Lauralee Alben
Jim Faris
Harry Saddler

Making It Macintosh: Designing the message when the message is design

An interactive product from Apple Computer, Inc. called "Making It Macintosh: The Macintosh Human Interface Guidelines Companion," uses over 100 computer-based animations to illustrate the principles of the Macintosh desktop interface. This product is designed primarily to teach Macintosh software developers how to build the Macintosh "look and feel" into their applications. It can also be of value to human interface designers, software product managers, and educators and trainers in the field of human interface design. "Making It Macintosh" is a companion to the book, "Macintosh Human Interface Guidelines." Both documents are published by Addison-Wesley (1993).

Lauralee Alben and Jim Faris, of Alben+Faris, worked intimately with Harry Saddler at Apple and a team of experts in a day-to-day, decision-by-decision development process over two years. This case study illustrates some of this design process, a process that included graphic and interface designers on the development team from the project's inception to its final implementation.

About the Authors:

LAURALEE ALBEN AND JIM FARIS *are principals of Alben+Faris, a firm specializing in graphic and interface design for interactive multimedia, software applications and emerging technologies.*

HARRY SADDLER *is a senior instructional designer and a member of the User Experience Architect's Office at Apple Computer, Inc.*

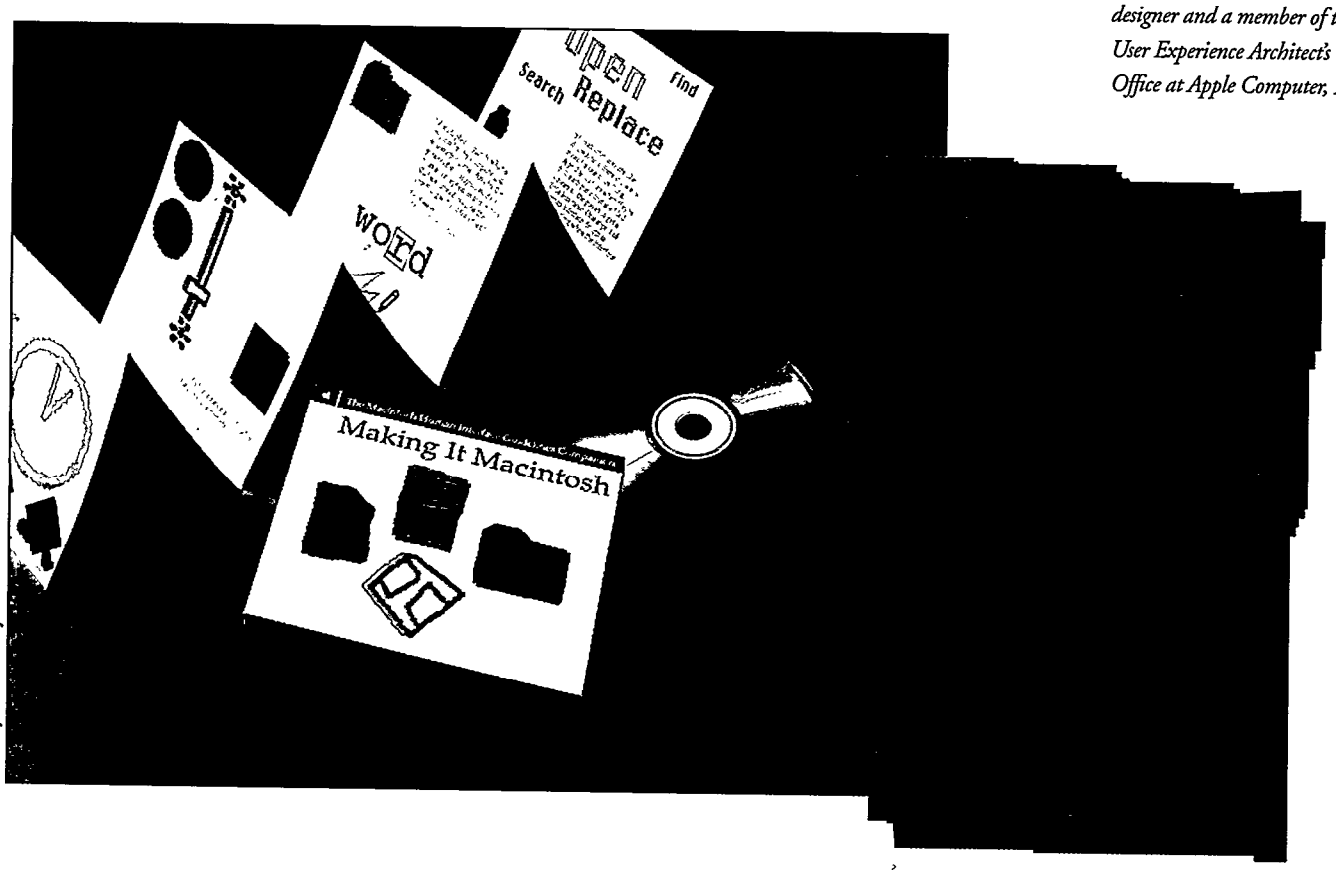


Photo by Bruce Ashley

Getting from here to there

Our design journey was an eventful progression that originated with a shared vision and culminated in the shipping of an interactive CD-ROM. Realizing the vision of supporting developers in creating effective interface design required the interdisciplinary collaboration of the team. Apple placed a high priority on achieving effective communication by including skilled and competent graphic and interface designers on the team.

Many complex issues needed to be solved in developing this product: content, technical, interface, instructional, navigational and graphic design issues. We were able to resolve these issues by designing iteratively and by looking simultaneously at details and at the whole. In the process we developed three very different interface designs, illustrated at right.

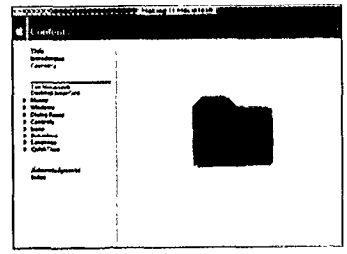
Having explored interfaces that were consistent with the Macintosh interface and others that were as different as possible in formulation, behavior and coloration we concluded that we needed an interface that was a compromise. The interface design also evolved as the scope of the content changed, from four parts to one, the Example part.

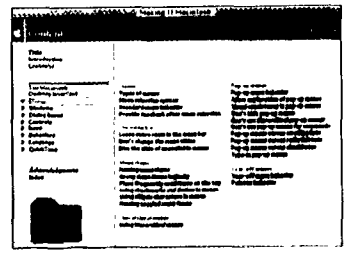
"How do we design an interface for a product that describes another interface?"
We created a simple and obvious interface that neither duplicated nor conflicted with the Macintosh desktop interface.

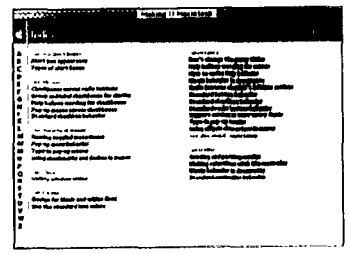
In the release version, we modeled the Contents screens after the System 7 Finder (right and below). When users click on the arrow to the left of the section titles, a display of the related topics appears on the right side of the screen. We expanded the Contents list to the right instead of down to eliminate the need for scrolling. The Index also follows this model. The gray background color connects the topic with its contents list on the right of the screen. The blue folder icon identifies the Example part.

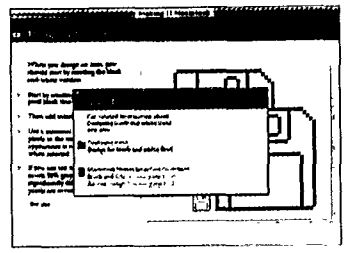
We also used a flat, linear structure, so users can navigate by buttons in the header bar to step forward and backward, screen by screen.

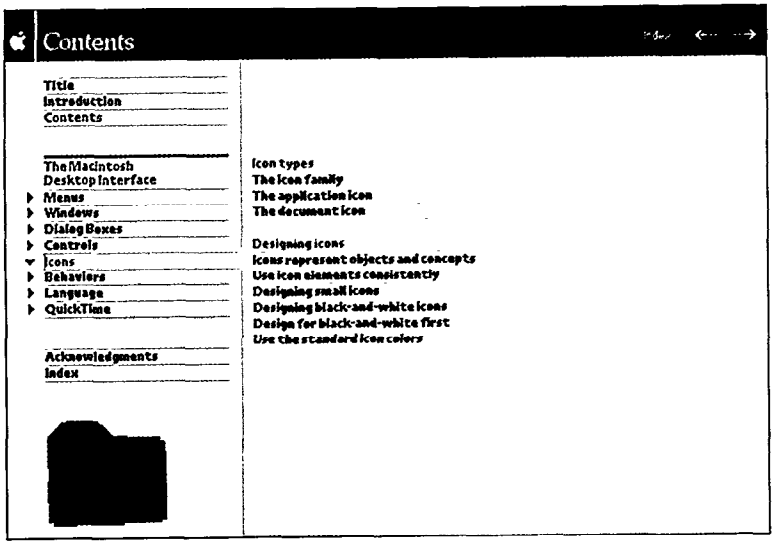
On the example screens, a See Also window can be summoned which allows users to navigate to related screens and back again. It also references related pages in the book.



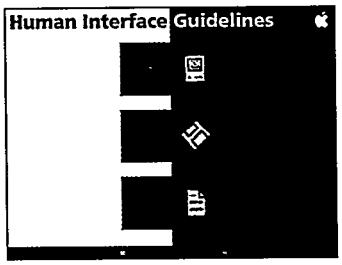


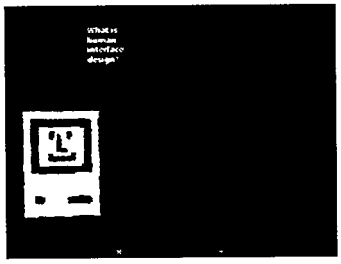


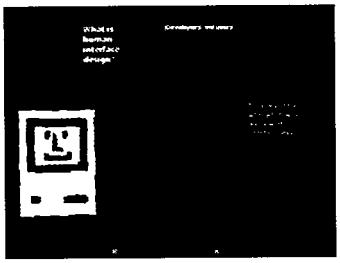


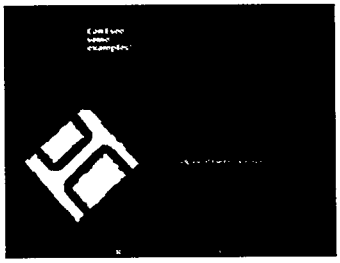


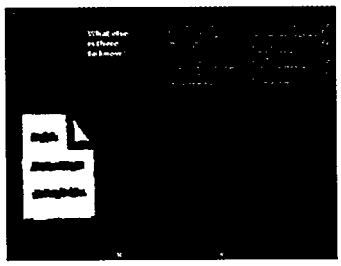
"Is this a multimedia presentation or a Macintosh application?"
Our answer was the former, but users disagreed.









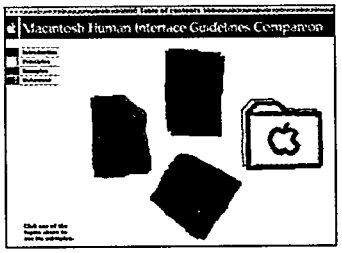


We initially thought of this product as a multimedia presentation. As a result, our first interface design (right) used the entire screen, eliminating any distraction of the desktop. We created a custom control bar at the bottom of the screen which provided general controls and direct access to the desktop. The bold use of color and large icons played a large part in creating an inviting world in which users could interact.

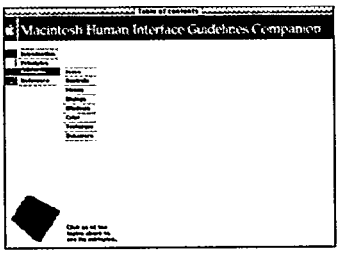
This interface was designed around the idea that there are two worlds developers need to bridge when designing applications: their own world and the world of users. The divided screen became a visual symbol of these two worlds.

Clicking an icon on the opening screen took users to the first screen of that section where a list of topics was displayed. Clicking on one of these topics displayed a list of examples and a "go to" arrow, which took users to that example screen. The visual appearance of this interface was very well received in formal user testing, but the tunneling nature of the navigation scheme proved slow and did not provide a good overview of the program contents.

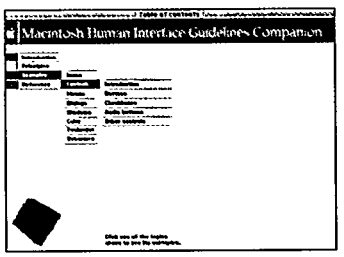
In the second interface design (right), we employed the standard Macintosh convention of running the program in a window on the desktop. By popular user vote the product became a Macintosh application.

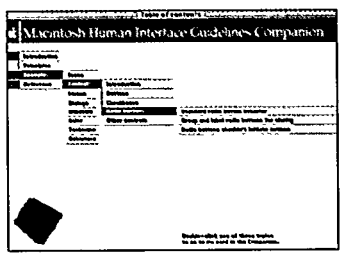


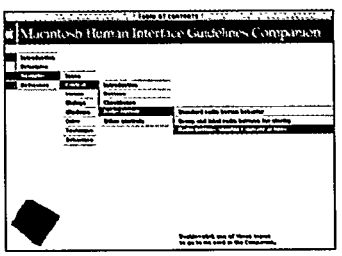
In this interface design we used a "way-finding" model for the Contents screens. The content is grouped in a hierarchy with four levels. Different paths through the hierarchy could be viewed and then selected by the user. Double-clicking on a topic sent users directly to the appropriate screen.



This interface gave users a visual map of the hierarchy of the product at a glance. It was initially surprising to users but easily understood. However, we eventually rejected this solution because of the extra clicks required to get to the lowest topic level. Another drawback was that the list of individual examples could not be displayed on a single screen without scrolling, which was undesirable from usability and technical perspectives.







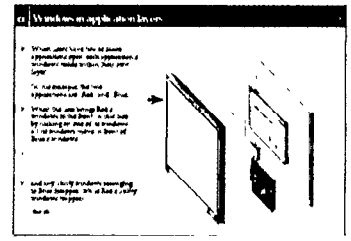
Example screen anatomy

At the heart of *Making It Macintosh* are the 100 animated examples of interface design. The design of the example screens was entwined with the instructional objectives of the project. The relationship between all the screen elements in terms of color, graphics, animation, composition, behavior and interaction were coordinated to achieve a conducive environment for learning.

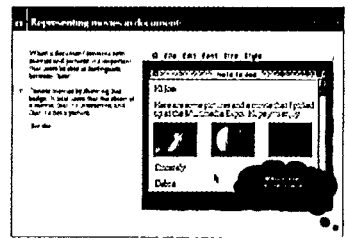
We used layering as a key organizing device in the visual presentation. At the lowest level are the standard screen components, such as the header band, the standard gray background, step buttons and text. The second level consists of the animated graphic depictions of the interface example. These graphics always have a shadow to separate them from the background. A torn edge indicates that a portion of the desktop is depicted. The top-most layer consists of annotations and callouts that comment on the interface depiction. All elements in this layer appear in the standard blue color and cast a shadow on the lower layers. These visual techniques help users distinguish between the objects in the examples and their own desktop.

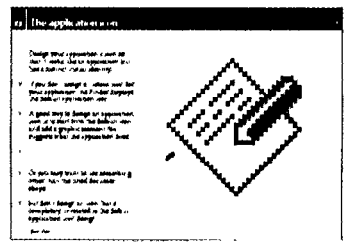
We designed a system for the team which specified the look, behavior and location (using an underlying grid) of all the standard screen elements. This created consistent screen layouts from example to example and provided a framework for writing, designing, animating and building the individual examples.

"How will interactivity influence the design?"
By using simple, "indirect interaction" we freed users to concentrate on the instructional examples rather than the interface.

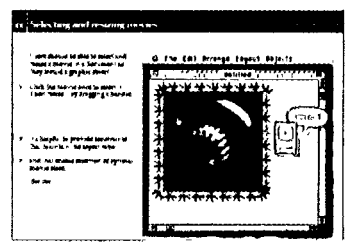


After experimenting with many different formats, we decided that the right side of the screen would not be interactive; rather the interface depictions placed there would be graphic animations. This greatly simplified the instructional design process by limiting the need to implement consistently interactive behavior in the depictions. Users could then control the pace and order of the animations from the left side of the screen, triggering the animation segments by use of the step buttons and textual narratives.





The animations generally depict Macintosh interface elements as they are manipulated by the user. In other cases, interface elements are juxtaposed for comparison, or transfigured to demonstrate an improvement or the selection of a more appropriate interface element or style.



"What is the optimal step button design?" *Using commonly recognized "play" and "pause" symbols clarified their functions.*

"How can we systemize the design and still allow for dynamic changes?" *Here is the system we developed for the team.*

We felt the way to view the examples should be obvious, without special instructions from the interface. The design of the step buttons went through numerous iterations to determine which size, color and symbols communicated most clearly to users. In the end, a black "play" arrow (blue in the selected state) worked best. A "pause" symbol was also used when an example step was playing. These symbols were placed inside a three-dimensional, gray shape, which integrated with the overall screen composition and color palette and yet was noticeable enough for the user to locate easily and use correctly. Both buttons and corresponding text are clickable.

Early designs using numbered step buttons were confused with multiple stages in the instructional content. Dots on the step buttons were confused with traditional bullet points; not every step represented a distinct instructional point. Grey dots were thought to be dimmed out, hence unavailable following the Macintosh convention. Bright colors like yellow and red distracted from the content. We also tried eliminating buttons and placing the text on three-dimensional rectangles, reversing the positions of the text and buttons, and having buttons in changing positions, all of which proved unsuccessful.

The icon for the example part always appears reversed out white in the header band.

The title for the section is in 9 point Espy sans serif bold, in black.

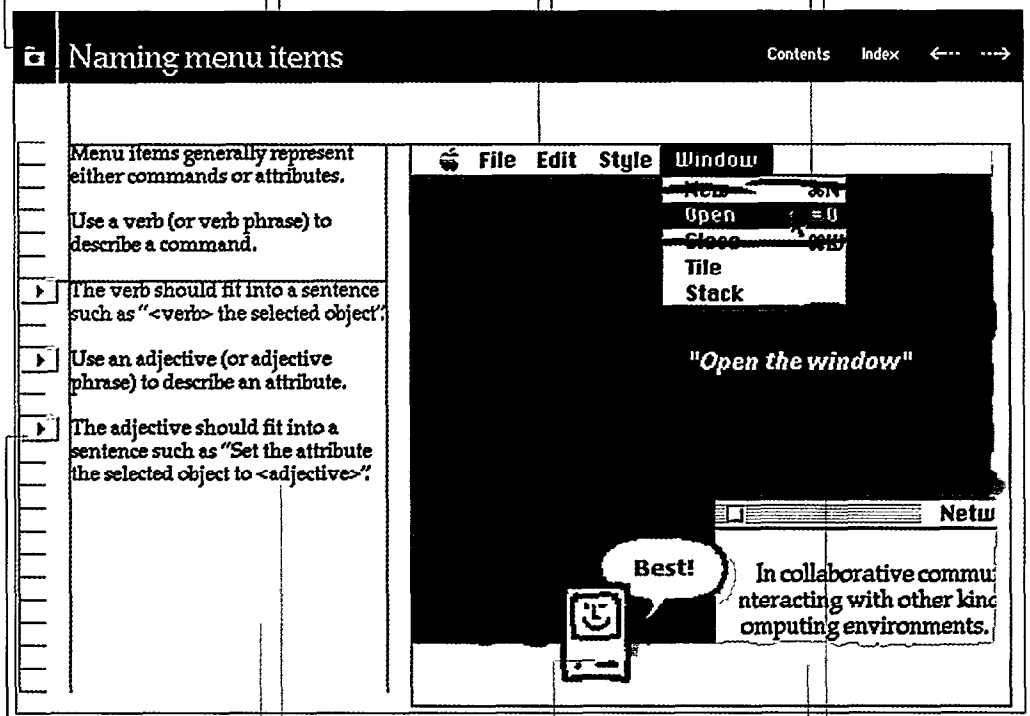
The example title is 18 point Palatino, in white. Initial capitalization only.

Electronic grid lines are red, showing exact positions for all screen elements and their optimum relationship to one another.

This is the header bar. Its position and size remain constant. Color is standard blue in the Example part.

Callouts are positioned on top of objects being referred to, in standard blue. Callout artwork is provided in various sizes.

These four navigation controls remain in this position and appear on every screen. Color matches the header bar.



Step buttons are positioned on hash marks, corresponding to the spacing of the text. Arrow and pause symbols appear in standard blue, in their selected state.

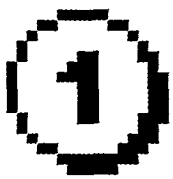
Text is in 12 point Palatino, flush left, ragged right, in black. One line space between paragraphs. Top of capital letters in the first line align with top of the arrow in the step button. Text appears in standard blue when selected. Minimum margin between text and graphics is 16 pixels.

"Luke" is positioned as close as possible or slightly overlapping the object to which he refers. Shadow position is 5 pixels down and to right. Luke always appears in standard blue. Alternate Luke expressions are provided for different instructional points.

User thought balloons touch a right or bottom edge of the screen Text is 12 point 86 Helvetica Heavy, in white, on a standard blue balloon. Shadow position is 5 pixels down and to right. Three balloon sizes are provided.

The background is always standard gray to provide a neutral environment.

Graphics depicting a portion of the desktop use a torn edge and shadow. Graphics (including shadow) should stay within grid lines and in most cases are centered right-to-left. Either "hang" graphics from top of grid, or center top-to-bottom.



The finer points of design

As we designed the interface elements, our primary consideration was communication, not decoration. In some cases, one pixel made a difference in achieving this goal. From the primary icons to the step buttons by which the user controls the interface examples, from the user thought balloons to Luke, we focused on the smallest details without losing touch with the larger context.

Interface design issues involve more than meets the eye. The issues we studied included: What ideas need to be communicated to the user? What functionality and content are represented? How will the interface elements relate to the look and feel of the application, product identity and operating system? How will the interface elements relate visually to one another and to the screen layouts, in terms of color, type, form, composition and so on? How can the interface be made extensible for later versions? What are the technical requirements and limitations (bit depth, memory processing speed, color palette, etc.).

"How will we depict the user's experience?"
The user's thoughts occasionally appear in cartoon-like thought balloons in the interface examples.

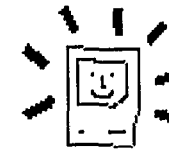
Depicting likely user reactions gave us a way to critique specific interface problems and solutions. The user thought balloons needed to be graphically differentiated from Luke's comments. We arrived at the solution of giving Luke an oval speech balloon and giving the user a cloudlike thought balloon. These balloons were designed in three standard sizes to allow for varying amounts of text.

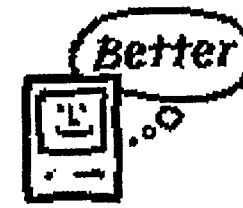
"How will we present evaluations of alternative interface solutions?"
"Luke" is an animated character who provides commentary in the interface examples.

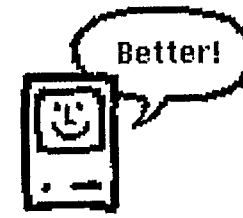
Luke is a humorous character with a colloquial patois. As the design of Luke evolved (below), we decided to employ verbal expressions, rather than abstract symbols, to convey the myriad distinctions we needed (some of which are shown at right).

Handwritten text (below middle) looked too much like the italic typeface in the user thought balloons. The Chicago typeface (below bottom) looked too much like the ubiquitous system font that it is, and it did not marry with the illustrative style of the Luke icon. Espy bold (right) gave the speech balloons a distinct look which fit the visual style.

After researching many typefaces in search of an appropriate visual voice (above), we chose a bold, italic Helvetica, which gave a stronger emphasis to the user's reactions (below). Both Luke and the user thought balloons are in the annotation layer and cast a shadow on the layers below. They give us a way to point out design flaws and successes without evoking the sense of an "interface police."



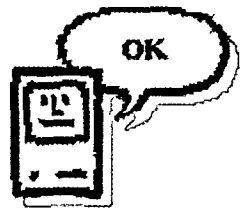


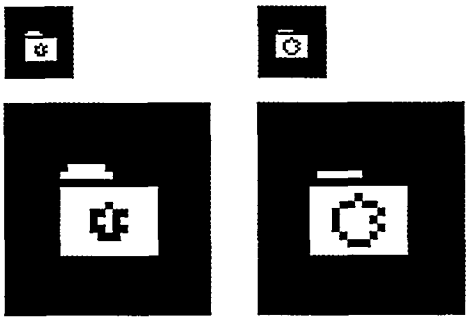


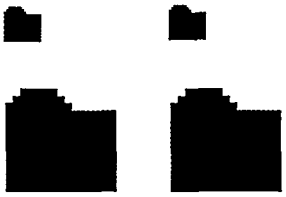
"What will the visual appearance of the icon family be?"

We leveraged the quirky illustrative style of the product identity, which we carried throughout the interface to give users a feeling of continuity, comfort and fun.

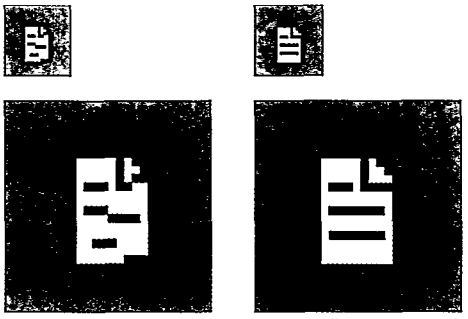
The folder icon (above) signifies the Examples part. It appears as part of the product identity in the opening title, as a central element on the Contents screen, and in a 16 x 16 pixel size in the header bar at the top of each example screen. Note how much the icon had to change to be recognizable in the smallest size (right). Important design details included how legible the icon appeared in different color contexts as well as the formulation of the apple. Making the apple solid black and adding one pixel to the height of the folder tab made a much more legible icon. This icon always appears in or upon the standard blue color.

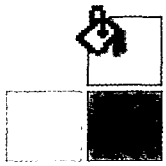






The document icon (right) was designed to represent the Reference part (not included in this release). The illustrative style of the large icon did not translate well in the small size for the header bar. Instead of looking quirky, it looked broken (below, in actual size and enlarged). A more straight-angled look worked better. This icon is always associated with its standard green color, whether as a fill-in color or background.



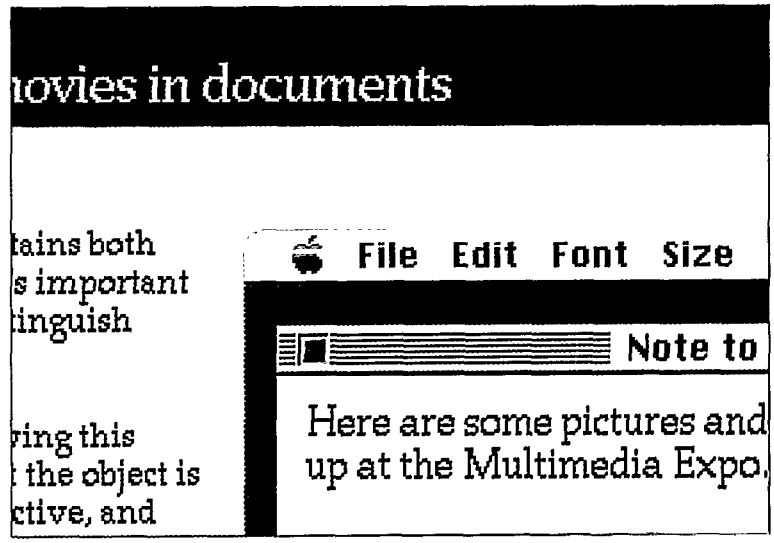


Color as communication

Color became an important decision point early in the design process. We were primarily concerned with color as a communicative element. But which colors? The selection was based on many factors, including visibility on black and white and other background colors, how well the colors worked as a set and as navigational cues and their meaning. Other criteria included choosing colors from the standard Apple color palette (256 colors), and picking colors that were not reminiscent of the limited color palette of early computer systems or of competitors' systems.

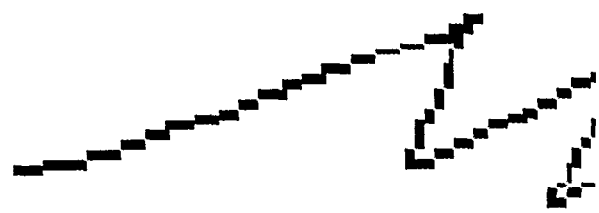
We used color to contribute to a distinct product identity, which contrasted with the more muted coloration of the Macintosh System 7 interface. This made it easy to distinguish *Making It Macintosh* from other windows on the desktop.

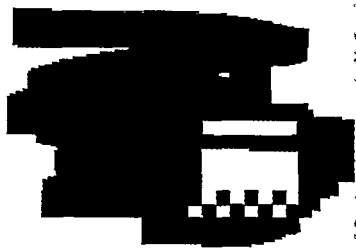
Our intention was to use color aesthetically and expressively; to give the product a spirit of liveliness and fun.



"How will color be used to support a learning environment?"
We created a cohesive learning environment by using color judiciously and consistently; in a way that made sense to users.

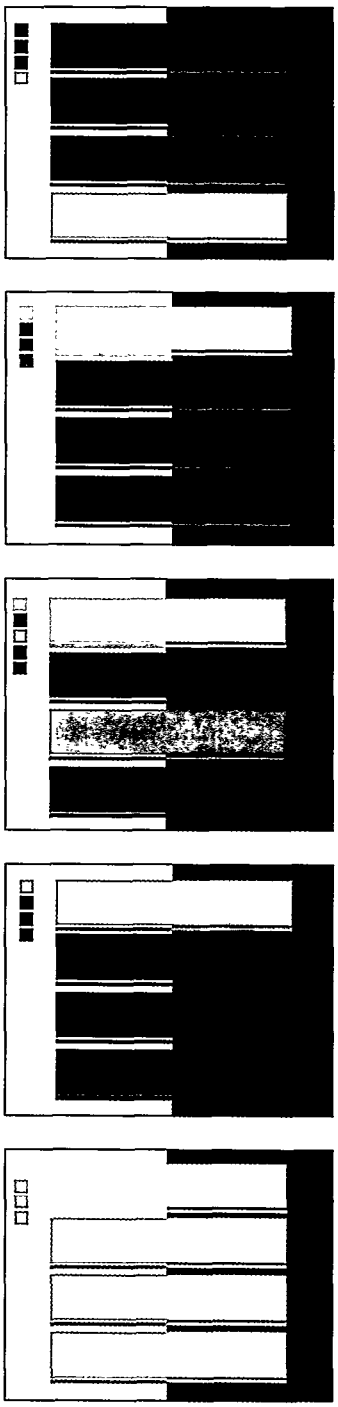
It was essential to create a minimum of visual noise on the example screens and yet provide enough cues to help users remember where they were. We decided to use color as a navigational aid to help users distinguish between the four parts, giving each its own color and icon. This called for colors that were distinctly different from one another and yet worked together as a family. The colors also needed to be of equal value (to have the same perceptual weight to users) and to be easily visible without being obtrusive. The header bands at the top of each screen were color coded as were the icons which appear throughout. Aside from this spot color, gray was the primary color in the Contents and example screens, where it helped set apart the largely black and white interface depictions from the background.





"What should the color palette be?"
Choosing colors was a process of experimentation and analysis. Colors were tested in graphic situations like those we expected in the product.

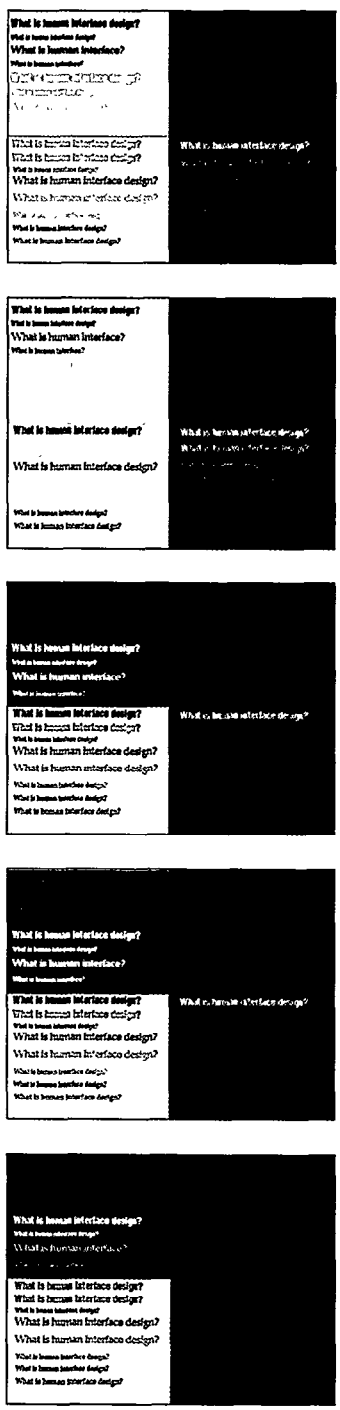
We conducted a rigorous study in order to test the appearance of the color groups in large and small quantities on black and white backgrounds. The final set of standard colors (top) functioned well in a variety of situations and met all the criteria. We also studied primary colors drawn from the Apple logo (second from top). These colors were very distinct and provided a strong contrast to the coloration of the Macintosh interface. However, they were too obtrusive in a working environment, especially when used in large quantities and for long durations. Highly saturated colors of a darker shade and pastel colors (bottom two screens) created visibility problems on some backgrounds and the colors were not as easily distinguishable from one another.



"How can we create synergy with typefaces and colors?"
Studying the impact of type and color on each other was one way to create an overall visual environment that was greater than the sum of the individual elements on the screen.

In selecting typefaces we were concerned with legibility and appearance in various color combinations. This time we made tests of typefaces in a variety of sizes and weights on various color and black and white backgrounds. Often, making a color slightly darker or lighter increased legibility. For example, the lemon yellow type (second from top, right) read well on black but was difficult to read on white. By "cheating" and using a darker yellow on the white background, the type appeared optically to match the color on the black background, but was much more legible.

We chose Espy sans serif, a screen font developed by Apple, for use in the Contents, See Also window and Index. For text on the example screens we used Palatino, a Macintosh standard font, which has a very legible 12 point size.



Search for

Are you sure you want to **Find**

Bringing users into focus

HARRY SADDLER
talks about Apple Computer's point of view on user-centered design.

Apple's goal in developing *Making It Macintosh* was to increase developers' interface design competence; our strategy was to wield a richer palette of instructional techniques than is normally used for technical documentation. Interactive multimedia was a natural choice: dynamic, interactive media is used to illustrate dynamic behaviors and interactivity. This enfolding of medium and message is at the heart of the design of *Making It Macintosh*: design conveys design principles, interactivity teaches techniques of interactivity, and human interface communicates the principles of human interface.

This conflation of design and meta-design, interface and meta-interface provided a satisfyingly rich design problem, for which we were able to find successful solutions. The task of implementing those solutions over hundreds of illustrations, animations, and interactive elements was an exercise in long-term project rigor and quality control (design, like invention, is 2 percent inspiration and 98 percent perspiration). The clarity of instruction in *Making It Macintosh* is the result of a carefully designed information framework and the discipline to work within that framework over the whole period of content development.

What did we expect to achieve from this attention to design? Of course, we needed to communicate our guidelines forcefully and unambiguously, but we also wanted to send the message to software engineers that design matters.

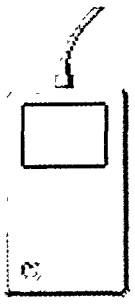
Does design matter to software engineers? Of course it does—they design software. For human interface designers to deny that they do is to misunderstand the nature of design.

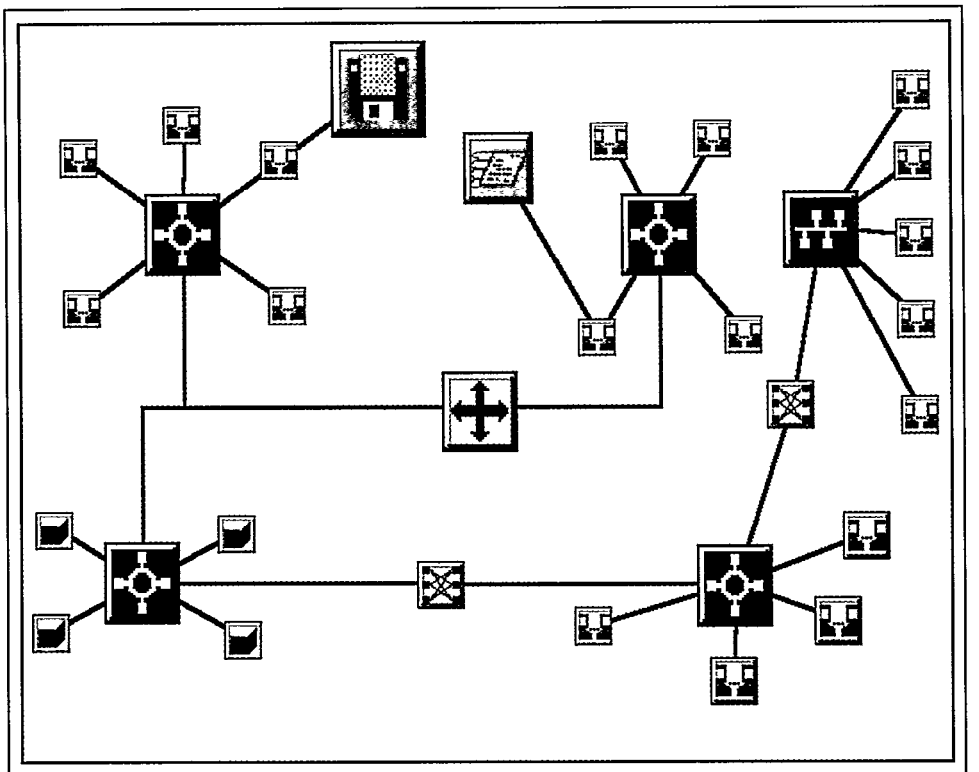
Design is about applying one's knowledge and skills—professional, personal, and cultural—to the creation of artifacts and systems that benefit people. Clearly, this is a domain shared by software engineers and human interface designers.

The question "Who is the designer?" often means "Who understands the whole problem, and who has all the skills needed to solve the it?" For software design, the answer is that no one person can be "the designer," because software design is too complex, requires too many skills, and crosses too many boundaries between disciplines. As a result, several different disciplines must be brought to bear. The difficulty in this is that each brings its own methodology and point of view, and the confluence of these can blur the vision needed to achieve good design.

Putting the user into the process helps put things back into focus. User-centered design methodologies may be innately interdisciplinary because no single repertoire of skills can adequately address the complexity of users and their tasks.

User-centered design is yet another meta-message of *Making It Macintosh*, the message being that we cared about its users—the software engineers—and attempted to discover and respond to their particular needs. We hope that they in turn are encouraged to learn about and design for their users, and not for themselves. Bringing users into the interdisciplinary process that is software design is not an option—after all, software is design in which the user participates long after the designers are done.





A typical application. **You** show a picture of **your** model with **your** icons -- your graphical, animated results are easier to understand.

After you've tried new MODSIM II, other object-oriented languages will seem primitive

Free trial and, if you act now, free training

MODSIM II is a modern high-level, object-oriented programming and simulation language with these important benefits:

- **Simulation.** All the features you need to do simulation and automatically collect statistics are built-in.
- **Object-oriented.** The powerful mechanisms that help you develop reusable, easily maintainable code.
- **Your program is portable.** Your programs are easier to read, and are portable.
- **Symbolic debugger.** Error detection simplified through steptracing, multiple breakpoints, and data viewing.
- **Graphics and animation.** You easily lay out your graphical user interface. You can draw icons and place them on a palette with no programming.

Free trial offer

The free trial contains everything you need to try MODSIM II on your computer. You receive everything you need for a complete evaluation.

For a limited time we also include free training.

Computers with MODSIM II

MODSIM II® runs on all popular workstations as well as PC's with Windows or OS/2. Your investment is preserved when you change computers, or operating systems.

MODSIM II is already in use at hundreds of installations throughout the world.

For immediate information

Call Hal Duncan at (619) 457-9681, Fax (619) 457-1184. In Europe, call Peter Holt, In The Netherlands, on 31 43 670780, Fax 31 43 670200. In the UK, call Nigel McNamara on 0276 671 671, Fax 0276 670 677.

Rush information on MODSIM II.

☐ **Yes, I want to learn the reasons for the growing popularity of MODSIM II.**

Special offer — return the coupon today and we will include a free course enrollment worth $950.

Name _____

Organization _____

Address _____

City _____ State _____ Zip _____

Telephone _____ Fax _____

Computer _____ O/S _____

☐ Send details on your University offer.

Return to:
CACI Products Company **MACM TRANS**
3333 North Torrey Pines Court
La Jolla, California 92037
Call (619) 457-9681 Fax (619) 457-1184

In Europe:
CACI Products Division
MECC Business Center
G. Martino Laan 85, 8th Floor
6229GS Maastricht, The Netherlands
Call 31 43 670780 Fax 31 43 670200

In the UK:
CACI Products Division
Coliseum Business Centre
Watchmoor Park, Riverside Way
Camberley, Surrey GUI5 3YL, UK
Call 0276 671 671 Fax 0276 670 677

CACI Products Company