Bforartists UI redesign Design document part 1 - the general concept

Content

Preface

This design document is about creating the new graphical UI for Bforartists. Note that this is just one step of many for the whole UI and usability development. The GUI is not the only aspect of usability.

And also this document is currently just a starting point and a rough concept that defines some corners. Most of the development will most probably happen in the prototype, at a case by case base. I'm a big fan of evolutionary prototyping. Which you can also see when you read this document here. It's pure evolution what i show here.

First, it is planned to redesign the UI with the very well known UI solution QT. But i would be also green when we could do the needed changes with the old UI solution. It's just that QT would bring so much advantages. Not just for the current changes, but also for future implementations. So it's a no brainer for me.

Before we go into detail let's analyze what goes well and what goes wrong with the current UI in general. I will not go too deep into detail here though. The goal of our Bforartists fork is to rewrite the UI completely anyways. So i save my energy, and explain the vital points in the design part.

When you want to know what in detail goes wrong with the Blender UI, then i suggest to google for Andrew Price and his UI Proposals, and watch the first two videos. He pretty much nailed it. Even when his own proposal in video 3 was not the best then. But it was already much better than what is currently in Blender in many areas.

Regarding design, i will not define and write about every button here. The exact details happens in part 2 of the design document, the theming. I will just define the concepts in this document.

This document first gives a rough overview over the new general concept. And then we will go into the details. Like look of buttons, functionality of toolbars, etc. From rough to fine.

We will also need some prototyping to make the concepts clearer and to play around with the look and feel of the new UI. This UI prototype will most probably not be made within Blender. Currently planned is UI prototyping in QT designer.

Last but not least, i want to avoid to make any big changes to the functionality while developing the new UI. We cannot fix all the design mistakes in Blender at once. This step here is to develop a useful graphical UI. Other steps will follow as a separate task.

But there are two or three things that we can iron out while developing the new graphical UI. Double menu entries for example. To name one, there is a create primitives menu in the tool shelf, and the same functionality in the menu.

Note that this document may change in more than one area over the time, to reflect the results of the prototyping then. This document in its current incarnation is just the first starting point of the development.

What went wrong with the Blender UI development

History

To answer the question what is wrong with the current UI we first have to dive into the history of Blender. Blender once started 1988 as an inhouse tool for a 3d company called NeoGeo. Inhouse means low resources for development. Graphical UI development is time consuming though. And there was no need to satisfy the masses with a general and standardized UI concept anyways. The useage was limited to a few people and to a few tools. The main UI concept was hotkey useage therefore, and to add graphical components where there was no other way, like for sliders and edit boxes. And they were more or less placed where there was free space.

Then Blender became open source in 2002. At this point there were lots of established 3D packages around. All of them much more advanced. So the main development focus for Blender became to catch up, featurewise. The useage remained hotkey centered though. With the graphical UI being a necessary evil. And so they continued to add graphical UI components where there was free space, plus minus a few adjustments.

It was foreseeable that this just had to end in a mess. And it did. With 2.49 Blender reached a point where even the hardcore Blender fans had big trouble to use the beast. They simply ran out of space, in both, the UI and the hotkeys. There are around 170 hotkeys from the core functionality. Add to that some vital plugins. And you reach easily 300 hotkeys to remember and to use. including some crazy ones where you have to hold 4 keys simultaneously.

The graphical UI was at this point never meant to be the main useage way. Mainly just to give visual feedback and to edit values. Blender was trapped. Hotkeys only was and is a dead end. And so, after many years, Blender became a new graphical UI with version 2.5 in 2009. The attempt was not this bad. It was definitely a big improvement compared to what Blender 2.49 and earlier had as a graphical UI.

But then happened two things. The UI development, wich wasn't really finished at this point, simply stopped. The "new" UI still suffers from some big design mistakes that needs to be ironed out. And this very necessary evolution step never happened.

The second thing that happend, and which is connected to the first thing, is, Blender developers started to go on as usual. They fell back into the old patters. Means, the main focus is again to add as many new killer features as possible. And the development of the UI gets ignored as much as possible.

And the new and old UI concept is again Hotkey centered. With a graphical UI that gets again more and more cluttered. It's the same pattern that we have seen with Blender before 2.5. The next UI crash lies ahead.

The general UI problems

So what is wrong with the Blender UI? In short: the usability.

It's not all bad. But the Blender UI is inconsistent in more than one area, doesn't always follow common standards, relies too heavy at hotkeys, gives poor feedback, and it is visually complicated designed.

The graphical UI is counter intuitive, nothing is self explaining. You need for nearly everything a tutorial. New users already fails at just moving the standard cube in the viewport because of the RMB select issue. And even advanced users are lost at new features without tutorials.

The UI is permanently overlapping each other by having tools in the menu bars of the windows, means you are always at pulling around the windows. There is too much stuff hidden in sub menus that should be top level.

Another main problem is that a tool can be virtually everywhere. Even in other windows. Some UV mapping settings opens in the 3D view window.

And so on. The list of issues is really long. Blender is for good reason known as one of the hardest to learn and to use 3D applications.

Most of the issues is a direct result of the hotkey centered useage philosophy. The graphical UI is still just a necessary evil in Blender. You feel this at every corner. And that's how it gets treaten by the developers. The UI development is de facto stopped since four years.

Regarding the hotkey philosophy, other software developers have long understood that the main useage of a software mainly happens by a graphical UI. Hotkeys are stil important though. They are there to speed up the workflow where needed. The fastest workflow is a mix of both. Graphical UI and Hotkeys. And it is individual who uses what in what case. In a well designed UI both should be possible.

There is another big point that went wrong in Blender UI development. The wished target audience of the Blender developers is not the user base. The target audience that they want to reach is the professionals. But Blender does simply not happen in the industry, asides from a handful enthusiasts. For various reasons. To name a big and unchangeable one: the GPL license. Which prevents a professional plugin pipeline. Since professionals fears the poisoning nature of the GPL license for their own code.

Hobbyists though have other needs than professionals. Especially in the usability area. A professional gets used to every tool in the one or another way. A professional also looks for fastest workflow in most cases. Which isn't necessarily always the best and most intuitive workflow for the masses. And a professional can also live with a wicked UI and workflow as long as the tool does the job. ZBrush is an excellent example here.

In reality myriads of wannabee hobby graphics artists failed already at just moving the standard cube in the Blender viewport, and gave up at it then. And that's the Blender user base, the hobbyists.

Let's enter another UI problem area. This time from the programmers point of view. The underlying code base. It is a crude mixture of C, Python and OpenGL. The idea behind was that Users should be able to change the graphical UI to their needs by changing the Python code part. In practice nearly nobody does this since it is ways to cumbersome, and will break with every new version. And the really vital parts are still in C anyways. That's why this approach for custom user menus more or less failed.

Then there was the consistency. We have four or five different looking dropdown boxes in Blender ...

So what is the conclusion?

To solve the current dilemmas we need to do some radical and big changings. Small adjustments will just not do the job. It is not done with putting a cluttered UI element just at another location, or with recolouring a button.

We need to flip the useage philosopy by 180 degrees. Forget about hotkeys. At least for now. The current UI philosophy was hotkey centered, and aimed at speed and professional useage. The result of this was the currently cluttered UI. The new UI philosphy needs to be graphical UI centered, with a intuitive, consistent and self explaining graphical UI design and layout. And the useage should be purely possible and done by the graphical UI elements in the first place. Except those cases where there is no other way, like navigation.

I want to provide just a most basic hotkey set with a handful hotkeys that can easily be remembered. Keyword homerow. Everybody who wants to use hotkeys can still set up their own then. It's one of the misconceptions anyways that professionals wants every tool as a hotkey. They set one up when they need it.

We can still provide the full Blender hotkey sets as an option. This would make the migration of long time blender users easier. But the useage focus must lay at the graphical UI. We simply need to develop the new UI with a main eye at the main target audience, which are the hobbyists. Not some imaginary professionals that will never touch the software for production. And most of the hobbyists are definitely no hotkey jockeys. They want it easy, not fastest. This makes also tasks like writing a tutorial much easier. A changed hotkey will not break the whole tutorial anymore.

Furthermore we imho need to rewrite the graphical UI with one of the common UI solutions, since it is very hard at the moment to change the look of the ui. Like spacing, turn rounded buttons into square etc. . At this point it is planned to use QT for the new UI solution.

If we use the Python code layer for the QT integration, or if we go directly down to the C code must be investigated further. I honestly miss the needed knowledge at this point to decide that.

On the one hand, when the QT UI would be working directly with the C base, then there would be no further need for the Python layer anymore. And the less code to maintain the better it is. On the other hand, most of the existing Blender Plugins relies at the Python UI code. And it would save us tons of work when we would use the Python layer for our QT UI. And so i would currently vote to use the existing Python layer, since there is also a Python api available.

The general concept

First, it may surprise you after reading the intro, and all the talking about radical changes, but the current Blender UI is already half of our mockup. I don't want to fall into the common mistake to create the wheel from scratch again just because the current wheel is not round enough. Which would be to go from one extreme to the other extreme. It's not all bad. I like many things the way they are. There are quite a few things a good base already. But i dislike many things. I simply want to make the wheel much rounder. Evolution instead of revolution is the key.

When an issue is not described in this document, then i want to deal with it like in the old UI. This means for example stuff like dragging the little triangle in the corner to open a new editor window. Or the general settings like font and stuff. That's our starting point.

To give you a rough idea where i want to arrive: think of a UI mixture of Unity (see tabbed windows), Maya, 3D Coat and trueSpace (those three are the base for icon buttons for example). They all have some great UI concepts and elements. So that's a good place to borrow from. As long as those concepts and elements fits into the main concept.

But at the same time none of them. To put an existing UI solution from another software on top of the existing core will not work.

What i want to achieve is the best of all worlds in one UI. With a fluid workflow, an intuitive surface, and an easy learning curve. But based at what we already have.

The new UI should and must be an EVOLUTION of the CURRENT Blender UI.

And i will lead you to this new UI by all the evolutionary steps that i show in the next pages in this document. I will explain the problem, explain why i want to change it, and i will provide a solution then.

The new UI will NOT be completely foreign and different. This will not work. But we need drastical changes here and there.

As told, the new UI will be an advanced version of the current Blender UI. But there just has to be some dramatic changes. It can not be one of those homeopathic versions that you can find so often as a proposal for a new Blender UI.

To put just a cluttered element at another UI location or just to recolour some UI elements will not do the job. Unfortunately that's what you mostly see when somebody makes a so called Blender UI proposal. Most of them thinks too short.

To think too short was already the problem with the introduced tabs for the Toolshelf. There is nothing really solved by it. Now we tab like crazy instead of scrolling like crazy. The main problem is not solved: there is simply too much wasted UI space. The elements are too cluttered in the single panels, so that you need tabs or scrollbars at all.

We cannot make all the needed steps at once though. This would kill us. For now the focus is at implementing the QT UI solution. And while at it we can also make the first drastical cuts. We are at redoing it anyways. But not everything. Some changings doesn't only need a change at the UI layout, but also how tools works. And that's a job for after the new UI gots implemented.

What will definitely arrive in the new UI is icon buttons besides the text only buttons. Since this solves quite a few space problems. And more Tabs. But implemented in a way that makes sense and brings really improvement. Introducing tabs to solve the scrolling dilemma was not really the best solution.

Since it did not fix the problem that leaded to the lots of scrolling at all. And this was caused by the lots of wasted UI space in the panel elements.

The menu bars of the editors should only contain menus or tabs. And not tools. It is planned to make some toolbars dockable at menu bars though.

There must also be a separation between tools and settings. A slider for example has much wider dimensions than a tiny button to call this slider. And this settings must all be found in a single location. Think of the Inspector in Unity.

What also is planned is a way to arrange and to tab the editor windows like in Unity for example.

But let's not overdo it. The current goal is just the UI rewrite, not a completely Blender core rewrite. This would kill us. Here we are at the point again that we cannot do everything at once. One step after another. We need evolution, not revolution. There is simply too much to fix to do it all in one gigantic step.

So there may be a few points missing in the first QT incarnation. For example, i am not this sure if we manage to have a scalable UI out of the box. And the first incarnation may also not be themeable. This all needs further investigations and development. But QT is capable of vector graphics. So there will be surely a scalable Bforartists UI at one point.

All in all we will have a new UI when we are done, but it will not too foreign from the old Blender UI. A former Blender user should still be able to jump into Bforartists without too much headache. The functionality remains unchanged after all. Those needed changes will happen in a later development cycle. But new users should be able to pick up Bforartists much easier then. And we will have a healthy base for further changes. That's the goal for now.

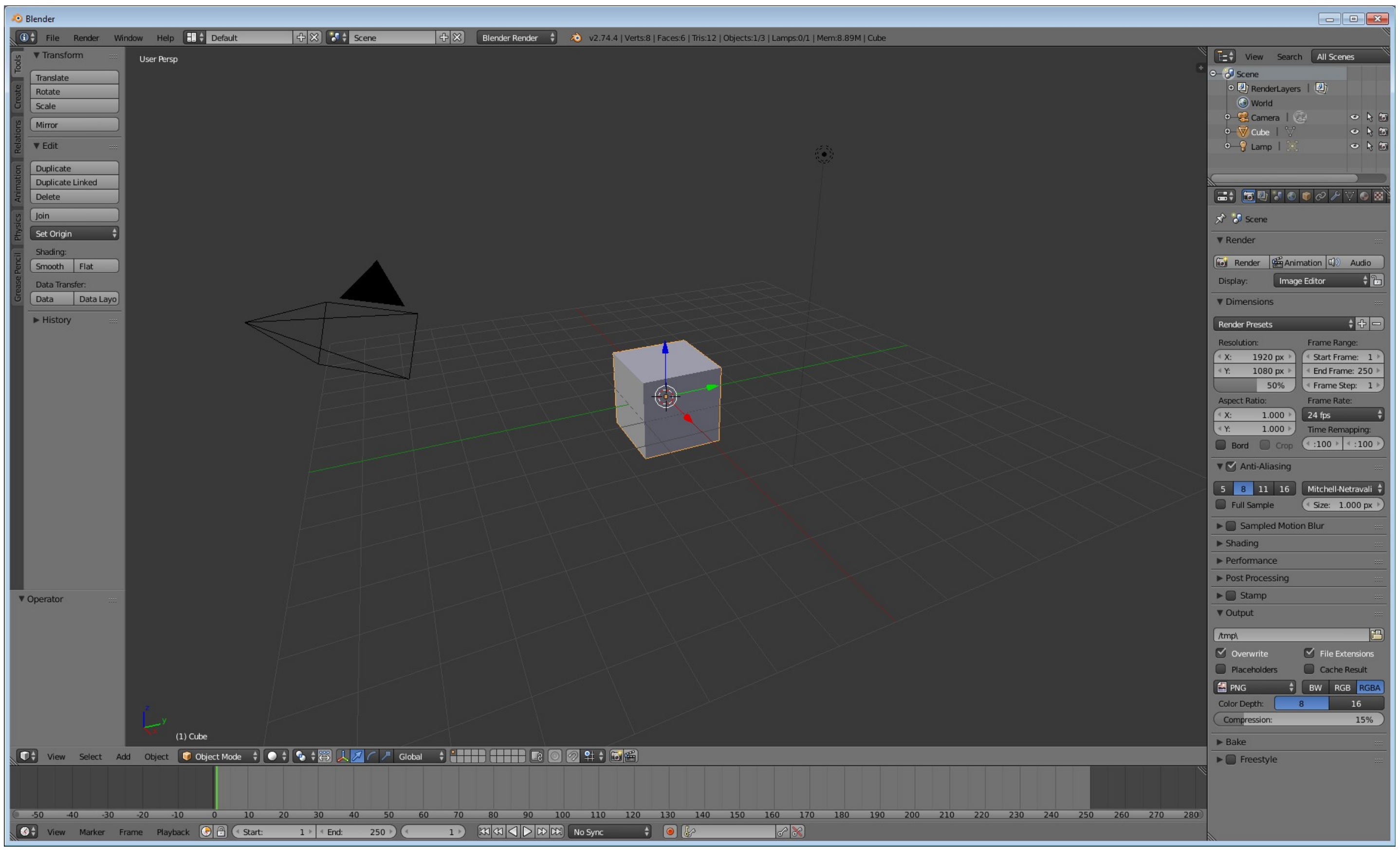
And i will lead you towards this goal in the next few pages. Step by step.

When you cannot wait to see the final result of the redesign, then go to page 38 immediately. You will miss the ideas behind the layout then though.

Default Layout - Change standard layout

There's nothing to code here. I simply rearrange a few windows here to have a base layout for our further evolution. But is a good first step to have a look at some problems.

This is what you get when you open Blender the first time. A hard to read UI with grey in grey. With too much elements that makes no sense. The eye finds no rest. You are usually lost here when you first have a look at it.



Let's do a few first steps. And clean up the standard layout that you will find when you open Blender.

One of the main problems is the grey in grey layout already. It's hard to find the element that you are looking for. Note that the colour that i use in the next shot is just an example for a better readable theme, since i have this theme already, it's my currrent favourite Blender theme. Recolouring the new UI layout, to theme it, is a task for later. But for now i use this theme here since the nearly unreadable grey in grey makes me crazy for even this job here.

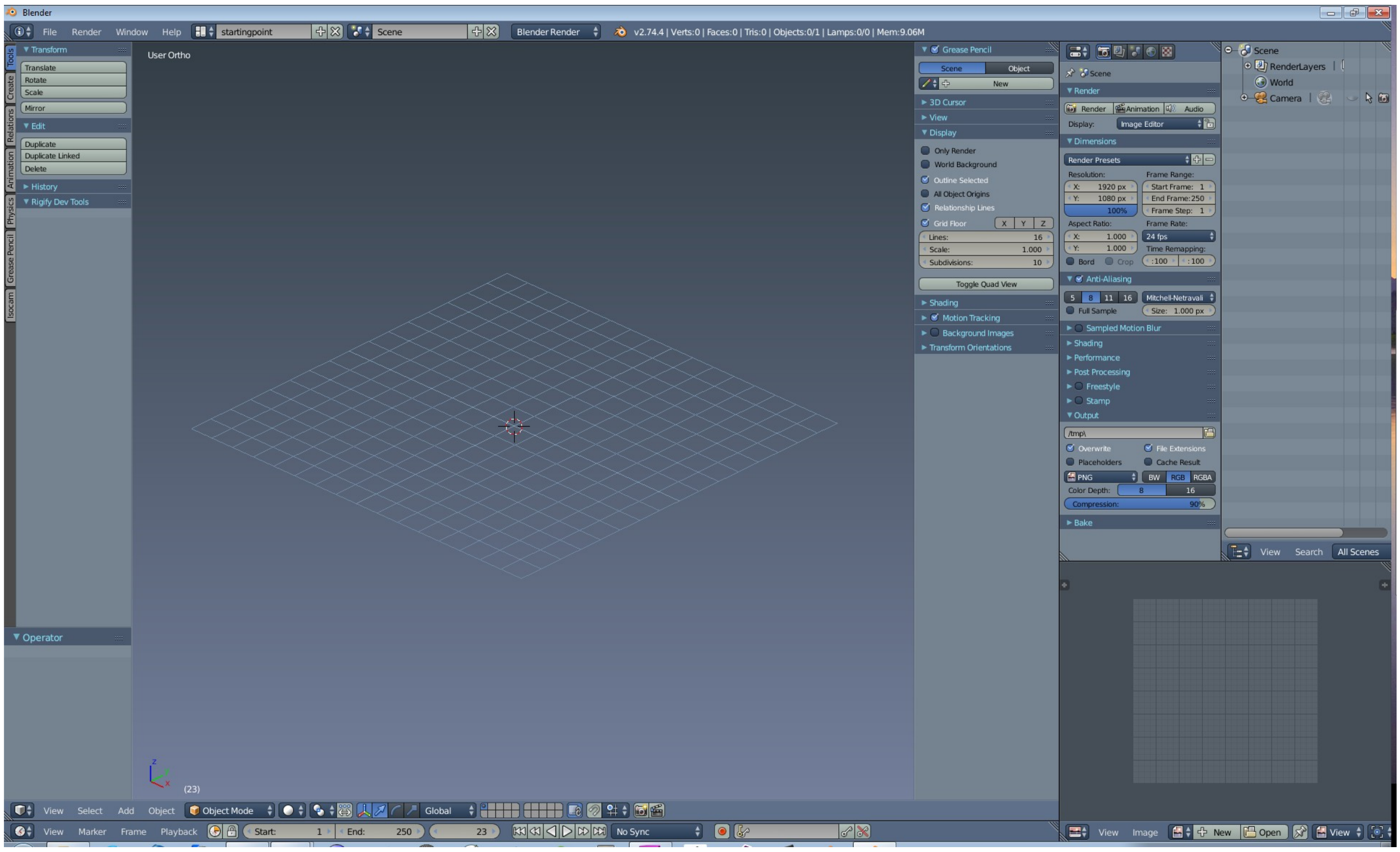
Important at this point is the arrangement, the layout, and the new designed UI elements. Not so much the colours. I just wanted to show that the colours are also a problem though.

Another nagger is the standard cube, a not neccessary light, and the visible camera. The outliner is tiny, with already a space wasting hotizontal scrollbar. And there is not this much space for new content left without adding a vetical scrollbar then.

And the timeline is open. That's a job for the animation layout though. We don't need to remove it completely. But we can close it. And pull it up when needed.

Let's change this. Now our layout looks like below. I have changed the colours. The outliner is now in an extra row. The timeline section is closed. There is an Image Editor window open as a place holder. I want to do something special in this area at a later point. Both, the Tool Shelf and the

Properties Bar is opened. And the standard cube and the light is gone.



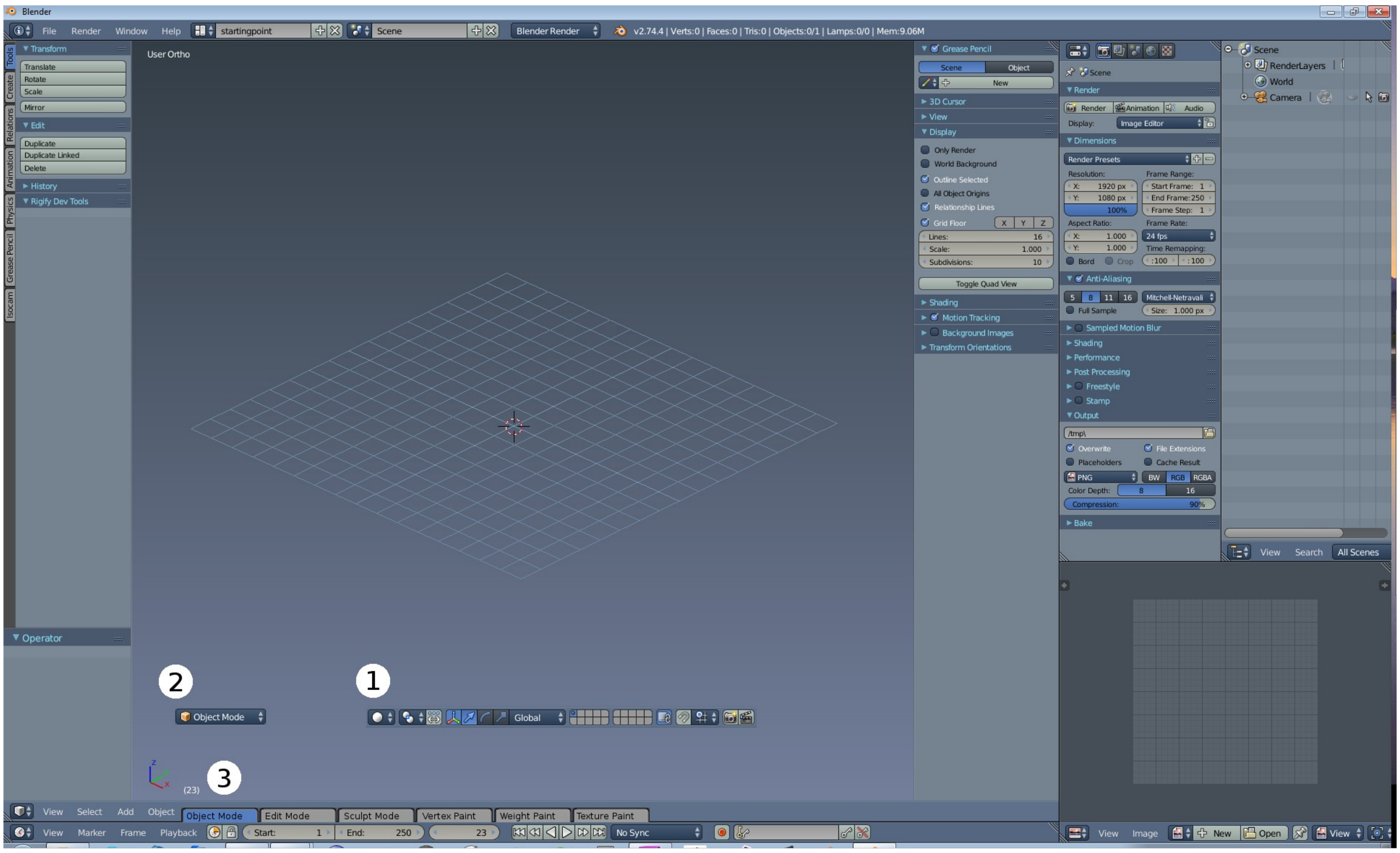
Looks much friendlier and a bit better readable now. And has solved the problem of a too cluttered UI already a bit. Even when it looks still pretty confusing to a newbie. But we will change that in the next steps.

3D View - The menu bar

Let's do a first cut. We start in the 3D View, since this is the most used editor window.

One of the problems in Blender is that the menu bars contains tools and settings. That's a big problem, since you cannot drag a windows smaller in width than the position of the very right tool. Or you will hide the tool behind the neighbour window. In practice this means that you are always at pulling the layout around to reach those tools then. And in the animation layout you cannot arrange the windows without to have hidden tools at even two monitors.

Let's put the tools into a temporary toolbar for now. We will refine the toolbar and the content of this toolbar at a later point. Some stuff will move into a toolbar, some stuff will go to other locations. And the toolbars will change their look dramatically in their further evolution.

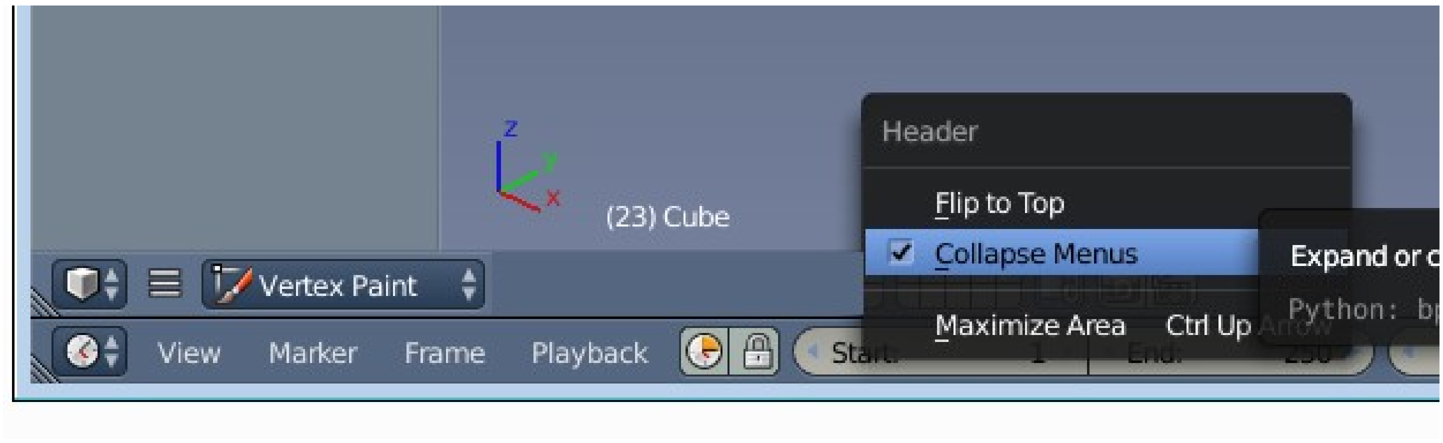


1) The tools that does not longer belong into the menu bar are separated now. We will tackle this issue and sort them when we have adjusted the rest of the workspace. What exactly goes where is a job for the prototype then.

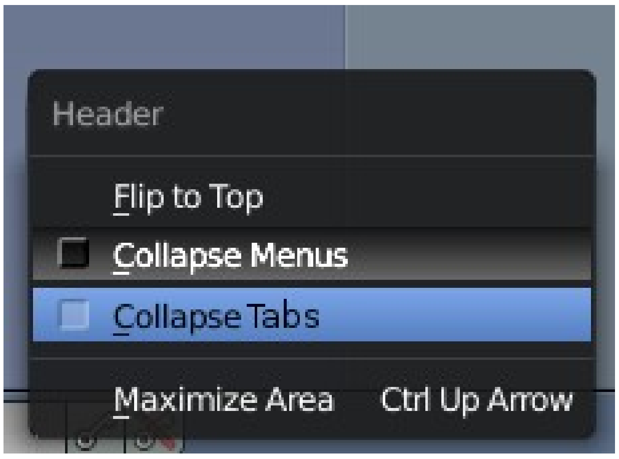
2) A Dropdown box to switch between modes is fiddly. Proper hotkeys doesn't exist neither. Just some crazy forward backward system with the tab key. Also something that needs a change.

3) Now we have enough free space to put tabs where before the tools were. Note that i have removed the icons here. Icons makes imho no sense at tabs that contains the full name. Alternatively we could also provide icon tabs. This would also be a slim solution.

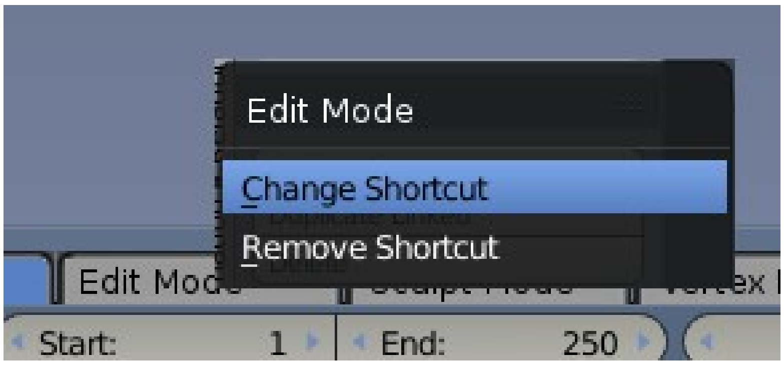
I would nevertheless prefer to keep the dropdown box for the collapsed menu. This would, in conjunction with the Collapse Menus lead to a super slim menu area too.



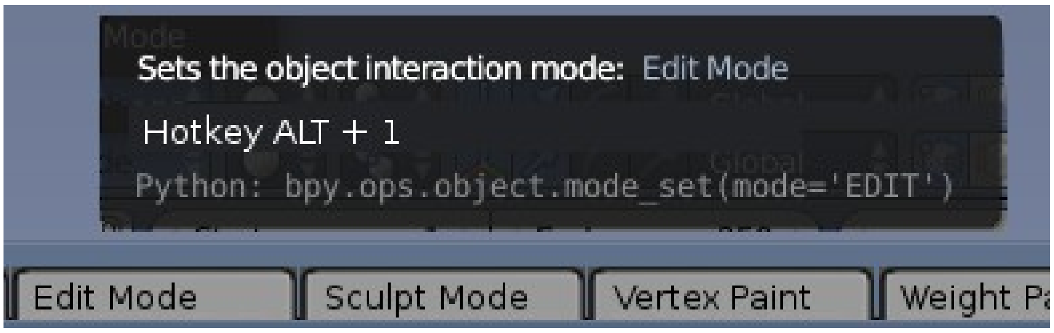
What we need to do though is to make the user decide if he wants tabs or not, independand if the menu is collapsed. So we expand the right click menu ...



Further functionality change: the Modes needs a unique hotkey, each. The current method with tabbing to go back to the previous mode is super cumbersome. And there is no other way than to click at the item at the moment. So let's add a RMB menu to the tabs where you can change the hotkey.

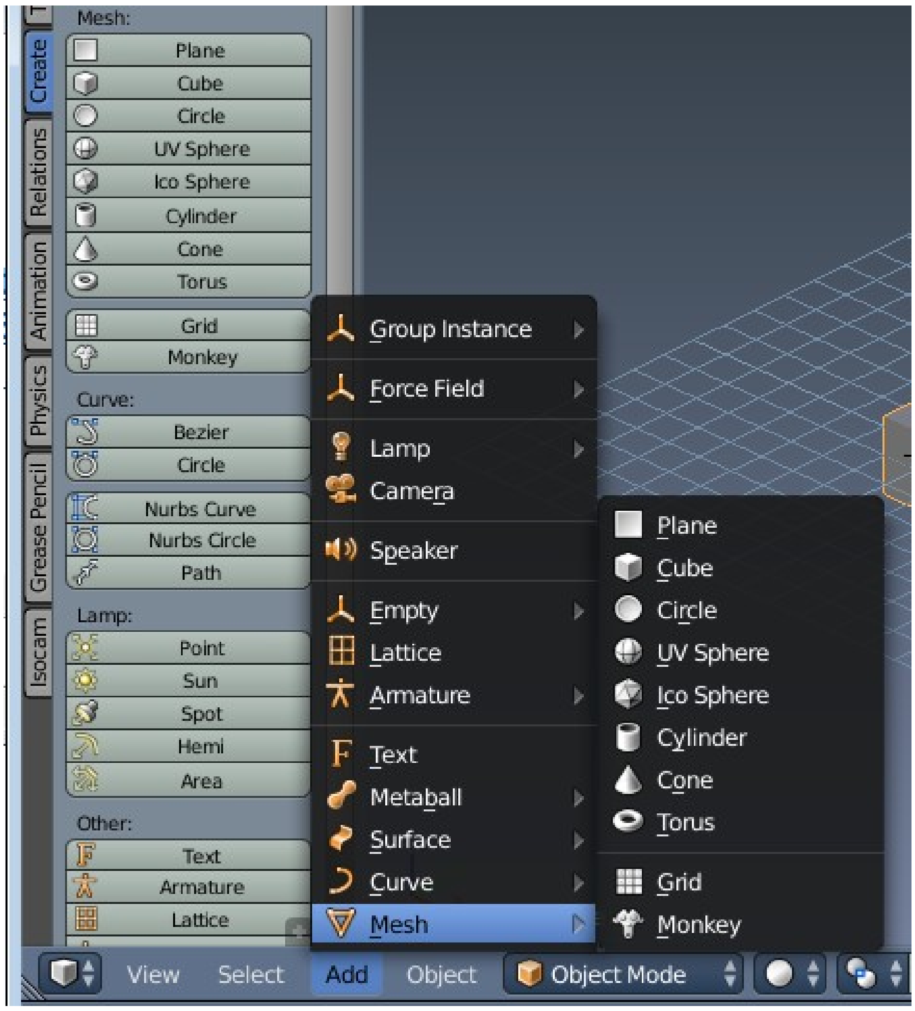


Now we need an entry in the tooltip for the hotkey.

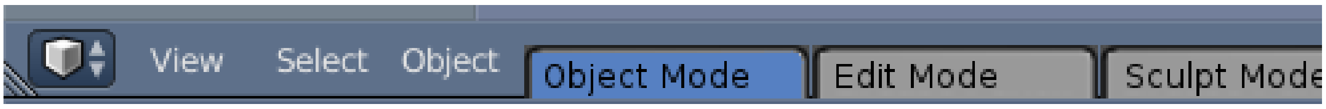

Sets the object interaction mode: Edit Mode
Hotkey ALT + 1
Python: bpy.ops.object.mode_set(mode='EDIT')
Edit Mode
Sculpt Mode
Vertex Paint
Weight Pa

Possible addition: display the hotkey and an icon also in the tabs. But this would bring a size problem again. Icon is not necessary since the tabs are already labeled with letters, and it's just a few. Icons would imho just add visual noise. And we have the hotkey already displayed in the tooltip. We could of course also use just icons for the tabs. Then there would be enough space for the hotkey too again. Icons or not, maybe even icons only, is something to evaluate in the prototype.

Further functionality change: The current layout contains a double set of ADD functionality. We have a create tab in the tool shelf. And we have a Add menu in the menu area.

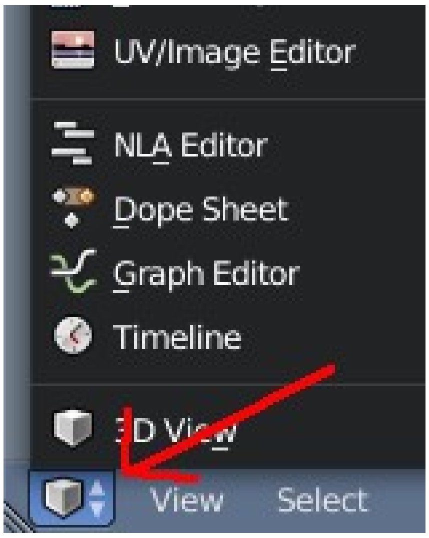


Both contains the same feature set. Both does the same. Means one of it is not longer needed. I prefer to add such stuff by a toolbar. So i remove the Add menu.



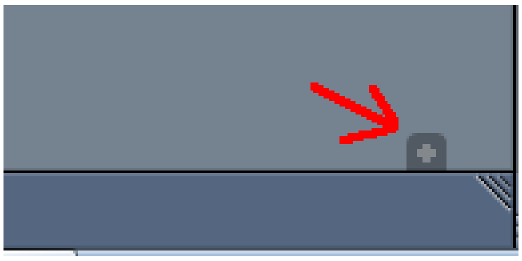
Now there is just View, Select and Object left in Object Mode menu.

What remains unchanged is the little icon down left and the with that connected menu to switch between the editor windows. It is perfect as it is.



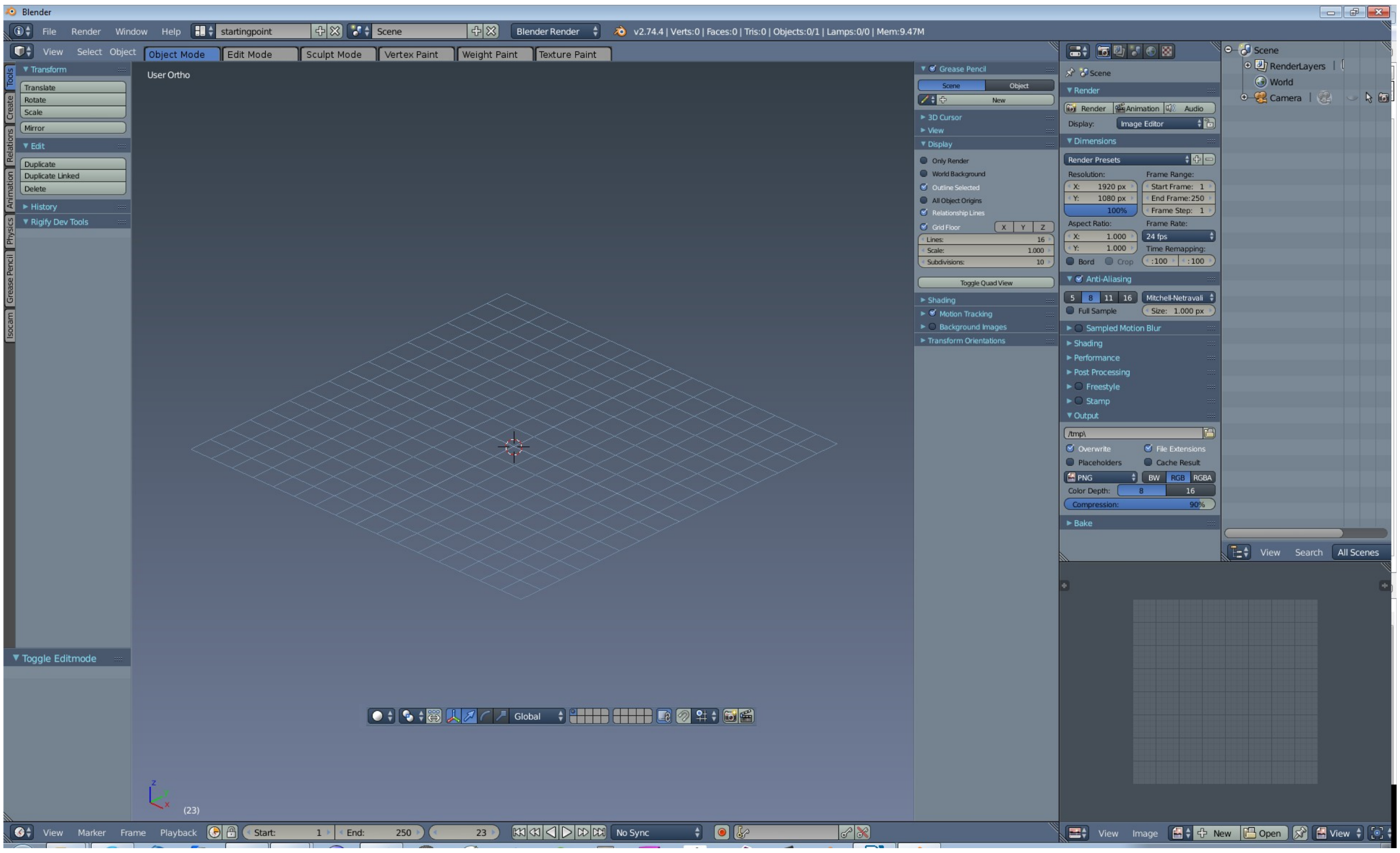
What remains unchanged is the RMB functionality to flip the menu bar to the top and to collapse the menu.

What remains unchanged is the functionality to hide the menu bar, and having a little icon to bring it back.



Nearly done with the Menu bar of the 3D View. People expect menu bars to be at the top of a window. So let's flip the menu bar to the top.

And after all those steps our temporary result looks like in the next shot:



```
v2.74.4 | Verts:0 | Faces:0 | Tris:0 | Objects:0/1 | Lamps:0/0 | Mem:9.47M
Start: 1   End: 250   23
Resolution: X: 1920 px  Y: 1080 px  100%
Frame Range: Start Frame: 1  End Frame: 250  Frame Step: 1
Aspect Ratio: X: 1.000  Y: 1.000
Frame Rate: 24 fps
Display: Image Editor
Output: /tmp\  PNG  RGB  Color Depth: 8  Compression: 90%
```

We have a clean menu bar now, with a few elements. It's at the top for UI standard reasons. What we also have is some tools now floating around where we don't know what to do with them. I will come to this toolbar at a later point.

The tools and Settings dilemma

One of the biggest problems in the Blender UI is the tools and settings. And there are various problems here.

Tools and settings can be virtually everywhere. There is a menu with tools and settings at the top, there is a menu with tools and settings at the bottom, there are tools and settings at the left, there are tools and settings at the right. Some stuff is even located in other editor windows. Like the pack islands settings from the UV Editor. And stuff like object settings and modifiers are completely separated anyways.

Sometimes the settings for a tool are directly asides the tool, sometimes they are far far away where you would never expect them.

Then there is the Properties and Toolshelf concept for the editor windows. In extreme cases you need to have the whole toolbar open for just three or four buttons. And the rest of the Tool Shelf covers the workspace for no reason then. This eats extremely lots of space. And that the buttons are text only buttons doesn't really make this problem any better. Since the text buttons needs to be as width as the longest word.

Having the settings asides the tool has a big plus when you look at it isolated: short mouse ways from calling the tool and adjust its settings. The problem with having the settings asides the tools is that you may have a icon button for the tool with 32x32. But the slider for it needs 128 pixel width to be useful and readable. And the problem with having the tools separated from their settings leads to searching since there is no central place for it. Plus it makes longer mouse ways overall. Because of the cluttered surface.

All in all this concept is highly inconsistent. And this all leads to the situation that even experienced users are always at searching.

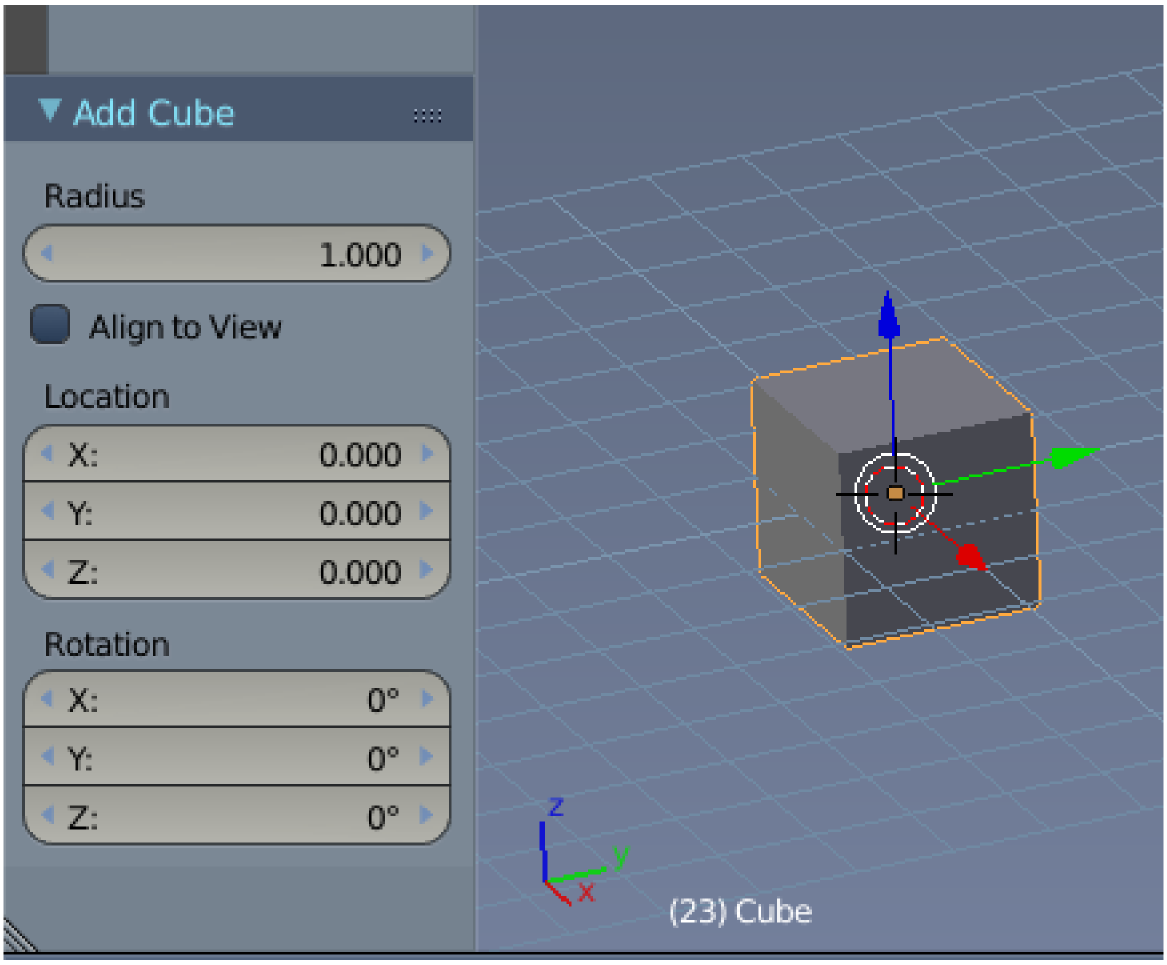
So we have quite a few problems to solve here. And we will.

New Editor Type - Inspector

Let's do one of the biggest cuts. We just need to separate the tools from the settings. And give them both a fixed place in the UI then. There is no way around it. You cannot make a toolbar that contains all settings for the tools already. Well, you can, Zbrus shows how. But that's a completely different UI concept, far away from any standard. We cannot use that concept in Blender without throwing out everything. And ZBrush is known for having the same steep learning curve than Blender. Now guess why.

There is of course a little caveat. As a consequence of separating the tools from their settings we need the inspector in every layout where you have tools and settings. But the price is well paid by getting a much simpler graphical UI. And currently we have something like the Inspector in every Editor anyways. The right toolbar ...

Anyways, i have chosen to go the separation way since it is much cleaner and easier. And for that we need a new editor type. The concept of separated buttons to call the tools and their settings can be found in many applications. Even in Blender. Here it's just very inconsequently designed, it just works for a very small fraction of functionality. Just have a look in the Tool Shelf at the bottom. There we have exactly that. When you create a primitive then you will find the settings for them there. And quite a few other things will appear here. Like external plugin settings.


Add Cube

Radius: 1.000

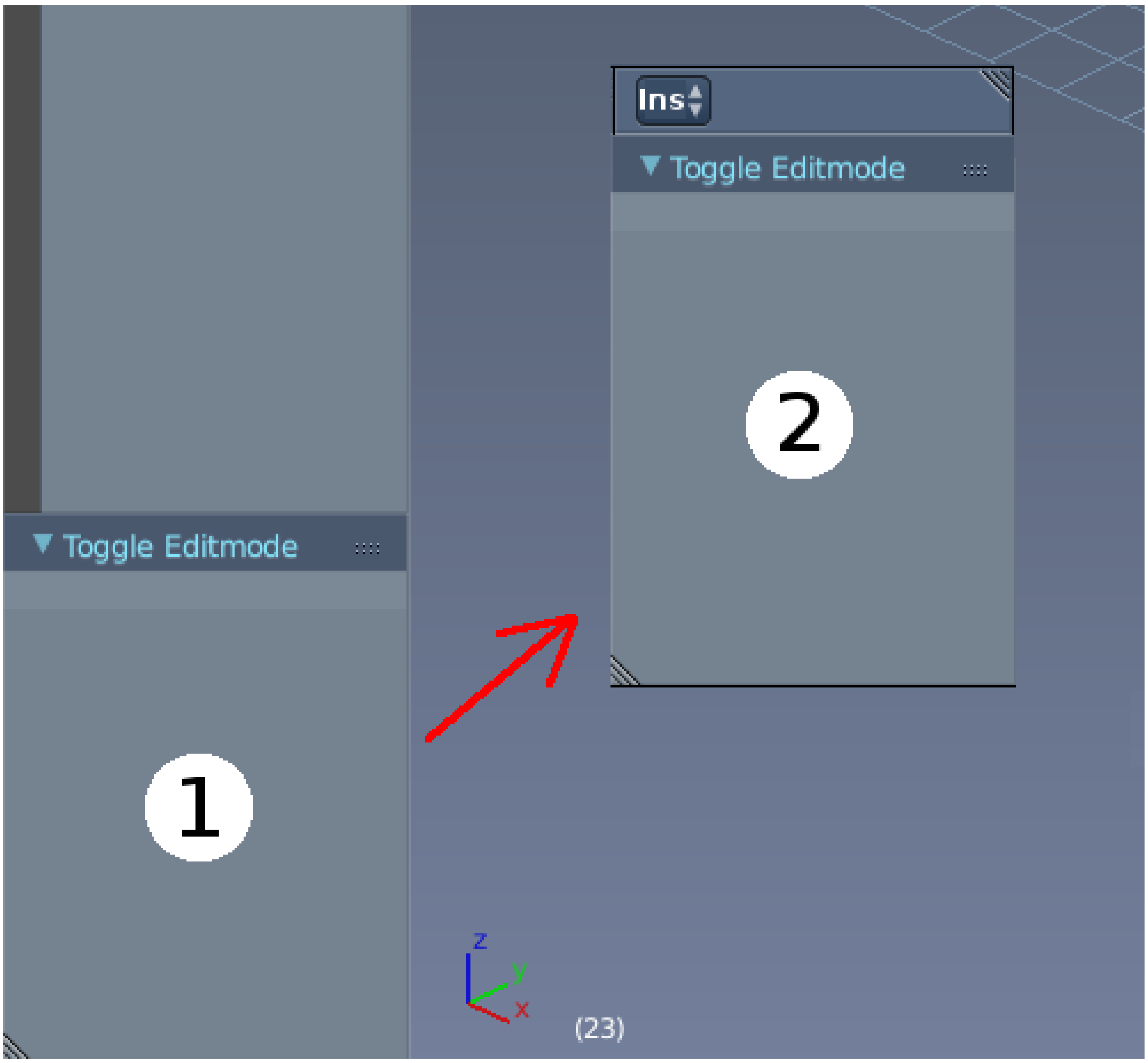
Align to View

Location
X: 0.000
Y: 0.000
Z: 0.000

Rotation
X: 0°
Y: 0°
Z: 0°

So let's do this cut and add a new editor type. I will call this new settings editor Inspector like in Unity. From now on all Settings for any tool has to appear in this Inspector window. And not longer at many different locations in the UI.

Let's create the Editor type called Inspector.



1) the former info area in the Tool Shelf becomes a separated Editor, and gets removed from the Tool Shelf.

2) our new Editor called Inspector. From now on all Toolsettings have to appear here.

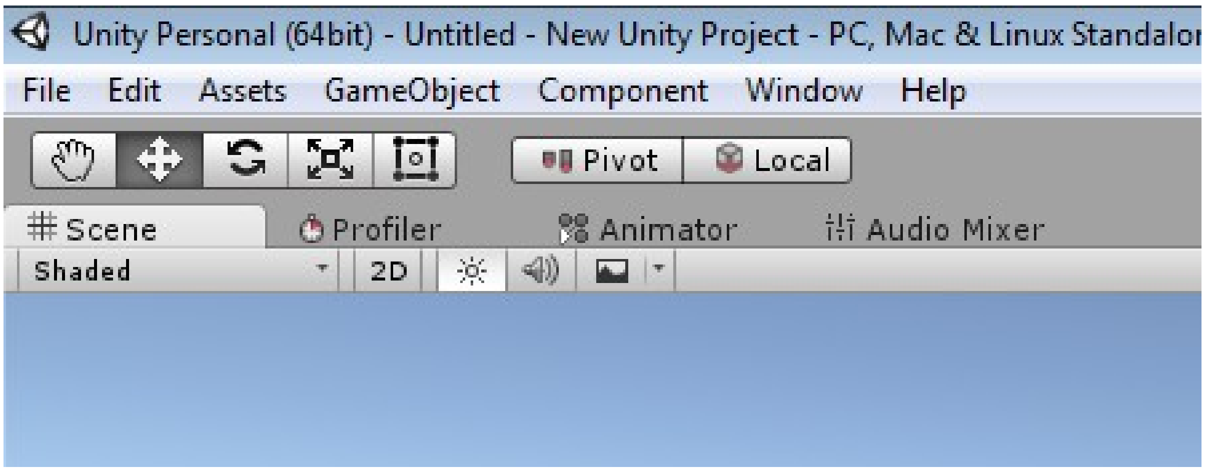
Concept note: a new called tool opens at the top of the list. And can be closed when not longer needed. Plus corresponding panels closes when you leave the mode they are used in.

See also Refining the Inspector. We do some further modifications there.

Tabbing editor windows

Our new created editor window has a problem at the moment. It is floating. Blender's UI concept is non floating though. Note that you CAN undock editor windows by holding shift, and drag the window at this little triangle handler in the corner. But the general UI concept is non floating. There is no way to add the floating window back into the non floating UI afaik.

What we can't do at the moment is to tab editor windows behind each other. That's a concept that you can find in Unity for example.



Tabbed editor windows is a great concept to organize many editor windows to which you need permanent and fast acess to. And would definitely be a workflow improvement.

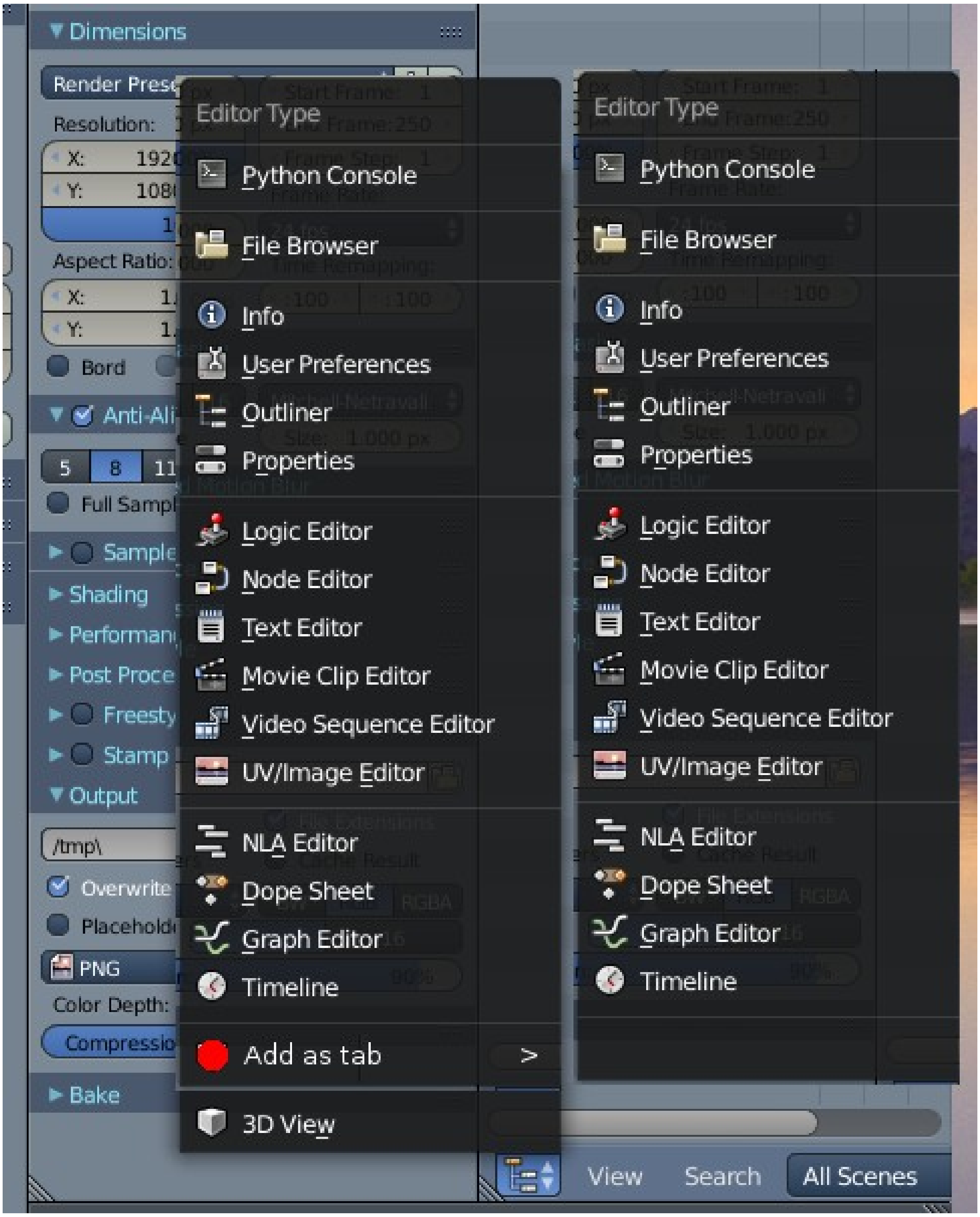
Think of a tabbed image editor behind the 3D view editor, that you can access with one click. Or the needed editors for animation. Dope Sheet, Graph Editor and Timeline. Without the need to change the complete layout, by just one click at the tab.

To use the concept from Unity does not work directly in Blender though. As told, the current concept is non floating windows. So how do we tab editor windows when there is no way to drag them out and dock them together at another location?

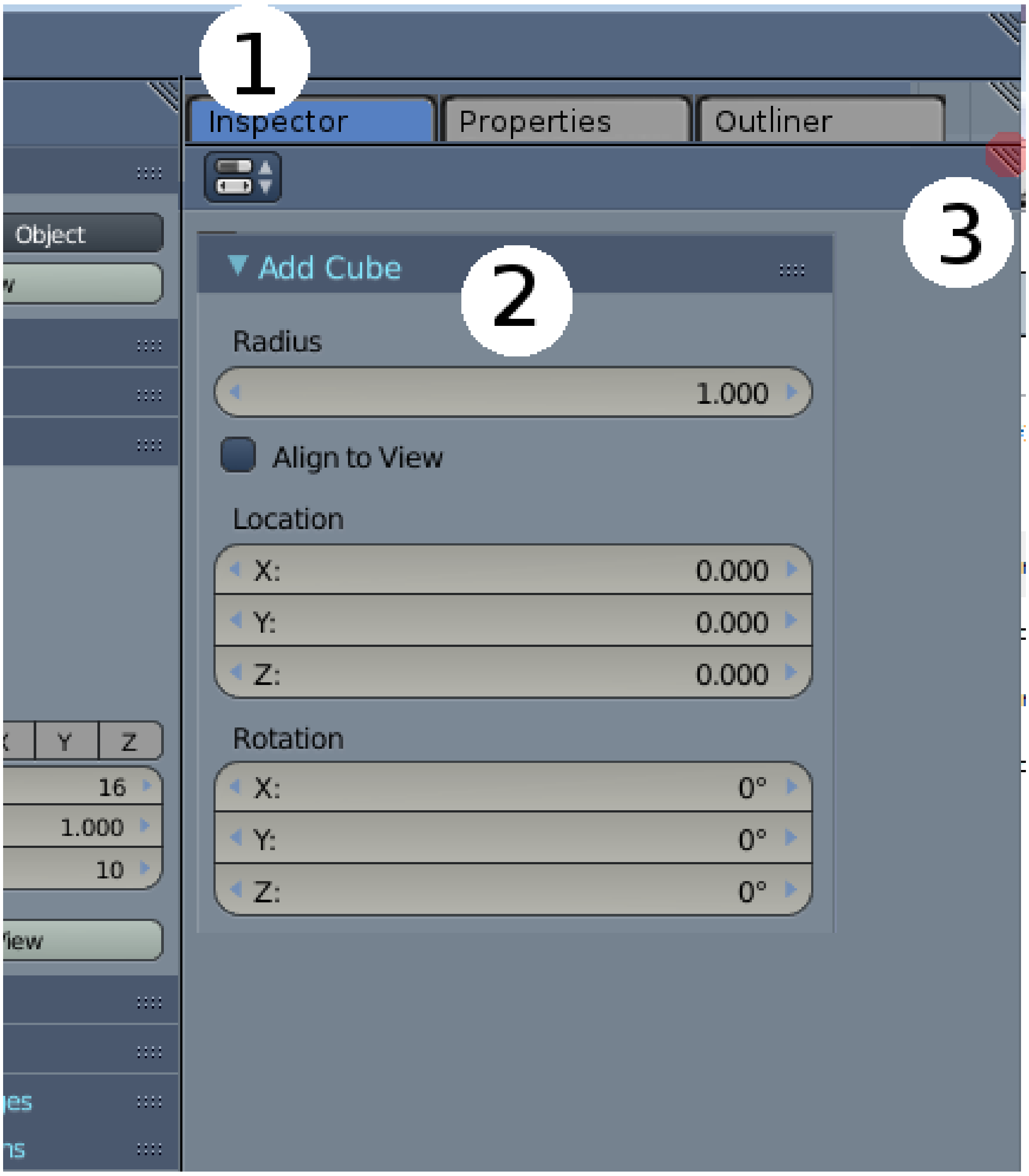
One way would of course to be to handle it like Unity, and make the editor windows simply floatable and dockable. But this means to add tabs to every editor window so that you have handlers to drag them around. Which is wasted space when you don't tab a editor window together with another editor window. And the tabs and their bars adds visual noise and eats UI space.

There is a more elegant solution. Show tabs just where there are tabbed editor windows.

Remains the problem how to add a editor window to another editor window as a tab. This could be done by a new menu entry in the existing editor type menus. See next image.



And now let's do some magic with our new created Inspector editor, the Outliner, and the Properties editor ...



1) Tabs. To easily switch between Inspector, Properties and Outliner. The editors are now very fast accessible with one click, without being visually in the way all the time.

Alternatively as for the Modes tabs, we could provide a dropdown box solution too.

We may need to develop two different tab styles to have a visual difference. One for the modes in the toolbars, one for the editor window tabs. The tabs in this shot are a bit too big at the moment for my flavour. Maybe we could limit the maximum text length here. Which would allow much shorter tabs.

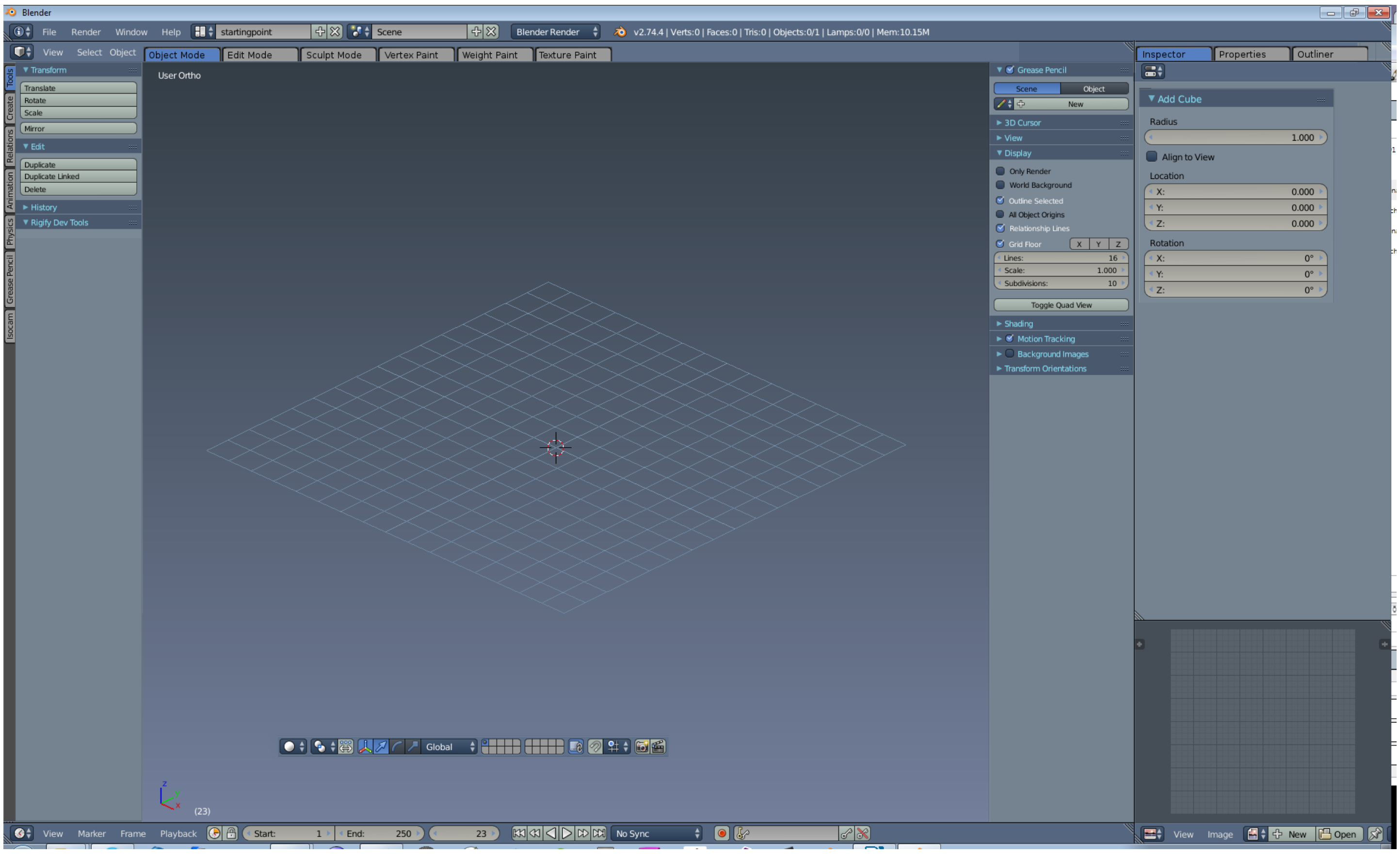
2) Inspector now showing the add cube properties when inserting a new cube. Without the need to open the Tool Shelf and to drag the small window at the ground big enough to see and to edit the values. Time saving. And now we have a central place for all tool settings.

3) By becoming a tabbed editor window we need to remove the drag triangle from the single editors. We can still split drag this editor window by the drag triangle asides the tabs. See above the red marked drag triangle.

Note, to deal with the case that tabs gets hidden i would suggest to deal with it like Blender does already in the Properties with the icon tabs. MMB + drag to scroll the tab area around.

For further modificatons see the point Settings Toolbar. We will add a few more things like a scrollbar, tabs, and a way to close the panels in the Inspector. That's why there is currently the space at the right side.

This task is done. Let's have a look what we have after this step now.



Only Render
World Background
Outline Selected
All Object Origins
Relationship Lines
Grid Floor X Y Z
Lines: 16
Scale: 1.000
Subdivisions: 10

Add Cube
Radius 1.000
Align to View
Location
X: 0.000
Y: 0.000
Z: 0.000
Rotation
X: 0°
Y: 0°
Z: 0°

Start: 1
End: 250
23

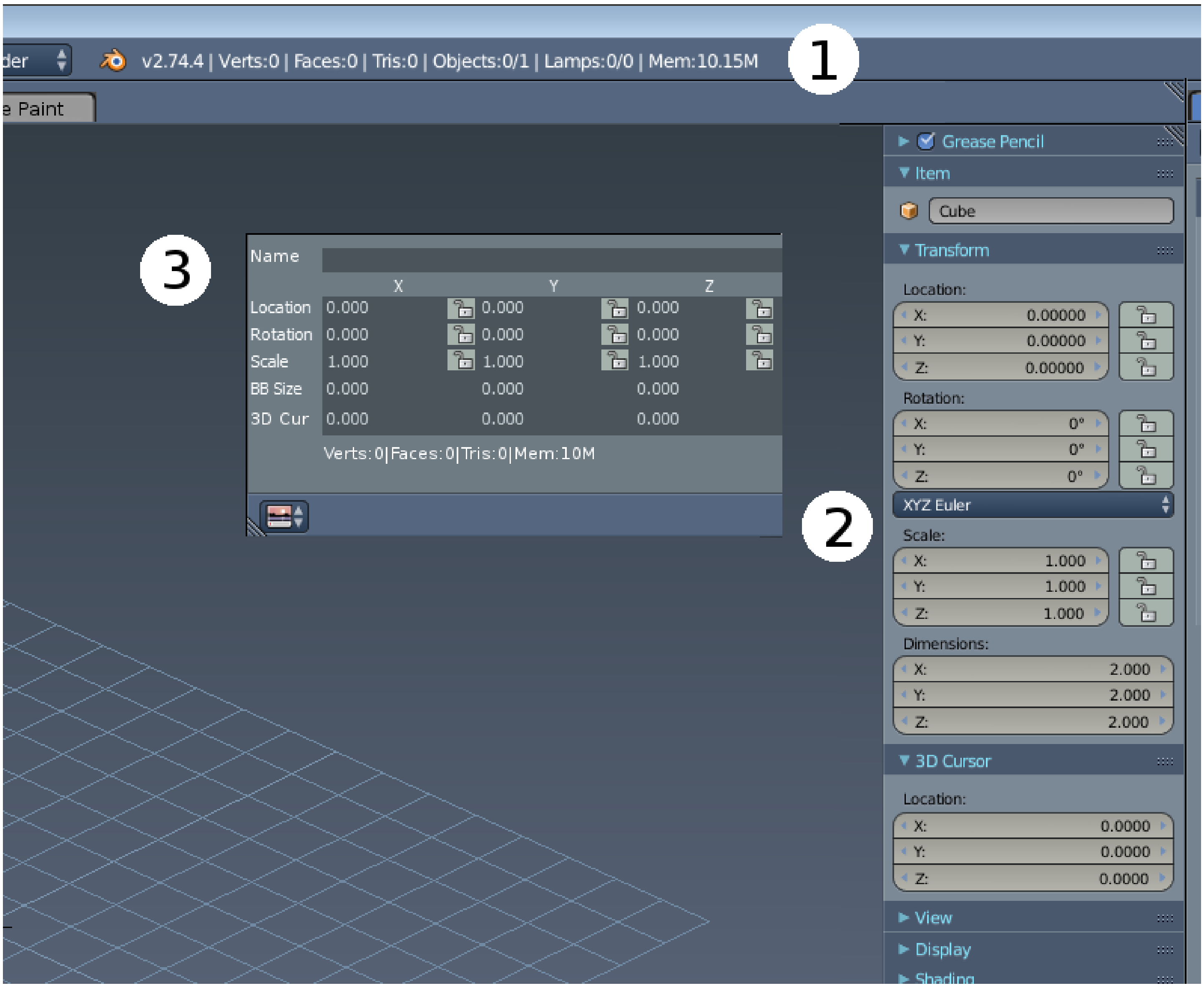
New Editor Type - Object Info

Let's add another editor type. One thing that always disturbs me is to keep the Properties Sidebar open to see or to edit the transform values of an object. So this in theory closable panel becomes in practice a panel that you cannot close really, since you need the values. And this Properties sidebar panel is full of visual noise too. You look permanently at settings that you may need from time to time, but that you rarely touch.

What i want to do is to separate the Transform, Item and 3D Cursor components from this properties sidebar, and put them into a new editor type. A Object Info editor that is permanently visible, and can be integrated into the current layout.

And while at it, we can also grab the info values at the top menu bar, and place it in our object info panel. As told before: menu bars have to contain just two things from now on: a menu and tabs.

Let's do it.


v2.74.4 | Verts:0 | Faces:0 | Tris:0 | Objects:0/1 | Lamps:0/0 | Mem:10.15M

	X	Y	Z
Name			
Location	0.000	0.000	0.000
Rotation	0.000	0.000	0.000
Scale	1.000	1.000	1.000
BB Size	0.000	0.000	0.000
3D Cur	0.000	0.000	0.000

Verts:0|Faces:0|Tris:0|Mem:10M

Item: Cube

Transform
Location: X: 0.00000, Y: 0.00000, Z: 0.00000
Rotation: X: 0°, Y: 0°, Z: 0°
XYZ Euler
Scale: X: 1.000, Y: 1.000, Z: 1.000
Dimensions: X: 2.000, Y: 2.000, Z: 2.000

3D Cursor
Location: X: 0.0000, Y: 0.0000, Z: 0.0000

1) The info area from the menu goes into the object info panel. What gets removed is the logo and the version number. This is no important information that needs to be displayed all the time.

What remains for now in the menu bar is the warnings.

2) The Item, Transform and 3D Cursor content from the Properties sidebar goes also into the object info panel.

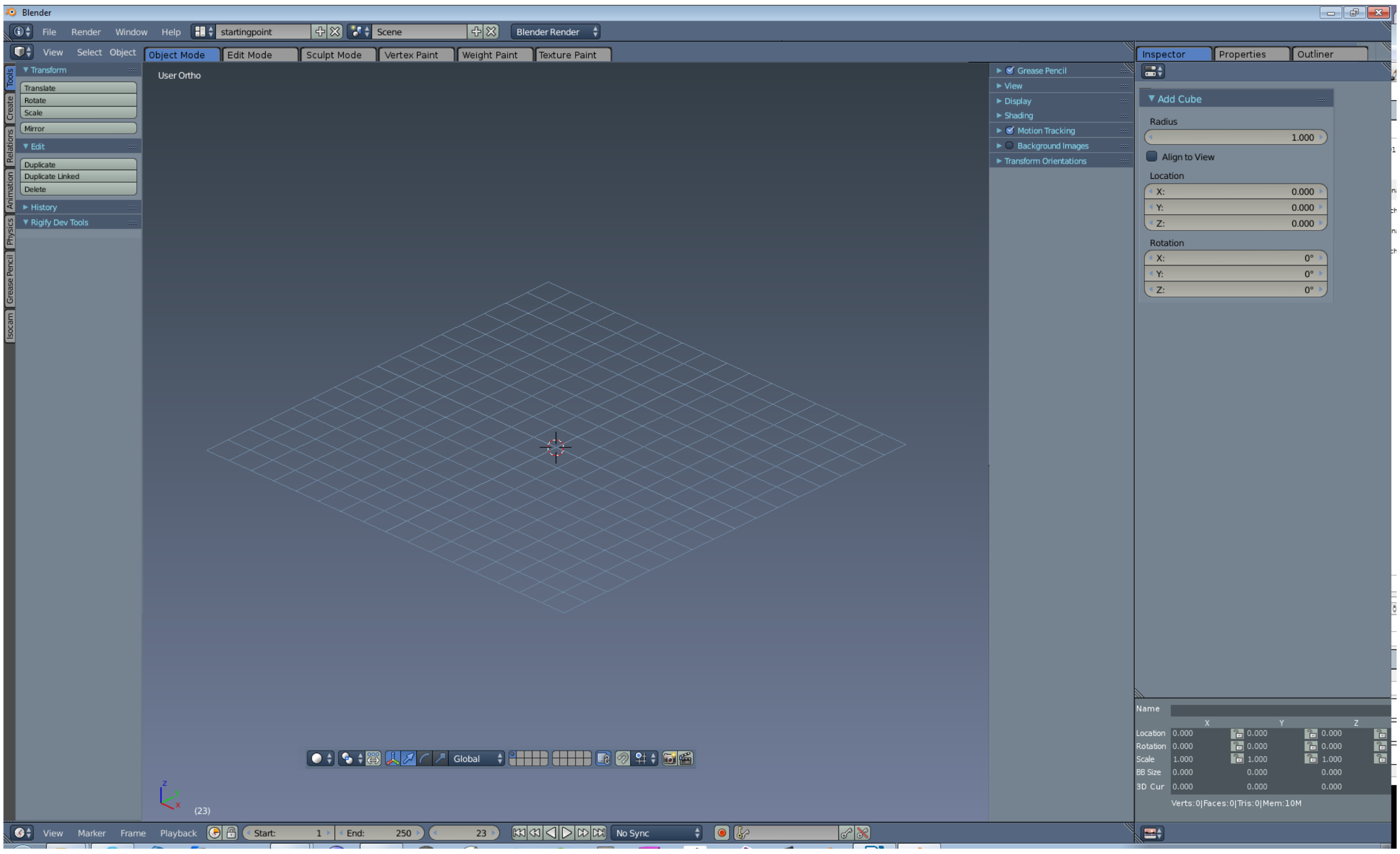
The dropdown box to switch between different rotation modes goes into the general settings in the Properties. This is a feature that you will rarely change. No need to keep it visible all the time.

3) Our new created Object Info editor window. It is much compacter, and integrates much better in the UI now. And when we collapse the menu bar then we win another 28 pixels in height ...

Let's place it in the layout. I had kept a UV Image editor window open as a placeholder down right for good reason. That's the new location for our object info panel.

Theming note: The lock icons needs to be made new. Much smaller so that they fit into the compact layout. Same goes for the little arrow functionality to scroll to higher or lower values.

This step is finished, and our result looks this way now:



Radius 1.000
Align to View
Location
X: 0.000
Y: 0.000
Z: 0.000
Rotation
X: 0°
Y: 0°
Z: 0°

Start: 1
End: 250
23

Name

	X	Y	Z
Location	0.000	0.000	0.000
Rotation	0.000	0.000	0.000
Scale	1.000	1.000	1.000
BB Size	0.000	0.000	0.000
3D Cur	0.000	0.000	0.000

Verts:0|Faces:0|Tris:0|Mem:10M

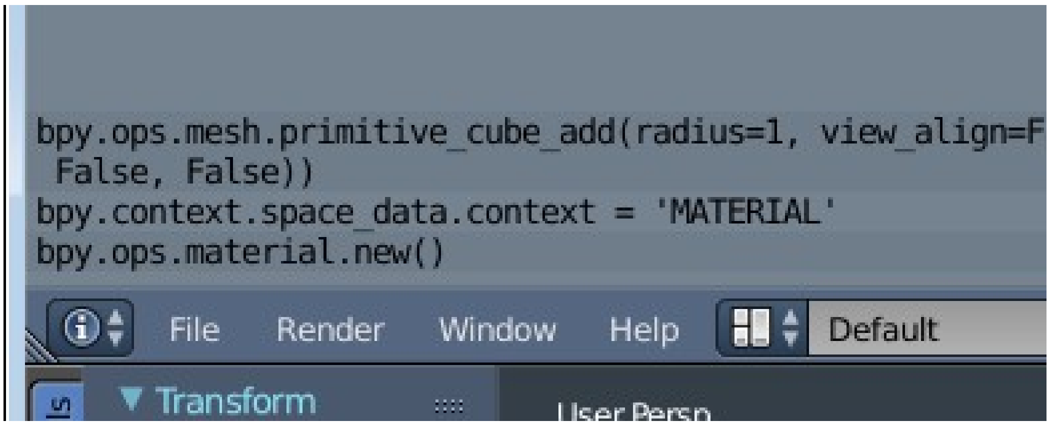
Info Editor - cleaning up the menu bar at the top

We have done some changings already at the top menu bar in the former step. And removed the info stuff, and have put it into the object info panel. So let's finish it. It's a menu bar. And remember, menu bars just contains tabs and menus from now on. Consistency.

We need to make a litte exception here though. But i will come to it later.

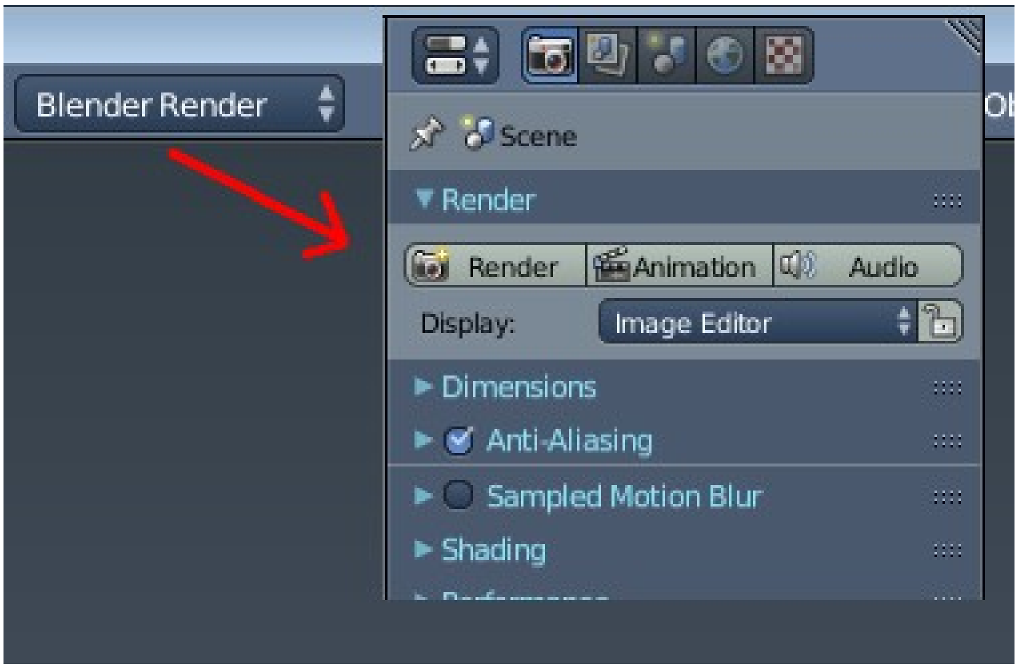
This menu bar is part of the Info editor that provides us with various informations. When you switch to the Scripting layout then you will find it twice. One time as a collapsed main menu bar at the top, one time as an expanded infor area for python scripting stuff.

You can also drag the top menu bar down, and the info area will appear there, with the Python commads for the last action. I want this to remain this way.

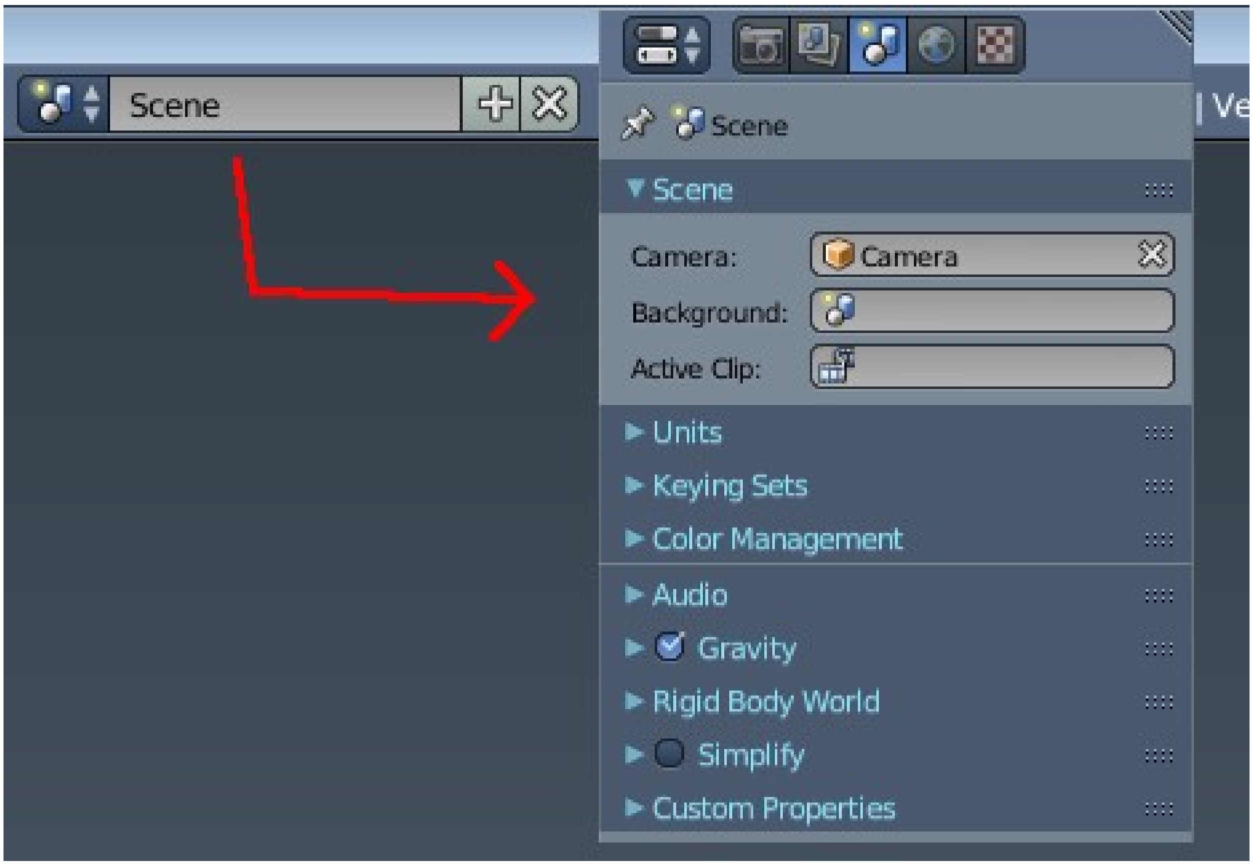


```
bpy.ops.mesh.primitive_cube_add(radius=1, view_align=F
 False, False))
bpy.context.space_data.context = 'MATERIAL'
bpy.ops.material.new()
```

Back to the menu bar. Here are still two items that doesn't belong into the menu bar. The dropdown box to choose the renderer. This belongs to the Properties, in the Render tab. Where the rest of the render setup happens. So let's put it there. It's not that you change the renderer every two seconds. And when you do, then you need to change other settings too.



And the other item is the Scene dropdown box, where you can deal with more than one scene at once. Something that i have never used so far. And i am pretty sure that nearly nobody else does. So it is at least not needed to have this item top UI level all the time. This belongs into the Properties, Scene tab.

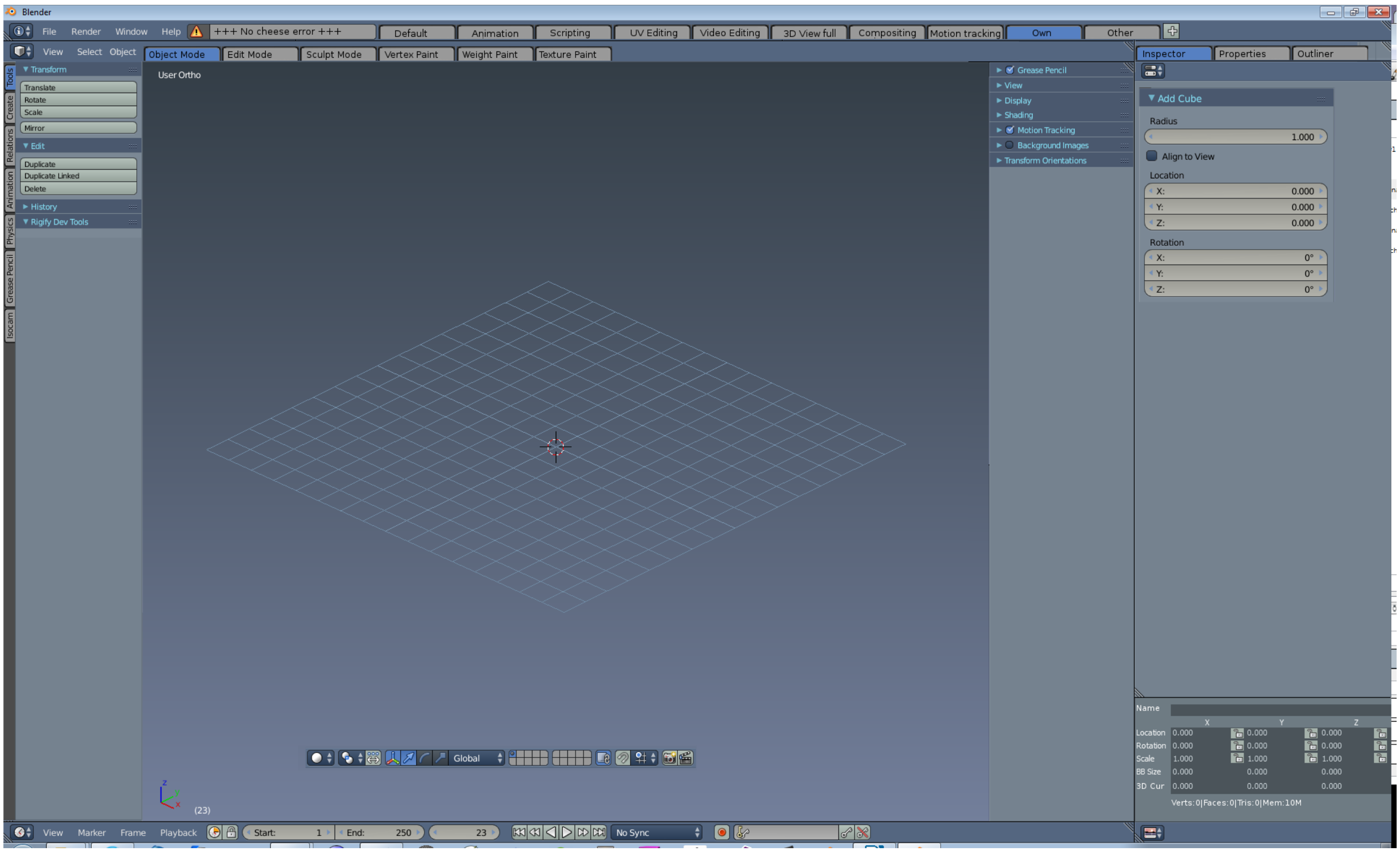


And then there is the Layouts dropdown box. I want to deal with it in the same way than with the menu bar that we have modified first. Make it tabs. But give it a way to switch to dropdown box to save UI space. The user can then decide by himself what he prefers.

The last item is about the exception that i have talked above. The Info editor menu bar shows the warnings. And we need a place in the UI that shows warnings. Our Object Info panel is not suited for this. This UI element may not be present in all layouts. The Info menu bar is. So the warning message remains in the menu bar. But we have tabs in the way now. Means we cannot place the warning right from the tabs. We need to put it left of it. And move the tabs to the right when a warning appears.

Tabs. For adding a layout tab we have an add button when tabbed. And it is thought that removing a layout tab happens by a rmb menu. When the user chooses the dropdown box method then it's as before.

Let's do this steps. Now our layout looks as follow. We getting closer.



Add Cube

Radius 1.000

Align to View

Location

X: 0.000

Y: 0.000

Z: 0.000

Rotation

X: 0°

Y: 0°

Z: 0°

Name

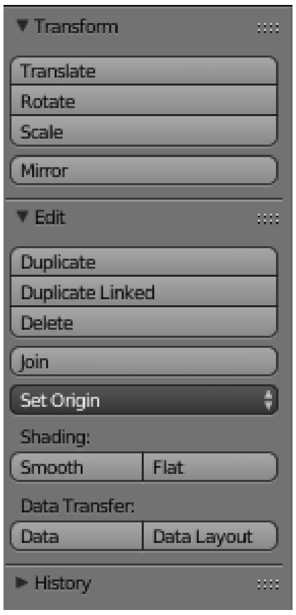
	X	Y	Z
Location	0.000	0.000	0.000
Rotation	0.000	0.000	0.000
Scale	1.000	1.000	1.000
BB Size	0.000	0.000	0.000
3D Cur	0.000	0.000	0.000

Verts:0|Faces:0|Tris:0|Mem:10M

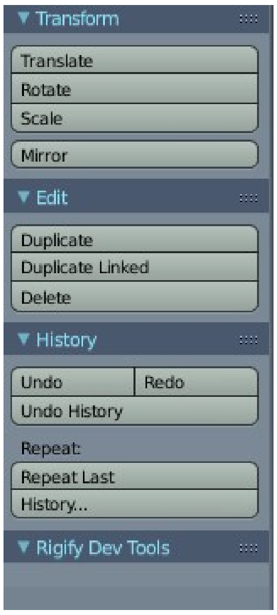
Start: 1 End: 250 23

Panels appearance

One of the things that makes it hard to navigate in Blender is that the single panels doesn't really separate against each other. When you have more than one panel open then it's hard to say where the one panel ends and the other starts.

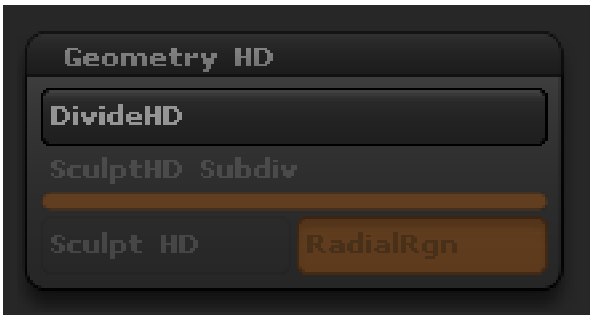


The preferences provides us with a way to make at least the bars of the panels in another colour. So that's a bit better then.

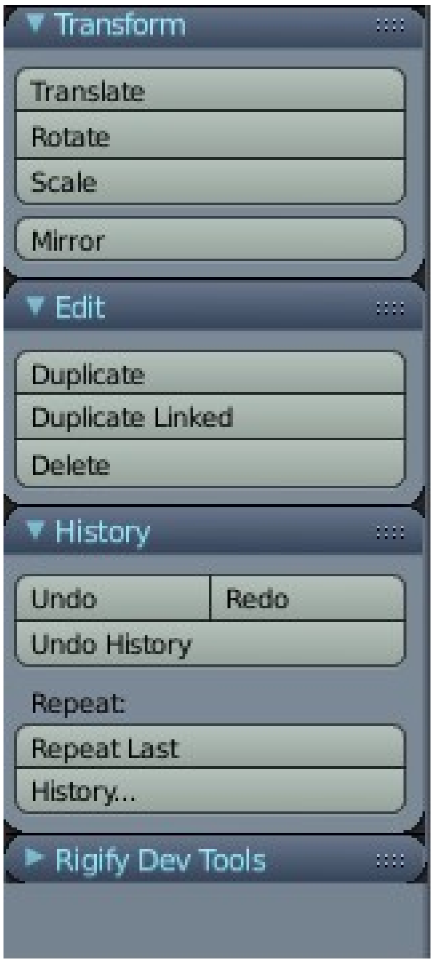


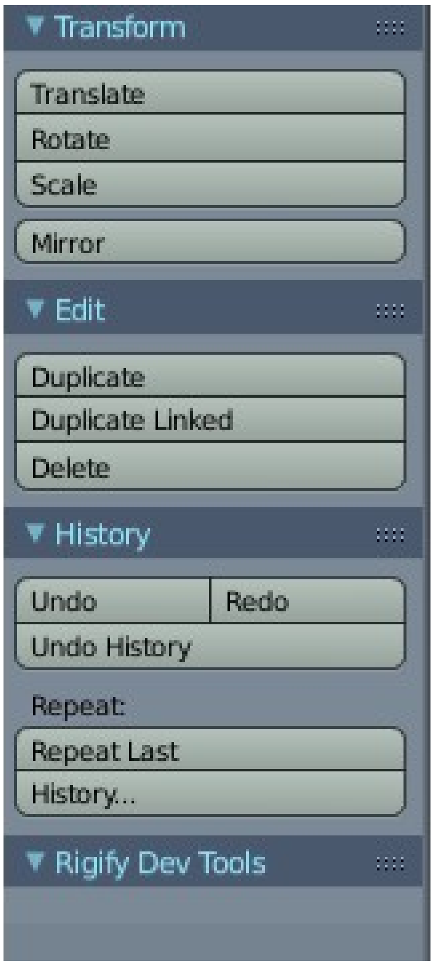
That's still far from optimal though. The bars are now visually disturbing when you make them strong enough to have a visual effect. And when you collapse them then you have the same dilemma again. They don't separate.

There are ways to separate sections better though. Just have a look at other 3D packages. Zbrush for example uses rounded corners, a shadow effect, a black outline, and a gradient here and there to define their panels.



The shadow is nothing for us. That's a great effect, but quite a few pixels wasted UI space. The gradient and the rounded corners is something that i want to grab here though. And a little black line at the top. Let's modify our shot from above ...





Small change, big effect.

The exact appearance like colours and contrast for the rounded corners will be defined in the theming document.

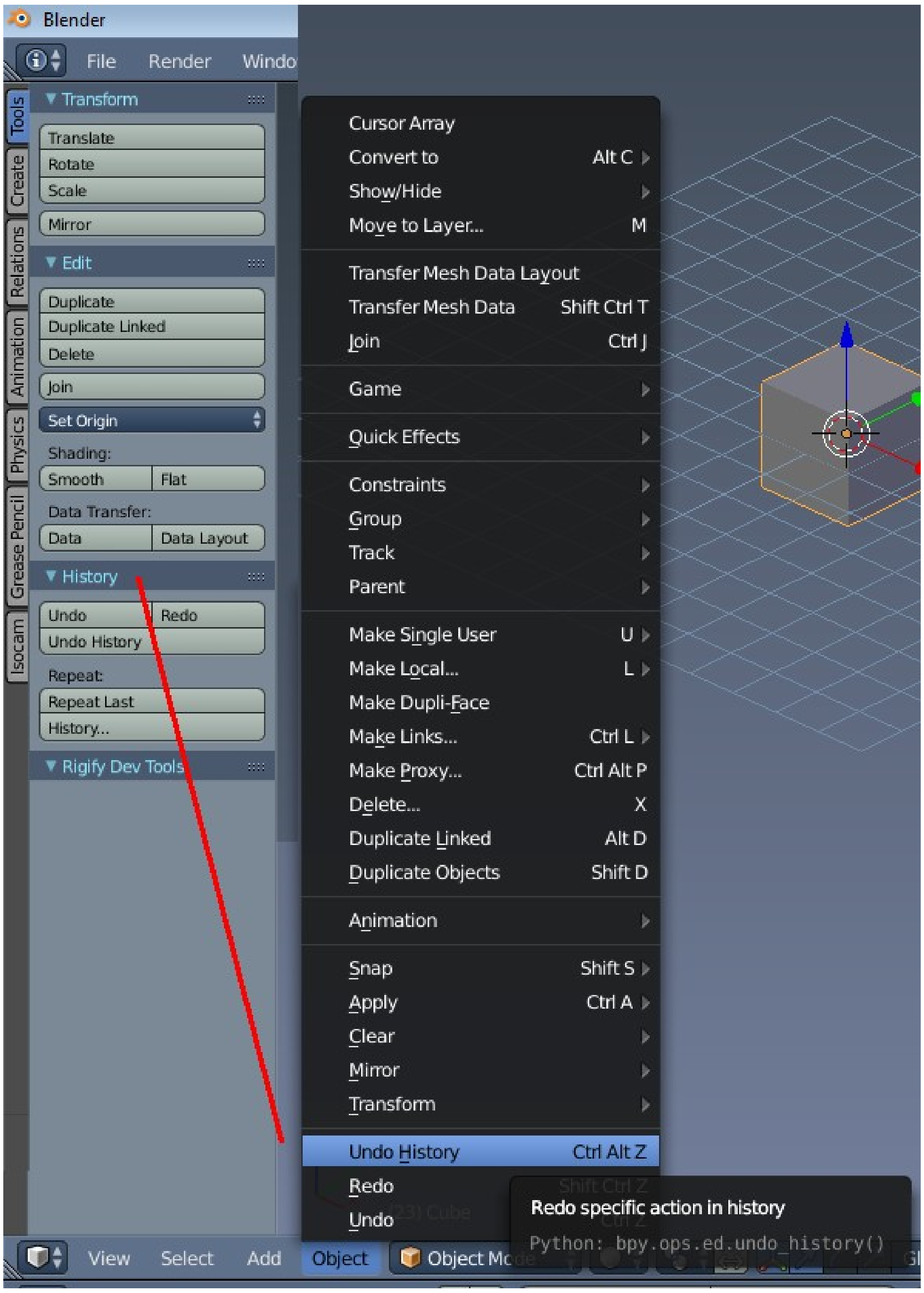
Double Entry - Undo History

The toolshelf contains a history. That's a double entry. And is one of those examples where the Blender programmers placed UI elements where there was space. Not where it makes sense. Instead of thinking through where those tools really belong to.

Undo history does neither belong into the Tool Shelf nor in the local Object menu of the 3D view. Undo does not only happen in the 3D View. This is a global thing. And it stores in fact also image operations from the UV Image Editor for example. The Image editor has no Undo menu at all. So in long term this needs to be in the menu bar of the Info panel. Undo in Blender is a special chapter though. This one needs a big change anyways. And we cannot solve this with the UI redesign.

So short term solution for now, since we just work at the graphical UI at the moment: remove History from the Tool Shelf. One entry is enough. And put the repeat action and history into the Object menu.

To place the Undo History in the Info Editor menu at the top is a job for a later task.



Redo specific action in history

```
Python: bpy.ops.ed.undo_history()
```

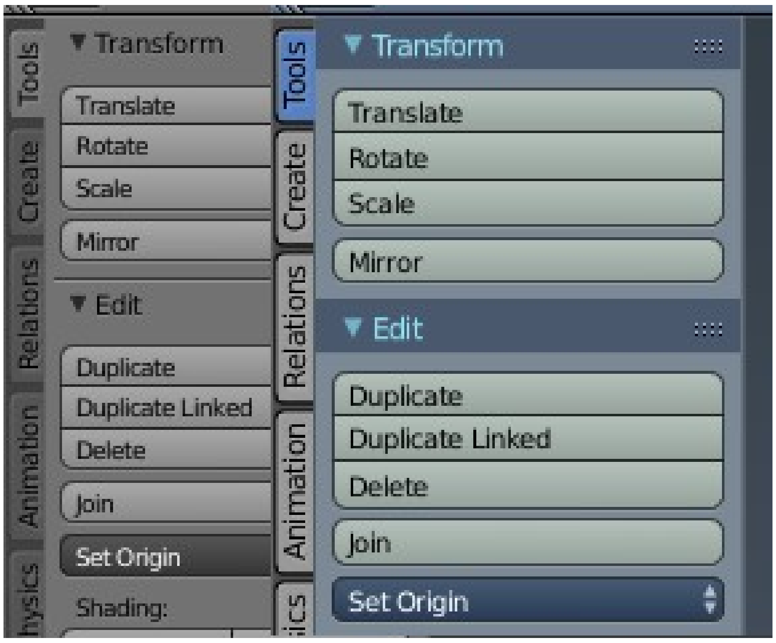
Tool Shelf redesign

The Tool Shelf. The place where just the tools should be, and where you find so many settings too. Like in Weightpaint mode. The place where you scrolled yourself to death until the tabs gots invented. And where you tab yourself to death instead now.

Well, we have already removed one little space eating nagger, and placed it as a completely new editor type into our layout. The inspector. That's the place where all settings should appear now, by calling them from a tool in the tool shelf.

What gets removed from the toolshelf is all the settings stuff. They gets called by the tools, and gets displayed now in the Inspector. What exactly is a job for the Prototype. Here it's just about the concept.

What i want to keep is the vertical tabs. The concept is good. It's just hard to read in the standard layout. Which was what quite a few users moaned about then, and what never gots fixed. But we have already partially solved it at the beginning of this document by changing the colours to make the tabs better readable.

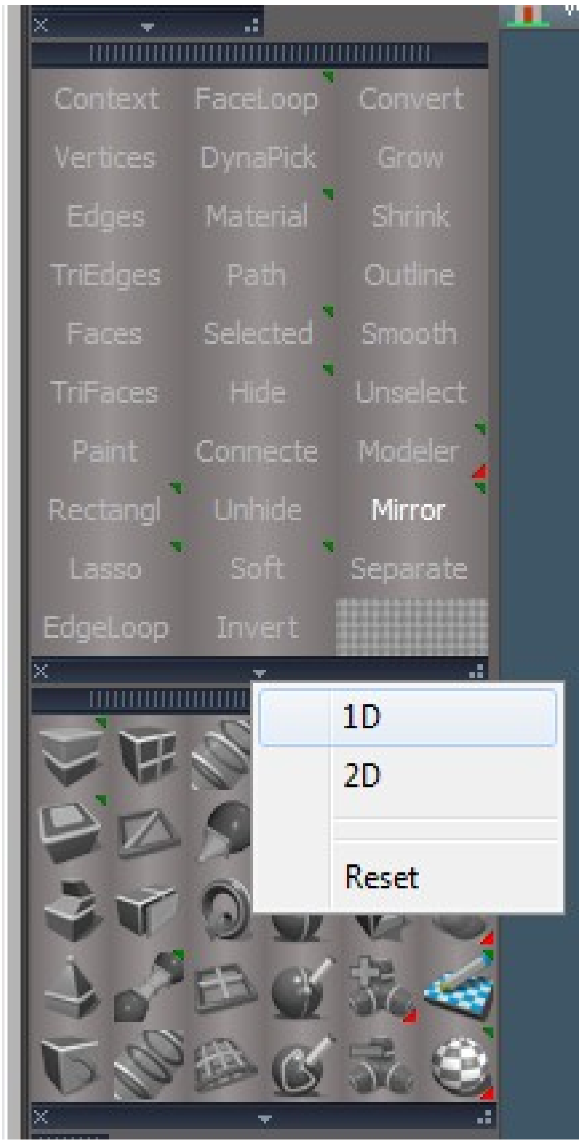


Now for the layout. It is fine in big parts as it is. What i of course want to keep is the separation of the single panels, so that you can expand and collapse them. So we still have sections like Transform, Edit, etc. in this toolbar.

What i want to keep is the ability to sort the panels by own needs. and to pin them.

The tabs needs to be equal long though. It is very bad for muscle memory to have always other patterns. I save me the job to modify our layout to have equal length in the document shots here, it's a job for the prototype to find out the best length.

Next big step. Let's introduce Icon only buttons in the Tool Shelf. The reason is simple, they are much compacter to arrange, this saves UI space. And they are faster to recognize than reading text, once you have understood what it is good for. But let's keep the text buttons as an alternative.



trueSpace 7 had it this way. You could choose what to display. 1D or 2D. Toolbar by toolbar.

We can of course not simply grab the trueSpace solution and place it one to one in Blender. trueSpace had fully customizable toolbars that could be placed everywhere in the UI. While the Blender tools and the panel contents are nailed. Something that i want to keep. And the Blender panels are much more cluttered. To clean that one up is a job for later. Same goes for fully user customizable user toolbars.

But what you can see here very clearly is how compact you can go with even the text buttons when you limit their length, and don't place them just underneath like in Blender. What we display here is 30 text buttons and 30 icon buttons. Blender displays in the same area just a small fraction. No wonder we have a space problem in Blender.

Let's define the size of an icon button first. The current existing icons in Blender have the size of 21x21 pixel. That's fine for the icons in the menus, the buttons and the tab icons in the Preferences. But is ways too small for tool button icons.

Let's have a look how other software handles it. trueSpace uses 27 pixel for its icons. Maya uses 26x26 pixel for a button icon. And Max has an icon size of 30x30 pixels. 3D Coat uses 31 pixel. I tend towards 32 pixels, which is including a button border of one pixel. Means the icon itself is 30x30 then. Because the standard size for icon only buttons since the displays becomes bigger and bigger nowadays.

The 21 pixel size will still be used for the small icon size for the mixed text and icon buttons. And of course in the menus.

Now let's define the length of a text button in standard zoom. At the moment the text buttons are mostly underneath in the Tool Shelf. And are as width as you pull out the Tool Shelf then. I want to be able to place them besides too, like shown in the shot from the trueSpace toolbars. Note, the Blender developers have chosen quite a few really long tool names. So we have to shorten a few names by dots " .. " when they get too long. Full name can be displayed in the tooltip then.

For now i would say that a text button should not be longer than let's say three times a icon button. That makes 96 pixels long in standard zoom. I want to fit a text button optically together with a row of icon buttons.

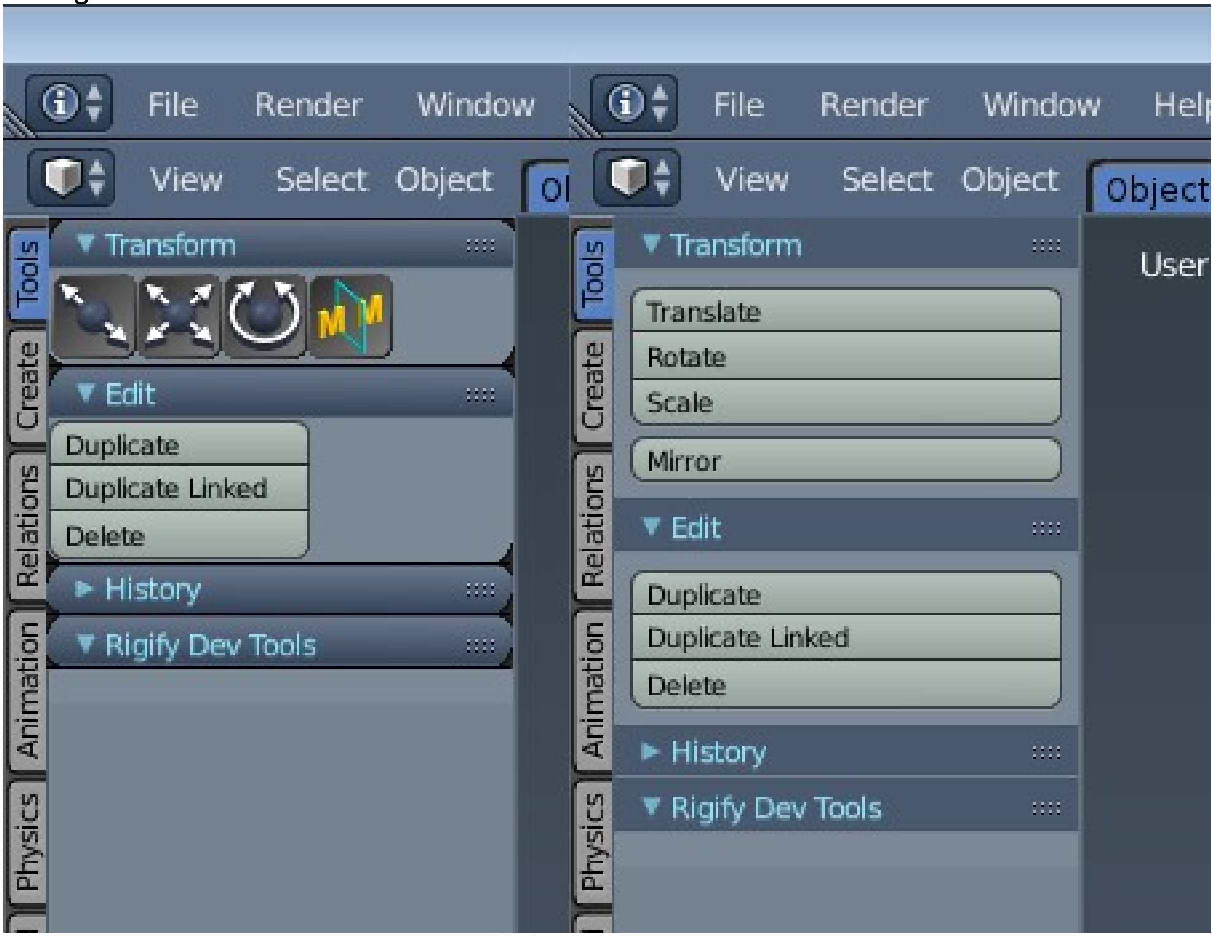
The layout should adopt to the size. Means when i pull the Tool Shelf bigger then the buttons should sort besides each other. When i drag it smaller then the buttons should sort below each other.

Then we can drag the whole Tool Shelf much smaller. so that it fits the 96 pixel text buttons. Or even just one icon button. Or on the other hand pull it out so far that every panel has just one row. Let's do this modifications.

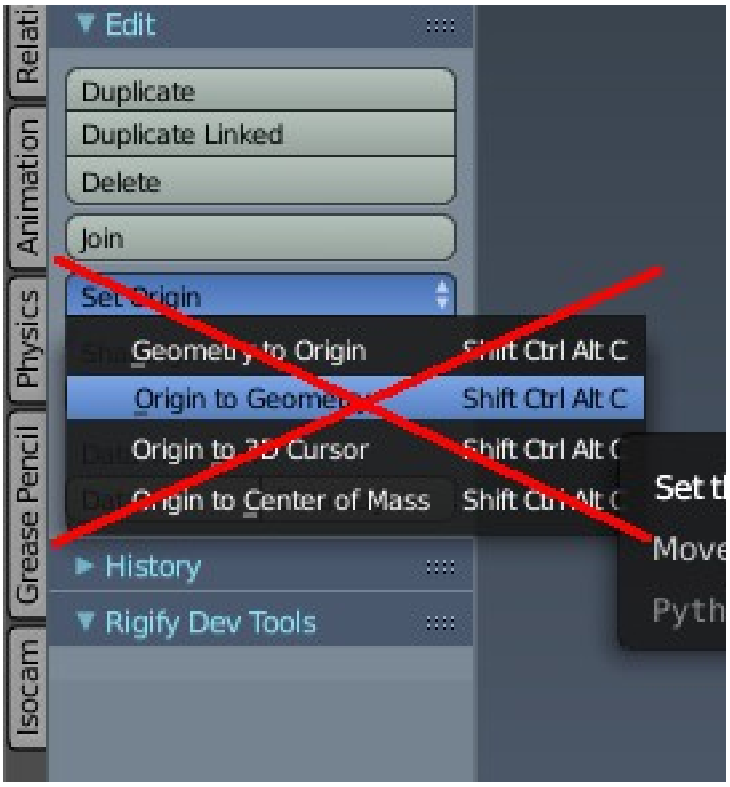
I have also added the rounded corners and the gradients in the next shot. And placed the text buttons much closer to the borders. There is so much wasted UI space in the panels for no reason. I think the shot makes clear why icon buttons and a fixed size for text buttons is a good idea.

The panel bars needs of course a rmb menu now, so that the user can decide if he wants to use icon buttons or text buttons. But we already have one for the nailing.

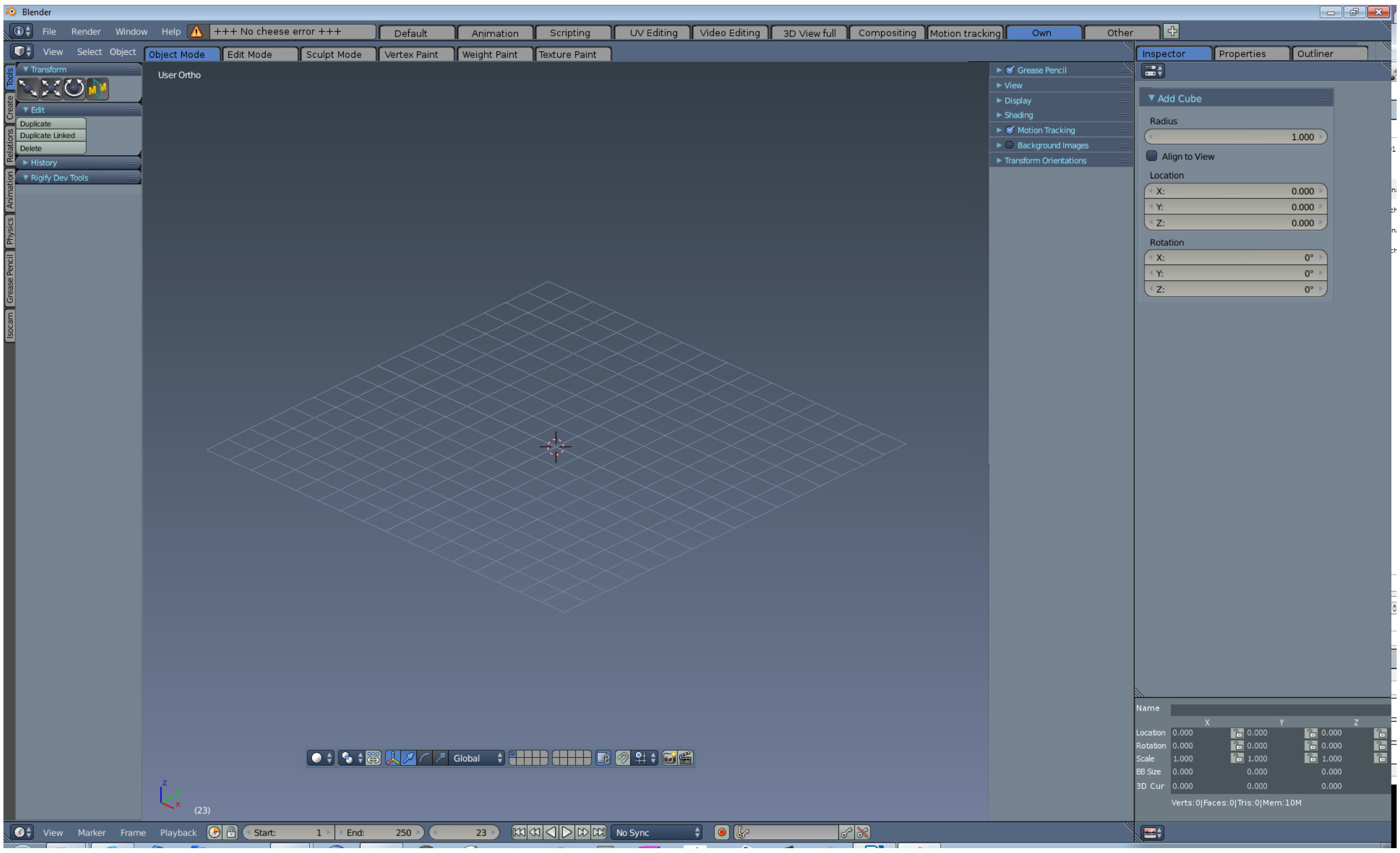
Note that the icons in this shot are just temporary. It's about the concept, not the final mockup. We need most probably a completely new icon set anyways. Vectorbased. And the standard theme will change too.



Another little thing that i want to change is the dropdown boxes in the toolbar. Away with them. No digging in sub menus here! It's a toolbar, not a menu. We have enough free space now after our redesign for a few more buttons! Especially when you can make them icon buttons. Set origin is a perfect candidate for icon buttons.



And we are done with this step. Now our baby looks like this:



What counts for this layout and mode counts also for all other layouts and modes. In Paint mode for example, you will find the panels with the pencils and stuff in the Inspector now. The toolbar will become a bit slim in this mode, since most of the paint stuff is settings, and will be displayed in the Inspector then. But that's the deal with having one central place to look at. And consistency is important.

The properties toolshelf becomes the settings toolbar

Let's tackle one of the last biggies in this current layout. At the right we have the properties toolshelf. We have also a editor window called Properties. So every time we talk about properties there is a chance that we talk about two different things. Let's rename the properties toolshelf into settings toolbar.

We have already removed the Item, Transfom and 3D Cursor panels from it. It still contains a tools and settings mish mash that doesn't really make me happy. And we have still our from the menu separated toolbar at the ground that we haven't sorted yet. This goes into one toolbar now.

And this toolbar can find its new place at the top then. Or at the right. Or at the ground. I want it to be dragable and dockable. And i want it to be flexible in size. Not that we end in the same dilemma than before, that the neighbour window hides parts of the toolbar. This is something QT allows, but the current UI solution not.

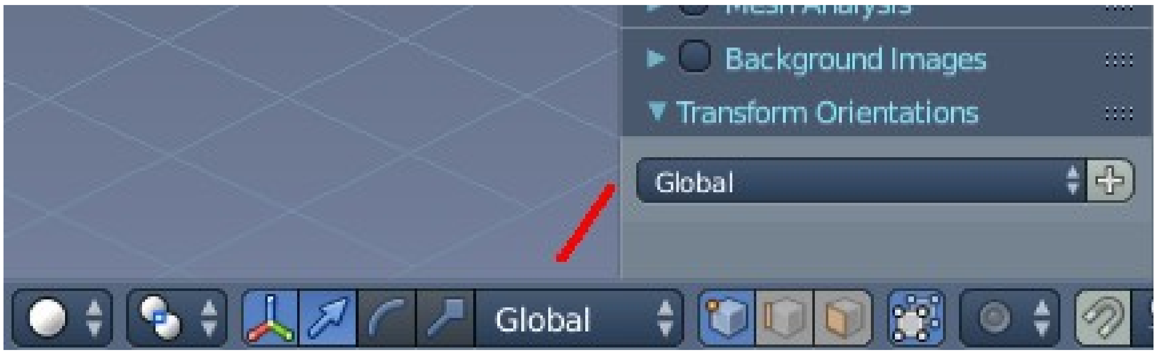
I would say, let's settle this toolbar at the bottom for now. Blender users are used to have stuff like shading and pivot settings at the ground. Prototype will tell us then if users are more happy with a bottom toolbar or a top toolbar. Or maybe having it even asides.

Our icon buttons in the toolshelf has currently 32 pixels. I could still live with the 21 pixels for our properties toolbar though. Maybe a bit bigger. Details follows in the theming document.

We have already removed the three items Item, Transfom and 3D Cursor panels from the Properties panel. Let's clean up the rest, and have a look what is left afterwards.

Having a closer look at the toolbar, we see quite a few double entries that exists at various other UI locations, A big problem in the Blender UI. Double entries wastes UI space for no reason.

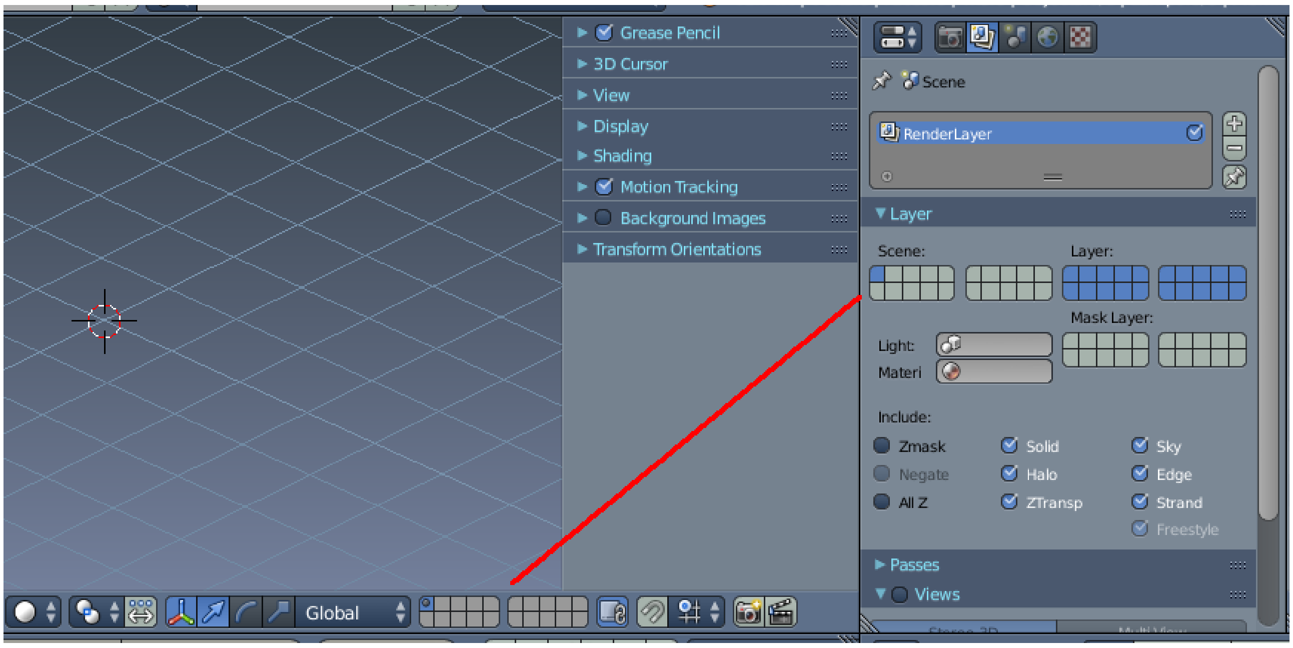
Transform Orientations already exists in the toolbar that was formerly located in the menu bar. So away with it.



Next double entry. The layers. They exists also in the Properties.

We could remove the one in the properties editor, and place a button in our toolbar to call the settings. Or we remove it completely. I do the second one. It's not that you change layer settings every two seconds.

When the demand is high enough then we can place a button into the Properties toolbar to call the Layer settings in the inspector at a later point. This is a task for the prototype or an alpha build then.



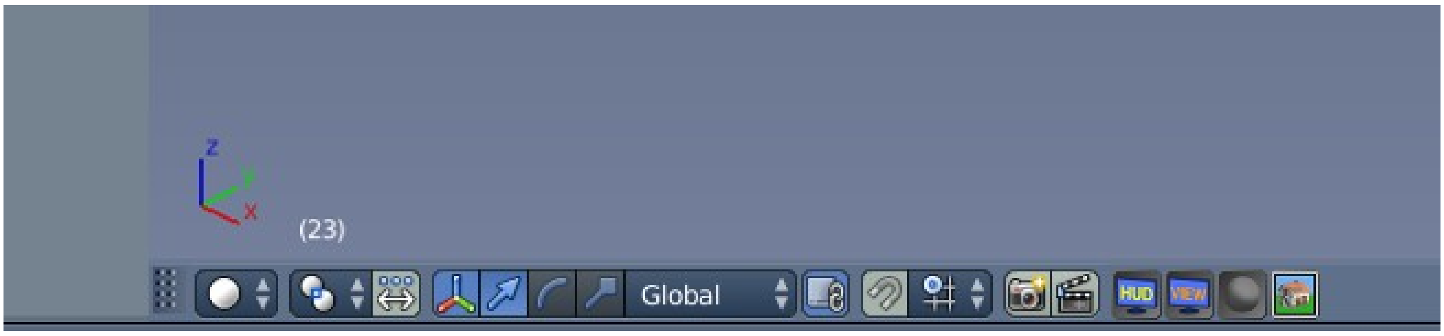
Motion tracking? That's too special for the modeling task. Let's put it into the Properties. The Scene section is a good place.

Grease Pencil is a tool. This belongs into the Tool Shelf. Its settings into the Inspector.

View is a render setting. This normally belongs into the Properties, in the render section. But let's keep it as a button, to call the settings in the Inspector.

So what remains of settings where we need quick access to is just a few things. In the current layout it's Display, Shading, View and the background image maybe. Four more icon buttons where formerly was a complete toolbar with the size of 200 pixel width and a height of the whole screen. Around 1000 pixels at a standard monitor.

The Properties Sidebar is no more. Now we have a tiny toolbar instead. Dragable and dockable at other locations when needed ...



The same cleanup and refinement steps that we have done here with the Properties Toolshelf and the toolbar in the menu bar applies of course to all the other editor types too. Like the Dope Sheet, F Curve and Timeline. And all the other editors where there are quick access tools in the menu bar.

Problem is, we loose a bit space by separating the quick access tools from the menu bar. Which was the reason why those tools gots placed into the menu bars at all. As a solution we could make our new settings toolbars also dockable above a menu bar where it makes sense. Remember, some of them have tabs now. That way we loose nothing, but win nevertheless neded flexibility and UI space.

I will do exactly that, and dock the new Animation Settings toolbar above the menu bar in the next shots ...

Refining the Inspector

The panels with their settings gets displayed in the Inspector when you click at one of the buttons. Tools also. Which raises another problem. It can happen that dozens of panels are open at the same time. And this brings the same dilemma that existed in the Tool Shelf before they invented the tabs there. Not this dramatic as before, since we will make the panels slimmer as shown in the Toolshelf already. But it remains a problem.

So, like in the Tool Shelf. we need a quick way to navigate. We will of course provide a scrollbar, but scrolling around like crazy is not really an option. There is another solution. Add a tab for every new panel that is open in the inspector. This gives quick access to the panels then. To make it slimmer i would limit the tabs names to four letters here. Maybe even icons.

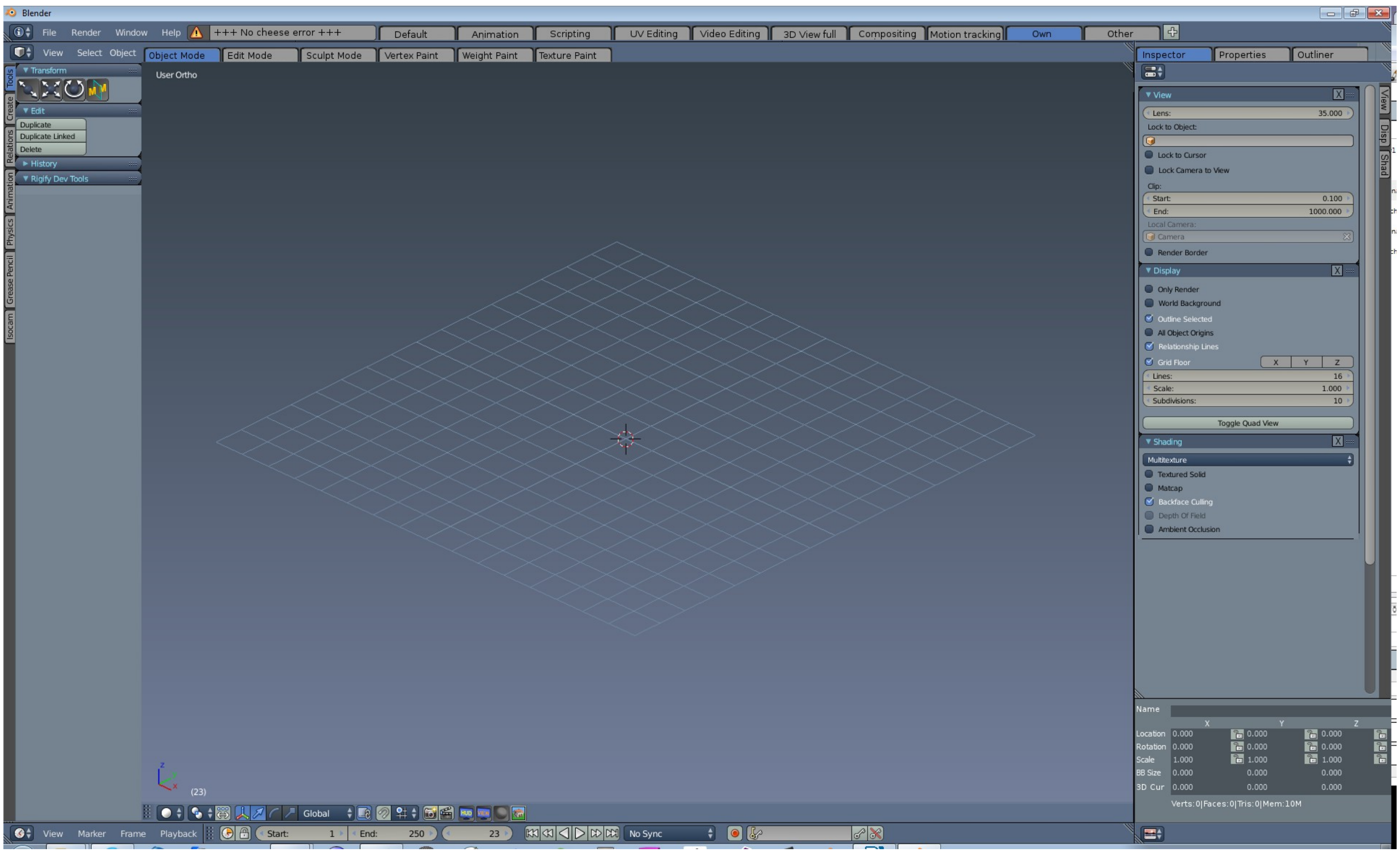
And we need a way to close the panels in the inspector that we don't longer need. A little close button. Not to forget a panic button to close all open panels.

It's also planned to remove all panels from specific modes when you leave this mode. Like switching from Edit mode to Object mode. Which will also prevent the Inspector from becoming a mess. This includes switching to another tool tab, and also switching to another layout.

Another road that could be gone is to close the current displayed toolsettings when another tool gets called. To avoid that you need to call a tool that you need for your workflow again and again we could add a way to pin important settings. So that just the unpinned settings gets removed when calling another tool.

I think a mixture of all of the abvove points will be the solution to find the right balance between a consistent and so easy to use UI, and the need to have a fluid and fast workflow on the other side.

Note that i have docked the Animation Settings toolbar in the menu bar of the timeline in this shot here.



View
Lens: 35.000
Lock to Object:
Lock to Cursor
Lock Camera to View
Clip:
Start: 0.100
End: 1000.000
Local Camera:
Camera
Render Border

Display
Only Render
World Background
✓ Outline Selected
All Object Origins
✓ Relationship Lines
✓ Grid Floor X Y Z
Lines: 16
Scale: 1.000
Subdivisions: 10

Shading
Multitexture
Textured Solid
Matcap
✓ Backface Culling
Depth Of Field
Ambient Occlusion

Name			
	X	Y	Z
Location	0.000	0.000	0.000
Rotation	0.000	0.000	0.000
Scale	1.000	1.000	1.000
BB Size	0.000	0.000	0.000
3D Cur	0.000	0.000	0.000

Verts:0|Faces:0|Tris:0|Mem:10M

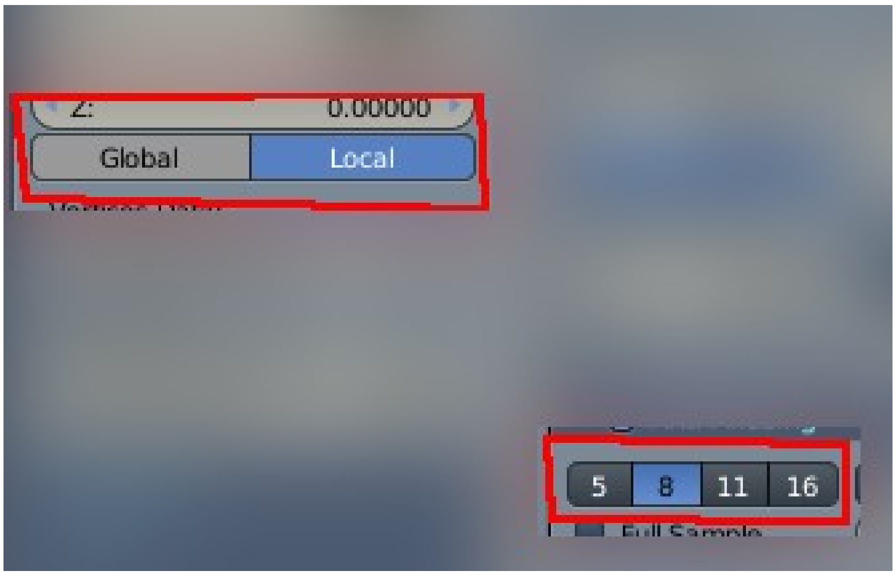
Start: 1 End: 250 23

We are nearly where we want to be. The surface becomes slimmer and slimmer, is not so clutterted aymore, but has still the same functionality.

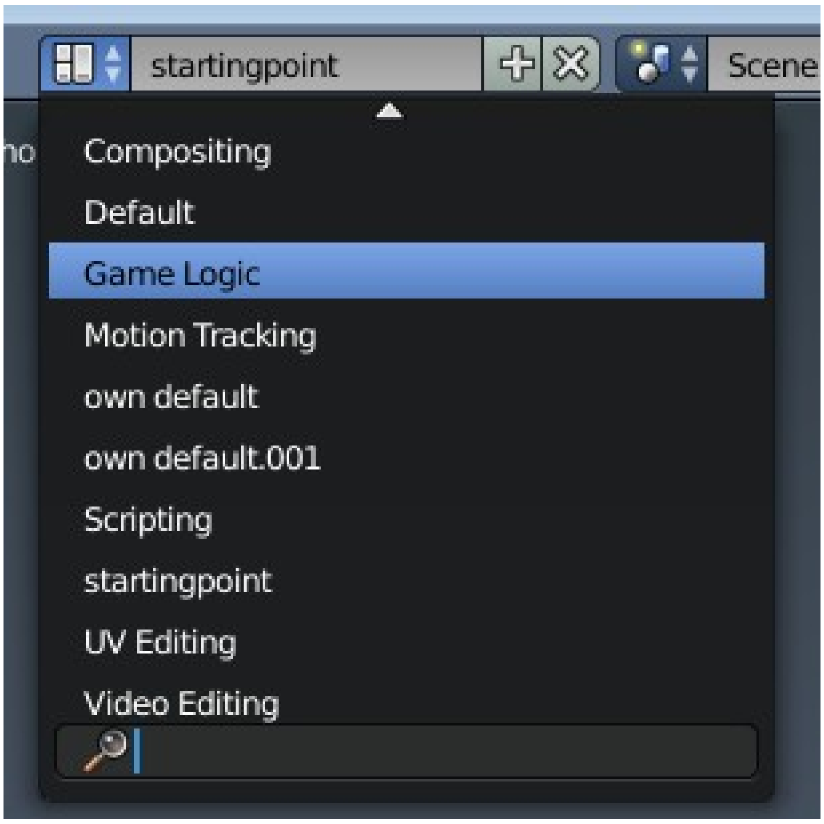
We are done with the redesign of the layout for now. The rest of the layout work happens in the theming document. And of course in the prototype across the alpha and beta versions. I don't expect that all ideas makes it into the final.

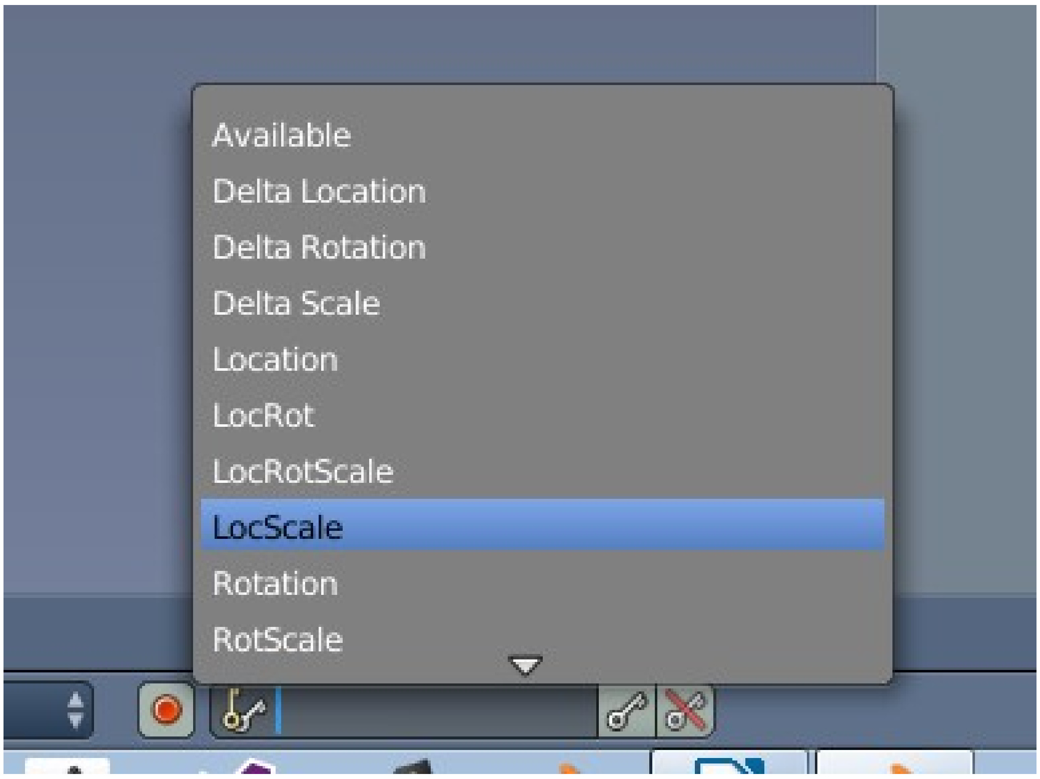
Same Elements, different appearance

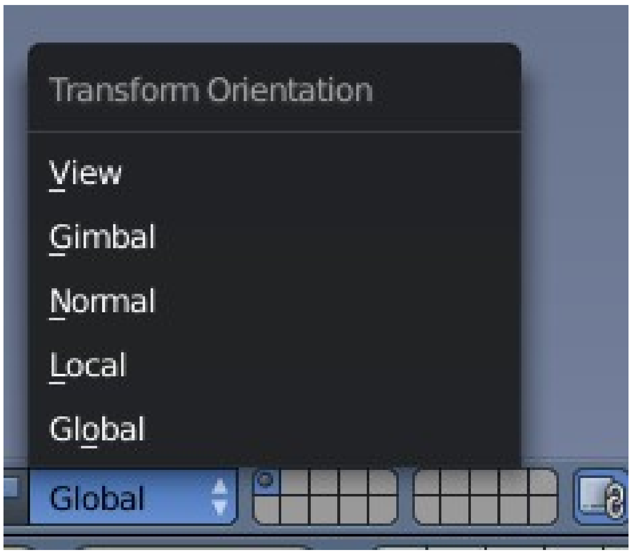
Blender has quite a few different appearances for the same elements. Two types of toggle buttons for example.

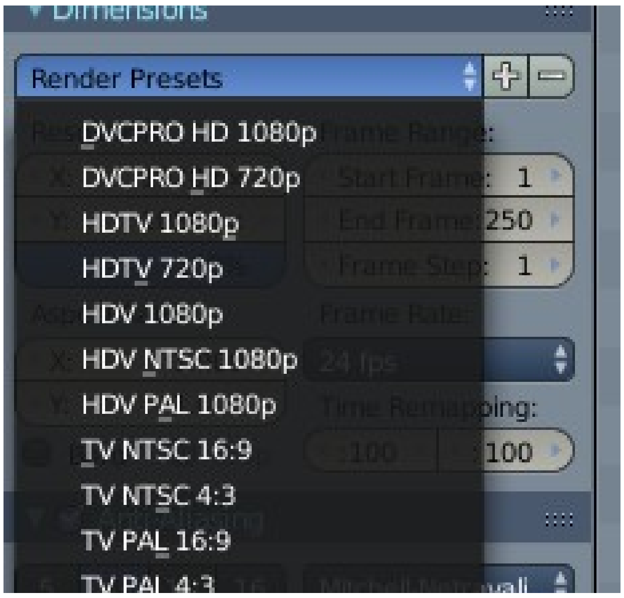


Dropdown boxes have a few more incarnations, unfortunately ... :




Available
Delta Location
Delta Rotation
Delta Scale
Location
LocRot
LocRotScale
LocScale
Rotation
RotScale


Transform Orientation
View
Gimbal
Normal
Local
Global
Global


Dimensions
Render Presets
DVCPRO HD 1080p
DVCPRO HD 720p
HDTV 1080p
HDTV 720p
HDV 1080p
HDV NTSC 1080p
HDV PAL 1080p
TV NTSC 16:9
TV NTSC 4:3
TV PAL 16:9
TV PAL 4:3
Frame Range:
Start Frame: 1
End Frame: 250
Frame Step: 1
Frame Rate:
24 fps
Time Remapping:
100
100
Mitchell-Netravali

I think there are even a few more. But the problem should be clear. There is not really a similar appearance. Some dropdown boxes even integrates into the UI in a way that you could think it's a title bar.

Regarding long dropdown box lists, i personally think that they should be in a own menu entry when it's a this long list. When it's after me then the keying options goes into the menu bar as a menu item. Since it makes me always crazy to scroll down to reach my needed setting. But the buttons and drop down boxes should at least have the same appearance.

Plugins integration

We may break standard Blender plugin integration with the UI redesign. Even when we will stay with the current custom UI solution, and not migrate to QT. Most of the Blender plugins relies at the Python UI code. So we need a solution here. Plugins are important.

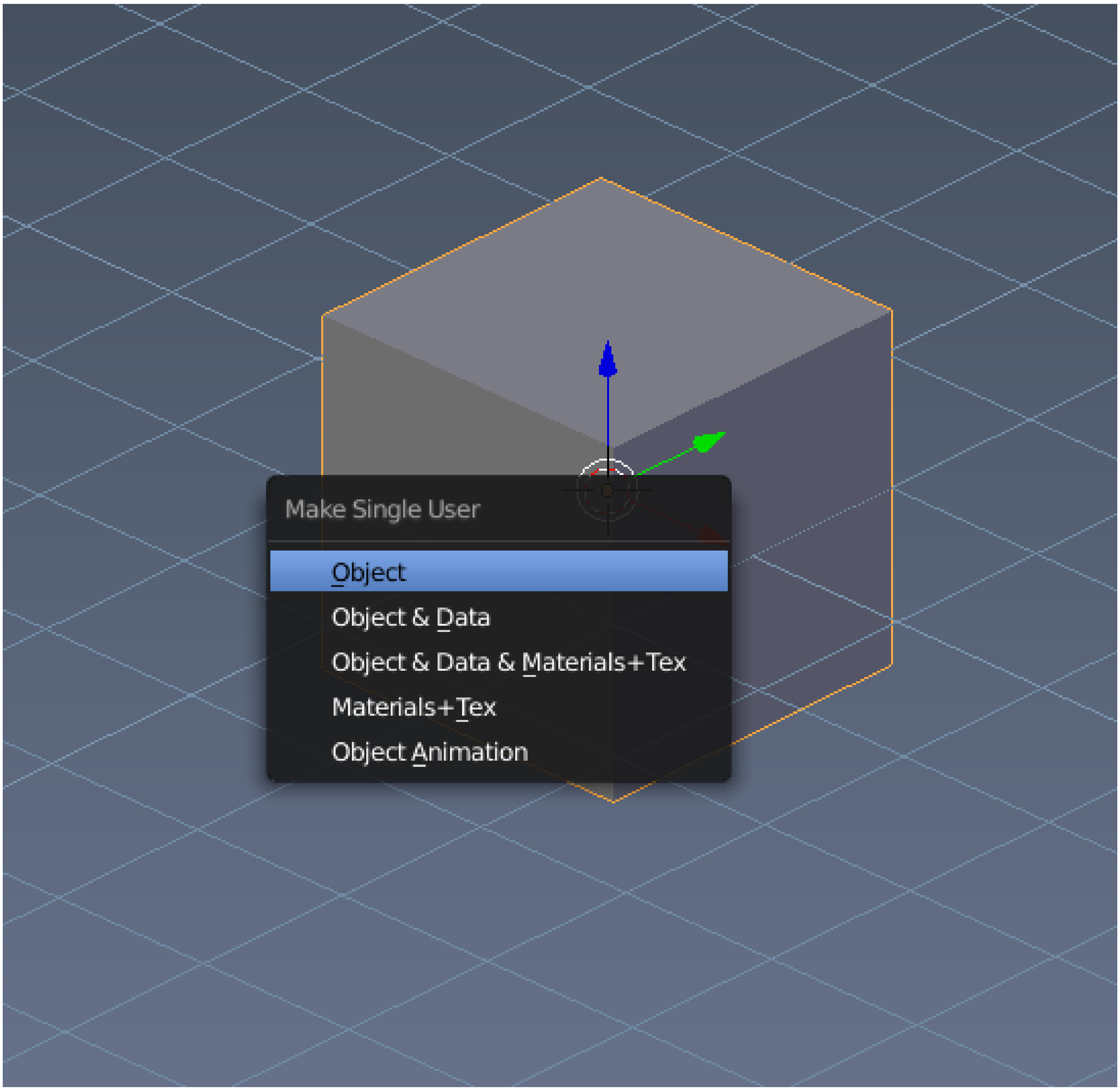
How exactly this problem gets solved is a task for later. Let's cross this bridge when we arrive at it. First we need a working UI solution.

Popup Windows

There are quite a few popup windows around that you can call by hotkeys. Some of them even still miss a menu entry.

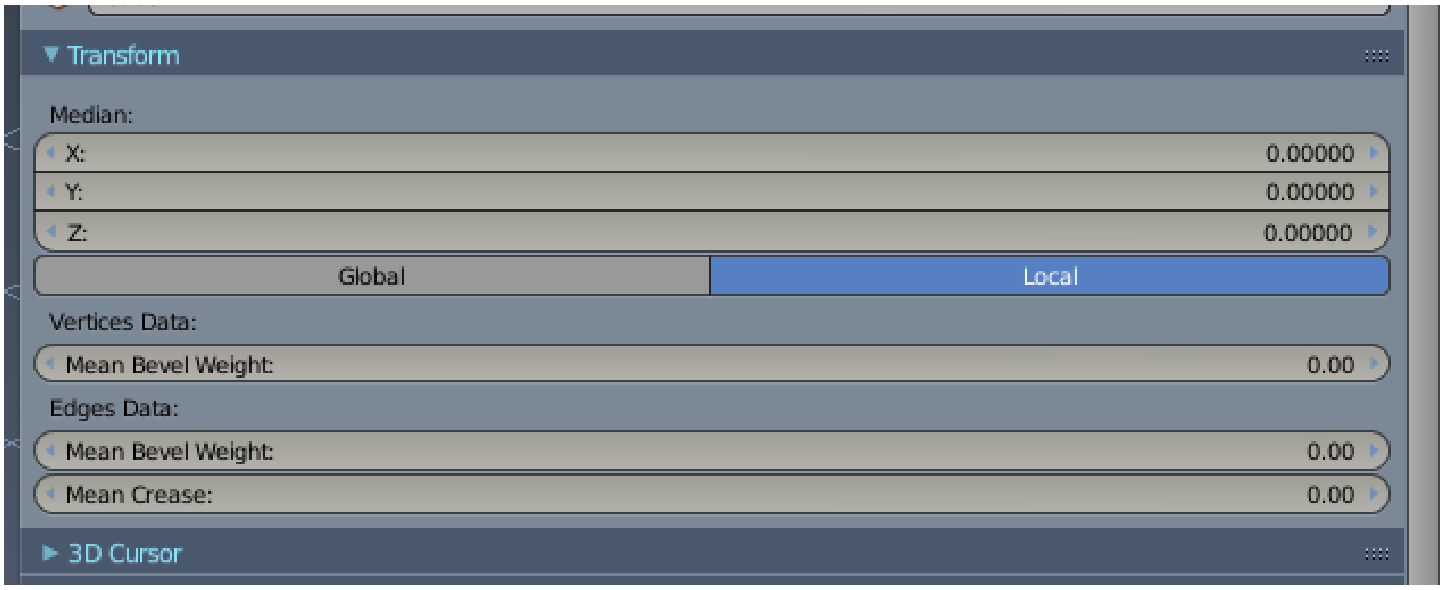
Those popups aren't necessarily evil. I want to keep this functionality. But the same functionality must also be accessible by a direct menu item. As told in the intro, we have to come away from the hotkey useage dogma. The primary goal is to make Bforartists compeltely navigatable by the graphical UI. Hotkeys comes on top of that. At the second place.

How to deal with the popup menus needs to be tackled one by one, case by case. We need to make a list of all available popup menus at one point. This issue doesn't necessarily be fixed while we develop the new UI already. For now i just mention them as a problem zone.



Wasted UI space

There is lots of wasted UI space in the current UI. Like buttons scales bigger or smaller with dragging the panel wider or smaller. Buttons, sliders, etc. , they all scale with the panel. And as a result they become hard to navigate in, since the text elements are far away from each other. Or you drag them too small and the values and words are overlapping. Means to find the best width is a job at its own. The user shouldn't need to battle with the UI this way though.



Transform

Median:

X: 0.00000

Y: 0.00000

Z: 0.00000

Global | Local

Vertices Data:

Mean Bevel Weight: 0.00

Edges Data:

Mean Bevel Weight: 0.00

Mean Crease: 0.00

3D Cursor

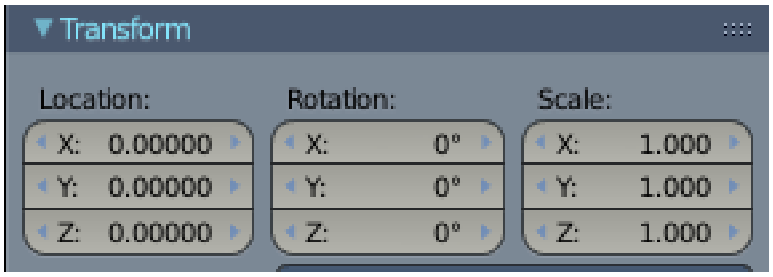
There should be a maximum width for sliders, buttons, and similar elements. This leads to a better readability. And also a better muscle memory.

We have already defined a fixed size of 96 pixels for a text button in the Tool Shelf. I would like to use this value also at any other UI element. This should be enough for sliders and stuff too. And when you drag a panel wider, then resort the items besides each other. This saves UI space.

Another thing that we need is a fixed width for panel elements in the Inspector and any the Properties. Maximum double text button size plus a bit of a border. Makes the design task a bit easier when you don't have to deal with flexible layouts and widths. And better readable, as shown above. So we have a text button of 96 pixels. Let's say maximum 200 pixels width for a panel element in the Inspector. This gives us space for stuff like paths too. Or better use 300 pixel already, since 200 pixel will not be enough for the Properties Editor.

The exact values is a job for the prototype.

Now let's have a closer look at some elements. Let's say the Transform values in the Properties Editor. We can make this much compacter, but at the same time much better readable.

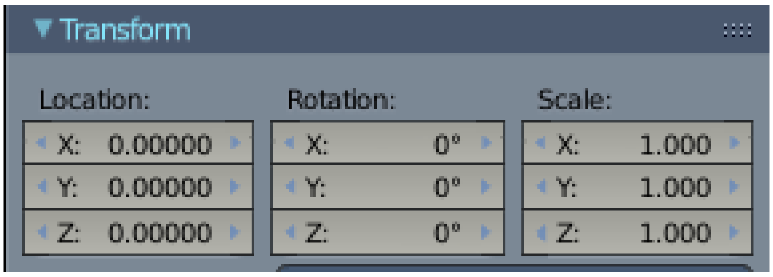


Transform

Location:	Rotation:	Scale:
X: 0.00000	X: 0°	X: 1.000
Y: 0.00000	Y: 0°	Y: 1.000
Z: 0.00000	Z: 0°	Z: 1.000

Step 1:

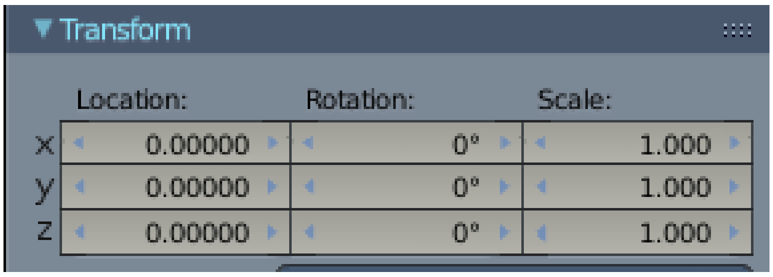
The rounded corners is space wasting. We need to go to the corners in a next step.


Transform

Location:	Rotation:	Scale:
X: 0.00000	X: 0°	X: 1.000
Y: 0.00000	Y: 0°	Y: 1.000
Z: 0.00000	Z: 0°	Z: 1.000

Step 2:

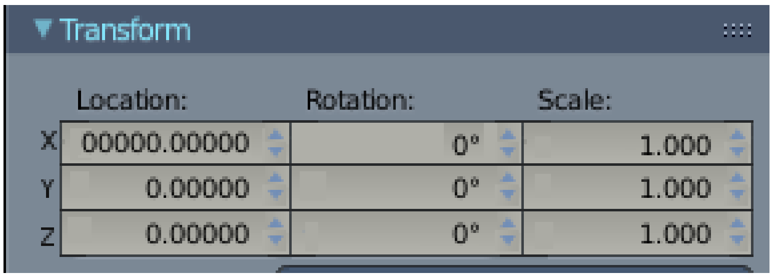
The gaps between the edit boxes is wasted space. And i have cleaned up the visual noise here. In general it's a good idea to write some text within the edit boxes. There was three times a x, y and z to read though. One time is enough. Even when this means we need to give them a space outside of the edit boxes.


Transform

	Location:	Rotation:	Scale:
x	0.00000	0°	1.000
y	0.00000	0°	1.000
z	0.00000	0°	1.000

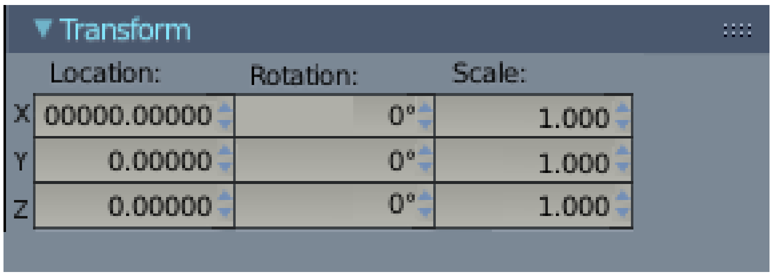
Step 3:

We have at the right side a slider element, and at the left side a little slider element. That's also visual noise. One side is enough:


Transform

	Location:	Rotation:	Scale:
X	00000.00000	0°	1.000
Y	0.00000	0°	1.000
Z	0.00000	0°	1.000

Step 4:

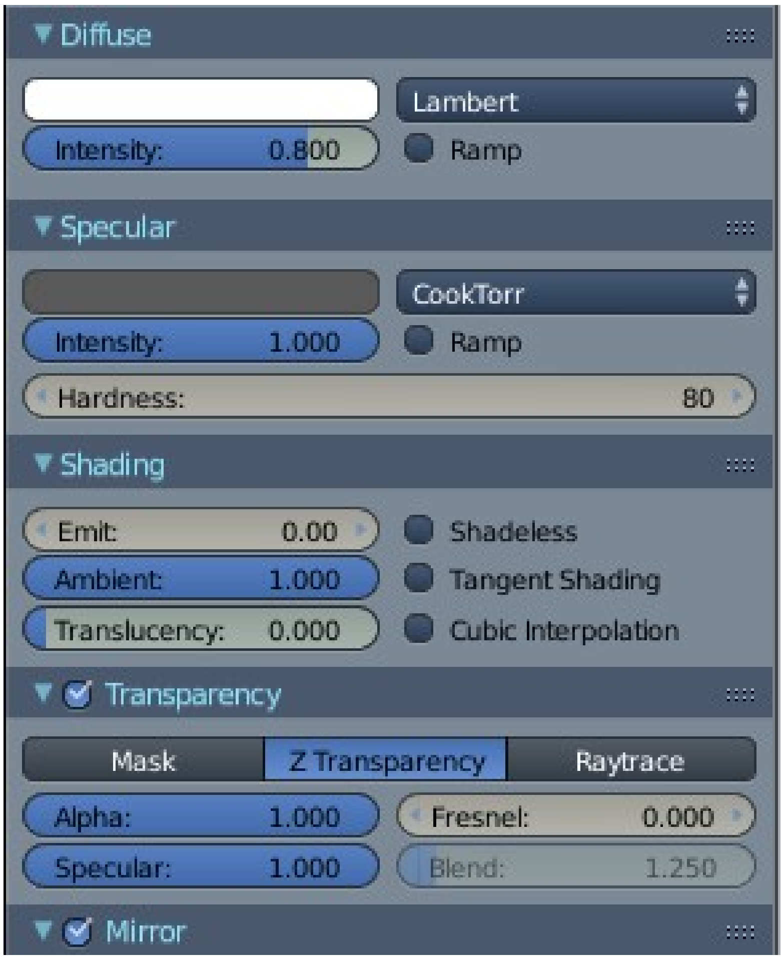

Transform

	Location:	Rotation:	Scale:
X	00000.00000	0°	1.000
Y	0.00000	0°	1.000
Z	0.00000	0°	1.000

And now let's arrange it a bit better. The edit boxes needs to display 12 digits, so there is a limit for how small we can drag them. But we have nevertheless freed quite a few pixels here.

A few simple steps, and the baby is much compacter. And even better readable.

It's not only the content of a panel. It's also that there are ways too many panels at all. Like for the shader settings. Let's make this one much compacter:



▼ Diffuse

Lambert

Intensity: 0.800 ● Ramp

▼ Specular

CookTorr

Intensity: 1.000 ● Ramp

Hardness: 80

▼ Shading

Emit: 0.00 ● Shadeless

Ambient: 1.000 ● Tangent Shading

Translucency: 0.000 ● Cubic Interpolation

▼ ✓ Transparency

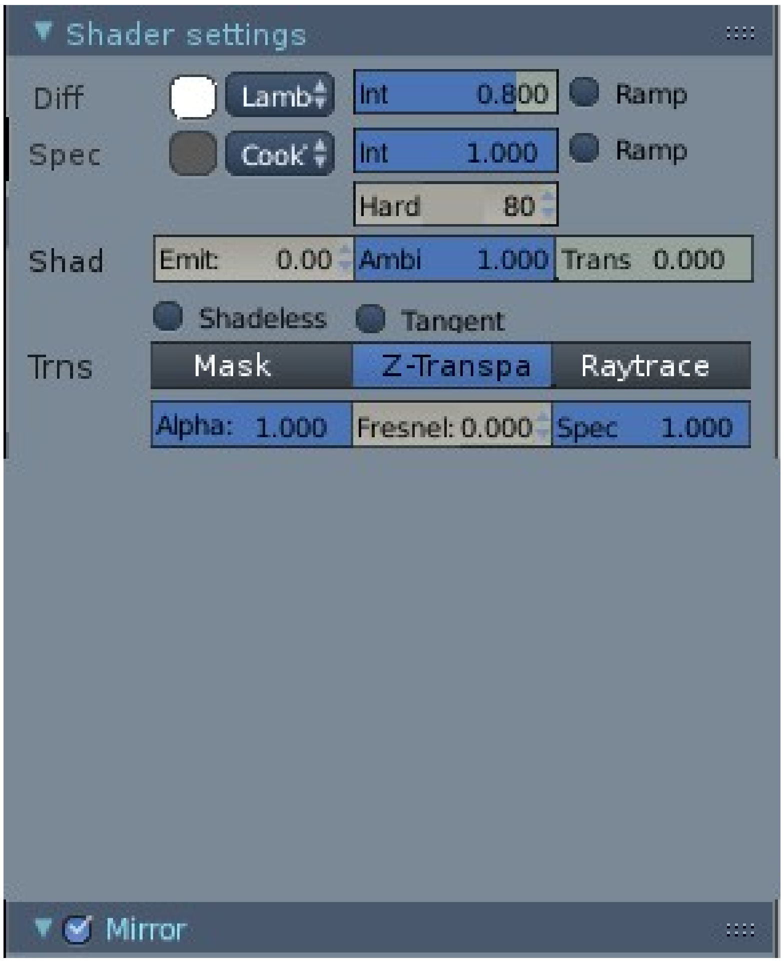
Mask | Z Transparency | Raytrace

Alpha: 1.000 Fresnel: 0.000

Specular: 1.000 Blend: 1.250

▼ ✓ Mirror

A few simple steps and some rearrangement later ...



▼ Shader settings

Diff Lamb Int 0.800 ● Ramp

Spec Cook Int 1.000 ● Ramp

Hard 80

Shad Emit: 0.00 Ambi 1.000 Trans 0.000

● Shadeless ● Tangent

Trns Mask | Z-Transpa | Raytrace

Alpha: 1.000 Fresnel: 0.000 Spec 1.000

▼ ✓ Mirror

Stunning, isn't it? It's the same content. But much compacter arranged. And it is at the same time much better readable. Lots of visual noise is gone. And we have freed lots of UI space. More than half of the space that gots used than before. Which reduces scrolling and tabbing.

What i have done here is to make the colour picking area much smaller. It does not need to have half the width of the panel. I have also unioned four panels into one here. Their panel bars are gone now. And i have shortened the names. There is no need to display the whole encyclopedia here. Four letters are usually more than enough to let you remember what this stuff is good for. For the full name there is the tooltip.

I could of course arrange it much better. And get rid of the three radio buttons in the transparency section by replacing it with a dropdown box. But this was just an exercise in a paint software to show the trouble areas, and what could be improved. The exact arrangement is a job for the prototype then. And cleaning up the Properties panels is a job besides the UI redesign anyways. See also the next point.

Long time goals

At the moment it is important to have the QT solution working. As told earlier, we cannot fix all the things that went wrong the last twenty years in one big step. We need a long evolution to arrive where we want to arrive. One step after another.

There are a few long time goals though that i want to write down here already. I wantt to tackle them when the QT migration is finished.

The one is to allow custom user toolbars. Where the user can add his favourite and most used tools. So that the user can set up his workspace in the way he wants it to be. QT already allows dragable toolbars natively. So it can happen that we will see this point already tackled in the first incarnation.

Another one is to clean up the cluttered mess in the Properties. The Blender developers did a great job here to throw in elements and menus where there was space. And especially the material and texture slots are a real mess. This can be arranged much compacter, but at the same time much cleaner. And more than one item will find a new home at another location.

That for the general issues. Document 2 will be about the theming. We will define in detail the needed graphical elements, theme all elements there, and generate our first dark standard theme.