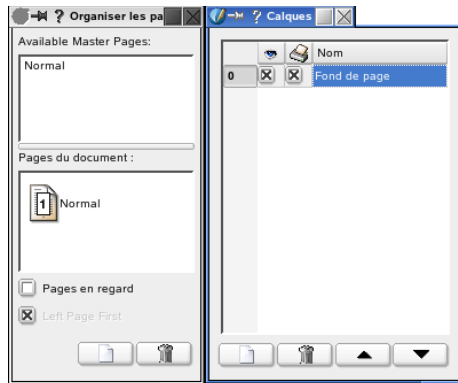
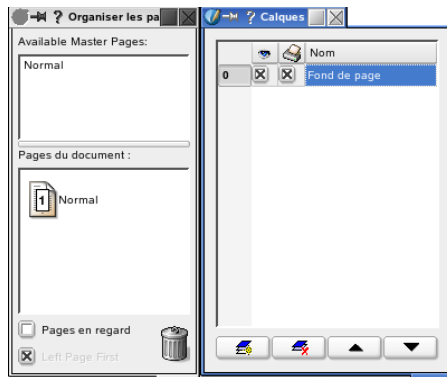
Follow the same design

When an interface is growing, it is getting complex for the user to find what he needs. Regularity is one thing that can help him.


For example:

Layer palette has a  command but not *Page* palette. Windows should have the same design to enhance immediate comprehension and accelerate the discovery of a command.

If windows have related similar actions, they should have the same buttons.



One simple proposal for Layer Window.

Here we can see that both windows have a  action. But they are not displayed the same way: one is an image/icon, the other a button. So we correct this and have modified the Pages windows to suit Layer window.

Note: we have another proposal for this window at the end of this document.

In the same kind of subject, all the design properties should be constant, such as margins, alignment, font family, font size, font color...

Window orientation

Just like at the eyes of your girlfriends, if you an eye above the other, drive fast to the hospital or to a psycho. The human visual field is horizontal, we are made like this. Of course, we can look at vertical things, but sky-scrapers give us torticollis while a landscape is comfortable

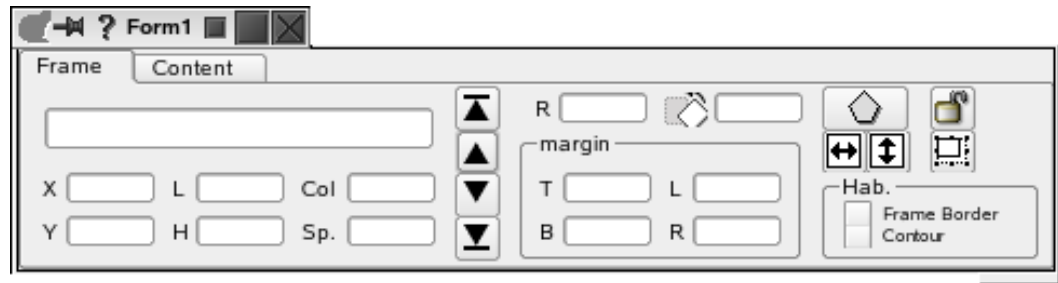
(following the horizon).

So when think of an interface element, we should prefer horizontal. Screens are horizontal (and now even 16:9 which is much more), alert dialog 99% are. Why not control windows ? They should too especially if there are many controls inside because one often reads horizontally, from left to right or right to left, rarely vertically.

For example:

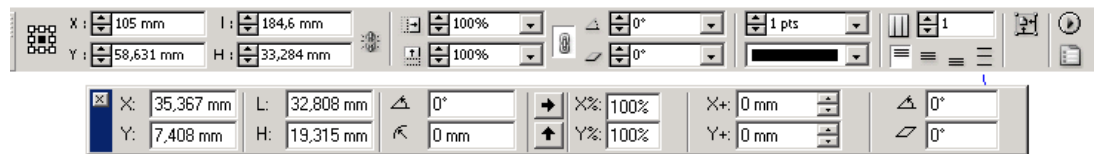
In Scribus, the Property window is displayed vertically. If the user can resize it, it doesn't change the position of the control. So the user has no possibility to get a horizontal one.

I have replaced QSpinBox by QLineEdit to save space



The more a window has control to display, the more it needs to be horizontal. The example above may be improved but seems to me much more convenient. Just look at what is done in concurrent softwares.

III IDQuarkProp.png



Horizontal property windows as used in InDesign and Xpress. But same is already used in Inkscape, for example.

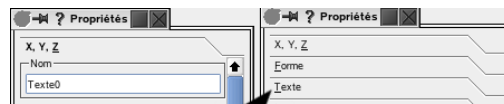
When the controls are really too numerous, a vertical can be used. In this case, controls are often display in lists, like this: III scroll_list.png

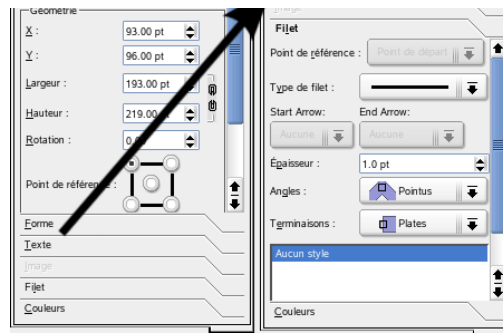
This is much more comfortable for the user and we'll see below, often easier to organize.

Tab orientation

As an extension to the preceding section, tabs should not be displayed one under another but aligned. This is to provide static places, much more usable for the user that can be fed up looking for the right tab.

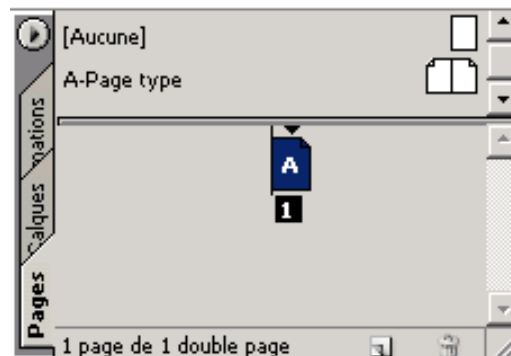
In the picture below you can see that Text tab position is modified when the tab focus is modified.





As an example, just have a look at what has been done in InDesign:

- classic horizontally aligned tabs;
- true vertically aligned tabs (shown above).



To conclude, a tab should have one position and only one. It shouldn't move in the box (Just think of what should be the interface if everything was moving like that: menus, icons... just as stairs in Harry Potter:)).

Contracting windows

What is important for the user? The need to have more space on screen. All other things have to:

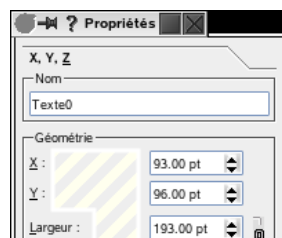
- be small and non-attractive
- be hidable.

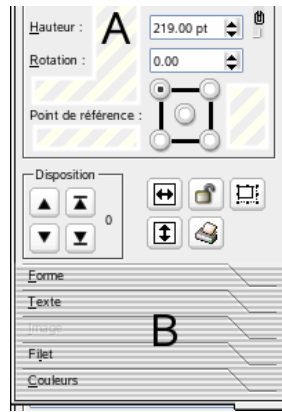
We have already told about the opportunity to resize the window to change the used surface. But we haven't talk about the content space.

In fact, some of Scribus windows contains empty space. This is in itself not so important: empty spaces gives air to element and is a way to organize and make groups more recognizable. But we shouldn't forget that the most important thing is the document. So each pixel placed in a window hide a pixel of the document.

For example:

Scribus Property window (again but it is so important and contains so many things that we couldn't miss it :)), is very large comparing to InDesign's. Fields are very long and empty spaces can be seen in different places.





In the suggestion above, we have put object-building information in a group. Fields are a bit smaller and labels are all abbreviated (same rule for similar items). Then condense Rotation fields that was too large. Finally, we have grouped tabs at the top.

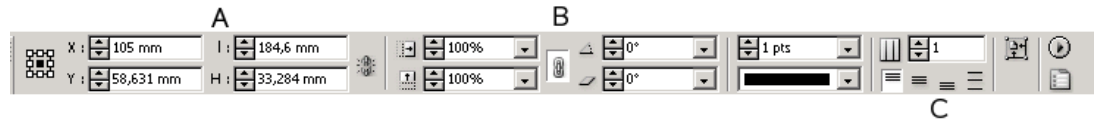
Other proposals for this window can be seen at other places, following the subject we are dealing with at the time.

Find the right control

It is not easy to choose between all available controls in Qt. Before any implementation, the developer should wonder how his control will be used ?

- is it for a novice ?
- does it need to be a very precise control ?
- what are the alternative ?
- will some value be often chosen ? If yes, which ?
- etc...

When the value has to be very precise, the user will write it directly. No need of helpers.



When looking at InDesign controls, we can see that they can have multiple behaviors:

- A: X, Y, L and H are highly customized value. They are related the the position and size of the frames. The user will use the tool to draw the frame, the spin to adjust it, or write inside to give a precise (rounded for example) value.
- B: Second group controls will have increments: the user can have the same behavior than with the preceding but he may also use predefined values that will be displayed as a drop-down list.
- C: We have only predefined values, and because they are not too numerous they can be displayed as toggles images.

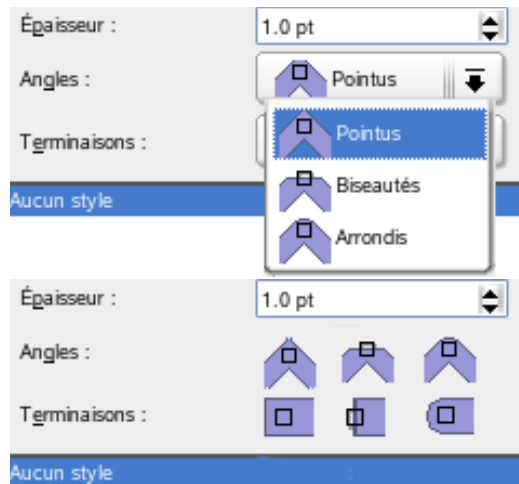
A such complexity is not necessary. Adobe always want to do software for any users. This is their success. But most of these possibilities are not known buy most users. Do Scribus developers have enough time for this ?

Since I'm not a a developer, I'm not sure Qt allows to put multiple behavior on one field.

Avoid drop-down lists when they are not necessary. This is true when the user won't have many choices.

For example:

In the Properties window, go to the tab containing Line parameters. Angle and ends lists is only containing 3 items. It should be possible with no more space to have a more efficient control.



Troubles with lists is that they:

- mask other properties
- don't display every item at a glance, i.e. the user has to make an effort

- as we can see here, there's trouble with the icon which shows a truncated and not a plain as it should (the picture is certainly good but too big to go in).

Using simple icons or buttons, we are sure to kill the two first troubles. Then, just have left to use pictures of the right size.

Buttons of different fields should have same size if possible and if they are related to the same kind of property. Pictures should also be created at the same size.

To end this section, we would like to talk about the opportunity of putting text and only text in controls. Anybody knows that text is not the most easy to read and is not as universal as picture. When possible, a picture is better. But there is another case where text should not be used: when it is placed after another control.

For example:

In Scribus Property window, Text tab, color can be defined, and with it the tint quantity. Trouble with that kind of organization is that the user may think at first glance that the `100%` is a unit or a label. Only a click on it shows it has a real effect.



In this case, this problem is added to the preceding. Of course, there are too many possibilities to display them all in buttons or icons. In this case, two alternatives seem to be more usable:

- add a QLineEdit in which the user will possibly write the value, or a drop-down with default values but editable for more preciseness.
- a slider as shown.

Best practice is certainly to have both: Slider+SpinBox as it is done in Opacity Layer Control in the Gimp. But in this case, the 'group' made by these two should be clearly identify at least by a label which increases the used surface.

To avoid partly an understanding problem and fasten window reading, labels should always be displayed before the control.

Editable fields content

Some fields, as QLineEdit or QSpinBox, are editable. The user can directly write the value he wishes regardless to the default values, or the listed ones.

It's evident that lists values should be the most common used.

In this case, he can also write the units he prefers. It is up to the application to understand the needs and calculated it in a inner unit. Sometimes, this is not working .

For example:

In the Property window, Text tab, the baseline field has a . But the text size is given in points. If for any reason, a user wants to write in the baseline SpinBox, it won't work and nothing will be done.

When a unit cannot be modified by the user, and that the user really have to use a predefined one, it should be written after the field. That way, the user understand he cannot change it and that he has to do the calculation himself or test possibilities.

Another lesson from user observation shows that they often don't like large fields. It seems that, in this case, this gives a non-professional aspect. To avoid this sensation, we propose to use these properties:

- field height: 12 or 14
- field width: 40 for LineEdit, 52 for SpinBox
- field font-size: 8px

Some fields can be larger if the content needs it but not smaller unless it won't be accessible enough.

Since icons should not be higher than the field, max size of icons is to be 16. But a bit less should be best.

Toolbox design

To mark its difference, the toolbox is often displayed in a different way. In Adobe's softwares as in Xpress, and also in Inkscape, it is made vertically. In this way it can be quickly identified and not be mixed up with Command bar or Options bar.

Where are tools? And I can't see any entry in window menu to display them !

The enhance Toolbox box use, it should not be consider a quick-access-menu. As we have already talk about, a toolbox is design for a very different thing to improve usability in graphic softwares.

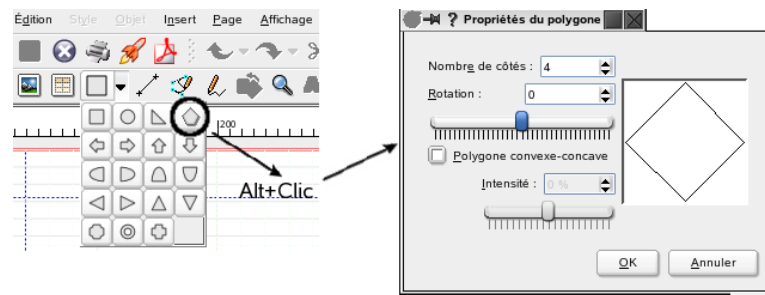
Toolbox display in some graphic programs

For example:

The Polygon tool in the Scribus Toolbox as a submenu which shows a specific property dialog. Two troubles are resulting of this: why should a shape be separated from others ? and why display property here instead of using Property windows.

menus should be used to group related command or object, not as shortcuts.

To reduce troubles introduced in the example, we would like to put the Polygon icon with other shapes, and , to keep the transient dialog, use a key modifier to make it appear.



Strangely, the Polygon shape does not appear in the Shape Tab of the Property window.

In both software just see that there is no command bar.

Icons and graphics

Words are very complex things:

- They all use a limited of shapes that we call letters.
- It is the order of these shapes that make sense.
- It needs lots of attention unless all items look the same at first glance.
- It needs time to point at.

As we have already said, it is much faster to work with icons. This is especially true when a user is accustomed to a software. Beginners will have difficulties in both cases, because they have to learn the meaning of the graphics.

To be efficient, an icons needs generally:

- To be different from others for easy recognition.
- To follow a common line for easy memorization.
- To keep the visual aspect clear, avoiding parasites and noise.

These 3 needs are evidently in a sort of opposition. Creating icons is a very difficult task. Sorry, creating good icons: clear, understandable, in purpose, and pleasant.

I'll discuss here of Scribus icons. As it is much easier to talk about than to draw them, I would like to apologize not to do as much as other contributors did. All my respect to them and I hope my thoughts will help them to improve their work.

General consideration

Creating icons is a very special part of graphic design. As difficult as a logo to conceive, an icons has to express as simple as possible a precise object or, more difficult, an action. The design of the icons as to be unique, different enough for other to improve recognition, but not too much to avoid misunderstanding and misinterpretation.

Don't forget icons will be displayed in place of words. But if an icon is a representation, a word is not less.

Some people like text display. Preferences should allow to set 3 types: icons only, text only, both.

Consistency again

Cross-Platform consistency

One of the difficulties we may find is that an application needs to be integrated to Window Manager design. That's why manufacturers as Apple, Microsoft, Sun and other do guidelines to help developer to build a standard UI.

LINUX HAS A GREAT PROBLEM: (hehe). There are many different window managers, and some of them are highly skinnable and customizable. So what place is left for the graphic designer ? Not much.

SCRIBUS HAS ALSO A PROBLEM: (hehehe). It is so good that it is now working on different OS so that it may also match their guidelines. In this case, icons are certainly the easiest as it they are a very visible part of an UI.

Plans could be like this:

- Each port of Scribus should have a set of icons matching the guidelines of the corresponding OS. This is to avoid habits for user that only use one OS.
- These icons could follow an internal guideline or template(s) and be kind of derivations of these. This is to help people using Scribus on different OSes.

Cross-Application consistency

Does anybody know other free software ? hmm. There are so many. I'm sure you're using some. May be Inkscape for vector drawings that can be perfectly imported into a Scribus Picture Frame. Or maybe GIMP, OpenOffice or whatever you need. I'm sure Scribus has common commands with those applications: File>Open, File>Save, Edit>Undo are used for a long time in software so that their icons have now typical properties. They have now a kind of standard design.

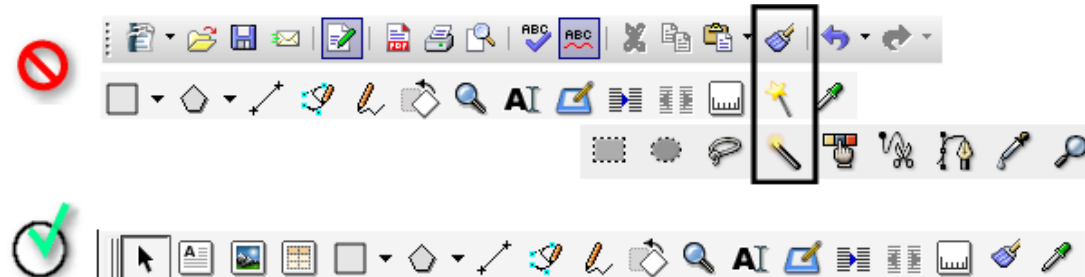
If other UI materials are very tiny, icons seem to be much more present in user spirit so that special attention must be taken to their similarity.

“I click on this icon to create columns but it doesn't work.”
“How can I put lines on the same basegrid ?” =>
questions related to link frame icon

It is impossible to change a 20-year accustomed user. So Scribus has to follow the same design.

For example:

Scribus has a magic wand icon which provide easy access to aspect copy/paste. The chosen icon may cause problems: it is use in other software as a selection tool. it would be better to have a brush in the same order at this one used in OpenOffice.



a more convenient way has to be found to represent this new tool.

But some other commands are not so universal: Adding a layer, Text tool, Draw shape... The question is if Scribus has to follow their designs. Of course not. But it could be a good thing especially for software that have to work together and that's the case between Scribus+Gimp+Inkscape. This would help to pass from one to another without becoming schizophrenic.

For example, look at how easy it is to work from Illustrator to InDesign. Similar command, similar icons. A very easy way to enhance productivity.

Trouble is:

Scribus is QT+KDE design

Inkscape+Gimp are gnome applications.

If general guidelines are sensibly similar, Gnome and KDE have in fact very different graphic styles. Especially, Gnome has very precise guidelines including use of palettes.

Scribus has some difficulties to get its own icons. The temptation is great to take them from other applications. But this may cause some trouble we'll talk about later. This should be done very rarely.

For the same reason, Scribus also integrates icons taken from other applications. The imported icons don't suit the initial use. This is to avoid expressly as soon as possible.

For example:

New version of Scribus has a preflight verifier. This is a very useful tool and we really need it. The icon chose for it is quite good: a rocket that takes off. A good homonym of what is preflight. Trouble is that icon is a KDE default for “launch” action. This may cause misunderstanding and make people think you can launch another application from within Scribus.

The user of the “flight metaphor” is OK but not right. Preflight Verifier purpose is not to make document fly, but check PDF conformity. Something like this should be better in meanings, but it is not very attractive and readable only because PDF icon is a logo, made to be complete in itself (everything added is too much):



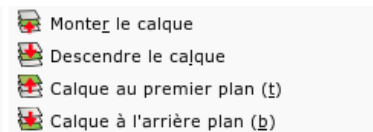
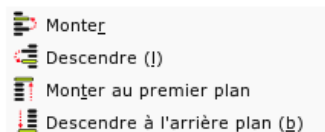
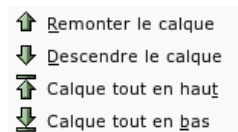
self-consistency

The use of icons have to be logic within the application itself. This in two main ways:

- designs for related commands, options or objects should have same approach
- a command should have the same icon everywhere

For example:

Stack commands don't have same icons everywhere. The screenshot below shows button in the Property window and icons displayed in the Object>Level menu. This should be unified, surely.



To unify, a choice is to be made. But which one? first icons are more consistent with the Gimp and the second with Inkscape. My preference to the firsts which are more simple, related to standard UI design.

Inkscape design seems not to be very consistent too. Look at the 2 ways to modify stack, the first on object, the second and layers. In both, an arrow shows the action. In fact, only the arrow is meaning something.

Finding the good metaphor

Don't try to be literal: think of the functionality. Users are accustomed to UI organization: they knows commands are groups by similarity of their action not by their name. That's why creating a new page is not in the File>New menu.

Icons have to work the same. They have to visually show what they are made for: saving is a floppy (from the time where floppies were used as hard disks) even if now we most commonly use CD, DVD or any other recent peripheral. But it is not a good idea to make a document fly :).

Several approaches are possible:

- icons as preview of the result: alignment in OpenOffice
- an object represents an action: for example, a bean means delete.
- a detail for the whole: question mark for ask your question (help)
- syntagmatic signs: arrows for up or down.

To keep consistency, it is sometimes necessary to make a choice.

For example:

File>New document and <Layer>New have different design.
When does the action prevail over object ?

In some cases a same design can have different meaning: for example an arrow turn to left can mean:

- ⤴ in editing software (here the arrow is often curved).
- ⤴ in editing software (here the arrow is often curved).

This has to be done cautiously as it may cause misunderstanding for uncommon icons.

We come back later on the context of icons

Keep the metaphor consistent

So we can see there are two main risks in icon design:

- to have similar icons that represent very different actions
- to have an over-detailed drawing which is perturbing the recognition and increasing cognitive overload.

Less is more but less has to be enough. Over-detailed icons have to be simplified step-by-step. But be careful not to introduce confusion.



in the above example, the left drawing could mean "home". The second also, but it should be more difficult to recognize. At least, the third is now going to mean "UP". Completely different.

Drawings should be very precisely defined, keep recognition and avoid likeness to no other related object.

For example:

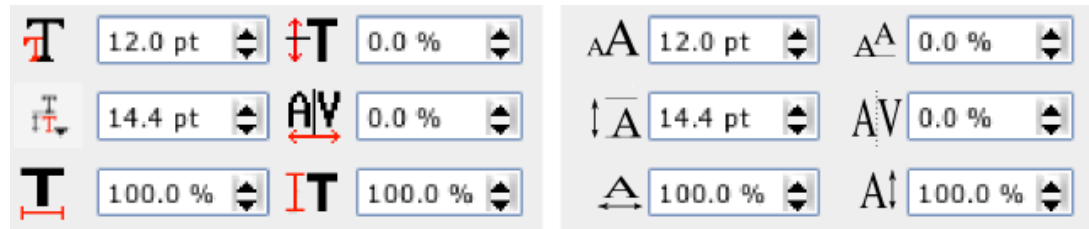
Typographic icons in the text Tab of the Properties window are great new things. But we can meet some trouble with them: all letters are not designed the same way; letter (T) is certainly too geometric and can cause confusion with other graphic application icons such as in CAD softwares.

Egyptians were drawing the most expressive view of an object. Keep that in mind. For example, a piece of paper, used for new document, is important for its surface, not for its thickness. So we show the surface.

Once again, the solution is to use a very common and well known shape, as differentiated as possible. Here, the designer chose T for Text. But he could also have chosen A, or ABC or AZ or any other letter. T was the most difficult. A would have been a recognizable letter in any situation. Plus, A is the letter used in the tool icon in the toolbar. If T has to be kept, it should design with more typographic attributes: for example, it could use a serif face.

As an example, InDesign uses T for character properties (this is related to the tool icon) and lines for paragraph

properties.




Here, we have used A letter, to refer to the letter user in the Text Frame icon and to make it lighter. Showing it following other rules we have already expressed before, it could be something like this:

Some options have been added but other still missing. As a matter of fact, this Content tab should also display Image property when that kind of frame is selected. But with new future options things will have to be changed here. But may be this is the way for icons, except for Edit one that should be improved.

If text properties go on that way, it should be necessary to think of split into 2 different tabs: one for character, one for paragraph.

Other changes made are relative to red markings. In fact, if I was saying above that some of them make me think of CAD icons, it is certainly due to these parts. Why? Just have a look at scale icons. Scaling is express with an I. This design is easier understood as a dimension as should be on a map or plan. I prefer arrow to show directions that things can go, and in this case, grow. Maybe the Up arrow should be enough for vertical and right for horizontal. Other tests have to be done to give more reliability.

Other icons have been added, associated with styles. Please look well at the Edit one, that use  as a meaning show properties. We thought that using system icon for that was too much because this is a very special part of advanced text applications.

I say text, we can see that in other applications, not only for laying out. OpenOffice is one of these.

Improving integrity

A UI has to be recognizable at first glance. Icons are not only for use, they should be designed very specifically for program specific commands. So, this is driving us in a way that is important that the UI gets a very identified style and unified all around the program interface.

Here it is up to designer ideas. But attention should be actively put on these design elements:

- strokes

- colors
- spirit
- professional habits
- expressiveness

Icon colors

If we have provided new icons for typographic options is that it is not only to spend time :). Of course, red is one of the most rapidly seen color. *It could be a good reason to create a Black/Red theme for Scribus. But we have other reasons that drive us in another direction.*

First of all, the most important thing on the screen is not the icon, nor the widows and palettes. The most important thing to display is the document. The user as the focus at any time on it, without being disturbed. Flashy colors in UI may cause disturbance by attracting the user's gaze. This is explicitly to be avoided.

After that, we can think of visual advantages to draw greyscaled icons:

- easier to draw
- importance of stroke design
- focus the design intention on the meaning, searching the most expressive drawing
- Unify the UI by avoiding color differences: this actually can be defined as a kind of theme in itself, even if it is not sufficient.

Icons and unavailable tools should not always be displayed.

We can have a look at other layout programs to have a look at their icon design.

In Quark Xpress, UI is very pure and simplified. There is very little information displayed. It uses a lot of transient windows. In the same order, Icons are only Black and White. But they are beveled outer when available, inner when activated with an extra lighter grey background.



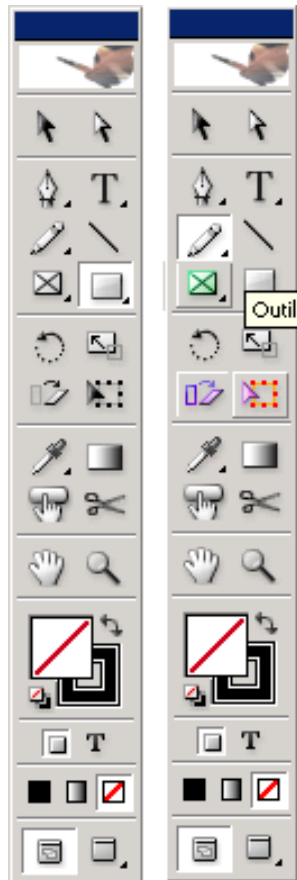
Xpress Toolbox containing BW icons

I'm sorry, this is only Xpress 5.0 UI. I've

never used more recent since I'm now using Scribus:).

In InDesign, Toolbox default position design is very simple: no bevel, no colors but separator between categories and icons are greyscaled. When the mouse hovers over an icon, this one gets colored (for the most part, mono-colored), and when activated, an inner bevel is made and a lighter background as in Xpress.

See that InDesign don't use colored icons for activated tool. Certainly to avoid color contrast and noise that we told about.



ID toolbox with it greyscaled icons changed with monochromed.

InDesign toolbox is much more complete in the sense that:

- *as in Xpress, black strokes are designed to enhance contrast within the icon but the use of greyscale gradients can be use to separate design element such as for Button tool (left to the scissors) and that different grey level are used to display different time of an object modification such as for Scale or Shear Tool.*
- *icons gives feedback to the mouse.*
- *icon groups are more visible thanks to the use of horizontal separators.*

Scribus icon design could be inspired by InDesign's, and find standards for modification of default displayed icons.

In any case, color misuse may cause

misunderstanding and what (an icon) is good in a context is not necessarily in another.

If designers want to keep colors in icons they should be aware to:

- handle their associative quality. They have to remember that the same colors are identified by the user to have related functionalities. In this case, it is important to define sub-themes on which icon hierarchy will have to be centered.
- create, in all cases, greyscaled or B/W icons first. We still think that color should be an additional data. The icon should "work" without that.
- use a very restrictive number of colors to avoid confusion, and that the user should be able to change color theme is the default one doesn't match is individual sensitivity.

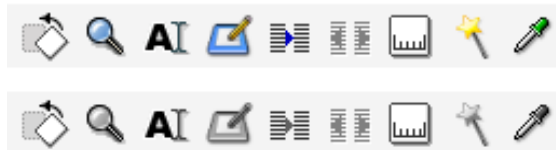
Icon contrast

If colors have to be avoided from icon design, other solution have to be found to improve their recognition. In this sense, strokes will be very important parts of the icon design. Strokes will have to express and show the most important things, those that make it identifiable.

For example:

for example of contrasted icons please refer to previous Xpress and InDesign toolbox.

But Icon contrast is not only made for common recognition, it is also very important for people with visual deficiency.



Gimp is providing two ways to test contrasts quality:

- For luminosity and value contrast, the icon should be transform to greyscale <IMAGE>Image>Mode>Greyscale.
- For color contrast validation, one may use <IMAGE>View>Display Filter>Visual deficiency (or similar)

Greyscaling actual Scribus toolbox seem to produce quite good display. Maybe the pasteStyle tool, because it doesn't use black strokes, could be more contrasted.

In the figure above, with the Frame linking tool you see trouble that may cause inactive icons. They are much too shaded.

"I click on this icons and nothing happens."

"How can I activate this icon ?"

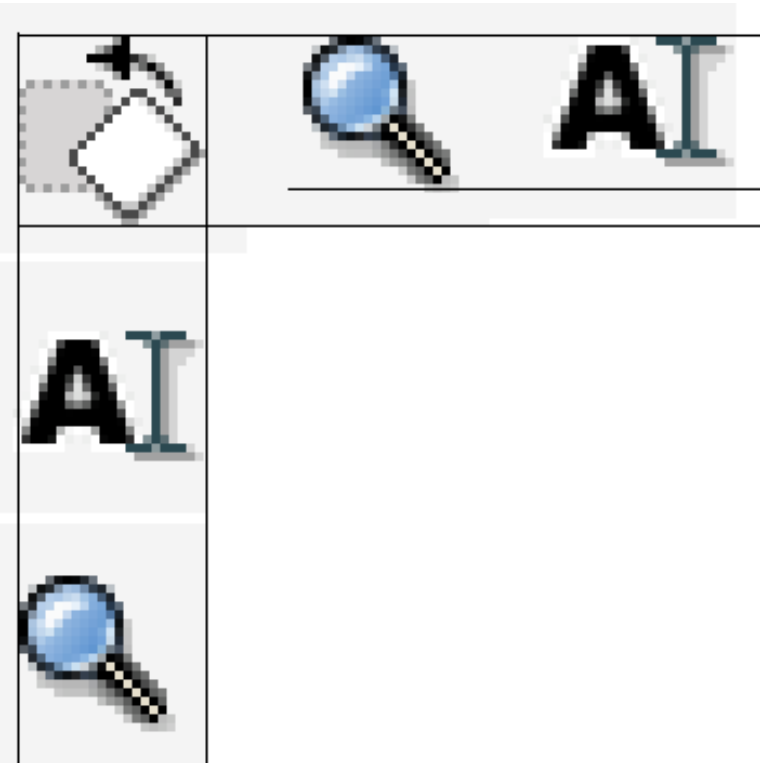
I can't understand why this is done this way, because menus are shaded the right way. This is may be due to the Window manager because this can be seen in other application. But very disturbing:

- Link icon is RGB: 48,48,48
- Inactive Unlink icon is RGB: 128,128,128, i.e. half grey
- Inactive Menus are RGB: around 200,200,200 for text, 230,230,230 for aliasing.

Strange thing, link icon and unlink don't have the same number of vertical lines. always have a look at integrity and consistency !

Icon sizes

Having a look at Scribus icons, it seems evident that they have different sizes. The figure below shows aligned screenshots of several of them.



Comparing size of some icons.

Even if some freedom can be accepted, icon design should be more constant in size especially vertically.

Graphical artists should draw several versions of each icon:

- for toolbox and properties
- 16x16 for fine display
- 20x20 for standard
 - for dialogs
 - 24x24
 - 32x32
 - 48x48
 - for extra accessibility
 - 96x96

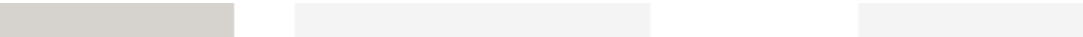
When an icon is a kind of menu for other tool (example for shapes), the triangle expressing that there is a sub-menu could be more discreet. It should at least not be vertically centered, so that it can be interpreted as a lonely tool.

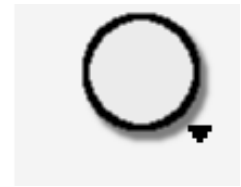
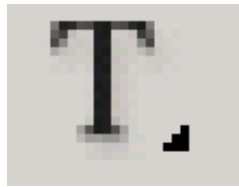
Why is the same tool displayed here and here ?

Best example seem to be Adobe's:

- takes less space
- is visible enough
- is not to attractive

Here is below an example of what could be apply in Scribus. The third icon we have made is a kind of contraction of Adobe's style and Scribus existing.





First is InDesign's, second actual Scribus, and the third, our proposal.

We have completely disapprove the Xpress way which displays triangles on top. Sub-options should also be put below. This way, their access has to be done from the bottom of the icon. Another bad thing is that the triangle is displayed within the button space and could make the user think it is only design and not menu indicator. One could say that it is the same for InDesign exception made that there are no button border that allows to find limits.

3d effect

One of last things is the aspect given to icons. Fashion is actually to do 3D styled graphics, as is a way to get deepness to the interface. So much that flat icons are now considered as unusable. This is actually an error. And, as for color/grey, it is better to have a correct flat icon than a bad 3D one.

We don't have in purpose to give advise here because it is up to designer to do their choice. But a choice could be made between different kind of deepness expression.

In this little non modified screenshot of the Scribus command and tool bar, we can see 5 ways to consider 3D.

1. Light coming from the upper left + shadow
2. Light coming from top (no shadow but stroke)
3. perspective view
4. flat design + shadow
5. flat design

It would be much clearer if a standard was chosen as a way to unify different used methods.

General Cross- Application consistency

The idea of a free Graphic Suite is growing in some peoples' minds. It means having three pieces of software (Gimp+Inkscape+Scribus) that should be similar enough to be considered as a whole. This is a kind of fashion: Microsoft did, more recently Macromedia, and Adobe also.

For the user, this can be a chance. The chance to have perfectly integrated features and interface. But, as Bryce Harrington said, we have to keep in mind a free software is not like proprietary. Most of companies wish to kill competitors: that is called concurrency. Free software grows everyday because people exchange ideas which is completely different from commercial practices. A graphic suite as an integration of different applications, may cause loss of openness.

As we have already told sometimes about Inkscape and Gimp, there will here just be some additional information.

It seems that between user and ethics, a choice has to be made. But in any case, this software can have common functionalities and these could use same guidelines, at least. Will look at some little things.

Icon File Format and management

Navigating the Scribus Icon directory, I've found some XPM icons. XPM, which is an old and historic Linux File Format should not be used anymore in user interfaces. XPM doesn't allow real alpha (in this way it is a kind of GIF). This can be easily seen because icons are less smooth.

PNG should at least be used. It allows true color imaging, indexed colors and will be used in any cases without memory overload. It has a real alpha support very useful for transparent background pictures. This helps doing pictures or icons that can be used in several OS.

A good idea is the one used in Inkscape. All icons are stored in a unique SVG file that is parsed to produce temporary PNGs on the fly. Good idea because vectors will certainly be future of UI design, because they can easily be modified and customized and because they are most precise. If needed, different icon sizes can be generated, in conformity to user preferences.

To go ahead, we could think of a common SVG file shared by different applications (or an SVG file including other ones). This would definitively avoid cross application inconsistencies.

Tools

Scribus has some tools in common with Inkscape and GIMP. Bezier, Text, Basic Shapes.

For the text tool, Scribus and Inkscape have some common things. But GIMP seems to have a different way to provide it. Scribus as a specific application for laying out will surely give much more attention to text controls that needs to be more complete and accessible. For this one, Scribus should be master.

Beziers have very different ways to be implemented:

- Gimp has a unique tool for creating and editing nodes
- As everything in Inkscape is made of nodes, there is a special tool for node editing. Options of this tool are displayed in the specific bar.
- Scribus keeps its Bezier options in a window called by a button in the property palette.

As Scribus can be an SVG editor/modifier, it should follow Inkscape's way. Maybe there is no need of a "Modify Shape" button. I guess there are two possibilities:

- a tab in the property dialog should be better. When this tab is activated, the shape automatically becomes editable.
- a node editing tool that displays a temporary window or activates a node editing tab.

Windows

Window enhancement has already been discussed at his time, I won't come back on it: http://inkscape.org/cgi-bin/wiki.pl?Common_Dialogs

Layer window

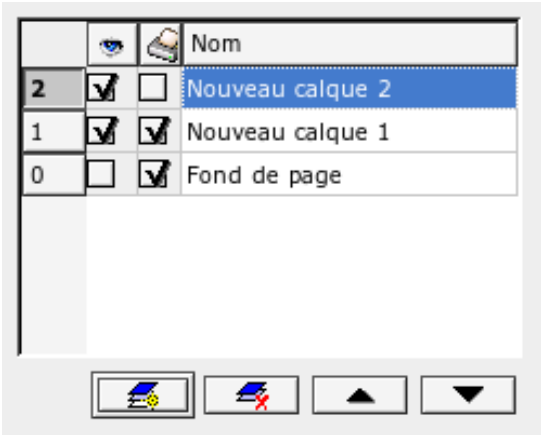
Layer window will soon be available in the 3 pieces of software. In the previous link, only a

simple comparison is made. I would like to make some proposals to modify the current Scribus palettes. I of course think that Gimp, that has years of experienced in the subject, has to be a model. As a specific application, Scribus also has specific options.

It already has some parts:

- Layer level
- Visibility
- Printability
- Layer name

The actual Scribus Layer Window

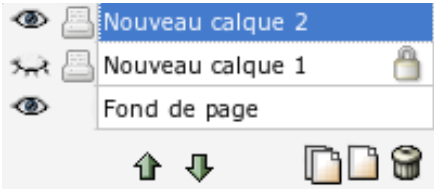


I can see some lock option could be added for each layer and I don't really understand the reason why the level number is displayed because layers seem actually shown in the dialog in the same order as they are in the layout.

Column headers are clickable but do nothing: I propose they toggle visibility and printability on all layers.

Row moving should be activated to enhance intuitive use..

Here is what could be done.



And a new one (thanks to prokoudine)

Text properties

As Scribus is a real text editor, it cannot have the same behavior as Inkscape and Gimp on the subject.

Scribus has many more properties, so that there is a need to iconify what can be and save space. The Scribus interface would surely be an example for other software. Inkscape is aiming to add Text options in the Option Bar so that it may fit the kind of proposal we've made above.

Conclusion

This is a first general glance I've made on the Scribus UI.

Many other things could surely be added. But one thing at a time.

I would like to thank all the trainees that wished to contribute and of course every usual Scribus contributor.