



Graphical User Interface Design

White Paper

Abstract

This paper describes the Design process for Graphical User Interfaces.

Tometa creates custom software for you

Tometa Software designs and develops robust software solutions for virtually all industries including in-house (vertical market) and retail software, some of which is on the shelves at your local software store. We focus our unique combination of creative, technical, and problem-solving skills on meeting our client's objectives. Because of our clarity of purpose, commitment to process, and broad professional skill sets, we are able to provide our clients with world-class solutions that are functionally superior and fully aligned with our client's strategic focus.

Balancing development speed, quality and cost is what we are all about. Tometa combines agile development practices with fixed pricing, so you know what the cost, end product, and delivery time table look like—up front. If we underestimate the effort, we complete the overrun on our dime. Simple as that. That's why large enterprise firms like Alcoa and NASDAQ choose Tometa.

Tometa's agile development expertise and low-overhead US location keep our prices comparable to offshore vendors – without offshore challenges. Using a fixed pricing model, we provide upfront visibility into a project's ultimate costs, end product and delivery schedule. Our clients like knowing that we have “skin in the game” – a fixed price that aligns our goals with yours, incenting us to get the job done right and fast.

Lastly, as a Microsoft Certified Gold Partner, Tometa Software, can customize its products or create custom web, client/server, and traditional applications. With programming experience in C#, C++, Visual Basic, CGI, HTML, RPG, Delphi, Java and many others; Tometa Software is uniquely positioned to meet your needs as a development firm.

[Check us out today](#)

Making a good Graphical User Interface (GUI) is quite often the make or break point of creating a brilliant application. The functionality of an application can be programmed perfectly, but if the GUI is hard to figure out or annoying to use, then the program ultimately will be a failure and the end user will likely choose something easier or more convenient. Since making a good GUI is vital to the success of a program, it can be the hardest part of creating a program.

In highly-funded projects, the teams that design the GUI can consist of many people with diverse skills. Graphics design artists, programmers, sound editors and even psychologists have teamed up together in search of the most intuitive and efficient GUI's possible.

This report will concentrate on the characteristics great GUI's have in common. It will also focus on why some GUI's fail. Finally, it will take a look at the difference in creating a GUI for a web application vs. creating a GUI for a desktop application. While this is a big topic, the majority of the detail will be about what to concentrate on when building a GUI and not a run down on how to build the GUI.

Key Ideas of GUI Design

GUI design is a relatively new portion of modern computing. The initial graphic user interface to bridge computers and users wasn't even available to the general public until the first Apple computers were released on the market. The biggest advancements in user interface happened when Xerox labs created the mouse and the modern point-and-click interface. For the purpose of this paper, GUI will refer to a point-and-click interface that is able to be used on the majority of PC's. Though more efficient GUI's are capable through the use of PC tablets and touch-screen devices, the majority of users still use a mouse and keyboard interface.

All OS GUI's have certain things in common: a method of starting programs from a list of available applications, a way of browsing the file structure, and if the OS supports multi-tasking then it will also have a way of accessing what programs are running currently. Most OSs do this pretty similarly. They use the same basic components—windows, icons, menus and taskbars make up a good portion of the default GUI experience. This is known as the WIMP interface. Applications expand on this and provide a variety of controls to make communication between the user and the application as smooth as possible.

GUI design and the WIMP interface has come a long ways since Apple took out 26 pages of advertising in several major publications to explain how a mouse works in an attempt to sell their new idea. Reading tutorials on GUI design from 5-10 years ago serves little value to a developer other than as entertainment value in comparing the way things used to be with the way they are now.

GUI design in today's world has an emphasis on being intuitive. An intuitive application is created by designing a predictable GUI that is consistent in its behavior to a users input. Obviously, looks are important but the feel is

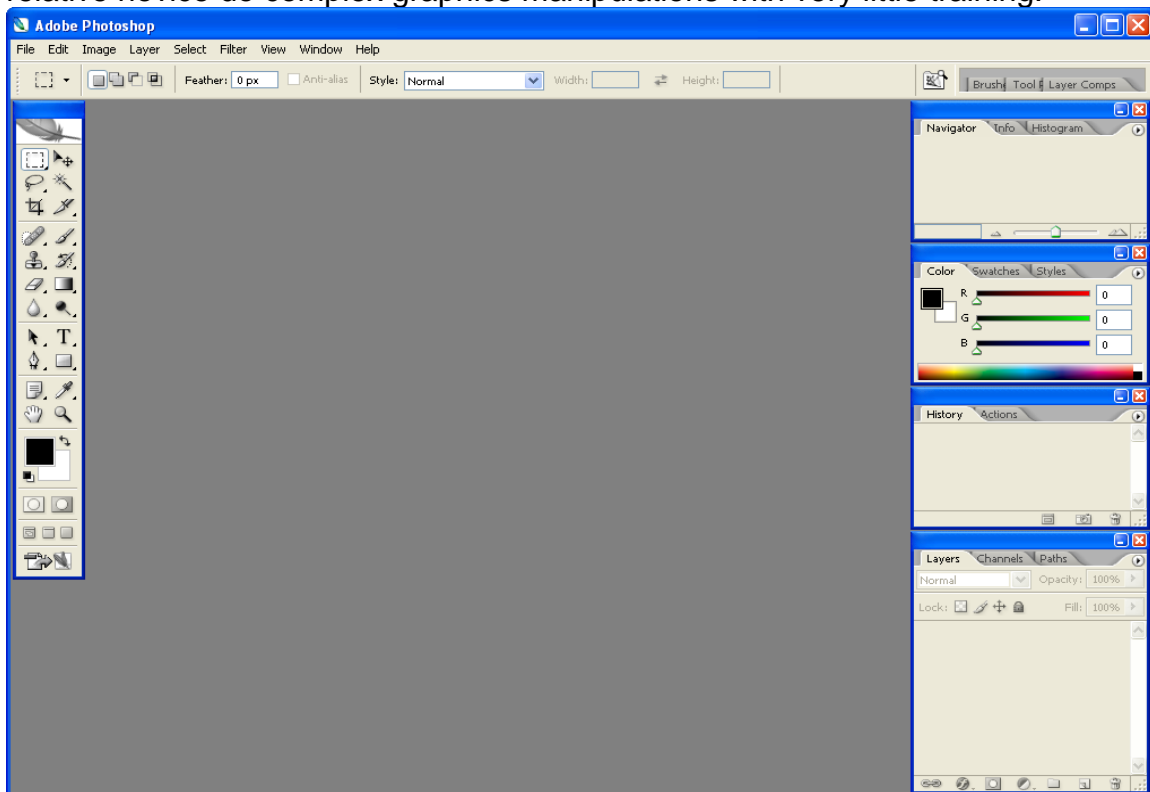
ultimately what the user deals with in a GUI. A simple GUI that is predictable and consistent provides the ideal experience for the user.

A simple GUI is geared towards the target audience to be within their realm of knowledge. If an application is being developed for senior citizens, then using a tree view is probably not the best way to display data because the majority of them don't have enough experience with computers to feel comfortable with that method of displaying data. The GUI should be appropriate to its audience and shouldn't be any more complex than is necessary to provide the ideal functionality. Ideally, a great GUI accommodates its users. It is as transparent as possible while allowing the user to work freely, with the focus on the content of the work while being visible enough to allow the user to freely access the controls being used to work.

A great GUI not only needs to be intuitive and simple at taking input from the user, but it needs to be responsive, so the user knows immediately what the result of the input was. This is often done with sounds and status bars.

The Good and the Bad

In the assumption that most users are familiar with computing in a GUI interface, designers are allowed to make things as simple as possible while remaining intuitive. This is the ultimate balance to strike when making a great GUI. A great example of a simple yet intuitive interface that still provides a lot of functionality over a large set of controls is the Adobe Photoshop UI. In the opinion of the author, Adobe Photoshop is a success as a GUI because it lets a relative novice do complex graphics manipulations with very little training.



The interface is made of components that are mostly familiar. The icons on the left represent do a good job representing the tool associated with them, and the arrows in the lower right hand corners of the icons mean that there are more related tools to select in the group. The windows on the right display information associated with a subject. The top window shows data about the whole image. The window below it shows information about color selections. The windows can be closed and opened using shortcut keys or the mouse.

What makes the interface worth mentioning is how intuitive it is to most users. If the controls are familiar to the user, the user can figure them out pretty quickly. The multiple icon interface used on the tool pane on the left is pretty unique to Adobe products, but once a user learns it, it provides a fast way to find a tool that the user isn't certain where to find. The intuitive user interface allows the user to learn as they get work done.

The final beauty of the Photoshop interface is that it represents simply a lot of different tools and methods to get things done. By having a simple interface that stays out of the user's way, the user doesn't feel overwhelmed. The workflow using a product with a simple but intuitive user interface is bound to be much higher than using something with a user interface that interferes with the user.

An example of an interface that interferes with the user is apparent in GUI's that use two list boxes to organize information. If they aren't designed with the utmost care, a simple mistake can completely ruin the usefulness of the product.



Here we can see in a screen from Microsoft Outlook, it is impossible to read the entire text of the selected member of the list. This is an example of a user interface being more complex than it needs to be and getting in the way of the user causing a slowdown in work flow or mistakes. In the case of the example, a mistake could be quite embarrassing.

Another example of a common and fatal error in GUI design is to use a scroll bar where a list box should be used. The user is forced to scroll through many options when a list box could be used to list all of the options and the user could select from the list box. Once again, this is an example of a user interface being more complex than it needs to be.

Using drop-down menus to provide input is more tedious than it needs to be more often than not. While it may be desirable when inputting data into a form for a database, it is probably easier on the designer and the user to create a

method of verifying the information and then making the user correct it or even offering a guess to the right answer. The control used to communicate with the user always needs to be the appropriate one for the job.

A lot of the confusion in GUI design happens when the designer attempts to take short cuts in designing the program. GUI programming takes a lot of attention to detail and a lot of tedious double-checking to make sure every last touch is added. Forgetting a small detail here or there can cause the end user tremendous frustration as they have to deal with a poorly programmed control or deal with a user interface that gets in their way and interrupts their work.

Desktop vs. Web

GUI design is important no matter what medium the application is being developed in. Although the differences between developing an application for a desktop PC and developing it for a web browser are becoming less and less they are still significant and certain things must be remembered when differentiating between the two.

The major difference is that web-based applications do not easily support manipulation of objects when compared to their PC based counter parts. It would be nearly impossible to find a good multimedia editor that runs as a web application because supporting the manipulation of objects in that situation would be nearly impossible.

When developing for a PC, the interaction between the user and the program is a lot more immediate and allows for a lot more complex object manipulation. There is no excuse to not acknowledge every action of the user. Something should visibly change or be audible to the user when an action is performed. In a web application it is somewhat excusable to ignore this, especially in form-based applications where pressing submit causes the browser to inform the user that an action was performed.

Another important thing to keep in mind when designing web applications vs. designing desktop applications is that web applications typically operate in one view. Traceable paths are very important so the user of a web application doesn't feel lost. This emulates having multiple windows open in a desktop application.

When developing for the web, it is helpful to mind the similarities than the differences between developing for the web and for the desktop. By replacing the functionality provided on the desktop in an intuitive or natural manner, a web application will provide great functionality without frustrating or alienating the user.

Conclusion

When it comes to great GUI design, several key points ensure that the developer concentrates on the right things. The main idea of a GUI is to accommodate an interface with the application's user in an intuitive and simple manner. The GUI should be out of the users way until functionality from the GUI

is needed at which point that functionality should be easy to locate. The GUI should be responsive and immediately let a user know that an action has been performed. Finally, no detail should be overlooked so that multiple ways to do each function are provided and the GUI has a feeling that it is well-put together.

This document is for informational purposes only. Tometa Software, Inc. MAKES NO WARRANTIES, EXPRESS OR IMPLIED, IN THIS DOCUMENT.

© 2004 Tometa Software, Inc. All rights reserved.