Glitter Widget Toolkit
Human Interface Guidelines

by Jan Jokela, latest review on mar 9

What's wrong

Aside from spell checkers, most graphical user interfaces.
Thus, lets speak about some of the most common user interfaces on mobile devices, big screens and desktop/laptop computers.

Desktop & Laptop computers.
In these creatures we use keyboards and mice as input devices. The first we use to type in stuff and run some commands through keys or combinations. The latter is used to point at stuff and select. Some mice have a ton of additional buttons.
With all this input power, one would expect computers to be easy to use right?

GTK, QT and the like.
These folks pretty much behave the same way. Whether you build an application in one or another, you will end up with an interface that shares the same methaphores and behaves quite the same way. In the case of QT and GTK, the most notable difference between applications is that GTK ones tend to be cleaner.
In both, application interfaces are supposed to be document centric, which is a good thing. Except they aren't. Lets take a look at the typical application:
First, you might have a view for your document, lets say an image. So far so good.
Now, you probably have a ton of 'bars' in your application. Status bars, tool bars, menu bars, side bars. The thing is those 'bars' are supposed to refer to your documents, but they aren't, for the most part. The tool bar is a nice example, while it may contain functions that manipulate or refer to your document, silly functions such as "New", "Open" or "Preferences" appear on them. Well, neither of those are relevant to your document. The other interface parts seem to follow this mess also. The menu bar, where you have your menus, combines all this eloquent mess into one, useless, hierarchical voodoo.
Another concept is the one of multiple documents in one app. This is a great thing, you can have multiple documents open at once without creating too many unnecessary windows for your window manager to handle. So, what do we do? Tabs of course! Tabs are great, you can easily switch between open documents just by choosing the tab you wish. But again, in these traditional interfaces, this great idea is hindered by the massive quantity of user interface around the tab view, which most often than not contains useless and irrelevant interface elements.

So, it is pretty clear we can improve the ergonomics of the traditional user interface. There is another thing that might just be as important. Being artisctically subtle.

These user interface toolkits have no art in them. They don't look natural, the interface elements align themselves in an inelegant way, they don't appear smoothly, they aren't fluid.

A last note on these. They aren't scalable. We aren't able to make them look exactly as we want on any screen size and dpi.

Mobile devices.
A first note, the iPhone OS interface toolkit is state of the art. It is in fact the only 'great' interface toolkit out there for mobile devices.
But since the iPhone interface toolkit is a proprietary one, here's something free, and an edge cooler.

Other toolkits just result in ugly interfaces with unelegant ergonomics and all the clumsiness known to desktop interfaces.
Most phones have about a ton of physical navigation buttons and function keys, plus a 9 or full sized qwerty keyboard. This is a bad approach because the buttons are fixed in plastic, they don't change depending on the app. And the applications, in they're turn, get messed up because they need to adapt to the clumsy hardware design.

Big screens.
There aren't really high level toolkits for user interfaces in big screens. Talking about full screen interfaces for media players, media centre apps and all of the kind.
Most people who write these just rely on low level toolkits such as clutter, Core Animation and the like. So they end up having to write most of the widgets themselves.

What matters
Taking a leap forward in user interfaces. Creating powerfull, ergonomic and artistically subtle interfaces that are comparable to an Alvar Aalto house near a Finnish lake instead of a container factory in the suburbs of Beijing.

On your hand
Using most of your screen real estate for the content you want to display. Minimal overhead when it comes to aditional interface elements like navigation bars and controls. Having the less relevant controls appearing only when needed. As an example we have media playback. If we are playing music or video, we don't need permanent control buttons on display. We can show them only when the screen is touched.

Having great ergonomics.
In mobile settings it shouldn't take more than a couple of taps to get where you need to be, nor should the interface be too complicated for you to intuitively get where you need to go. The centre piece needs to be your content. Whether it is a high resolution photo, or a simple item list. It needs to have most of the screen real estate. That's crucial.

When it comes to features, use the 80% rule. Implement only those that are most likely to be used by 80% of your audience. After all, we are talking about mobile devices. These shouldn't have as much weight as in a desktop environment.

Visual cues.
Interfaces need to be fluid. You can't afford to have interfaces that are sluggish, that don't appear natural. That don't offer visual cues in the form of subtle transitions and effects. Glitter offers you a leapfrog advance in that department. Base widget already implement the fluidness you need. So when writing your own widgets, take the base ones as an example.

Jaw dropping sexiness.
Don't surrender to mediocrity. Use the best icon packs you can possibly get, and when stock items are not available, design pretty darn good ones. Depending on performance compromise, use SVG icons or at least icon sizes sufficiently big to display pixel perfection.

On the desk, on your lap