Maintaining Consistency in a Changing World:

Apple® **Human Interface Checklist**

Consistency is our edge: the central core of established objects and behaviors in the Apple Desktop Interface can be and should be maintained. We have compiled this Checklist to offer designers and reviewers a concise source for the most important, consistent elements of the Apple Desktop Interface.

The Checklist discusses strategies for designing and testing, reviews the main principles behind the guidelines, and offers a list of important items to keep in mind while testing.

The material is intended to supplement the book, *Apple Human Interface Guidelines: The Apple Desktop Interface,* published by Apple through Addison-Wesley.

Apple Human Interface Guidelines covers the principles in greater detail; gives the specifications for windows, menus, and other elements; and describes the special problems of developing software for international markets. *We urge you to get and read this book.*

 APPLE COMPUTER, INC.

Written by: Debra Brackeen

Deborah Grits

Barbara Mackraz

Bruce Tognazzini

20525 Mariani Avenue
Cupertino, CA 95014-6299
(408) 996-1010

Simultaneously published in the United States and Canada.

Contents

A strategy for designing

Users are most comfortable and productive in an environment which remains familiar and predictable. They are confused by interesting but unnecessary variations in visual or behavioral presentation. They become upset when the system suddenly interprets their own, traditional behavior in new and unexpected ways.

The power and performance of applications continue to grow. We urge developers to create entirely new objects and behaviors to support emerging ideas and capabilities. At the same time, we caution against simply changing existing interface elements, particularly when that change is either subtle or not absolutely necessary to the user's operation of your application. Therefore:

- Make differences obvious: users do not notice subtle visual or behavioral changes. They are confused when the system no longer seems to interpret their desires in the same way.

- Make changes only when they will result in significant increases in user-satisfaction or productivity. Changes to standard interface elements that are made for the sake of personal aesthetics or simply to differentiate a product always confuse: our users have come to expect every element of the design to communicate information about the task. If a scroll bar is different, for example, our users will assume that there is some new ability inherent in it and will set about the task of finding out what it is.

- Remember that users often run several applications simultaneously: unnecessary differences among them cause continuing frustration and a high error rate.

Design Hints

The Apple Desktop Interface relies on some distinctive models for design. The Human Interface Guidelines provide six tips to help developers adhere to these models. It is important for testers as well as developers to be aware of these points:

- With few exceptions, a given action on the user's part should always have the same result, regardless of past activities. In other words, don't use modes unless there's a very good reason to.

- Applications should be prepared for the user to do anything at any time. This means that the event loop should be the central routine of an application.

- Users should be able to reverse any action. Always provide a way out for them.

- The screen is the stage for human-computer interactions; it should represent the "world" that the computer creates for the user, showing all the alternatives, the requested activities, and so on.

- When the computer must display textual messages, use simple and concise terms.

- Test early, and test often.

International Considerations

We have entered the era of the World Market. By planning early for eventual localization, you can quickly and easily translate your application into many different languages.

The rules are neither difficult nor cumbersome, as long as you plan from the beginning. Most center around concepts no more difficult than allowing for extra white space, so that words can expand in length and characters can be larger or display diacritical marks.

Appendix B of Apple Human Interface Guidelines discusses some of the technical details necessary for eventual painless translation. The rules change as the Apple interface continues to grow. Always check the latest Human Interface Updates.

The Checklist covers those few elements of internationalization that could be picked up by an external review of working American-

language software: most of the planning for internationalization is internal, and can be seen only by the programmers.

A strategy for testing

The book, *Apple Human Interface Guidelines,* describes the principles of the Apple Desktop Interface and the particular specifications of all standard elements. It is a good place to begin for testers who evaluate the high-level aspects of an application. In addition to this book, updates are occasionally distributed that describe the interface of new features such as hierarchical menus.

In areas not covered by the specifics in the guidelines, standard applications serve as useful models. For example, in graphics programs a tool palette interface similar to the pioneering applications of MacPaint or MacDraw would make most users immediately aware of the capabilities of the application.

Consider ease of learning

How often a product will be used should have bearing on how easy it is to learn. If the product is likely to be used only occasionally, it should be apparent how to use it in all its power; a user is not going to be willing to spend much time learning it. In this case, an environment that seems intuitive and predictable is all the more crucial. If the product is likely to be used very often, a user can be productive right away but can grow into the product's full powers—in other words, learn in stages.

For example, a tax program used once a year should be easy to understand the first time and easy to remember a year later, but if a user works with a word processing program every day, he or she can pick up more and more of its features while using it. Of course, most products fall somewhere between “once a year” and “every day.” You should have some idea of where the product you’re working with falls on the scale.

This also applies to the parts of a product. If a particular feature of a word processing program, such as line numbering, is likely to be used only rarely, it should not require as great an investment in learning time as, for example, text formatting.

Use common sense

Common sense and sound judgment should come into play as you determine how literally to apply the guidelines. For example, it would not be wise to take metaphors from the real world to their logical extreme. A paint program might have a thumbwheel to change line size, but it would take unbearably long to scroll on it from 1 to a large number. In this case, a type-in field would be more sensible.

Even though the best programs are usually close to the guidelines, sometimes developers modify an aspect of interface in a way that is reasonable and works well. It is important for testers to be sensitive to this. The goal of evaluating the interface is to see that a program essentially conforms to the guidelines so that users don’t have to relearn the basics and can immediately feel at home in the environment—but at the same time not thwart innovation that might improve upon the interface for a particular need.

The Human Interface Guidelines are just that—guidelines, not steadfast rules. Use common sense and sound judgment when determining how literally to apply them.

The Principles

This section summarizes the fundamental tenets of design and graphics in the Human Interface Guidelines and discusses accessibility to disabled people.

General design principles

Ten principles lie at the heart of the guidelines:

Metaphors from the real world

Use metaphors based on real-world counterparts and make them plain, so that users have a set of expectations to apply to the computer environment. Carefully craft a visual, aural, behavioral illusion to support the metaphor, so the user can operate in a stable artificial reality.

Direct manipulation

Users want to feel that they are in charge of the computer's activities. They expect their physical actions to have physical results, and want their tools to provide feedback.

See-and-point (instead of remember-and-type)

Users select actions from alternatives presented on the screen. They rely on recognition, not recall; they shouldn't have to remember anything the computer already knows. Most programmers have no trouble working with Boolean logic and with a command-line interface that requires memorization, but the average user is not a programmer.

The general form of user actions is noun-then-verb, or "Hey, you—do this."

Consistency

Effective applications are both consistent within themselves and consistent with one another. Users like to rely on familiar ways to get things done. (See the section, A strategy for designing)

WYSIWYG (what you see is what you get)

There should be no secrets from the user, no abstract commands that only promise future results. There should also be no significant difference between what the user sees on the screen and what eventually gets printed.

User-initiated actions

The user, not the computer, initiates and controls all actions. (People learn best when they're actively engaged.) This is different from the more traditional model, in which the computer acts and the user responds with a limited set of options.

Feedback and dialog

Users appreciate immediate feedback on the progress of an operation. Communication should be brief, direct, and expressed in terms of the user's point of view.

Forgiveness

Users make mistakes; forgive them. Forgiveness means letting users do anything reasonable, letting them know they won't break anything, and always warning them (but not restricting them) when they're entering risky territory.

Even actions that are risky should be reversible—let users know about any that aren't.

Perceived stability

Users feel comfortable in a computer environment that remains understandable and familiar rather than one that changes randomly. Consistent graphic elements provide visual stability; a finite set of objects and actions to perform on them provide conceptual stability.

Aesthetic integrity

Visually confusing or unattractive displays detract from the effectiveness of human-computer interactions. Messes are acceptable only if the user makes them—applications aren't allowed this freedom.

Users should be able to control the superficial appearance of their computer workplace to display their own style and individuality.

Principles of graphic communication

The real point of graphic design, which comprises both pictures and text, is clear communication. In the Apple Desktop Interface, everything the user sees and manipulates on the screen is graphic. As much as possible, all commands, features, and parameters of an application, and all the user's data, appear as graphic objects on the screen.

Good design must communicate, not just dazzle. It must inform, not impress.

Three principles in the guidelines govern graphic communication:

Visual consistency

Visual consistency constructs a believable environment for users. Because such concepts as storing documents in folders are the same both in the real world and in the Apple desktop, users don't have to relearn them.

Simplicity

Simple design is good design. The screen should never have complex icons and never be cluttered with too many windows or dozens of buttons in a dialog box.

Clarity

Good graphic design begins with an understanding of the situation the user is in or the problem being solved. A picture isn't always the answer—sometimes words are more appropriate. In any case, graphics should be clear and readable.

When its time to get help, hire an artist. The few hours a free-lance artist will take in making your application understandable will be well worth the small investment.

Designing for disabled people

A primary tenet of the Human Interface Guidelines is to make the interface accessible to everyone possible, including disabled people. Computers hold tremendous promise for people with disabilities, but all too often computers have ended up instead as obstacles—when a few modifications in the hardware or software could have made things very different.

Developers and testers should keep this in mind when working on applications. For example, software with an enlarge feature can make it easier for people with vision problems to read the screen; audible messages with visible cues such as a flash on the screen can make it possible for people with hearing problems to take notice.

The Checklist

This section lists questions about the Apple Desktop Interface that you can ask yourself while reviewing software. These questions will help bring to mind the particulars of the guidelines.

The questions here apply to all the specific sets of standards except selection. The selection standards of the guidelines are detailed, so refer to “Selecting” in Chapter 3 of Human Interface Guidelines for information about them. The reasoning behind most of these questions can also be found in Chapter 3.

Every question must be answerable by “yes” for conformity with the guidelines.

General considerations

- ☐ Does the application have the “look” of the Apple Desktop Interface (including, but not limited to, desktop, windows, and menus)?
- ☐ Does the application have the “feel” of the Apple Desktop Interface (including, but not limited to, pointing, selecting, and keyboard input)?
- ☐ If a metaphor is being used, is it suitable for the application? Does the metaphor have a “real” visual and behavioral representation, as with the desktop, so that the users do not have to carry a “map” in their head?
- ☐ Does the application always provide some indication that an activity is being carried out in response to a command?
- ☐ Does the user always have the option of finding an object or action on the screen (as opposed to having to remember and type)?

- ☐ Are the operations consistent with the standard elements of the interface; that is, if a user is familiar with such applications as MacPaint, MacDraw, and MacWrite, will the application seem like familiar territory?

- ☐ Is a printout a replica of what the user sees on the screen (that is, WYSIWYG)?

- ☐ Is suitable feedback provided during task processing? Is the completion of a processing task indicated somehow? Is the duration of the task indicated?

- ☐ Is an explanation offered if a particular action cannot be carried out? Are alternatives offered?

- ☐ Are there warnings about risky actions? Are there different warnings for different levels of risky actions? Are there enough warnings without being too many? Are users allowed to back away gracefully from risky territory?

- ☐ Is there a feeling of stability? Are there enough landmarks to remind the user what area of the application he or she is in?

- ☐ Can an operation be interrupted with Command-period? Can Escape be used to cancel an operation that has a Cancel button?

Graphic design

- ☐ Do the commands, features, and parameters of the application, as well as all of the user's data, appear as graphic objects on the screen (as much as possible)?

- ☐ Are the graphics believable?

- ☐ Does the screen look "clean" and free from clutter?

- ☐ Does the user have control over the design of the workplace, allowing him or her to individualize it?

Window standards

- ☐ Does the standard state of the window seem appropriate on a 9-inch display? On a larger display?
- ☐ Does the preliminary user-selected state seem appropriate on a 9-inch display? On a larger display?
- ☐ Does the choice made to open in either the standard or the user-selected state make sense?
- ☐ Can each sizable window be made as large as the smaller of either the maximum document size or the maximum size of the displays, including multiple monitor displays?

Scrolling standards

☐ Does the window use either the standard scroll bar mechanism or the hand for scrolling? If it uses the hand, does the pointer either always become a hand in the window or appear highlighted in a tool palette?

Figure A–1. A hand highlighted in a tool palette

☐ Does clicking on a scroll arrow cause the document to “move” a distance of one unit in the chosen direction? (The unit should be appropriate to the application.)

☐ Does clicking in the scroll bar below the scroll box advance the document by a windowful? (A “windowful” is the height or width of a window, minus a one-unit overlap.) Does clicking above the scroll box move the document back by a windowful?

☐ If the user drags the scroll box and then moves the pointer well outside the scroll bar, does the scroll box snap back to its original position?

☐ Is the function of the arrow keys different from the function of the scroll bar? (Remember: all these questions should be answerable with "yes": arrow keys should not substitute for scroll arrows.)

Dialog box standards

A modal dialog box is one that the user must explicitly dismiss before doing anything outside it—these boxes should be used sparingly. A modeless dialog box allows the user to perform other operations without dismissing the box first.

- ☐ Is the question posed in a straightforward and positive way? For example, “Do you want to erase everything on this disk?” rather than “Do you not want to alter the contents of this disk?”
- ☐ When appropriate, do buttons have descriptive labels, such as “Destroy Power Supply” rather than “OK”?
- ☐ Does the default button (if any) have a bold outline around it?
- ☐ Does pressing Escape indicate Cancel in a dialog box? (Pressing Escape should never cause the user to lose information.)
- ☐ When a command key equivalent is used, does the button hilite?
- ☐ Do modal dialog boxes not lead to other modal dialog boxes?
- ☐ Do modal dialog boxes that can be moved have a drag region (title bar), as well as the two-pixel-wide outline within the content region to signify that it is a modal dialog?
- ☐ Has room been left to allow the dialog box to grow during localization? Most languages require more characters than English to convey equivalent messages.

Alert standards

There are three classes of alerts—Note, Caution, and Stop—each represented by a different icon.

Figure A–2. The three alert icons

- ☐ Do the alert icon and message fit the situation?
- ☐ Is a beep alert accompanied by a flash (rapid inverting) of the menu bar so that people who can’t hear don’t miss the message?
- ☐ Does the alert message not only tell the user what is wrong but offer suggestions as to what to do to correct it?
- ☐ Is this alert necessary? Often, the user can be prevented from making an error. Example: if the application cannot handle an 80 character file name, don't offer them an 80 character field in which to enter it.

Menu standards

- ☐ Does the menu bar contain only menu titles?
- ☐ Are the standard menus—Apple, File, and Edit—present, with at least the standard items? (Needed for Desk Accessories, even when the application doesn't use them.)
- ☐ Has enough room been left on the right side of the menu bar for the menu that some desk accessories add to the menu bar? Is there also enough extra room to allow for the expansion that almost always occurs during translation into other languages?
- ☐ Do the unique menus of the application have names that are appropriate? Are the names sufficiently different from the standard menu names? Can the user understand and remember their meaning?
- ☐ Are frequently used menu items available at the top level rather than in a hierarchical menu or a dialog box? If not, can the user move them up?
- ☐ When an item in a menu is currently disabled, is it dimmed in the menu rather than missing from it?
- ☐ If all the items in a menu are currently disabled, is the menu title dimmed? Can the user still pull down the menu and see the dimmed names of the operations?
- ☐ Are the names of menu items appropriate? Can the user understand and remember their meaning?
- ☐ Are menu titles and items in caps/lowercase unless there is a compelling reason to have a different style, such as “ALL CAPS” in a Style menu?
- ☐ Do menu items have an ellipsis (…) if more information is required from the user?

☐ Do hierarchical titles in a menu have a right-pointing triangle? Are hierarchical menus used only for lists of related items?

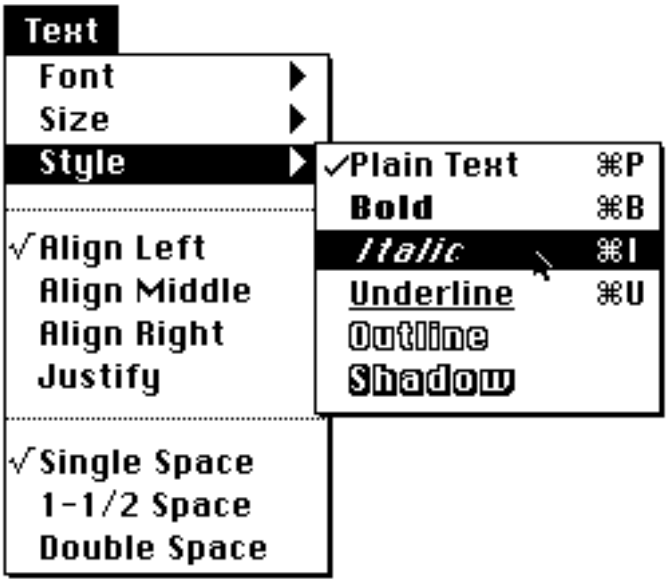

Figure A–3. A hierarchical menu

☐ Can the user see all the commands, items, and hierarchical titles in a menu without scrolling? Scrolling should be necessary only for menus that users have added to or for menus that spill over because the user has selected a large system font.

☐ Is the indication of a pop-up menu a drop-shadowed box around the current value? While the menu is showing, is its title inverted and is the current value checked? If the menu must be scrolled, is this indicated by up- or down-pointing triangles?

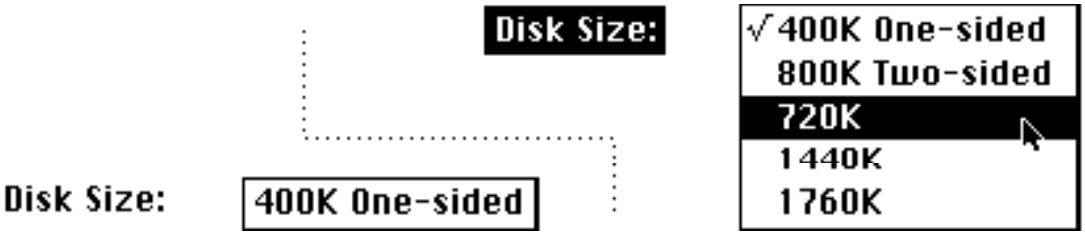

Figure A–4. A pop-up menu

- ☐ Do keyboard equivalents appear where appropriate? Are the keyboard equivalents case-independent. (This second rule does not apply if the product uses both cases in the keyboard equivalents and enables the user to predict which case to use.)

- ☐ Does the application support Undo, Cut, Copy, and Paste?

- ☐ If the application is text-oriented, can the user change the font and style by using menu commands? Usually, the fonts and style are in menus are called Font and Style, but there may be a good reason to put them in another menu.

- ☐ If a palette is present, is the selected symbol (icon, pattern, character, or drawing) highlighted?

- ☐ If a menu has been torn off and moved, can the user still get access to it from the menu bar? When it is torn off a second time, does the first instance disappear?

Mouse standards

- ☐ If the user initiates an action by pressing the mouse button, does the action take place only when the button is released?

- ☐ Are there ways other than double-clicking to perform a given action?

Programming strategies

You may want to refresh your memory about the main programming issues, especially modes and the event loop. See “A Strategy for Programming” in Chapter 1 of Human Interface Guidelines.

- ☐ Is there a clear visual indication of the current mode? Does the visual indication of the mode appear near the object most affected by the mode? For example, the MacPaint pointer changes to a pencil in draw mode and to a paint brush in paint mode.

- ☐ Is each mode absolutely necessary? Do the modes within the application properly track the the user's own modes? Do users consistently avoid the kind of errors caused by the program being in a mode other than what the user wants or expects? Making a mode visually apparent is no guarantee that the user will track it: test the application on users and find out what sorts of mistakes they are making. If the errors are modal, eliminate the modes.

- ☐ Can the user save a document or quit an application at any time, unless he or she is in a modal dialog box?

- ☐ Are the widest possible range of user activities available at any time? The user should spend most of his or her time in the event loop.

- ☐ Will a user unable to distinguish colors be able to use the application? Will someone without a color monitor be able to use it? The information conveyed by color coding should also be presented in another form, such as text, position, highlighting, gray-scale variations, or pattern. (These questions do not apply to programs in which the task to be carried out requires full-color vision on a color monitor.)

- ☐ Will a user with a hearing disability be able to use the application? Audible messages should be supplemented with visual cues or should allow the user to choose visible instead of

audible messages. (This question may not apply to music programs.)

- ☐ For those who cannot handle book-form manuals, is any part of the manual available in electronic form?

Documentation

- ☐ Does the manual include a glossary of potentially confusing terms that relate to the application or to the application's topic?

- ☐ If the manual refers the user to another document, is the reference more appropriate than having the information in the manual itself?