Every pixel is precious
(or, "How I learned to stop worrying and love interface design")

By Phillip Kerman

Overview:

To gain awareness of the fundamental "rules" for good interface design we will explore common flaws and look for solutions. The goal is to create interfaces that communicate transparently—as opposed to creating interfaces with which the user must compete.

In addition to traditional human-computer interface theories from such pioneers as Edward Tufte and Donald Norman, insights from the presenter as applied to computer based training and specifically Authorware, will be included. Finally, case studies and Authorware models provided will attempt to supply the tools necessary to create intuitive designs which maintain a high level of integrity.

Outline:

"It's hard to make something easy"
- Interface design's first priority is the user.
- (Other important issues: accuracy, clarity, and consistency.)

Why should you care?
- Or, more importantly, how can you convince others it's worth the investment to make something right.

How to?
- You *can* learn by doing, however we can also learn from others.
- Edward Tufte's graphical rules (graphical integrity, data-ink maximization, unintended optical art, chart junk).
- Don Norman's usability theories.

Critique bad interfaces:
- QuickTime 4's movie player.
- Flash's action script window.
- Others too… but realize it's easy to criticize—the criticism should offer solutions.

Guidelines applied to multimedia and computer based training —with examples (see list on other side).

Summary

Credits:

And/Or example
Brandon Blank of The Human Element, Inc.
http://www.thehumanelement.com

Compaq Non-stop systems and
Microsoft Windows NT4 Information Center
Produced by New Interactive
http://www.newi.com

Columbia Sportswear Custom Catalog Builder
Produced by Oswego Group
Graphics: Randy Keener
http://www.oswegogroup.com/

Race track interface
Graphics: Andy Schlabach, Split Diamond Media

HP 2100 & HP 3150
Produced by Waggener Edstrom

Side Effects
Produced by Nathan Lucas

Revolver (from 1997)
www.gabocorp.com

Dutch Train Schedules
http://www.ns.nl

You Don't Know Jack
www.bezerk.com

FAST! Actionscript tool
Grooveware Multimedia
http://swifftools.com/stools/

Bibliography and suggested reading:

The Art of Human-Computer Interface Design by Brenda Laurel, Published by Addison-Wesley (ISBN: 0-201-51797-3)

The Design of Everyday Things
By Donald Norman
Published by Doubleday Books (ISBN: 0385267746)

3 Books by Edward Tufte / Published by Graphics Press
The Visual Display of Quantitative Information (ISBN: 0-9613921-0-X)
Envisioning Information (ISBN: 0-9613921-1-8)
Visual Explanations: Images and Quantities, Evidence and Narrative (ISBN: 0-9613921-2-6)

Quick Time article:
http://www.iarchitect.com/qtime.htm
Brian C. Hayes of Isys Information Architects Inc.
http://www.iarchitect.com

Guidelines to follow and common flaws to avoid:

If you can't figure it out, how will they? If you ever find yourself (or anyone in your team) misunderstanding an interface component, treat this as a sign that you must fix it! For example, if you keeping clicking the "Back to Main" button when you're intending to only go back one page… if it happens to you even once realize it will likely happen to your user too.

Multiple routes to same destination is okay—try "teaching" with repeated images and words. Those hung-up with consistency sometimes recommend against providing the user two different avenues to the same information (like two buttons to reach the "main menu", for example). There is nothing wrong with this! Different people like doing things differently. When you use this technique it is recommended to use the same words or images (for example, always call the main menu "main menu").

Make sure labels match. Often the best way to assure labels are clearly associated with the thing they're labeling is with *proximity*. That is, simply place the label close. Other ways include: *color-coding* (maybe both the label and the thing being labeled are temporarily outlined with the same color); *call out* (by drawing a line to connect the label to the thing); and *repeating* the sub-title (a "mini-label") in the body of the explanatory text.

Overcome the "1 of 2 dilemma". An issue often arises when trying to highlight something (when there's only two total)—like highlighting *True* when both True & False are available. No matter how clearly to attempt to indicate the highlight, you're always left in a situation where one thing is highlighted and the other isn't. The difficulty for the user arises when they try to de-code your graphic treatment for "highlighted". For example, if you decide to draw a bright green box around the selected button—in the end, the user only knows that one is green and the other isn't! If the user remembers the un-selected state then they might be able to figure it out. However, your graphic treatment must go over and beyond subtlety to be effective in this situation (of highlighting "1 of 2"). Try to make your graphic treatment pass the following test: can any user ascertain which is highlighted even when they are not given the benefit of viewing the screen in the pre-highlighted state?

You can repeat text, but not at the top of a paragraph. When you find yourself repeating a block of on-screen text, it's important to assure the repeated text does not appear at the beginning of each paragraph. Generally, people don't read. Even the most thorough readers quickly "tune-out" and skip text which is familiar or obviously repeated. By placing such repeated text at the end, you assure the new or unique text has a good chance of being read.

Less is more. In the case of "You Don't Know Jack", for example, there are practically no graphics—just a simple interface.

Leverage off standards when you can—certainly never contradict. Standard computer interface conventions (e.g. menus, checkboxes, radio buttons)—for better of worse—are easy to apply to your advantage. If the user has bothered to learn a convention, you're wise to use that knowledge to your advantage. If you want to teach them something new—that's okay. According to Donald Norman, it's okay to make the user learn some new convention—just remain consistent so they don't have to keep re-learning. Finally, regardless of whether you use standard conventions or create your own—never contradict traditional conventions. For example, radio buttons should only be used in cases where there's an exclusive choice (e.g. male/female). But don't use radio buttons for a multi-select situation—use checkboxes instead. There are several other standard conventions with which you should follow. For example, underlined text on web pages denotes hyperlinks.

If it's not adding anything, then it's distracting. Edward Tufte calls it "Chart Junk" and it's anything that's not supporting your message. This is not to say aesthetics have no value… just that everything should have a purpose (your purpose could include being "pleasing to the eye"). Your interface has precious little space for superfluous content. The concept of "Data-Ink Maximization" (by Edward Tufte) attempts to calculate the information density as a ratio of information communicated compared to pixels (or "ink") used.

There's never enough room—try designing your interface on an index card. A computer screen can only hold a fraction of what a piece of paper can (the visual resolution of paper is hundreds of times greater than the screen). In an attempt to put more and more information on the precious screen space that you *do* have… you will eventually run out of space. Therefore, a good trick is to attempt to draw your interface on a small index card. (For example, just this paragraph barely fits on the projected screen.)

Rules like "Navigation should never take more than 11 clicks" or "7 plus or minus 2" can be misinformed. Different content can be organized differently. Tables are great for reference material, but maybe not for sequential information. In the case of the rule not more than 7 items (plus or minus 2) was based on a study of how much someone can *memorize*. For users "scanning" a list, they can handle many more items.

When testing, don't instruct just watch. This is more a usability concern than interface design. It's important, however, to test your product's usability with un-biased, un-directed users. When you test your own product there's a form of "group-think" that occurs—where you incorrectly convince yourself a design is valid.

Make sure the program is fun or engaging (even for you while testing). If running through your program is not compelling to you—how will it be for the intended audience? Measure a program's "playablity" based on how fun it is for you.

Phillip Kerman phillip@teleport.com www.teleport.com/~phillip/ +1 503 236-7721