A scribus UI analysis *Part II*

Table of Contents

Everything is going on. Because some users give their minds about many things i was talking about in the previous document, i'm able to present here a new stuff including them and my own proposal on their subject.

If the preceeding document have been made with some trainee support, please read this one as an personal only, since it is scholar holidays here. I won't be able to give more feedback from common users before october or november.

So special thanks here to Avox, Prokoudine and a_l_e who have been very active during the first reading. One more time, i apologize for my crap language and i hope it will be understandable.

Who is this document made for ?

This document has been written in first part to give Scribus contributors a user feedback that was made in an professional teaching context.

If you are a Scribus User, and just want to get informations on how to use Scribus, don't read this. It may be confusing if you're not already used with Scribus Interface.

This is only a working document made by an individual. THIS DOCUMENT IS NOT OFFICIAL DOCUMENT OF THE SCRIBUS DEV TEAM. Reading it you won't get any knowledge of what is or what will be Scribus UI.

About this document

```
This Preformatted Text style has been used to quote user proposals.
```

> tips

> warnings

Beginning with little things

Frame tool

A very simple thing. Many people knows how to use MS Word (and soon OpenOffice). And it is often hard to explain beginners that Scribus does something else. I can remember many of my trainees very disappointed at seeing that Xpress could do so little things for them and that it was much longer to do a layout with it than it should be processed with OO.

The strange thing is that a frame has to be place before any content addition. Normal : the advantage is that we can put things exactly where we want.

But they are more surprised when (i.e. always) there is a tool for a picture and another for text.

To help complete beginners with page layout, proposal would be using a unique tool (frame or even text tool despite if of trouble that may cause superposition). A dialog could ask for type or simpler the type is defined when something is added inside, regarding the type of what is added.

But in fact, the user will have at the end to understand that everything is separated in frames?

Sliders

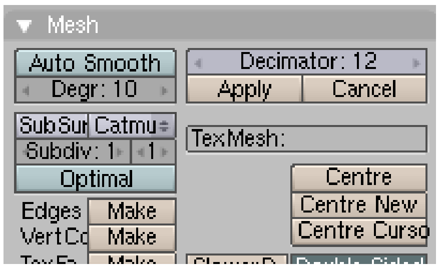
I had a denial of spinbox which are for me sometimes not very usable as many clicks have to be done to reach a 'distant' value. I was proposing the use of sliders in some cases. But it seems that some softwares have overpassed this difficulty.

Shift pressing allow more preciseness in spinbox control by giving access to decimals. But it sometimes looks hard to the user to go back to the default modification value.

```
LA has a spinbox that looks like an underlined label. you drag the value up or
down with a mouse
```

Where “LA” stands for “Logic Audio” that I don't know. But i can take example of Blender.

Of course, Blender UI is very special and don't really follow usual guidelines. But that's the reason why it is interesting : different things inside that could be interpreted in Scribus way.



Blender Mesh Panel of Editing context as it is diplayed at lauch.

First overview of it : controls behavior are distinguished with colors. We re interested in purple one (for example where Decimator is written). You can also see that there are two triangles at left and right of the control. Control+Triangle show that you can change the value by :

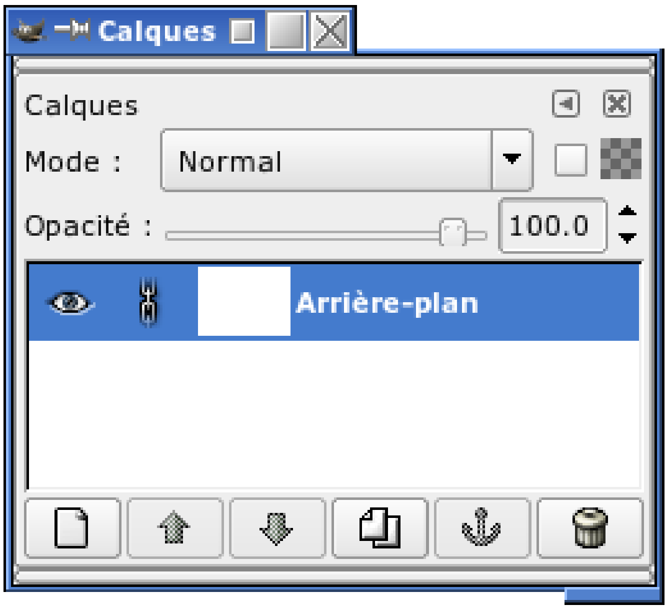
- dragging the mouse over the control
- Ctrl+drag : larger increments
- Shift+drag : edit the value for numeric modification

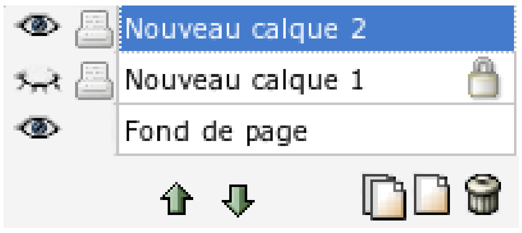
As 2nd and 3rd are already implemented, special attention to the first that could be added as an extra method of QSpinBox Object (if possible).

Layer Palette

```
i don't think that toggling the visible/printable layers by clicking on the eye
is a good idea.
That's just my experience. If i have several layers, i don't want to hide the
visible ones and show all the others: that doesn't make sense in my workflow.
What i could be interested in, is having all and/or none of the layers showed
after clicking, in order to fasten the selection of the layers i want to show
for the next editing action.
```

Just to remind things here are some Layer Palettes : left is Gimp's and right the one proposed last time.

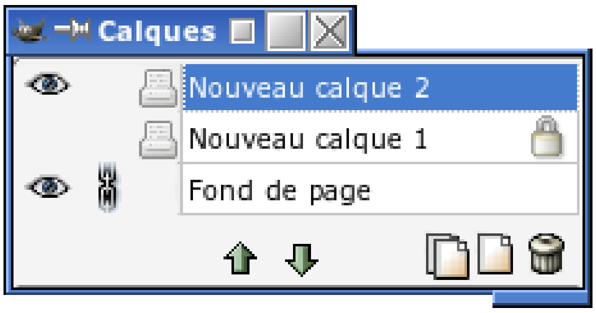




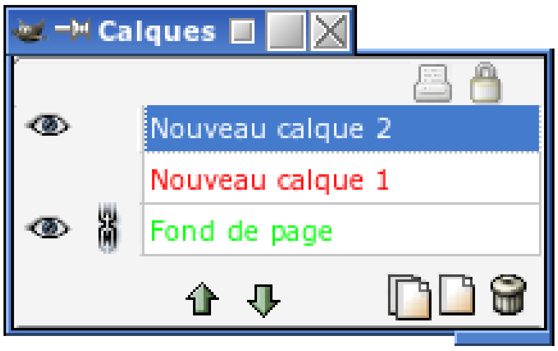
Not many things to add. Wish is to have quite similar palette to improve consistency between both applications. But here are some things that could be done :

- Add a link layer feature
- Add a color context for each layer. This is done in Adobe's vector apps and is very useful : a color is given to a layer and this color is the one used to display selection borders when an object of that group is selected.
- Eventually, remove the eye-closed icon which is not needed : visible when an eye is displayed, in other cases, not visible. This could avoid cognitive overload.

So something like this could be given :



Many there are too many options on the same line. In this sense, we could keep visibility and link on layer line and put printability and future features above all :



In any cases, options that have to be often manipulated should be on layer line, others can have a different place. Icon display (for example here the printer) has to be updated when another layer is selected.

I think the idea to show/hide all layers is good, as an additional option. We have to allow single layer hiding : just think at the situation you want to align object placed in 2 different layers but masked by another one : you just need to mask this one.

Toggle all layer visibility proposal : Ctrl+Click on eye (as in Gimp). This could also be applied for other options. In this case, column heading don't have any sense left and can be removed.

Modifier and Cross OS consistency

```
Ctrl-Click is rigth mouse button on Mac...
```

but

```
you make some examples with alt+click ... the problem is that under X11
alt+click is captured by the system for moving the windows
```

Of course, this not easy to make a software for everyone. Looking at Adobe's softs which can be used on Win and MacOs, We can see that modifiers are not always similar : Ctrl + Click on Win becomes Option + Click.

It will always be hard to be OK for everyone. But Scribus should follow OSs guidelines.

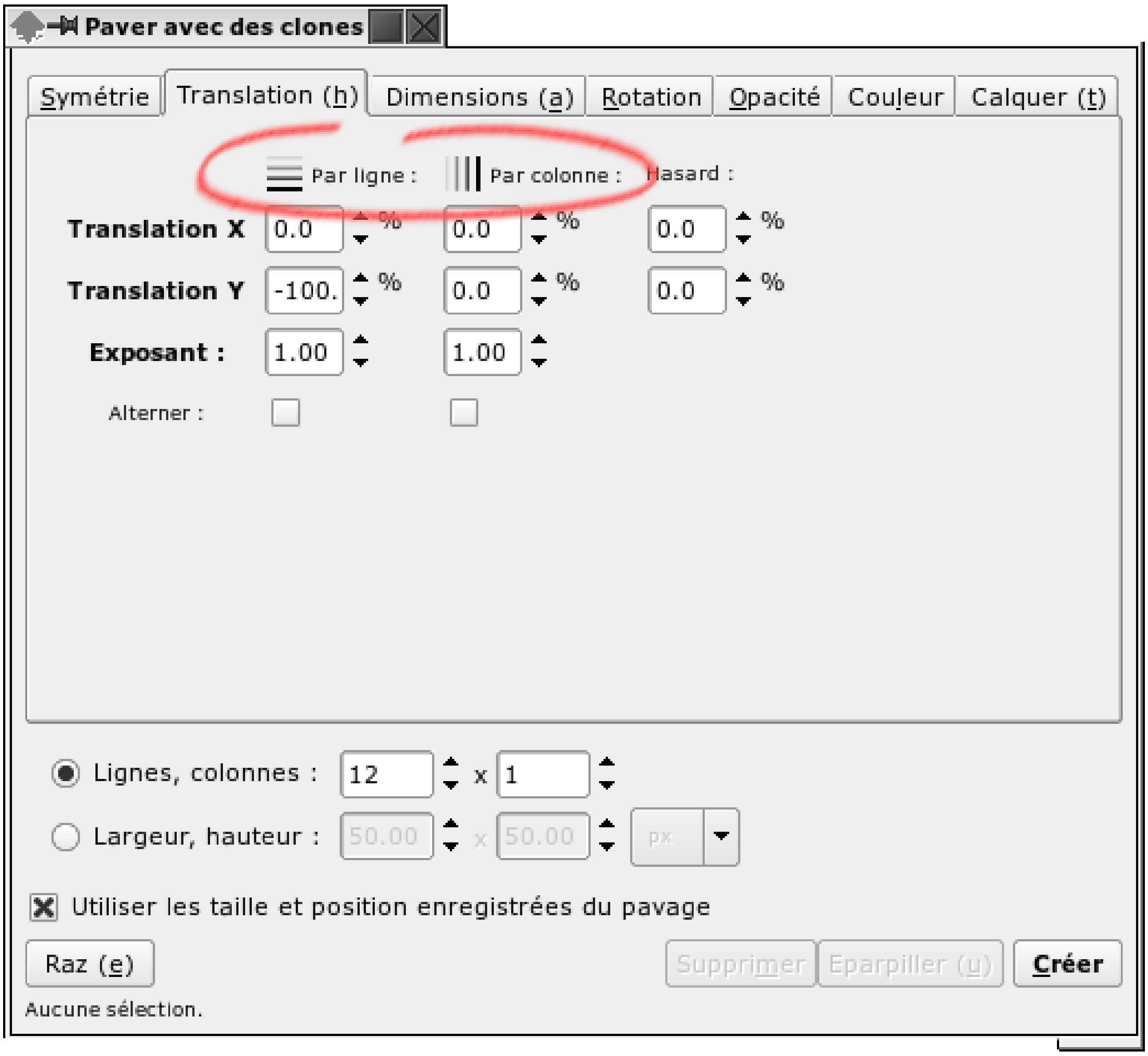
Two solutions :

- Our nice Apple dev make a Find/Replace (this could be done in a bash script before committing new files)
- Put an extra test on event handler which should say : if Os is .. and that those key pressed are ..., then key is ... else ...
- Surely simpler for devs : add in the Preferences a way to configure key modifiers.

Just for info, Alt+Click on X11 can be used with an additional Ctrl : Alt+Click for X11, Ctrl+Alt+Click for application. I guess Alt+Click can also be deactivated in some X or WM config.

Icons or Text

A nice discussion with Bulia (Inkscape) made appear that, if he is very attentive to usability, he dislikes Icons. He made only 2 for is Clone Tile dialog:



Paver avec des clones

Symétrie | Translation (h) | Dimensions (a) | Rotation | Opacité | Couleur | Calquer (t)

	Par ligne :	Par colonne :	Hasard :
Translation X	0.0 %	0.0 %	0.0 %
Translation Y	-100. %	0.0 %	0.0 %
Exposant :	1.00	1.00	
Alterner :	☐	☐	

(●) Lignes, colonnes : 12 x 1

( ) Largeur, hauteur : 50.00 x 50.00 px

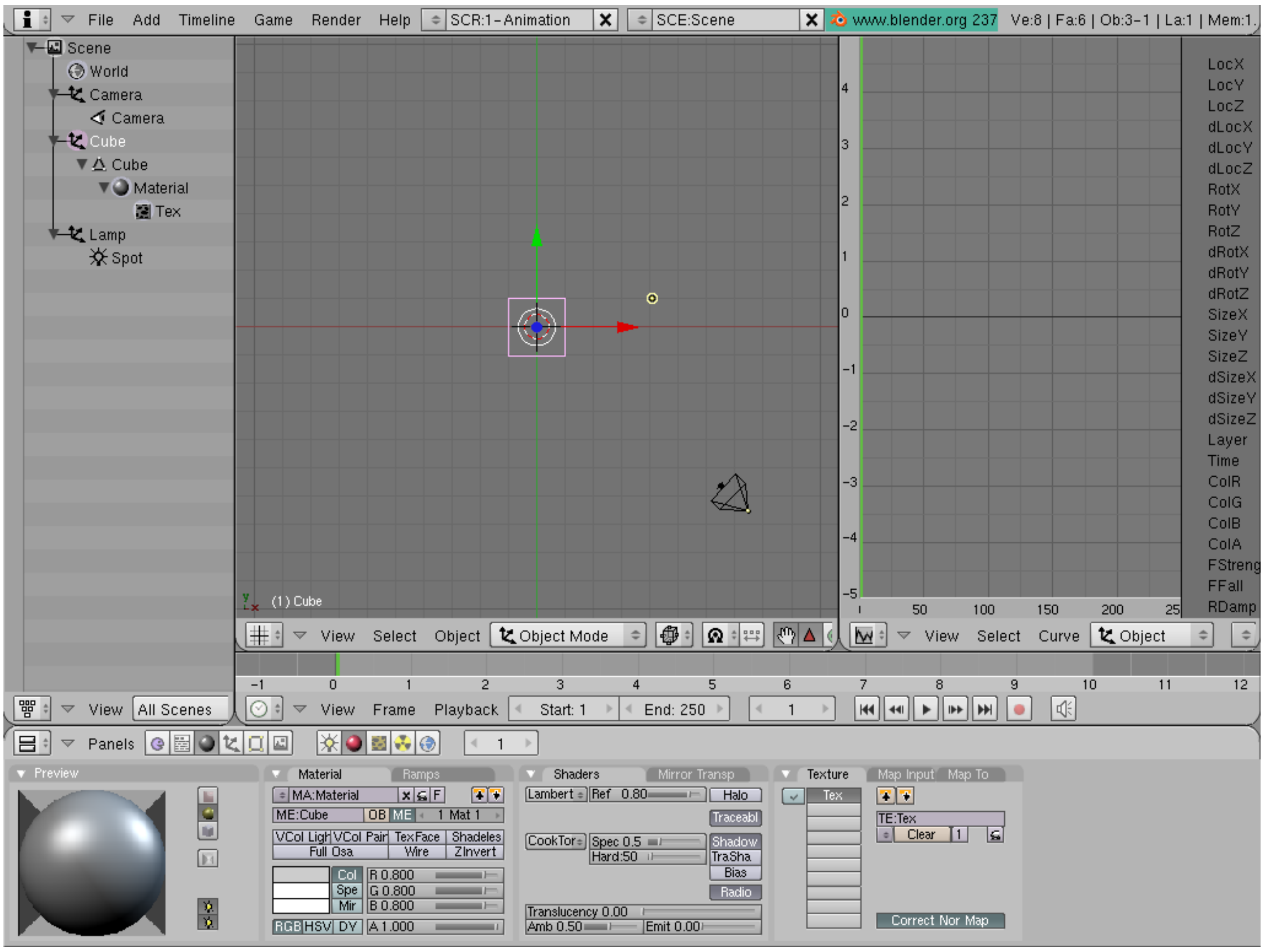
☒ Utiliser les taille et position enregistrées du pavage

Raz (e) Supprimer Eparpiller (u) Créer

Aucune sélection.

One can easily see Bulia's mistrust in icons : he draws some but kept labels near it. As he explain, icons is brought as an extra information but is not essential.

In fact icons are not necessary in software's User Interface. Blender is another good example of application that have very few and still efficient. But Blender has its own UI guidelines based on Raskin's thoughts which are not Gnome's or KDE's. So Blender UI is very logical, gives a lot of space to controls and lots of possibilities to customize it. It is also quite long to learn (we will come back later on that point) :



Screenshot of the Blender UI.

Of course it would be long to describe all the icons. But compared to all controls given ,they are not very numerous. In fact, most of 3D options would be very hard to display in a little icon. So, only contexts and some general options are iconified. We can see that the column on the right uses the same icons than the second group right to “Panel” (lamp, material, texture, env).

What are icon advantages ?

Gnome-HIG :

```
“• Icons can assist the user in rapidly scanning a large number of objects to
select the desired item. Particularly af-
  ter a user is acustomed to an icon's appearance, they can identify it more
rapidly than a text label.
• Icons can augment text by providing visual suggestions to accompany the
descriptive text. Some things are eas-
  ier to communicate with a picture, even a very small one.
• Icons can compactly represent a large number of objects when there is
insufficient space to display textual de-
  scriptions (such as in a toolbar).”
```

I would like to add :

- icons give more stability to the UI : text can have different length in several language (for example : “save” is “Enregistrer” in french which is quite longer and should modify UI)
- icons are more recognizable in small sizes
- may be we forget that too often, but text is also a representation good at representing concepts but not visual things. In this case a picture is more expressive.
- Despite some people would like to convince us that pictographic representation is a new way, just look at Egyptian Temples to put head on your shoulder and things in the right history :)

When using icons ?

That's the real question :

- For tools
- For frequent actions
- For frequent options
- For options that cannot be misinterpreted particularly in foreign context (to avoiding translation inconsistency).

Rare options don't need icons. Icons are provided to fasten control recognition. But the user have to learn them. It is possible only if he uses them a lot. In this case, icons are a bit like shortcuts, but easier to memorize because displayed : user just have to choose in the given shown possibilities.

Dialogs Vs Palettes

It seems that two kind of logic appear on the subject : quantity and quality.

Quantity

```
modal dialogs should be reserved for actions which need more parameters (file
open etc.)
```

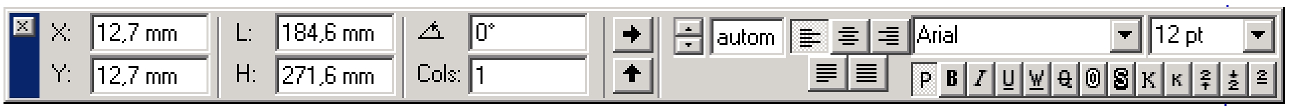
In this case, It seem to be a problem of quantity. We would have :

- Modal dialogs : when many things have to be proposed
- Palettes : when few parameters are to be displayed.

My opinion on this is that it won't be very clear for the user. The UI has to be organized regarding to user habits and mind. This solution is much to centered on software itself, just as if the trouble was to find a place to properties. But the trouble is not here. Scribus UI is already somewhat good. Following this logic, would decrease its efficiency.

Why ? Just an example. Text properties which is certainly the most complete property set in Scribus would be the biggest modal. Quark Xpress makes a great part of Modal dialogs. Users can access Caracter and Paragraph format in such dialogs. But two notes have to be added.

- In Xpress, modal is used for temporary things and palettes for permanent (ie temporary windows for temporary and localized props and permanent windows for permanents props and objects). So in this context, modal may never be used by some users (like me, because i always think a local prop can be reused again later).
- Some properties (even temporary) are available in property window which is considered as a palette.


X: 12,7 mm
Y: 12,7 mm
L: 184,6 mm
H: 271,6 mm
0°
Cols: 1
autom
Arial
12 pt

Quark Xpress's Properties palette

As shown above, Xpress Properties palette displays most frequent options on frames, paragraphs and caracter. For more properties the user to :

- Display a specific window (Frame, caracter, Paragraph, Tabs, Lines...)
- Define a style which will be displayed in the homonym palette

I guess Caracter styles are on the way and i'm waiting for them very impatiently. But, as an idea, it could be interesting, in some cases to have Frame styles in which we could define size, margin, colors and eventually main default text style. This way, it would be possible to use common frames even if they are not to be put in templates and to earn precious minutes. This could be an alternative to Album use which seems not to be well known.

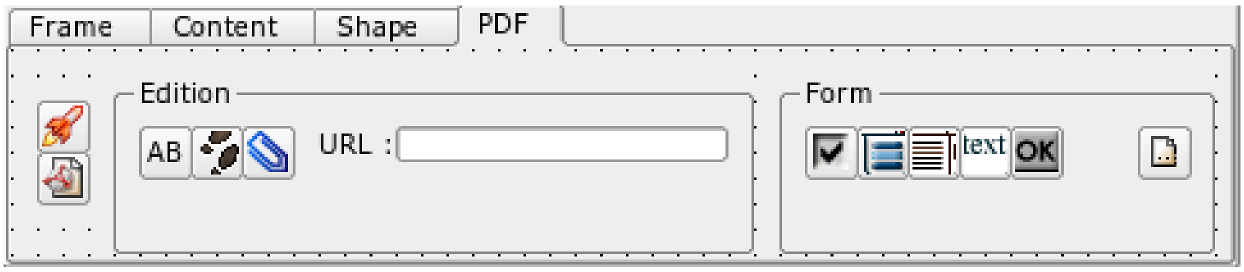
Quality

```
parameters(properties) for objects should be on palettes, parameters for actions
(like open, new, print, export, import, etc.) should go into modal dialogs
```

I prefer this proposal. Here, we have windows organized regarding to objects quality or status.

- modal dialogs : Actions parameters
- palettes : Object properties

I don't remember in which context it has been made but i guess i can be in opposition to my proposal of a PDF palette in which i put some actions : Preflight...



Souvenir, souvenir

This proposal was made to avoid transient windows which could be very boring we having tens of controls to be individually defined.

In fact, this palette includes Three Things :

1. PDF objects
2. Objects properties
3. PDF actions (or some of)

It was a try to condense on a place very dispatched options in actual Scribus UI. Background of this was that the user will first understand the interface has objects options because objects are the more visible and usable things in it. And i wanted to take advantage of this. But in fact, PDF is a very complex things that is also related to printing, with particular options that will be displayed there. If a PDF tab as to be, it should group most common options for document creation only.

And for me, Preflight is also creation part since a trouble in verifier can cause layout modifications in some cases.

Come back to main subject, putting all properties in palettes can be invasive. This is the case in Adobe's software. Trouble is to define what is a property. When talking about text, it is easy to say that height or color is a text property. But we could also think that the Manage Picture dialog could be a kind of document (not page but document content) property as valid-link or colorimetric modes or visiblity is really a property of imported object (and in fact is not the picture a property of the image frame ?).

Of course, when talking about objects, i'm not referring to programing objects but objects on the layout. Both can of course be related but not necessary and new contributors should be aware not giving controls to each programming object properties or method.

As we can see, talking about properties is not enough and more preciseness has to be brought if this way is to be followed.

My opinion

is to follow one way, the second. Use of palette is more convenient than modal dialog for object properties and some controls/verifiers.

User should never be blocked in action : this is the forgiveness guideline. Inkscape have important use of so palettes (that looks like dialogs).

And always opening and closing windows can be boring as we exposed for our PDF tab.

But the number of palette will grow. Some things have to be done :

- Each palette should have a predefine but customizable shortcut
- Palettes should be grouped when possible and groups should be customizable (from a window or may be one day as does GTK by dragging)

I'm also assuming that the F10 shortcut should be changed to Tab :

- This is a common shortcut in many proprietary softwares (in fact, because Scribus will be ported on MacOs and Win many users won't use only free software, sniff)
- This is a logic shortcut for navigating inner windows that can be related to the Alt+Tab (navigating through applications).

. User action

The users may want to use the Interface by different ways. Some will like menus, others keyboards and some others toolbars.

```
I can perfectly change the value in spinbox from keyboard(...) arrow up/down
```

In Graphical User Interfaces, each element is made, since the eighties, to be reached via a mouse. For the common non professional user, this is the only way to talk to an application. Of course, the mouse is not the only peripheral that can be used in that purpose. The following :

- keyboard
- tablet
- reactive screens
- vocal synthesis

are some others. Two last are not often used and can be ignored. Vocal synthesis will be taken in charge by the OS or a service. This one will use some element of the UI to understand what is demanded.

Reactive screens and tablet will have in most time the same behavior as the mouse. We can notice that tablet even if they are now well-known and cheap are not used even by some professional. A friend of mine, teacher on MS stuffs (there are too many), has been very surprised to discover that at my home and see how easy it was in use.

Keyboard will be a very specific peripheral. But strange, as i think that the most useful peripheral to write text would be a microphone, most of users just relate keyboard to text typing (surely in comparison to ancient typewriter machines). Many of my trainees, even accustomed in computing things for years, have felt difficulties to learn a simple Ctrl+S (which mean S*** as everybody knows when forgot to use it).

What i mean in describing this is that it is not sufficient to say that best or more practical can be do by a mean, whatever it is, because most of the users won't navigate through these means. May be 80% never use keyboard for something else than writing text. Shortcuts, Modifiers, out of mind.

But this is not a bad thing. This fact can be exploited. At this time, as a graphical software (user shouldn't forget Scribus is not a text processor!!), Scribus make a great part to graphical objects manage with a mouse or similar material. Menus, Toolbars, Windows and Palettes and simply Page document, all these things take their senses by the mouse. Many notes we have made here and in the previous document are related to this kind of use.

Some shortcuts are provided to fasten the work and menus have access keys so that the user don't always need to move the mouse. This is very useful : it is so heavy, when i point a precise place to have to go somewhere else and come back exactly at the previous place to do the thing i wanted. But in all time, all these means are thoughts as replacement or alternatives, not as assistance or complementarity.

Example 1 : Quark Xpress

I could take 2 examples of softwares that have a very particular use of keyboard.

Quark Xpress does the same job : page layout. Xpress at the beginning was a competitor to Aldus PageMaker (bought later by Adobe, if i remember). As soon as Xpress was stable enough, Aldus uses the party. Xpress was made for professional : not graphists, not communication consultants. It was made for Print shops and typographers. Xpress is a kind of boring software, and that's what felt many of my trainees after a Photoshop session (for the history, i don't still teach Photoshop, but Gimp now).

Many of Xpress functionnalities are not displayed in the user graphical interface, at least until v5.0 (the last i used before turning to Scribus). The user have to know by heart some keyboard shortcuts.

In these some very useful :

- Adding page number
- Image constraint to a frame ...

Only a very attentive reading of the user manual divided into 2 parts (help and learning) could help in this.

Even if Xpress is not a model of customization, some nice tricks can be extracted. Particularly the ability to define a user shortcut for user defined style. A very could example on how not wasting time in going every minutes in the same palette clicking in the same place.

In fact, the mouse is not needed in Xpress, everything can be done by keyboard and it is very fast for accustomed users, most of them professionals, and sometimes much more that InDesign.

Example 2 : Blender

We have already told about Blender and show how it is rough in display (no nice icons ...)

But Blender User can tell you how efficient it is :

- buttons are displayed very logical (grouped by functionnalities and context, a color code explaining who to use it)

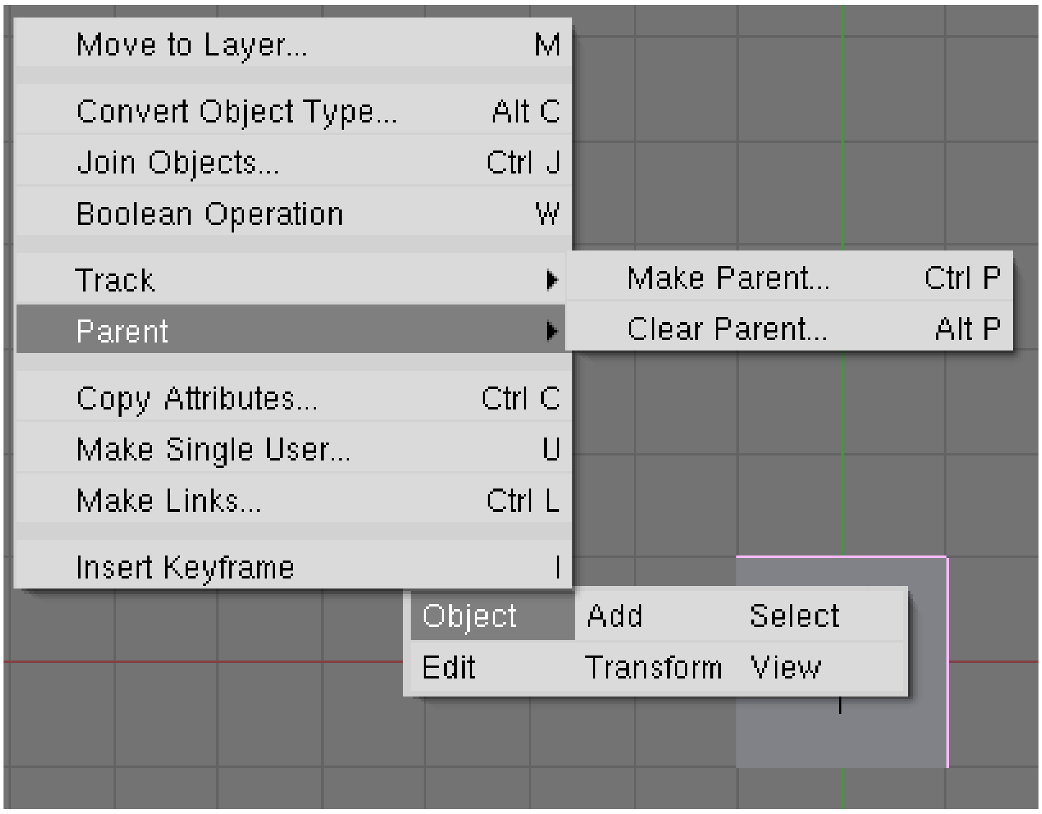
- a principal property/object window display at bottom (non floatable)
- apparently nothing in menus.

A nice thing is transparent windows that can display options without hiding the document and even leaving capacities to work on object through the widow itself depending on the place clicked : empty places are free for document focus)

In fact, Blender UI is the exact opposite of what is a standard Graphical User Interface as defined in Apples or Microsofts HIG and even in Gnomes or KDE which are evidently inspired by the previous.

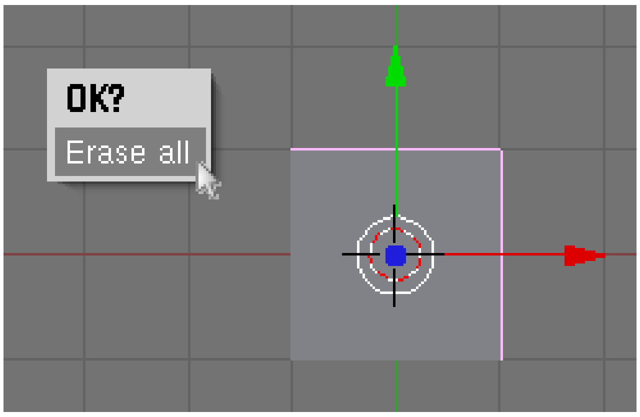
Blender UI tries to find the exact complementarity between Menus, Mouse (windows and controls) and keyboard.

- Functionnalities related to Blender itself are found in menus (that's also Gimp's way in some part)
- Functionnalities related to Scene creation are displayed in a contextual menu. Of course, some part of this menu are modified with the object type or the mode used (Edition, or Object). In fact, only the “Add” menu is often used. Others, as shown above, give access to functionalities that all have shortcuts which are systematically used.



- Options are displayed at the bottom in contextual drawer windows.

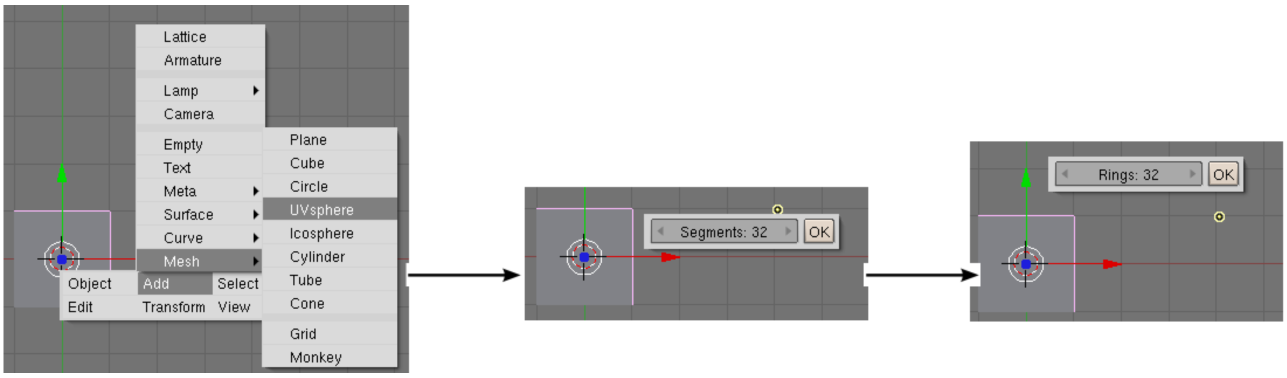
As a graphical software, the use of the mouse in Blender is very important : it is the design tool. So everything is to be done to use it only in that purpose. For example, dialogs are displayed at the mouse position so that he don't have to move.



Dialog displayed at the mouse position to increase productivity.

Dialogs hate containing several fields. Single field window are displayed when needed. That way, the user don't have to navigate between options and just validate the one he doesn't want to modify.

The window is automatically closed after a delay so that the mouse is kept free.



Creating a sphere in Blender and use of simple dialogs.

Conclusion

These example show how it is possible to have a different view of the user interface : not only this one that says that a functionality has to be available by different means but in a way that the use of the software is defined by the purpose, the needs, and the global context.

Official guidelines are made most common apps which are not graphical one due to their specific use of the mouse. Never forget one thing : the most important is not software and its possibilities but the document. And in fact, 90% of layouts are made in rectangles.

So should Scribus use Blender's way. Of course not, but there are two moment when a special attention should be placed on alternative usabilities :

- when a task is very often repeated
- when it is very rarely use. :)

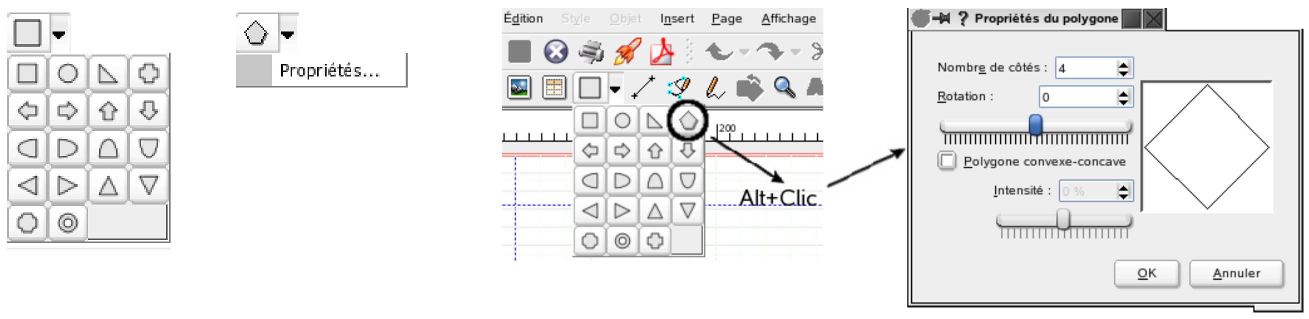
I should do this :

1. Avoid long context menus and see how blender organizes its own in 6 blocks. Delete rare entries, replacing them by an esoteric shortcut and put infos on them in a help.
2. Give very simple access to common tools and options. For example a frame always has to be of a type, at a place, and sized. T-key adds a frame at cursor position, with default size customizable (or buffer it and wait for a click when use with mouse). S-key gives focus to bottom-right point and display this point position for changes during a frame specific ruler is displayed to help.

The second position is not made to replace existing but just improve productivity for some kind of users that like keyboard a lot.

Shape Palette

`As for Polygon_Property.png, I would prefer a palette with those settings.`



Nombre de côtés : 4
Rotation : 0
Polygone convexe-concave
Intensité : 0 %

Just to make things come back, left, actual Scribus Polygon Tool, right, my proposal in Part I.

I completely agree this remark. In fact, too moments have to be considered in creation of a UI element :

- creation time of that element
- later modifications

For the first, i would use dialogs. For the second, Property Window and in this case Shape Palette. Note that my proposal was to use a key modifier to make the dialog appear so that there is no obligation to use this method.

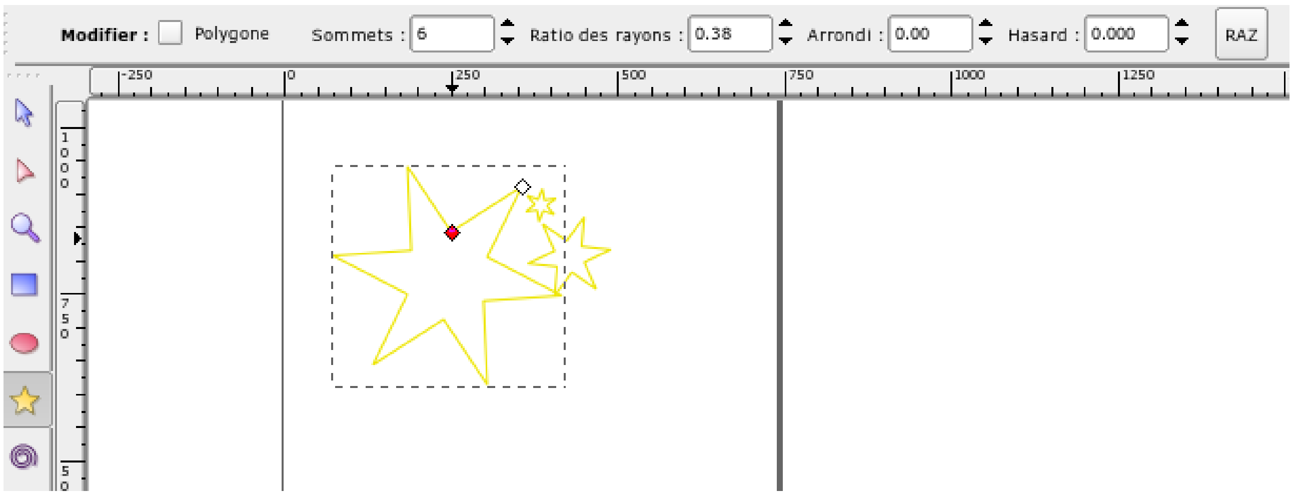
Why giving this possibility ? because many user seem to think of the creation as a whole in which the object is near to its characteristics. Something like this : I want a pentagon and not i want a polygon that could have five or 6 or more/less sides.

Why i made this proposal ? Because it could, with only very little changing, have a different way to think polygon drawing. Not only being defined in two steps, actual Scribus ways suffers of 2 troubles :

- Properties need to be defined and then tool has to be activated for drawing : it seems to be completely unnatural for many users tested.
- There is no shape preview during drawing time

Following the first concept, Adobe Illustrator has a nice way to implement these options : for example, to increase to number of sides, the user just have to press key-ArrowUP while dragging the mouse during creation of the corresponding object. But trouble is that the shape cannot be modified afterwards.

Inkscape can be a good example of what could be an interface for shape modification.



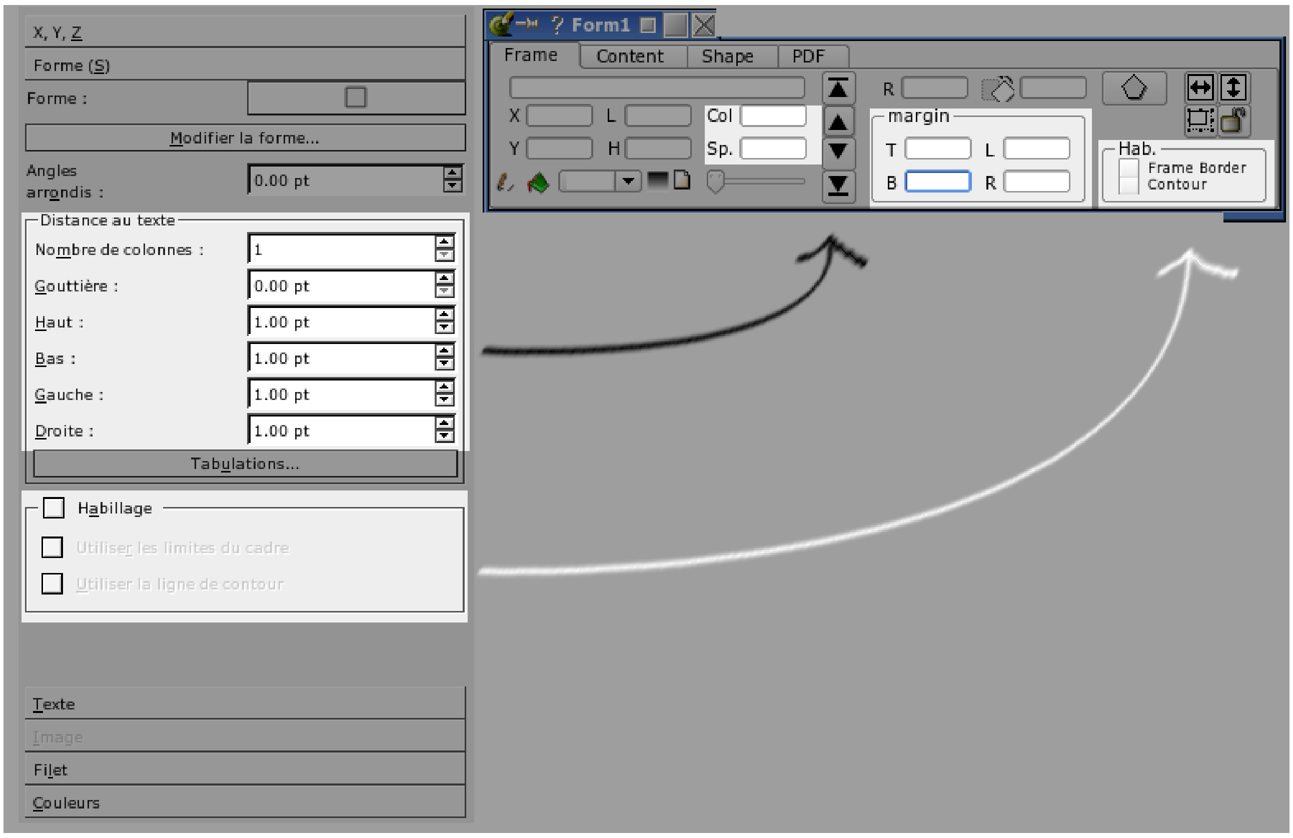
Inkscape example of completely contextual Tool/Objects Options and handles on the shape itself.

Inkscape, as a drawing tool, is very attentive to shape properties. Here, tools and object structural properties have a contextual toolbar. When a tool is activated, the content of the toolbar is automatically updated. The user can modify options and then draw or draw and then modify. Plus the drawn shape has 2 handles allowing the changes some properties more visually (Point and see principle).

Of course, using contextual windows is more efficient and usable. That's why we had proposed a tabbed property palette in which the user wouldn't need to navigate between tab for a single object. For example, we put some options like color or style creation in Text Tab and allow the user not having to go to a specific tab or window.

As i had displaced Inner Margin and columns properties to Frame Tab because :

- it is more consistent with such existing apps as Quark Xpress ans Adobe Indesign
- they were not related to the shape drawing but the text behavior within the shape, i.e. shape as a container. I thought it logically meant "Frame";


X, Y, Z
Forme (S)
Forme :
Modifier la forme...
Angles arrondis : 0.00 pt
Distance au texte
Nombre de colonnes : 1
Gouttière : 0.00 pt
Haut : 1.00 pt
Bas : 1.00 pt
Gauche : 1.00 pt
Droite : 1.00 pt
Tabulations...
Habillage
Utiliser les limites du cadre
Utiliser la ligne de contour
Texte
Image
Filet
Couleurs

Some options of the actual Shape tab have already been moved to other tabs.

As some other feature could be soon be implemented such as an external margin, particularly when using Text Flows;

a shape Tab could be kept with a new organization. It the same time, the Polygon button of the Frame Tab could be removed, place be used by following buttons an give more space to flow options.

This shape tab could contain actual properties :

- shape
- rounded angle

but also Edit Shape controls that saves a window.

On this addition, trouble could appear if Edit shape window is considered as containing tools. Personally, i don't consider them like this simply because they cannot create but just modify just as other controls even they can be used much more visually.

Warning, in addition i would change the “Reset Contour Line” label to “Update contour line” or “Contour line fits shape”. For many users, reset means go back to an anterior state, which is visibly not the case here.

I cannot actually determine myself about the opportunity to display all default shapes as button instead of drop-down menu in a such palette.

But surely, it should be divided in 4 parts :

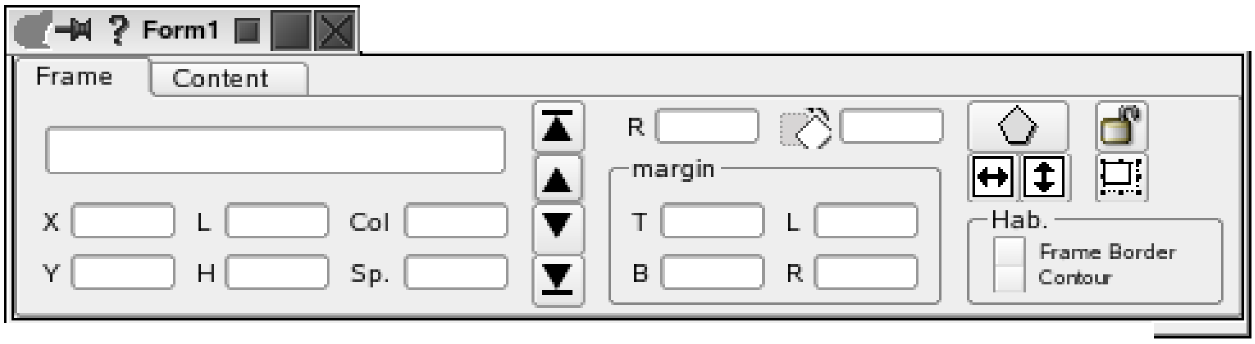
- default shape
- shape drawing options contextual to the shape (for polygon they are already made but we could imagine other options to determine a Cross proportion or a rounded side amplitude).
- Node editing
- Other value as rounded corners, rotation, toggle Absolute/relative coords...

May be that Scribus wishes on drawing tools has to be more expressed before applying such modifications. As i'm already used to navigate between 3 softwares (i've done it for years between with Xpress/Photoshop/Illustrator), i think a better integration with Inkscape could do a better effort. But Node editind is still a very strong, interesting and original possibility that could also be more visible in the Scribus UI. In this sens, our proposal could have a reason to become reality.

Property Window

This is certainly the most used palette in Scribus. It contains major params on objects. This palette very have to be usable, the most usable of all kind.

In the previous document, as a response to window tabs trouble, i proposed to give it an horizontal look and to move some properties to make them match user habits, especially Xpress users.



First proposal made. Now other tabs have been added as show Page 20.

Some people have tell their disappointment. They most have 2 reasons :

- Most of layout are made as Portrait so that place can be available on sides of screen but rarely on top or bottom;
- This kind of palette will often be moved so that the user will have to look for it.

Referring to other publishers experience :

- Xpress has an horizontal Property palette
- InDesign has many, but most important in on top, just under the menu, horizontally
- Macromedia (sorry!!) also has a large use of horizontal of an horizontal property palette,

I still think this is one of the best way :

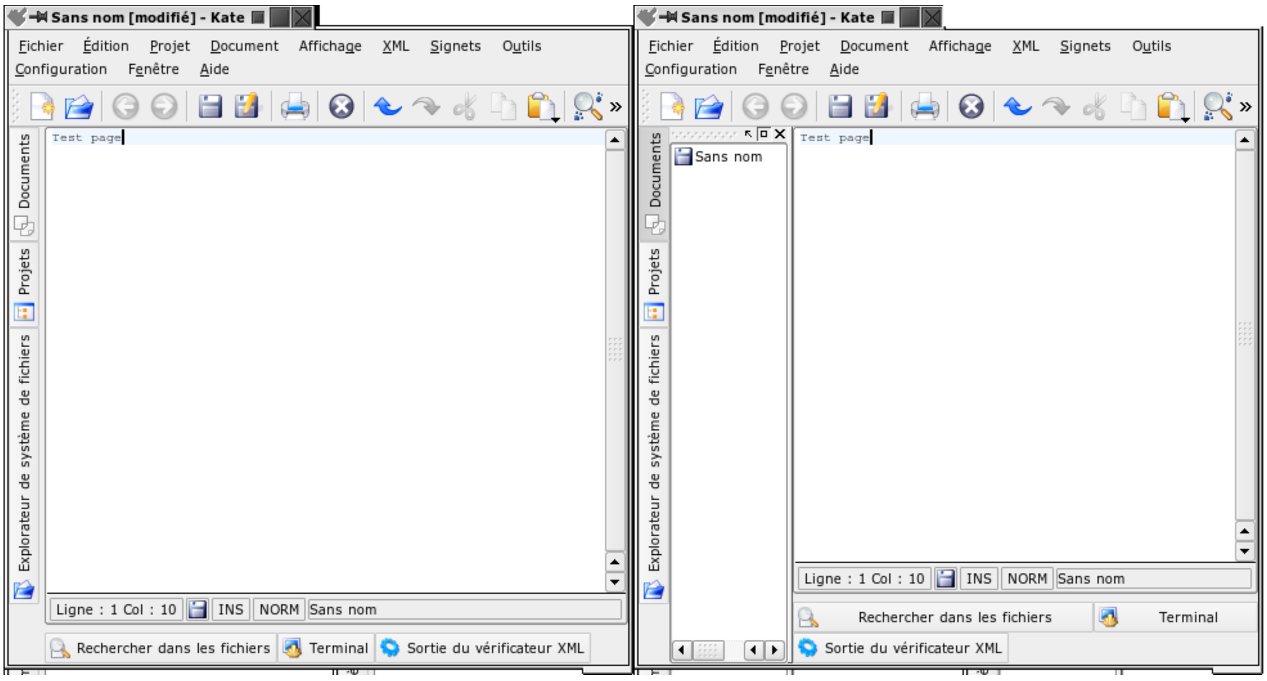
- it takes less place
- it is more easy to find controls
- it is more easy to navigate in...

Of course, the best should be giving the user the choice. But it may be hard to develop.

In fact, to avoid trouble, the palette could have several behavior.

- Floating and hidable with a default predefined shortcut (Adobe uses Tab key)
- Docked and retractable (Macromedia's model)
- As a unique contextual option bar (Adobe's model)
- As separate palette (Adobe's model)

If verticality has to be given for any reason, I suggest to make it retractable as it is done in Kate for file navigator...



Navigating model used in kate that could be used for objects options

Each button could be a tab of the actual property window, but details are placed right of the buttons and not between so that they are not moved by a tab activation. The options should always be available at the same place. It is displayed vertically so that document space is preserved.
Surely actually the simplest.

Cedric