EXHIBIT 63 PART 1

Apple
PRESS

# Newton 2.0 User Interface Guidelines

**Addison-Wesley Publishing Company**

Reading, Massachusetts Menlo Park, California New York
Don Mills, Ontario Wokingham, England Amsterdam Bonn
Sydney Singapore Tokyo Madrid San Juan
Paris Seoul Milan Mexico City Taipei

ISBN 0-201-48838-8
1 2 3 4 5 6 7 8 9-MA-0099989796
First Printing, May 1996

**Library of Congress Cataloging-in-Publication Data**

Newton 2.0 user interface guidelines / Apple Computer, Inc.
p. cm.
Includes index.
ISBN 0-201-48838-8
1. User interfaces (Computer systems) I. Apple Computer, Inc.
QA76.9.U83N49 1996
005.265—dc20 96-20168
CIP

# Contents

# Figures

P R E F A C E

# About This Book

*Newton 2.0 User Interface Guidelines* describes how to create software products that optimize the interaction between people and devices that use Newton 2.0 software. The book explains the whys and hows of the Newton 2.0 interface in general terms and in specific details.

*Newton 2.0 User Interface Guidelines* helps you link the philosophy behind the Newton 2.0 interface to the actual implementation of the interface elements. Examples from a range of Newton software show good human interface design. These examples are augmented by descriptions and discussions of the reasoning behind the guidelines.

This book also contains examples of how *not* to design human interface; they are marked as such and appear with a discussion that points out what's inappropriate and how to correct it.

## Who Should Read This Book

This book is for people who design and develop software for Newton devices. If you are a designer, a human interface professional, or an engineer, this book contains information you need to design and create software that fits the Newton model. It also provides background information to help you plan your software product's design.

Even if you don't design and develop software for Newton, reading this book will help you understand the Newton interface. This understanding is useful to managers and planners who are thinking about developing Newton software, as well as to people who are studying human interface design in general.

This book assumes you are familiar with the concepts and terminology used with Newton devices, and that you have used a Newton device and its standard applications.

## What's in This Book

This book begins with a chapter that describes Newton devices such as the Apple MessagePad, what people do with them, and how they differ from personal computers. The first chapter also presents important principles you should keep in mind when designing Newton software, and explains how to involve users in designing the interface. The rest of the chapters define various parts of the Newton 2.0 interface. They describe each interface element in general language and show examples of how to use the elements correctly. For the more technical reader, the book specifies dimensions, spacing, and other specific implementation details for the Apple MessagePad. The book concludes with a list of common interface mistakes and a glossary.

## Related Books

This book does not explain how to create Newton software with Newton Toolkit, the Newton development environment. For that you'll need to refer to these other books, all of which come with Newton Toolkit:

- *Newton Programmer's Guide.* This set of books is the definitive guide and reference for Newton programming. This book explains how to write Newton programs and describes the system software routines that you can use to do so.

- *Newton Toolkit User's Guide.* This book introduces the Newton Toolkit (NTK) development environment and shows how to develop Newton applications using Newton Toolkit. You should read this book first if you are a new Newton application developer.
- *Newton Book Maker User's Guide.* This book describes how to use Newton Book Maker and Newton Toolkit to make Newton digital books and to add online help to Newton applications. You have this book only if you purchased the Newton Toolkit package that includes Book Maker.
- *The NewtonScript Programming Language.* This book describes the NewtonScript programming language.

## Visual Cues Used in This Book

Throughout this book you'll see visual cues to certain types of information.

- **Boldfaced text** indicates that a new term is being defined and that a definition of the word appears in the glossary.
- This symbol indicates an example of the correct way to use a Newton interface element.
- This symbol indicates an example of the wrong way to use a Newton interface element. It specifically calls out common mistakes.

## Developer Products and Support

APDA is Apple's worldwide source for hundreds of development tools, technical resources, training products, and information for anyone interested in developing applications for Apple computer platforms. Customers receive the *Apple Developer Catalog,* which features all current versions of Apple development tools, as well as popular third-party development tools. APDA offers convenient payment and shipping options, including site licensing.

To order product or to request a complimentary copy of the *Apple Developer Catalog*, use the following information:

APDA
Apple Computer, Inc.
P.O. Box 319
Buffalo, NY 14207-0319

| | |
|---|---|
| Telephone | 1-800-282-2732 (United States)<br>1-800-637-0029 (Canada)<br>716-871-6555 (International) |
| Fax | 716-871-6511 |
| AppleLink | APDA |
| America Online | APDAorder |
| CompuServe | 76666,2405 |
| Internet | APDA@applelink.apple.com |

If you provide commercial products and services, call 408-974-4897 for information on the developer support programs available from Apple.

C H A P T E R 1

# Newton and Its Users

Before you can begin to design an application, it is crucial that you have a clear picture of what a Newton device can do and how people will use your Newton software. This chapter introduces some high-level concepts that will help you clarify that picture. In addition, this chapter presents some basic principles of user interface design that apply to all types of software. The chapter concludes by detailing how to conduct user tests of your product during its development.

## Understand Newton

Newton is a software and hardware technology designed for a family of products in the category of personal digital assistants (PDAs), such as the Apple MessagePad. The goal of Newton technology is to help people and businesses become more productive by simplifying basic tasks and making it easier for people to manage bits and pieces of information while on the move. Information entered on a Newton device can be moved to a desktop machine or a mainframe computer, where it can be manipulated in powerful applications.

Newton is not a small portable computer with another graphical user interface. There may be similarities between portable computers and Newton devices, but the differences summarized below are more important than the similarities when it comes to designing a user interface for an application.

| **Newton** | **Portable Computers** |
|---|---|
| Focused function | General purpose |
| New architecture optimized for mobility and communications—use it anywhere, any time | Derived from desktop computer architecture, which is optimized for stationary operation |
| Tapping, writing, and drawing with a pen | Typing, pointing, and clicking with mouse and keyboard |
| Intelligent assistant | Scripting and macros |
| New and custom applications | Existing desktop applications |
| It's a communications assistant | It's a personal computer |
| Simple | Complex |

To take advantage of its distinguishing features and capabilities, Newton has distinctive user interface elements.

## Know Your Audience

Identifying and understanding your target audience are among the most important first steps when you start designing your product. To create a product that people can and will use, study the people who make up your target audience.

It's useful to create scenarios that describe a typical day in the life of a person you think uses the type of product you're designing. Think about the different work spaces, tools, and constraints and limitations that people deal with. You can also visit actual work places and study how people do their jobs.

Analyze the steps necessary to complete each task you anticipate people wanting to accomplish. Then design your product to facilitate those tasks, using a step-by-step approach by thinking of how a person might get from one place to the next in a logical fashion.

Involve users throughout the design process and observe them working in their environment. Use people who fit your audience description to test your prototypes and development products. Listen to their feedback and try to address their needs in your product. Develop your product with people and their capabilities, not computers and their capabilities, in mind. For more information, see "Involve Users in the Design Process" on page 1-13.

## What People Do With Newton

The features and capabilities that make Newton what it is also strongly influence what people want to do with Newton devices. These expectations indirectly affect the user interface of Newton software. An application must make it easy for people to accomplish the following tasks on demand:

- Capture information fragments—write, sketch, pick from lists, specify dates and times, and select options
- Organize information—file, sort, schedule, prioritize, copy, delete, and format
- Retrieve information—find, recall, browse, skim, read, and view
- Send and in some cases receive information by various means—print, fax, mail, and direct transfer

## Accessibility

Your software needs to appeal to and be useful to people with a wide range of abilities and backgrounds. There are likely to be members of your target audience who are different from the so-called average user that you envision. Users will undoubtedly vary in their ages, styles, and abilities. They may also have physical or cognitive limitations, linguistic differences, or other differences you need to consider. Identify how the individuals in your target audience differ and what special needs they may have.

Make your application accessible to people around the world by including support for worldwide capabilities in your designs from the beginning of your development process. Take stock of the cultural and linguistic needs and expectations of your target audiences.

## Observe Basic Human Interface Principles

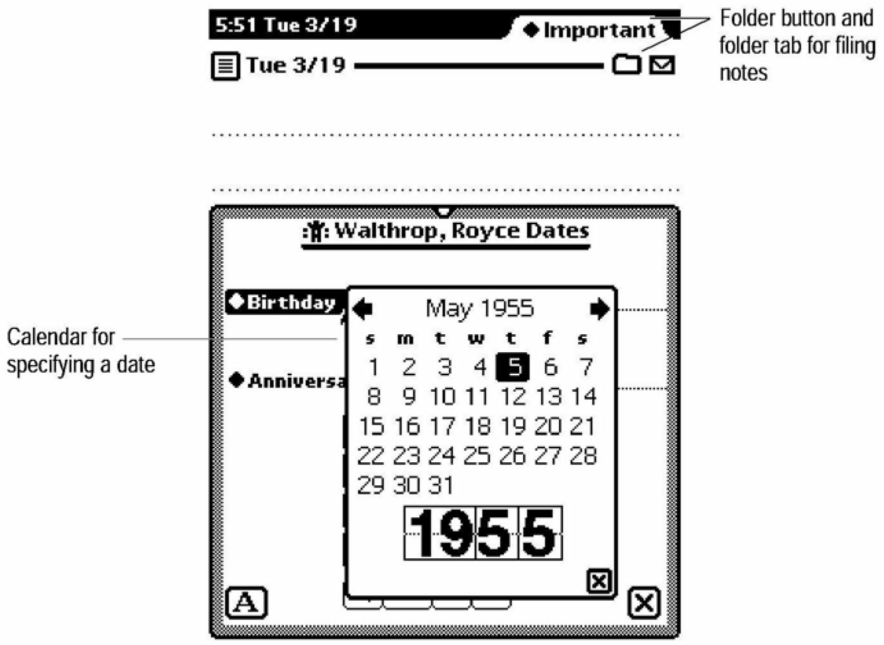
Effective software adheres to certain basic principles no matter whether it runs on a Newton PDA, a personal computer, or a high-powered computer workstation. These principles are based on the capabilities and processes not of the machine but of the human operator—how people usually think, act, and work.

### Metaphors

Wherever possible, model the actions and objects in your program on something from the real world. This trick especially helps inexperienced users quickly grasp how your program works. Folders are a classic metaphor. People file things in folders in the real world, so they immediately understand the concept of filing data items in folders on a Newton. Other common metaphors include scrubbing to delete data, tapping buttons to make things happen, sending and receiving things through an in box and out box, setting dates and times on calendars and digital clocks, and homing in on information with alphabetic index tabs. Figure 1-1 illustrates some Newton metaphors.

Metaphors suggest a use for objects and actions in the Newton interface, but that use doesn't define or limit the implementation of the metaphor. For example, a paper folder has a limited storage capacity, but a folder on a

**Figure 1-1** Metaphors help people quickly grasp how software works



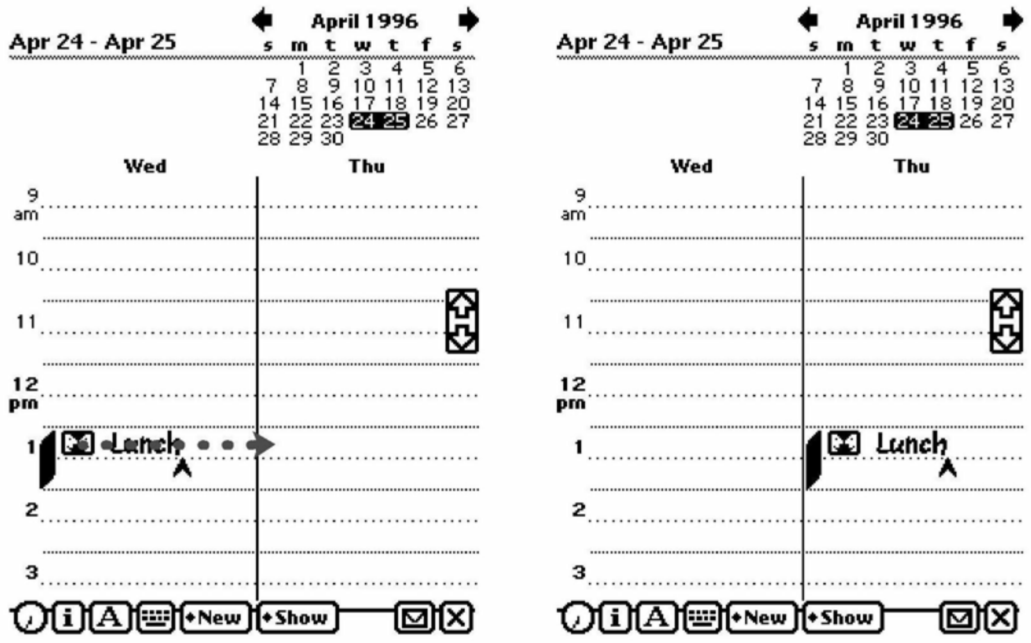
Newton doesn't have to be constrained by the same limitation. Newton folders can hold a limitless number of items (up to the storage capacity of the hardware), and this is an advantage that the Newton can offer. Try to strike a balance between the metaphor's suggested use and the ability of the Newton to support and extend the metaphor.

Naturally you can't find a metaphor for everything. Be sure to use the established metaphors, but if you can't come up with a solid metaphor for another object or action, then do without. Don't distort the real world into a caricature in a slavish attempt to find a metaphor.

## Direct Manipulation

Your product should let users feel that they are directly controlling something tangible, not abstract. Make sure objects on the screen remain visible while a user performs actions on them, and make the result of the user's actions immediately visible. For example, a user can reschedule a meeting in the built-in Date Book application by dragging the meeting's icon from one time to another. Figure 1-2 illustrates direct manipulation.

**Figure 1-2** Users should feel they are directly controlling something tangible



1. User drags a meeting icon to a new time

2. Icon appears at new meeting time

## Feedback

In addition to seeing the results of their actions, users need immediate feedback when they operate controls and ongoing status reports during lengthy operations. Have your application respond to every user action with some visible change. For example, make sure every button highlights when a user taps it. Audible feedback also helps, but can't be the primary or sole feedback because people may use Newtons in places where they can't hear or where they must turn off the sound.

The system automatically provides feedback when it's temporarily busy by displaying the busy cursor. During operations that last more than a few seconds, your application should display explanatory messages and show elapsing progress.

## See and Point

A Newton application is better than a person at remembering lists of options, commands, data, and so on. Take advantage of this situation by presenting lists and letting users choose from them. People can concentrate on accomplishing tasks with your program instead of remembering how to operate it. As a bonus, your program controls its inputs and doesn't have to check as many error conditions.

## Consistency

It's likely that people will use other Newton software besides yours—at the least they will use some of the built-in applications and services. You can turn this likelihood to your advantage by designing your application's interface to be consistent with other Newton applications. Both you and your application's users benefit if they can build on prior experience when learning how to use your application.

You can make your application consistent visually and behaviorally by incorporating standard Newton interface elements in it. Visual consistency helps people learn and then easily recognize the graphic language of the interface. For example, users learn to recognize a black diamond as the source of a pop-up list of choices. Behavioral consistency of the interface means people only have to learn once how to do things such as erasing and scrolling. Then they can explore new functions and applications using the skills they already have.

## User Control

Allow users, not your application, to initiate and control actions. Keep actions simple and straightforward so users can easily understand and remember them. Provide ample opportunity to cancel operations before they begin, and wherever possible allow users to gracefully stop an operation that's underway. Be careful about unleashing agents, experts, or wizards that will do things behind a user's back.

## Forgiveness

People make mistakes, so your program should make it easy for them to correct their mistakes. Let them use the Undo button to reverse their last action. People need to feel they can experiment without damaging the system or their data. Create safety nets for people so they feel comfortable learning and using your product.

Always advise people when they begin an operation that has potentially dire consequences. Display a warning and have the user confirm the operation before proceeding. This doesn't mean you should have users confirm every action. Frequent warning messages suggest something is wrong with the program design—obviate some of the warnings by making more actions reversible.

## Stability

Personal digital assistants introduce a new level of complexity for many people. To cope with this complexity, people need some stable reference points. The Newton interface is designed to provide an environment that is understandable, familiar, and predictable. It defines a number of regular interface elements to foster a perception of stability, including view borders, view titles, folder tabs, standard buttons, and standard button locations. Each of these elements has a specific look and a regular, predictable behavior. In addition, the interface defines a clear, finite set of basic data objects—text, ink text, shapes, and sketches—and a clear, finite set of editing commands with which users can create and manipulate the objects. Your application can share and enhance the stability by using the regular interface elements and handling data objects in the customary manner.

## Aesthetic Integrity

People primarily see software as a functional product, not a fashion product. This means you want them to notice what your product does, not how it looks. Don't succumb to the temptation to load up with the latest interface fads; they'll quickly become dated. Since people will spend a lot of time with your product, design it to be pleasant to look at for a long time. A spare, clean interface will stand up to repeated viewing much better than a highly decorative interface. For example, the built-in Setup application has lots of non-functional decorative elements to make a user's first Newton experience a friendly one, but the built-in applications that people use daily have none of that decoration.

Make sure you follow the graphic language of the interface. Don't invent new interface elements to replace existing ones, and don't change the function of standard interface elements. If you change the look of standard interface elements, people will actually try to make up functional reasons for the differences. If you square off the corners of your buttons, use unique view borders, or use a different symbol to designate a pop-up, people will waste time trying to figure out what your custom elements do that the standard ones don't. It won't occur to people that you merely have your own notion of how the interface elements should look.

## Design for the Newton System

In addition to the general user interface principles presented in the previous section, you should keep in mind the guidelines in this section as you design software specifically for the Newton system.

### Observe the Built-In Applications

Your software will coexist with built-in Newton applications and services. On an Apple MessagePad they include the Notepad, Names File, Date Book, In/Out Box, Filing, Routing, Find, Assist, and others. If your application has functions analogous to those in the built-in applications and services, use the same mechanisms. Users will be accustomed to them and will expect other software to work in the same way. You can most easily match the built-in applications and services by using the system proto templates in the Newton Toolkit.

You can extend the Newton interface if you need to, but make sure your extensions retain the original look and feel.

### Use the Common Pool of Data

All built-in Newton applications and services can access a common pool of data, and so can your Newton software. This pool is the information a user enters into the Newton device. Since all applications have access to this data, a user can work more efficiently—because each piece of information needs to be entered only once. Thereafter, the data can be accessed and used in many different ways. Your application can read and write this data. Put it to the user's advantage.

## Keep Applications Simple

Newton isn't designed for complex tasks or applications that require viewing a large area or multiple windows of data at a time. Applications that require the user to keep track of several pieces of information at once probably won't work well because the user must either move around a lot within the application, or deal with many simultaneous or layered views. Studies show that users become confused in those situations.

Remember that people will use Newton while on the move, in places where there's no place to sit or to set it down. In such settings, it's easiest to use applications with simple, straightforward screens and an obvious path through the information. Make sure that your application's controls are clearly identified, that there aren't too many "places" for the user to navigate through, that you don't display too many container views at once, and that the user can easily see what to do. Minimize writing; tapping to pick from a list of alternatives is easier.

## Use Screen Space Wisely

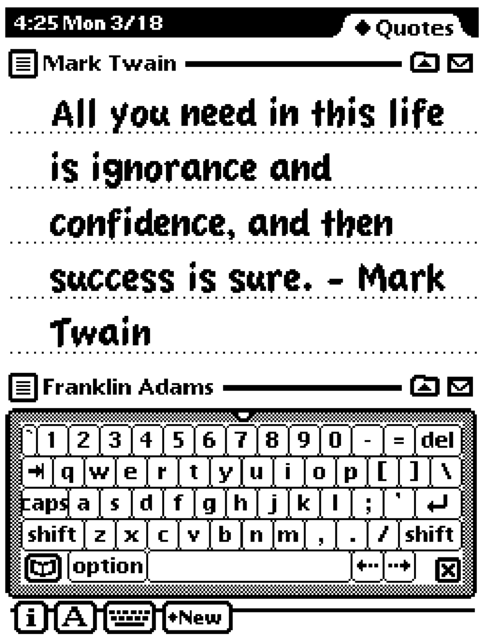
Because the user's hand is usually held close to a Newton device, it's best to keep tappable controls at the bottom of the screen, have the user enter data in the middle of the screen, and display titles and other descriptors at the top of the screen. This way, the most important information (the user's information) isn't obscured each time he or she taps a button. If you need to display controls on the side, make sure your application allows users to move the controls to either side of the screen, according to whether they are right- or left-handed.

## Check the Screen Size

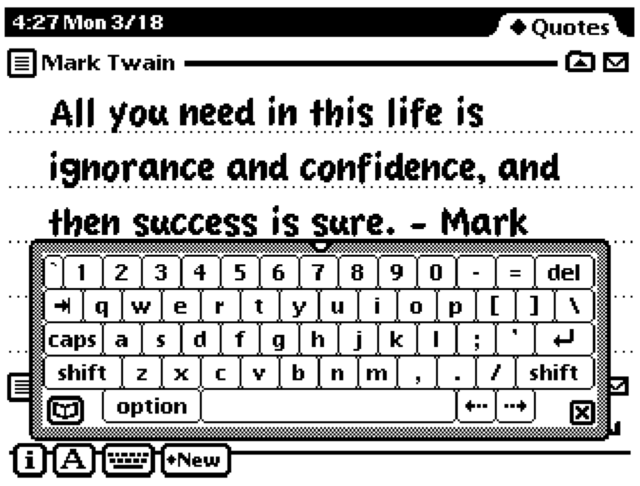
A Newton application's **main view**—the visual object that serves as the application's base of user operations—can be any size. If your application's main view does not fill the entire screen, keep in mind that whatever is visible behind your application will be operable. In this situation, users can get confused about what's frontmost—and therefore about what will be scrolled when the scroll arrows are tapped and which view is currently in use.

Also keep in mind when designing your application that future Newton devices may have larger or smaller screens than current Newton devices. To work with different screen sizes, a Newton application must check the screen size and make adjustments as needed to the size and location of the things it displays so that everything fits. If you want your application to work in either of the two display orientations available on an Apple MessagePad 120, your application needs to be able to adjust the position and configuration of everything it displays for regular or sideways orientation of the display. Figure 1-3 shows how the built-in Notepad application and the on-screen keyboard adjust their size, position, and layout when a user rotates the display.

**Figure 1-3** An application adjusts its size, position, and layout to fit the screen


4:25 Mon 3/18
Quotes
Mark Twain
All you need in this life is ignorance and confidence, and then success is sure. - Mark Twain
Franklin Adams
option
New

Regular orientation on a MessagePad 120


4:27 Mon 3/18
Quotes
Mark Twain
All you need in this life is ignorance and confidence, and then success is sure. - Mark
option
New

Sideways orientation on a MessagePad 120

## Involve Users in the Design Process

The best way to make sure your product meets the needs of your target audience is to show it to the kinds of people you hope will buy it. Do they understand what it's for and what to do with it? Can they use it? Can they keep track of where they are? Does it help them? You can do this during every phase of the design process to help reveal what works about your product as well as what needs improvement.

When you give people an opportunity to use your product or a mock-up of it, they will inevitably find some undiscovered flaws. You can implement significant changes to your product during its evolution and thereby save yourself lots of time and money and save your users from frustration. By identifying and focusing on users' needs and experiences, you can create products that are easier to assemble, learn, and use. These improvements can translate into competitive advantages, increased sales, and enhanced customer satisfaction.

### Define Your Audience

There are several steps to involving users in your design process. The first step, done at the beginning of a project, is to define the users and then do an analysis of the target audience. You want to determine what these people are like, how they might use a product like yours, if they have any similar products, and what features they would like to see in your product. By doing some research on your target audience, you can find out if what you're including in a product is desirable and useful.

### Analyze Tasks

The second step is to analyze the tasks people will be doing with your product. You need to do a task analysis for each task you anticipate that your users will do. Look at how they perform similar tasks without a Newton.

Then look at how the Newton can facilitate the tasks. To help plan a task analysis, imagine a scenario in which someone uses your product. List each task a person might perform in that scenario, then break each task apart into its component steps. This allows you to identify each step that a person goes through in order to complete the task. Order the steps according to how people do them. When you feel you have all the steps listed and ordered, read the list back to someone and see if that person can use the steps you've listed to accomplish the task.

## Build Prototypes

For the third step, apply the information you've collected about your users, their skills, and the tasks you envision them performing to create a prototype of your design. Prototyping is the process by which you develop preliminary versions of your design to verify its workability. You can use a variety of techniques to construct prototypes of your design. Creating storyboards is one technique—you draw out the steps your users will go through to accomplish a task. Another technique is to build a simulation of the product in prototyping software that animates some features or demonstrates how the product will work.

## Observe Users

Once you have a prototype drawn or mocked up, you can begin to show it to people to get reactions to it. The fourth step, called user observation, lets you test the workability of your product design by watching and listening carefully to users as they work with your prototype. Although it is possible to collect far more elaborate data, observing users is a quick way to obtain an objective view of your product. Before you do any testing, take time to figure out what you're testing and what you're not. By limiting the scope of the test, you're more likely to get information that will help you solve a specific problem. You can use the information you gather about your target audience to help you pick participants for your user observation; find people who have the same demographic background and experience level as the typical user in your target audience. Your participants will work through one or more specific tasks. These tasks can be based on the task analyses that you performed earlier in the design process. After you determine which tasks to use, write them out as short, simple instructions. Your instructions to the participants should be clear and complete but should not explain how to do things you're trying to test. See the following section, "Ten Steps for Conducting a User Observation," for more information; it includes a series of sample steps on which you can base your own user observation.

During the user observation, record what you learn about your design; you'll be using this information to revise your prototype. Once you've revised your prototype, conduct a second user observation to test the workability of the changes you've made to your design. Continue this iterative process of creating prototypes and conducting user observations until you feel confident that you've fully addressed the needs of your target audience.

## Ten Steps for Conducting a User Observation

The following steps provide guidelines that you can use when conducting a simple user observation. Remember, this test is not designed as an experiment, so you will not get quantitative data that can be statistically analyzed. You can, however, see where people have difficulty using your product, and you can then use that information to improve your product.

Most of these steps include some explanatory text with sample statements that you can read to the participant. Feel free to modify the statements to suit your product and the situation.

1. **Introduce yourself and describe the purpose of the observation (in very general terms). Most of the time, you shouldn't mention what you'll be observing.**

   Set the participant at ease by stressing that you're trying to find problems in the product. For example, you could say something like this:

   - "You're helping us by trying out this product in its early stages."
   - "We're looking for places where the product may be difficult to use."
   - "If you have trouble with some of the tasks, it's the product's fault, not yours. Don't feel bad; that's exactly what we're looking for."

- "If we can locate the trouble spots, then we can go back and improve the product."
- "Remember, we're testing the product, not you."

**2. Tell the participant that it's OK to quit at any time.**

Never leave this step out. Make sure you inform participants that they can quit at any time if they find themselves becoming uncomfortable. Participants shouldn't feel like they're locked into completing tasks. Say something like this:

- "Although I don't know of any reason for this to happen, if you should become uncomfortable or find this test objectionable in any way, you are free to quit at any time."

**3. Talk about the equipment in the room.**

Explain the purpose of each piece of equipment (hardware, software, video camera, tape recorder, microphones, and so forth) and how it will be used in the test.

**4. Explain how to think aloud.**

Ask participants to think aloud during the observation, saying what comes to mind as they work. By listening to participants think and plan, you'll be able to examine their expectations for your product as well as their intentions and their problem-solving strategies. You'll find that listening to users as they work provides you with an enormous amount of useful information that you can get in no other way.

Some people feel awkward or self-conscious about thinking aloud. Explain why you want participants to think aloud and demonstrate how to do it. For example, you could say something like this:

- "We have found that we get a great deal of information from these informal tests if we ask people to think aloud as they work through the exercises."
- "It may be a bit awkward at first, but it's really very easy once you get used to it. All you have to do is speak your thoughts as you work. If you forget to think aloud, I'll remind you to keep talking. Would you like me to demonstrate?"

5. **Explain that you will not provide help.**

   It is very important that you allow participants to work with your product without any interference or extra help. This is the best way to see how people really interact with the product. For example, if you see a participant begin to have difficulty and you immediately provide an answer, you will lose the most valuable information you can gain from user observation—where users have trouble and how they figure out what to do.

   Of course, there may be situations in which you will have to step in and provide assistance, but you should decide what those situations will be before you begin testing. For example, you may decide that you will allow someone to struggle for at least three minutes before you provide assistance. Or you may decide that there is a distinct set of problems on which you will provide help. However, if a participant becomes very frustrated, it's better to intervene than have the participant give up completely.

   As a rule of thumb, try not to give your test participants any more information than the true users of your product will have. Here are some things you can say to the participant:

   - "As you're working through the exercises, I won't be able to provide help or answer questions. This is because we want to create the most realistic situation possible."
   - "Even though I won't be able to answer your questions, please ask them anyway. It's very important that I capture all your questions and comments. When you've finished all the exercises, I'll answer any questions you still have."

6. **Describe in general terms what the participant will be doing.**

   Explain what all the materials are (such as the set of tasks, disks, and a questionnaire) and the sequence in which the participant will use them. Give the participant written instructions for the tasks.

   If you need to demonstrate your product before the user observation begins, be sure you don't demonstrate something you're trying to test. For example, if you want to know whether users can figure out how to use certain controls, don't show them how to use the controls before the session. Don't demonstrate what you want to find out.

7. **Ask if there are any questions before you start; then begin the observation.**
8. **During the observation, remember several pointers:**
   - □ Stay alert. It's very easy to let your mind wander when you're in the seventh hour of observing users. A great deal of the information you can obtain is subtle.
   - □ Ask questions or prompt the participant. Make sure you have a tester protocol that spells out how frequently you prompt and what you say. Your interruptions shouldn't be frequent, but when a participant is hesitating or saying, "Hmmm," ask what the participant is thinking about.
   - □ Be patient; it is very easy to become impatient when someone is taking a long time. The participant is doing you a favor and is probably somewhat nervous. Anything you can do to alleviate the participant's insecurities and put the participant at ease will provide you with much richer data.
9. **Conclude the observation.**

   Do the following when the test is over:
   - □ Explain what you were trying to find out during the test.
   - □ Answer any remaining questions the participant may have.
   - □ Discuss any interesting behaviors you would like the participant to explain.
   - □ Ask the participant for suggestions on how to improve the product.
10. **Use the results.**

    As you observe, you may see users doing things you never expected them to do. When you see participants making mistakes, your first instinct may be to blame the mistakes on the participant's inexperience or lack of intelligence. This is the wrong focus to take. The purpose of observing users is to see what parts of your product might be difficult to use or ineffective. Therefore, if you see a participant struggling or making mistakes, you should attribute the difficulties to faulty product design, not to the participant.

Be sure to schedule time between your sessions to make notes and review the session. Jot down any significant points. If you used videotape or audio cassette tape, mark in your notes the specific parts of the tape that you may want to review.

To get the most out of your test results, review all your data carefully and thoroughly (your notes, the videotape or cassette tape, the tasks, and so on). Look for places where participants had trouble and see if you can determine how your product could be changed to alleviate the problems. Look for patterns in the participants' behavior that might tell you whether the product was understood correctly.

It's a good idea to keep a record of what you found out during the test. You don't need elaborate video equipment; a hand-held video camera will work. In fact, you don't even have to use video equipment. You can use a tape recorder to record what is spoken during the session. The important point is that you create some kind of objective, factual record of the session that you refer to later. That way, you'll have documentation to support your design decisions and you'll be able to see trends in users' behavior. You might want to write a report that documents the process you used and the results you found. After you've examined the results and summarized the important findings, fix the problems you found and test the product again. By testing your product more than once, you'll see how your changes affect users' performance.

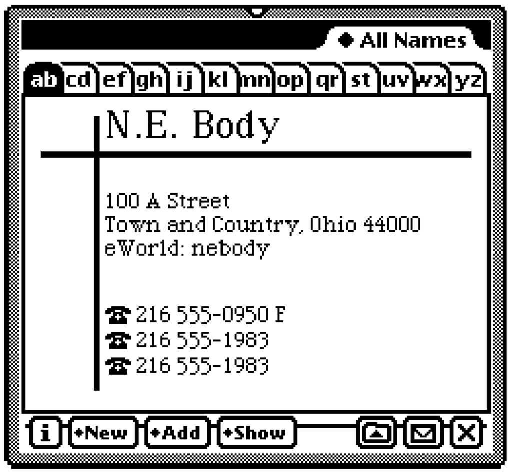
C H A P T E R 2

# Container Views

pictThis chapter describes **container views,** in which an application shows the user text and graphic information, and in which the user interacts with the information and the application. The chapter presents specifications and recommendations about the appearance and behavior of these container views, including how to display them on the screen, how users interact with them, and how they interact with each other. There are several kinds of standard container views whose regular looks enhance the visual stability of Newton applications. The standard views provide predictable ways to see and interact with all the different kinds of information people can create and store on Newton devices. Figure 2-1 shows examples of container views.

There are conventions for opening, closing, moving, scrolling, and getting an overview of container views. This means that no matter which application people use, they know how to control container views on the screen and how to adjust container views in the available screen space.

**Figure 2-1** Examples of container views

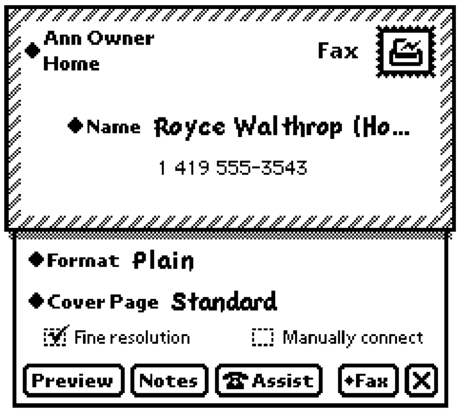


N.E. Body

100 A Street
Town and Country, Ohio 44000
eWorld: nebody

☎ 216 555-0950 F
☎ 216 555-1983
☎ 216 555-1983

Main view



◆ Ann Owner
Home

◆Name Royce Walthrop (Ho...

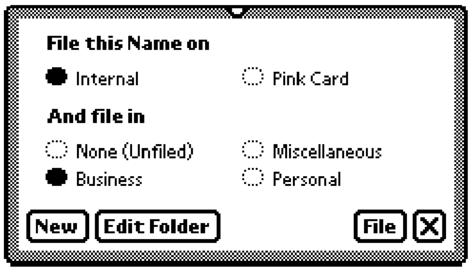
1 419 555-3543

◆Format Plain

◆Cover Page Standard

☑ Fine resolution ☐ Manually connect

Routing slip



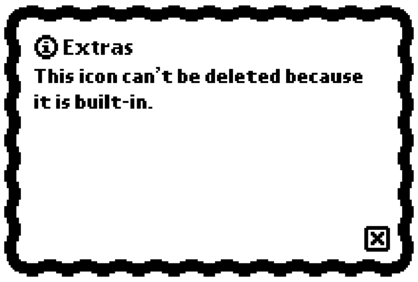
File this Name on

● Internal ○ Pink Card

And file in

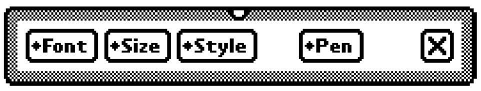
○ None (Unfiled) ○ Miscellaneous
● Business ○ Personal

Ordinary slip

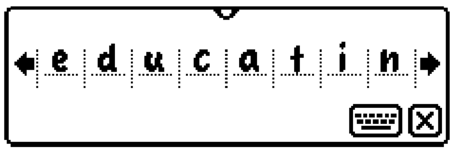


Extras
This icon can't be deleted because it is built-in.

Alert box



Palette



Corrector view

When people manipulate container views on the screen, they see immediate visual feedback. As a user drags a movable container view, the view keeps up with the user's pen, reinforcing the user's sense of direct manipulation. When people open and close container views, they see a representation of such actions. These mechanisms emphasize that the user is in control and can directly manipulate "real" interface objects such as container views.

## How Views Look

Nothing makes an application look more like it belongs—or less like it belongs—on a Newton device than the appearance of its container views. This section describes the key visual attributes of container views:

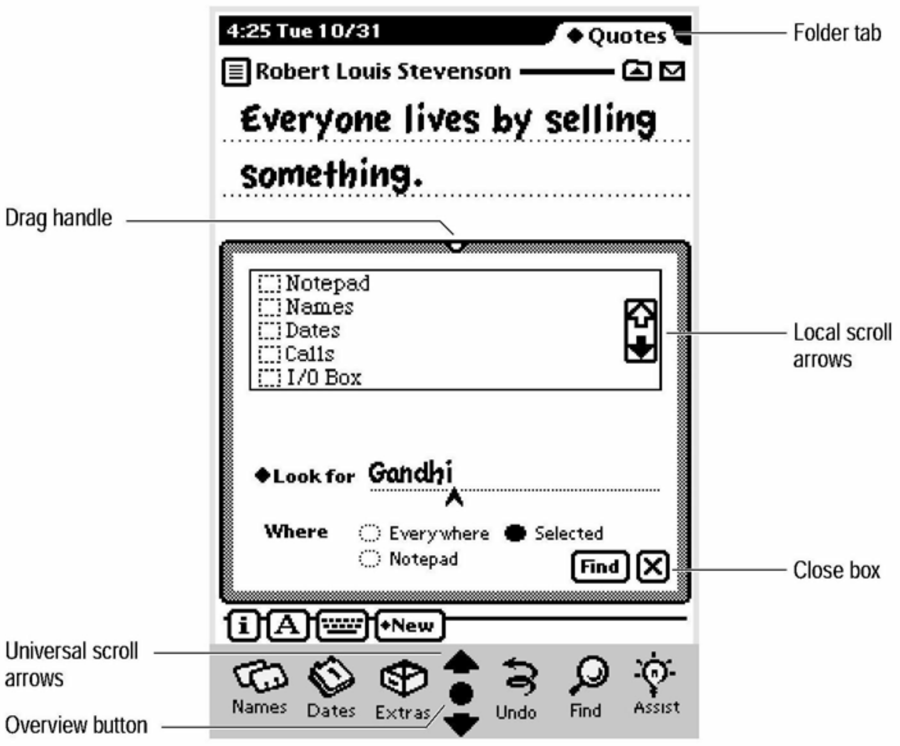
- controls
- title style
- border style
- fill pattern

### View Controls

There are several standard controls for manipulating container views. These controls include the drag handle, folder tab, Close box, local scroll arrows, universal scroll arrows, and Overview button. The first four controls are part of the container view they affect. The latter two controls are not part of the container view they affect. Figure 2-2 points out the standard view controls.

For details on container view controls, see "Moving a View" on page 2-33, "Folder Tab" on page 8-19, "Close Boxes" on page 3-14,"Scrolling" on page 2-36, and "Overview" on page 2-44.

**Figure 2-2** Standard controls for manipulating views

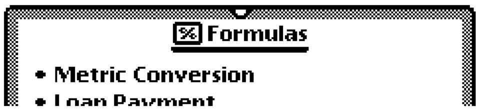


## View Title

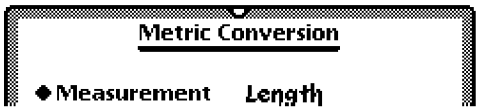
A container view should have a title at the top unless the view's identity is obvious from its contents. Ordinarily a title consists of text in the bold style of the system font, an optional small icon, and a triple underline, all centered at the top of the view. If the title of a subordinate view is long or instructional, you can left-justify it and omit the icon and the triple underline. Figure 2-3 compares different types of titles.

**Figure 2-3** Various title styles

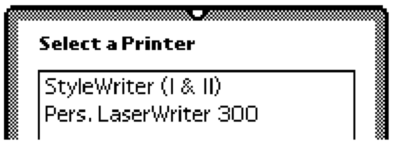
Ordinary title with icon



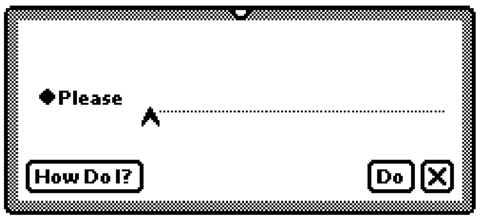
Ordinary title without icon



Subordinate view title—left-justifed



No title—view's contents make its purpose clear



The title only identifies the container view's contents. The title is not a control that the user can tap to change a setting, alter a state, or initiate an action. Controls that do these things are described in Chapter 3, "Controls." For example, if you want users to be able to change a view's title, have them tap a button or choose from a picker in the status bar.

A view title should not end with a colon. The title's position, size, and font make a colon unnecessary and distracting.

You capitalize view titles according to conventional rules for book titles. That is, you capitalize the first word of a title, and you capitalize all other words except articles (*a, an, the*), coordinating conjunctions (for example, *and, or*), and prepositions of three or fewer letters.

On an Apple MessagePad use 10-point text for the title. The optional icon must be no more than 11 pixels tall.

## View Border

Every container view is framed by a border. (A border is not visible if its view fills the screen.) Primarily, a view's border serves to demarcate what's in the view and what's not. Secondarily, certain borders identify special types of container views.

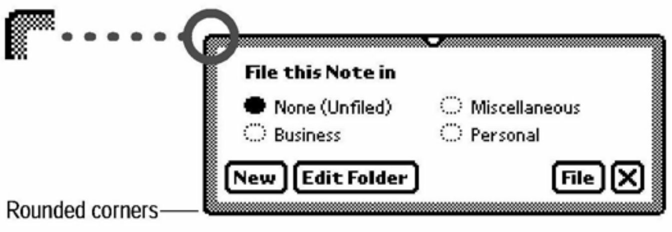
In general, Newton views are rectangular and have rounded corners. Use square-cornered borders only when you have a specific need for a particular look.

A view's border is not visible if the view completely covers the screen. For example, a MessagePad 120 user does not see the border of a view that measures 240 × 320 pixels. The view has a border, but it is off-screen.

### Matte Border

The most common type of view border, called a **matte border,** consists of a thick gray band edged on the outside by a thin black line. Users expect views with matte borders to be movable (see "Moving a View" on page 2-33). Figure 2-4 shows the matte border.

**Figure 2-4** A matte border indicates a movable view

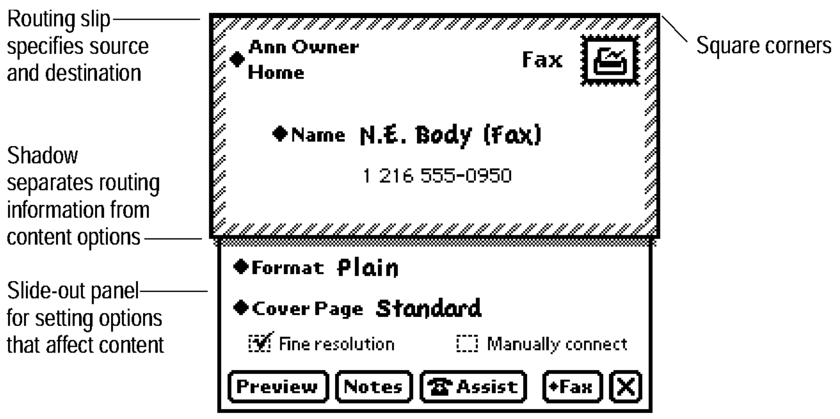


On an Apple MessagePad a standard matte border is five pixels thick with a corner roundness of five pixels and an inset of one pixel.

## Striped Border

A border made of pairs of short, slanted lines edged by a thin black rectangle is used around views known as **routing slips** (see "Routing Slips" on page 7-12). It's no accident that this border looks something like the border traditionally printed on airmail envelopes, because routing slips are analogous to postal envelopes. Figure 2-5 shows a routing slip border.

**Figure 2-5** A striped border suggests routing

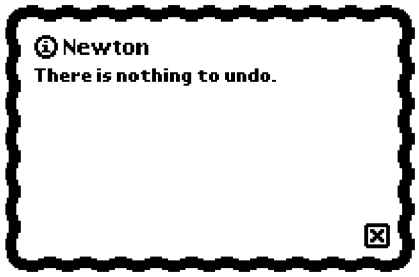

Routing slip specifies source and destination
Shadow separates routing information from content options
Slide-out panel for setting options that affect content
Square corners
Ann Owner
Home
Fax
Name N.E. Body (fax)
1 216 555-0950
Format Plain
Cover Page Standard
Fine resolution
Manually connect
Preview
Notes
Assist
Fax

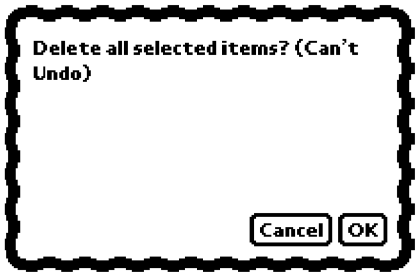
The paired short lines in a striped border slant 45 degrees to the right.

## Wavy Border

A view with a heavy black wavy border is called an **alert box.** It contains an important message that a user must acknowledge. There are two types of alert boxes; they are described in "Notification Alerts" on page 2-17 and "Confirmation Alerts" on page 2-18. Figure 2-6 shows the wavy border of an alert box.

**Figure 2-6** An alert box has a thick wavy border

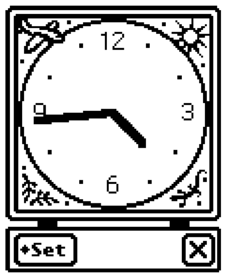


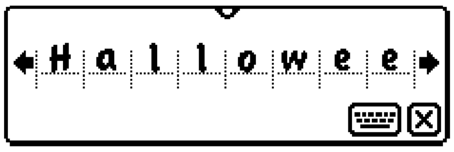


## Plain Border

For simplicity, some container views require a plain black border made of medium-weight lines. Figure 2-7 shows examples of views with plain borders.

**Figure 2-7** Some views need the simplicity of a plain border

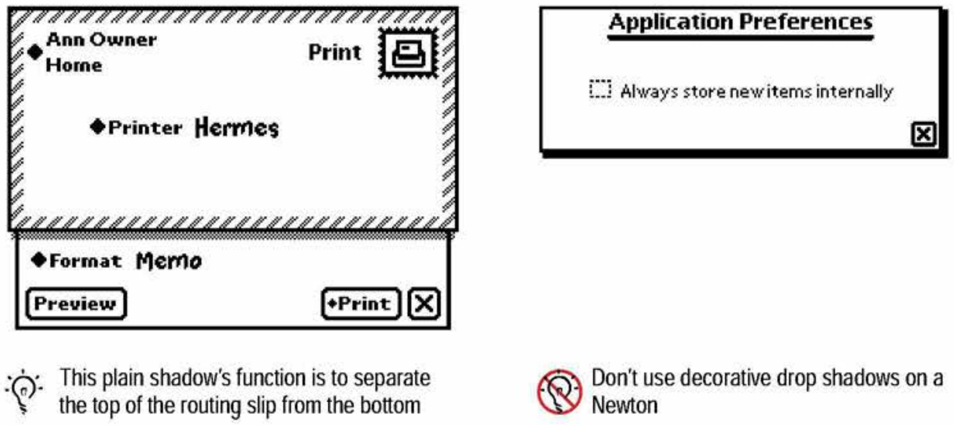




## Drop Shadows

It's possible to add a drop shadow to a view's bottom and right borders, but this ersatz 3D look is not appropriate for Newton applications. Don't use drop shadows just because you like the way they look or because you want to make a Newton application look like a personal computer application. Although you shouldn't use drop shadows, you can use another type of shadow that tells users something about a view. For example, a shadow reinforces the notion that there are two parts to a routing slip—an outer part above the shadow and an inner part below it. Figure 2-8 shows acceptable and unacceptable uses of shadows in the Newton interface.

**Figure 2-8** Sparing use of some types of shadows is OK


Ann Owner
Home
Print
◆Printer Hermes
◆Format Memo
Preview
◆Print
Application Preferences
Always store new items internally
This plain shadow's function is to separate the top of the routing slip from the bottom
Don't use decorative drop shadows on a Newton

## View Fill

Standard container views by default are filled with white, not with black or a pattern. If you want users to see through a container view to the views beneath it, you can make it transparent.

## Main Views

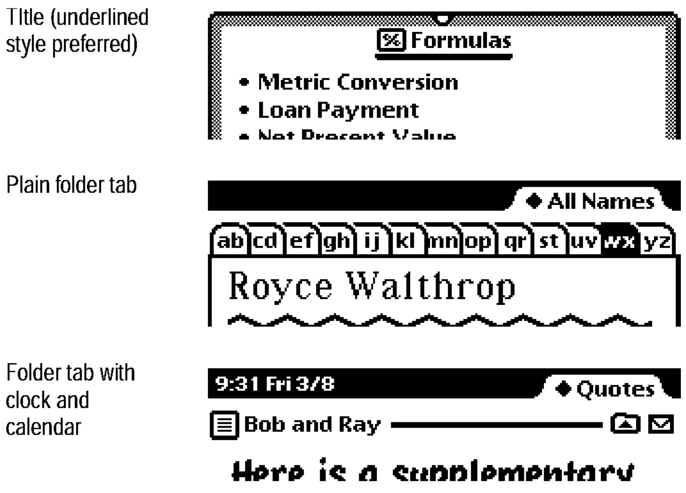
Nearly every application has a main view that serves as a base of operations. An application's main view may also be called the **application base view.** But strictly speaking, the main view is a user's concept and the application base view is a programmer's concept. An application base view is the view that contains all the other views that make up the application. A main view is a center of user operations.

Applications are not limited to one main view. The built-in Names File and Date Book applications, for example, have several main views each.

## Title or Folder Tab

An application's main view should have an ordinary, underlined title at the top unless the view's identity is obvious from its contents. An application's main view cannot have an ordinary title at the top if the application allows users to file information in folders. In this case a **folder tab** must go at the top of the main view. A folder tab shows the name of the folder whose data is currently displayed in the view, and a user can choose a different folder by tapping the folder tab. A folder tab can include a view title or a digital clock and calendar, but does not have to include either of them. (For more information on folders and folder tabs, see Chapter 8, "Newton Services.") Figure 2-9 compares a main view with a title to another main view with a folder tab.

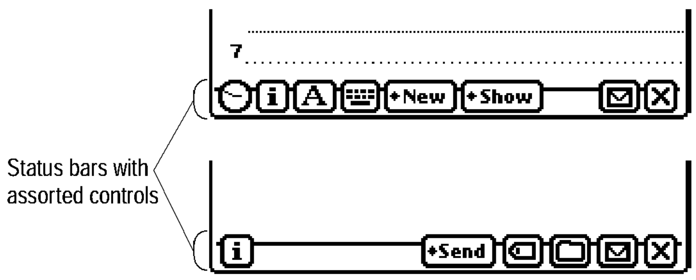
**Figure 2-9** A title or a folder tab tops a main view



## Primary Controls and Status Bar

An application's primary controls go at the bottom of its main view, usually on a **status bar.** A status bar is not strictly required, but it helps to visually anchor the controls. Figure 2-10 shows sample status bars with assorted controls.

**Figure 2-10** A status bar anchors primary controls at the bottom of a main view



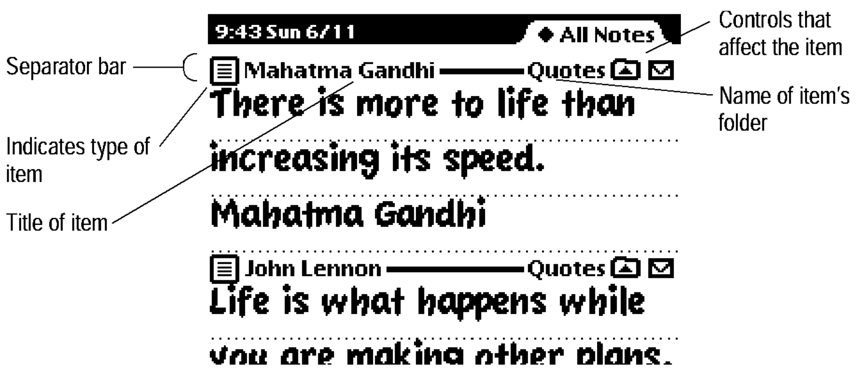
Each application can have a different set of controls, but an application's main view must have a Close box unless the application is the backdrop (see "The Backdrop" on page 2-29 and "Closing a Main View" on page 2-32). Close boxes and other standard status-bar controls are described in "Close Boxes" on page 3-14 and "Standard Newton Buttons" on page 3-22.

On an Apple MessagePad, the status bar is a black line two pixels thick, with end points two pixels from the right and left edges of the application's main view.

## Separator Bars

In a view that may display more than one variable-sized item at once, like the notes in the Notepad, a **separator bar** heads each item. A separator bar identifies the item below it and carries controls that apply only to that item. Figure 2-11 shows some separator bars in the Notepad.

**Figure 2-11** Separator bars separate multiple items in a scrolling view



A user creates a separator bar, also called a divider bar, by drawing a line across the view or by tapping a New button on the view's status bar. Tapping the New button always scrolls to the last item and adds a new blank item below it. Making the line gesture adds a new blank item below the line, before the following item.

A separator bar is a heavy black line with various buttons and text. At the left end of each separator bar is a picture button, called the **Item Info button,** which indicates the type of item below it. Next to that button is the item's title, displayed in the bold style of the system font. For more information on the Item Info button, see "Item Info Button" on page 3-29.

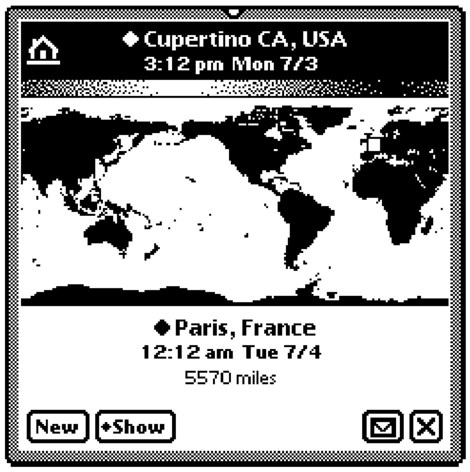
At the right end of each separator bar is an Action button for routing the item. If the application allows users to file items in folders, a Filing button appears on each separator bar next to the Action button, and the name of the item's folder appears next to the Filing button whenever the view is showing the items of all folders. For more information on those buttons, see "Action Button and Picker" on page 7-8 and "Filing Button and Slip" on page 8-14.

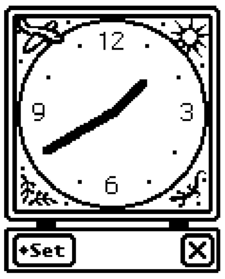
On an Apple MessagePad, the separator line is two pixels thick and the title is in 9- or 10-point text.

## The Main View's Border

Every application's main view must have a border, even if the border is not visible because the view fills the screen. Generally, an application's main view should have a rounded-corner matte border (as described under "View Border" on page 2-6). Alternatively, a main view can have a plain rounded-corner black border. A matte border is a better choice if the view is movable (or might be movable on a large screen), because users historically have associated matte borders with movable views and plain borders with stationary views. Figure 2-12 shows the two border styles.

**Figure 2-12** Main views have matte or plain borders with rounded corners





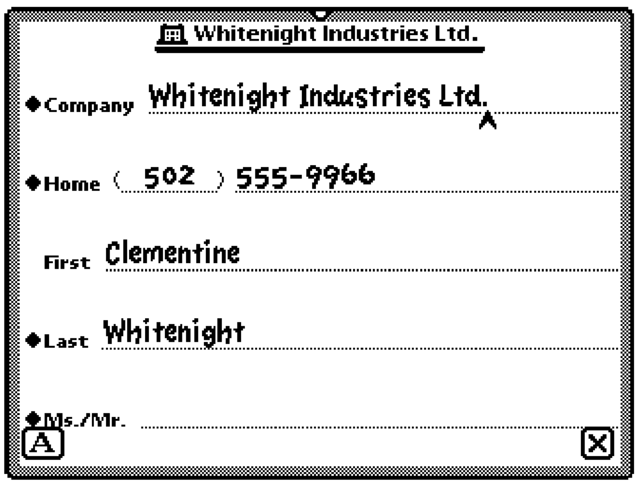
A movable main view is preferable to a fixed view. If users can't move your application's main view, they may have to close your application to work on another application beneath it. If you want a user to be able to leave your application open, make its main view small, matte-bordered, and movable.

It's possible for a stationary main view to have a different border style and still look like it belongs on a Newton device. You need a strong reason—something more than personal preference—to give your application's main view anything other than a rounded-corner black border or a rounded-corner matte border.

## Auxiliary Views

When an application needs to display and input more information than will fit in its main view, it displays an auxiliary view. There are several types of auxiliary views, as shown in Figure 2-13 and detailed in the following sections.

**Figure 2-13** Examples of auxiliary views



Whitenight Industries Ltd.

◆Company Whitenight Industries Ltd.

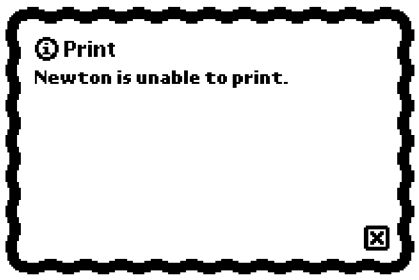
◆Home ( 502 ) 555-9966

First Clementine

◆Last Whitenight

◆Ms./Mr.

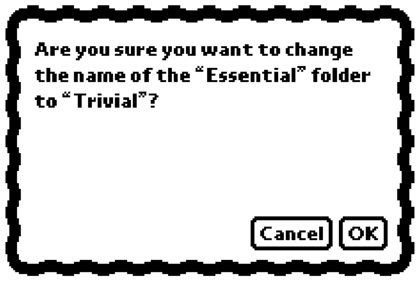
Ordinary slips give users the space they need to make detailed settings and to input or change data



Print

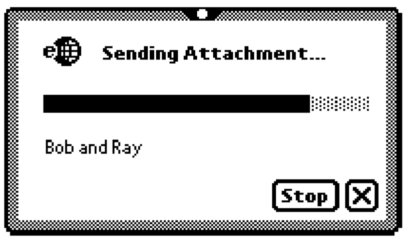
Newton is unable to print.

Notification alerts communicate important messages to users



Are you sure you want to change the name of the "Essential" folder to "Trivial"?

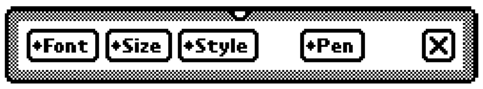
Confirmation alerts ask users to authorize a far-reaching or dangerous operation



Sending Attachment...

Bob and Ray

Status slips tell users what is happening during lengthy operations



Palettes give users handy access to useful settings and information

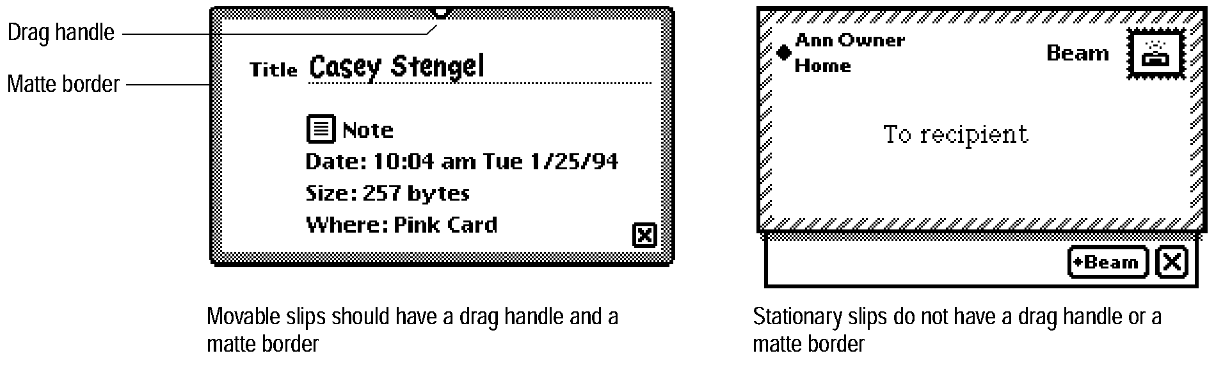
An auxiliary view appears in front of the view to which it is subordinate. For details on the customary position of a slip and the front-to-back ordering of views, "How Views Work" on page 2-28.

## Slips

The most common type of auxiliary view is called a **slip.** An application can use slips to get detailed user input. For example, the Date Book application displays essential information about meetings and events in its main view but has users input or change details in meeting and event slips. In addition, an application can use slips to display and allow users to change incidental and infrequently accessed information such as the title of an item or preference settings. Slips can also request responses and present alternatives that specify how an action should be completed. For example, a slip for routing e-mail should insist the user enter an e-mail address, without which the e-mail cannot be sent, and the slip offers numerous options that affect what the e-mail message includes.

Most slips are movable, but some are stationary. Movable slips provide more flexibility for someone using your application. If a user wants to see something under a movable slip while the slip is open, the user can drag the slip out of the way. To see something under a stationary slip, a user has no choice but to close the slip. Figure 2-14 compares slips that move with slips that can't.

**Figure 2-14** Users can move most slips

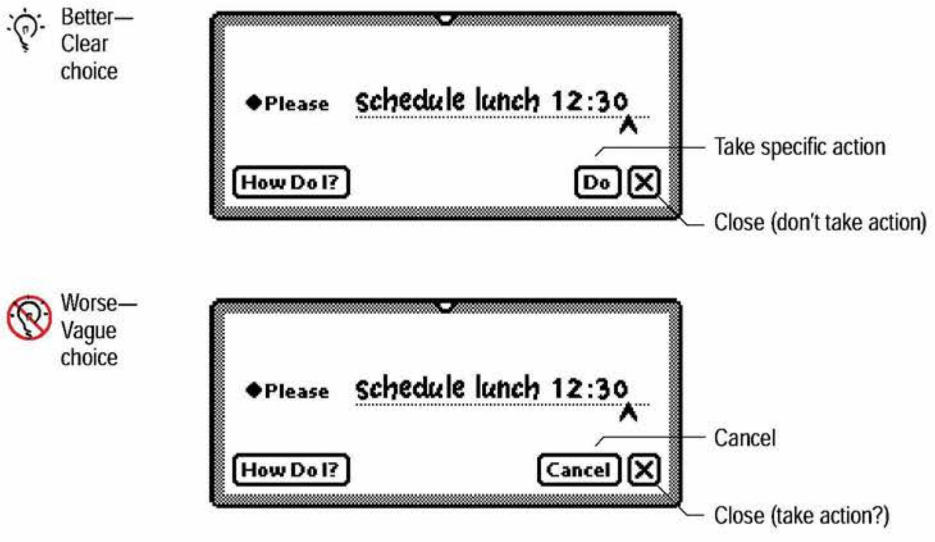

Drag handle
Matte border
Title Casey Stengel
Note
Date: 10:04 am Tue 1/25/94
Size: 257 bytes
Where: Pink Card
Ann Owner
Home
Beam
To recipient
Beam
Movable slips should have a drag handle and a matte border
Stationary slips do not have a drag handle or a matte border

Movable slips should have matte borders, and stationary slips should not. For instance, routing slips are stationary and have special striped borders. Border styles are described in "View Border" on page 2-6.

A slip contains text and controls and may contain icons, pictures, and input fields. Each slip contains some text to indicate the purpose of the slip and what caused the slip to appear. In some cases this text is a title for the slip.

Most slips have a Close box or large Close box in the bottom right corner, and some slips have additional primary controls at the bottom. For instance, a Close box alone is not enough in a slip whose purpose is to prepare for and initiate an action. In this case users must be presented with a choice for dismissing the slip: take action or cancel. A text button named with a verb such as Do or Find clearly means "Take this action with the settings I've made in this slip." A large Close box located next to one of those take-action buttons thus means "Ignore these settings and cancel the action." The alternative combination of buttons—a text button named Cancel to mean "cancel" next to a large Close box to mean "take action"—is ambiguous. Figure 2-15 compares these two alternatives.

**Figure 2-15** Dismissing slips that complete actions



In the absence of a take-action button, a Close box means simply, "I'm done with this task."

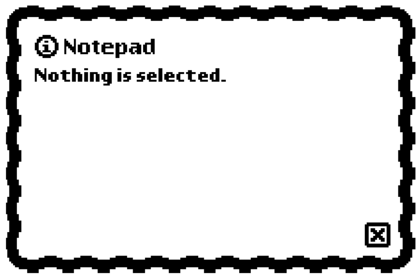
Close boxes and text buttons are covered in Chapter 3, "Controls." Input fields follow the guidelines given in Chapter 6, "Data Input."

## Notification Alerts

An application uses a particular type of auxiliary view, a **notification alert,** to provide messages about error conditions, warn users about potentially undesirable situations or actions, and announce alarms. Use a notification alert to convey information that is useful to the user but doesn't present any threat such as a loss of data.

The notification alert's wavy black border visually distinguishes it from operational slips. A user responds to a notification alert by tapping its Close box, thereby acknowledging the message and putting away the slip. Figure 2-16 shows a notification alert.

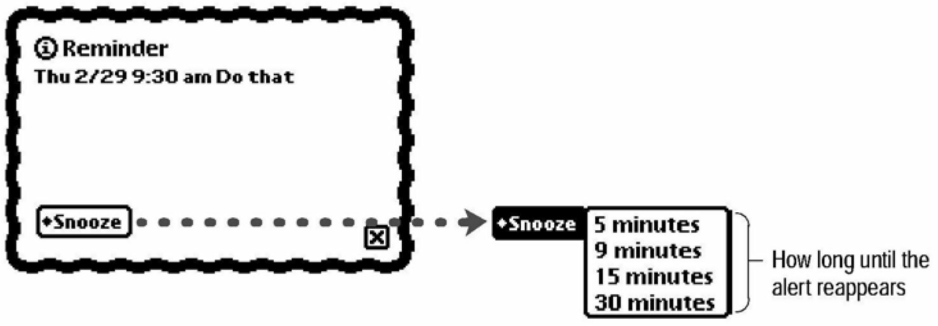
**Figure 2-16** A notification alert tells the user something important



Notepad
Nothing is selected.

Some notification alerts have a Snooze button, which enables a user to temporarily dismiss the alert. The alert reappears after an interval of time chosen by the user. For example, this is the type of notification alert the built-in Date Book application uses for its reminder alarms. Figure 2-17 shows a notification alert with a Snooze button (for more information, see "Alarms" on page 8-4.).

**Figure 2-17** A Snooze button enables a user to dismiss an alert temporarily



Before closing a notification alert, a user can tap the small circled *i* in the upper left corner to display the date and time at which the notification appeared. While any notification alert is open, the user can scroll through recent messages by using the universal scroll arrows (as described under "Universal Scroll Arrows" on page 2-38). The Newton operating system logs notification messages, handles notification message scrolling, and provides the small circled *i* and its functionality.

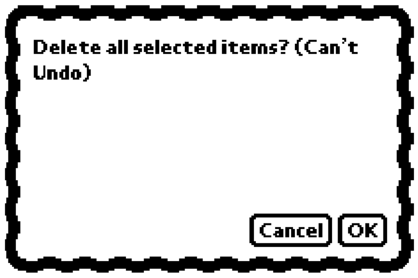
When you write notification messages, use phrasing that make sense to users. Use simple, nontechnical language; don't provide system-oriented information that a user can't relate to. When possible, give users information that helps explain how to correct the problem. Be as specific as possible; if appropriate, use the name of the application or the name of the view to which the message applies. For example, "The Expense Report slip doesn't scroll" is more helpful than "This view doesn't support scrolling."

## Confirmation Alerts

An application uses a **confirmation alert,** which has the same wavy border as a notification alert, to have the user confirm or cancel an action that may have far-reaching consequences. For example, confirmation alerts appear before the Newton puts into effect changes the user has made to folder names. Use a confirmation alert to warn the user in advance of a potentially dangerous situation. For example, a confirmation alert appears before Newton restores anything from the backup on a storage card.

A confirmation alert has no Close box. Instead, it has labeled buttons, usually one named OK and another named Cancel. The user taps OK to continue the far-reaching or potentially hazardous action or taps Cancel to cancel the action and do something else. Figure 2-18 shows a confirmation alert with OK and Cancel buttons.

**Figure 2-18** A confirmation alert tells the user about a grave situation


Delete all selected items? (Can't Undo)

Take care to phrase the confirmation message so that it makes sense with either the Cancel button or the OK button. For instance, the message "You've modified one or more items. Do you really want to cancel?" is not as clear as "Disregard all changes? (can't undo)"

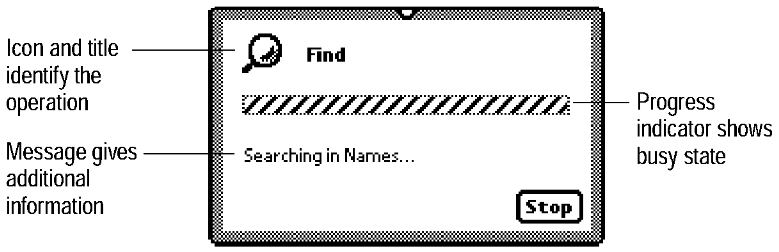
Instead of an OK button, you can use a button whose label describes the result of accepting the message in the confirmation alert. For example, in a confirmation alert that warns about the consequences of restoring from a card, you could have a button named Restore instead of OK. Likewise, you could replace the Cancel button with one that more precisely describes the action, such as Don't Restore.

Confirmation alerts are modal. While a confirmation alert is displayed, the system restricts users to interacting primarily with that confirmation alert. The system ignores all taps outside a confirmation alert. (A user can write outside a confirmation alert, however.)

## Status Slips

When an application begins an operation that takes more than a few seconds to complete, the application should display a message describing its busy status. The application can display the status message in a view that's already displayed, or it can display the message in a **status slip.** Figure 2-19 shows a typical status slip.

**Figure 2-19** A status slip reports on a lengthy operation


Icon and title identify the operation
Find
Progress indicator shows busy state
Message gives additional information
Searching in Names...
Stop

A status slip contains some or all of the following items:

- An icon visually identifies the application or the operation in progress.
- A title names the application or the operation in progress.
- A message provides additional information about the operation in progress (for example, "Searching Names...", "Connecting to desktop", and so on).
- A progress indicator shows that the system is busy, and may show elapsing progress.
- A Stop or Cancel button allows a user to halt the ongoing operation.
- A Close box allows a user to put away the status slip without halting the ongoing operation.

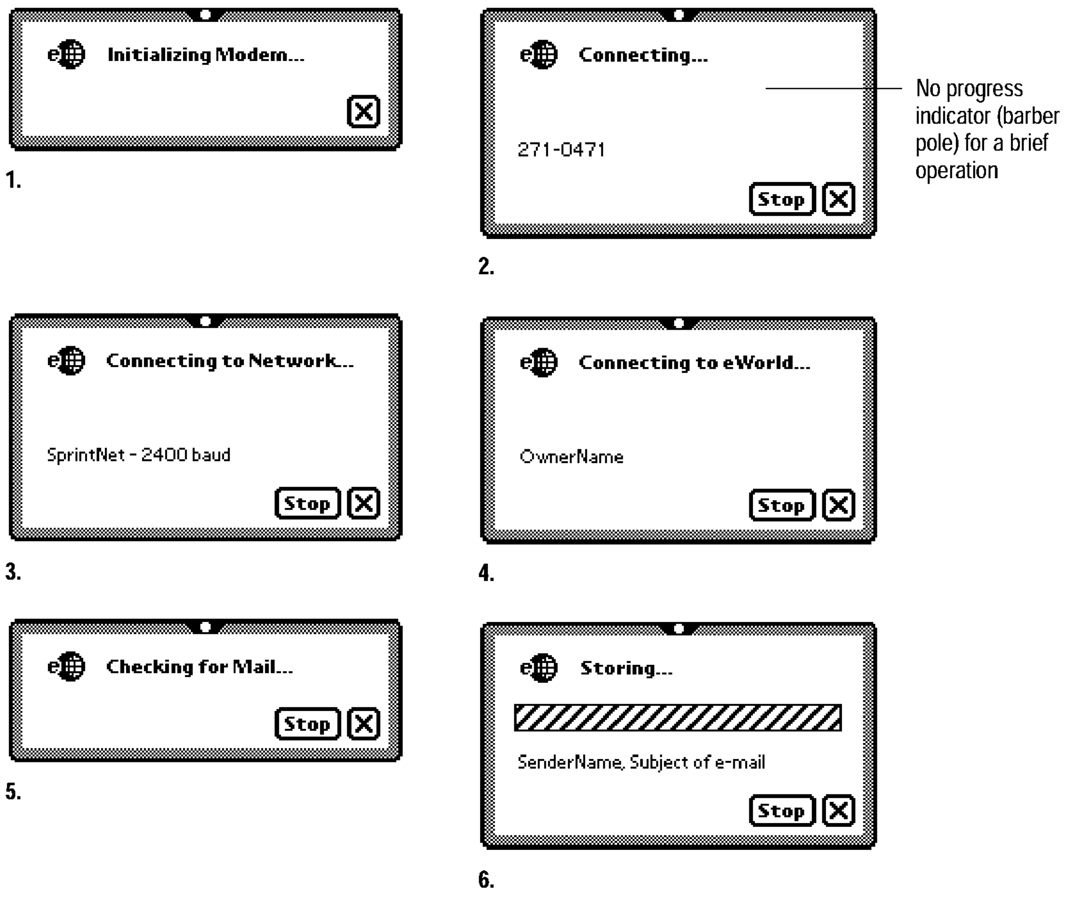
A status slip does not take the place of the Newton busy cursor, which appears automatically at the top center of the screen when the system temporarily cannot respond to user input (see "Automatic Busy Cursor" on page 8-2). Your application should display a status slip when it begins an operation that takes more than a few seconds to complete.

## Title and Message

Either the title or the message text in a status slip should begin with a gerund, such as *preparing* or *sorting*, and end with three periods (. . .), not a single period, a hyphen, a dash, or an ellipsis character (…). For example, when searching more than one application, the built-in Find service displays "Find" on the first line and a message similar to "Searching in Names..." (the message is continuously updated with the name of the application currently being searched). Other applications use the title and message text differently. When receiving e-mail, for example, the first line could display the current phase of the operation and the second line could display specific information being used in that phase. Figure 2-20 illustrates.

The title should be in the bold style of the system font, and the message should be in the plain style of the system font. The icon size should correspond to the title height. On an Apple MessagePad, use 9-point text for the title and 9-point text for the message.

**Figure 2-20** A sequence of status messages traces the steps of an operation



1. Initializing Modem...

2. Connecting...
271-0471
Stop

3. Connecting to Network...
SprintNet - 2400 baud
Stop

4. Connecting to eWorld...
OwnerName
Stop

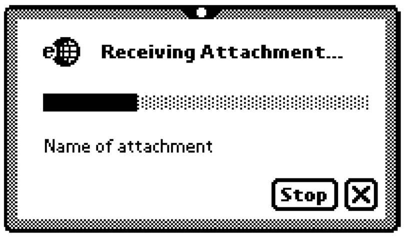
5. Checking for Mail...
Stop

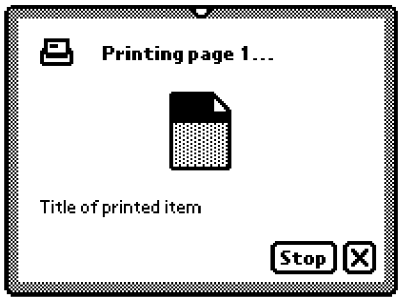
6. Storing...
SenderName, Subject of e-mail
Stop

## Progress Indicator

The progress indicator, if present in a status slip, can take different forms. It can be a simple "barber pole" gauge, which animates a set of diagonal stripes while the operation progresses but does not indicate how much of the operation has been completed. Alternatively, if it's possible to quantify the progress of the operation that's underway, then a status slip should include a progress gauge that indicates relative completeness of the current operation as a shaded portion of the entire gauge. Figure 2-21 shows two examples of progress gauges.

**Figure 2-21** A gauge in a status slip measures elapsing progress





## Close, Stop, or Cancel

A status slip usually has a large Close box and a Stop button or Cancel button. Tapping the Stop button or Cancel button halts the operation that's in progress. If halting the operation takes more than a few seconds, the application should change the status message to "Stopping..." or "Canceling..." while halting is in progress.

Make an effort to choose the button—Stop or Cancel—that most accurately describes the action that will occur. *Stop* suggests halting an operation that has already begun. In contrast, *Cancel* suggests a user has decided to take a different tack—without any action having taken place.

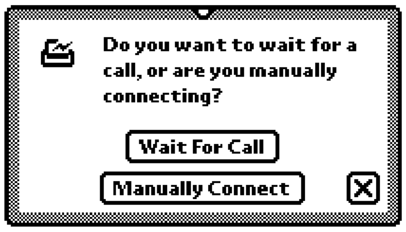
Tapping a status slip's Close box closes the status slip but does not stop the operation it was monitoring. To alert the user that an operation is in progress, the application registers the operation with the system's Notify service, causing the Notify button to blink at the top of the screen (see "Notify Button and Picker" on page 8-2). The user can open the status slip by tapping the Notify button and choosing the operation from the Notify picker that pops up. If an application completes an ongoing operation while the status slip is closed, the application must unregister the operation so the Notify service will remove it from the Notify picker.

Close boxes and text buttons are covered in Chapter 3, "Controls."

## User Decision

Besides reporting on the progress of an ongoing operation, a status slip can report a condition that requires a user to choose one of two alternatives. This type of status slip contains an icon, a message of up to three lines, and two text buttons. This type of status slip does not have a progress indicator, Stop button, Cancel button, or Close box. Figure 2-22 shows an example of a status slip that demands a user decision.

**Figure 2-22** A status slip can report a condition that demands a user decision


Do you want to wait for a call, or are you manually connecting?

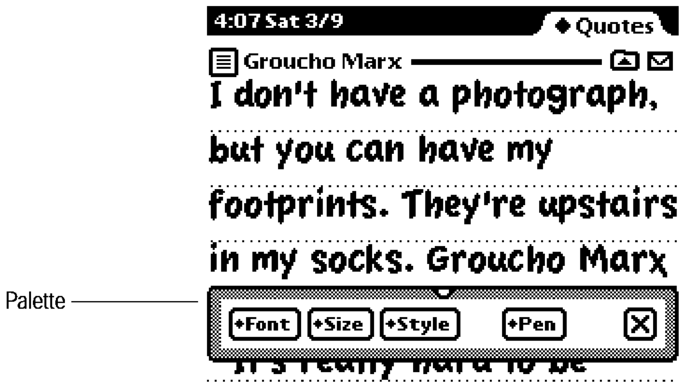
## Palettes

A **palette** is a small container view that gives a user instant access to useful settings or information. For example, a user can use the Styles palette (from the Extras Drawer) to set the font, size, and style of any selected text or to change the line weight of a drawing. Figure 2-23 shows the Styles palette.

A palette must be movable, so it has a rounded-corner matte border and a drag handle, like a slip. A palette can have a title, but typically a palette's contents make its function clear without a title.

A palette floats on top of main views and on top of slips that are already open. For details on the customary position of a palette, the front-to-back order of container views, and how container views move, see "How Views Work" on page 2-28.

**Figure 2-23** A palette provides handy access to useful settings


4:07 Sat 3/9
◆Quotes
Groucho Marx
I don't have a photograph, but you can have my footprints. They're upstairs in my socks. Groucho Marx
Palette
◆Font
◆Size
◆Style
◆Pen

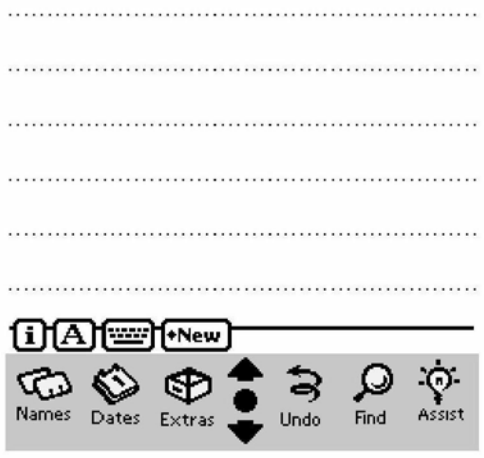
A palette has a Close box, or a large Close box if a text or picture button is adjacent, but the user may leave the palette open indefinitely. This has several ramifications, one being that a user must be able to move the palette to get at whatever it is covering. In addition, changes a user makes in a palette should take effect right away. Immediate feedback lets the user see that the input had the desired effect. If your application doesn't respond immediately to new settings of checkboxes, radio buttons, and other controls, it's less clear to the user when the input goes into effect. (For descriptions of close boxes, radio buttons, checkboxes, and other controls, see Chapter 3, "Controls.")

Palettes that remain open take up screen space, a valuable commodity on smaller screens. Therefore, use palettes sparingly. Don't use a palette where you can use a slip instead (such as in situations where the user can make the appropriate settings and then close the slip). Don't use a palette in place of controls in the status bar.

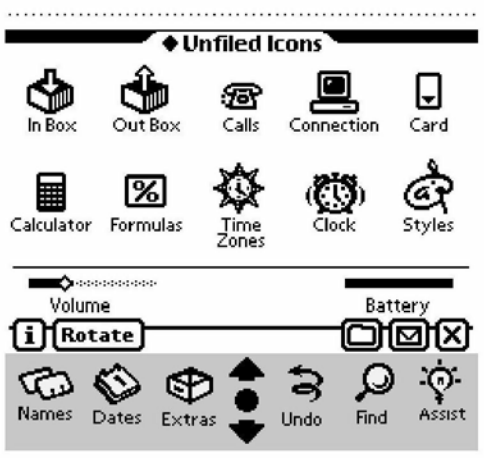
## Drawers

A **drawer** is a container view that slides open and closed at the bottom of the screen or at the bottom of another container view. Figure 2-24 shows the Extras Drawer.

**Figure 2-24** A drawer slides open and closed



The Extras Drawer closed



The Extras Drawer open

A drawer can be used for the main view of an application or for an auxiliary view. It can have a a title, a folder tab, or neither, depending on its function and contents. A drawer must have a Close box.

## Roll Views

In a **roll view** several discrete, fixed-size subviews are arranged one above another like pictures on a filmstrip. A roll view invariably contains more subviews than can be displayed in full detail at once. To see a subview that's not currently displayed, a user can scroll through the subviews. Alternatively, a user can see an overview consisting of one-line subview titles.

In most applications, users don't find roll views useful. Studies show that users tend not to use scrolling to access different subviews, finding it easier to pick from a list than to remember the relative position of subviews. Usually it makes more sense to implement the subviews of a roll view as individual slips, and list their titles in an overview. For example, the overviews of the built-in Preferences and Formulas applications list the titles of individual slips, and users cannot scroll from slip to slip.

For more information on scrolling and overview, see "Scrolling" on page 2-36 and "Overview" beginning on page 2-44.

Roll views bear some resemblance to the paper roll structure of the Notepad, but there are several major distinctions. For one, users can create new notes in the Notepad but cannot create new subviews in a roll view. Users can also vary the length of notes in the Notepad, but a roll view's subview sizes are fixed. Moreover, a roll view's summary cannot scroll and can display at most 16 one-line subview descriptions. There can be more than 16 subviews; to see beyond the 16th subview, a user must scroll subview by subview, starting with the 16th one.

## How Views Work

Container views provide immediate feedback about actions a user may take, such as opening, closing, moving, and scrolling. The remainder of this chapter describes these behaviors.

### Opening Container Views

Opening a container view makes it visible and gives the user access to it (unless it is partly or completely obscured by another container view that's already open). Some of an application's container views open in response to user actions. Tapping an icon in the Extras Drawer may open an application's main view; tapping a button, tapping a text label marked with a black diamond, or choosing from a picker may open a plain slip, a confirmation alert, or a palette. In addition, an application may open status slips, notification alerts, and other views on its own.

### View Display Order

A Newton user can keep more than one application open at a time (memory permitting). Each open application has its own pile of container views. At the bottom of an application's pile of views is its main view. An application's auxiliary views appear on top of the main view in the order in which they were opened. (Technically, it is the application base view that contains all of an application's other views. Usually the main view is the base view, but you can organize your application differently if necessary.)

A user can bring a view with a matte border and drag handle to the front by tapping the drag handle.

## The Backdrop

A Newton device always has at least one application open, and it is called the **backdrop.** The backdrop's main view is at the bottom of the display order. The backdrop cannot be closed, so its main view has no Close box. For example, the backdrop on an Apple MessagePad 120 is initially the Notepad. A user can change the backdrop with the Extras Drawer application.

## What Is Active

On a Newton device there is no single active view or active application, because all visible views and applications are active. Most views, both movable and stationary, allow users full access to the visible parts of other views. If a user can see a place to tap, write, or draw outside a view, the user can usually do it. A user can even interact with some applications that aren't visible by using the system's Find service or its Intelligent Assistant service (described in Chapter 8, "Newton Services").

Naturally, a small movable view affords the most access to other views behind it. A stationary view only allows access to what a user can see around it. A large movable view—larger than half the height or width of a Newton device's screen—will always block some part of what's behind it. (A user can't move a view partially off the screen.)

It is possible for an application to take over the screen, putting users in the state, or mode, of being able to work only inside one view. The application temporarily suspends access to other views that may be visible. It forces the user to make decisions before doing any other actions, such as adding or changing information in a visible application. Users can cancel a modal view, they can respond to a message in it, or they can use a modal view to set parameters or assign values that become content in a view to which the modal view is subordinate. If users tap in other views, nothing happens. Users must attend to the one modal view and must close it before they can use other views. Users can put away a modal view only by tapping one of its controls.

Although modeless views give users more flexibility, modal views have the advantage of being less ambiguous. Nothing a user does in a modal view should take effect until the user taps a button to confirm the state of the modal view. A modal view avoids intermediate states that can occur with a modeless view, where a user's changes take effect without the user being aware that this is happening.

You can use modal views when your application needs information before it can continue. A modal view is fairly simple to implement, but that doesn't mean that you should use modal views too freely. You should rarely restrict the user's actions by forcing the user into a mode.

## View Position

When designing an application, you must decide where to position the application's main view, ordinary slips, and palettes. The system takes care of positioning routing slips, status slips, notification alerts, and confirmation alerts. In making these decisions consider the type of view, its size in relation to the main view (or the screen), what other views you know will also be open, and how the view's content relates to the other open views.

The positions at which views open affects the usability of your application and of the whole Newton device. Each view that opens may obscure part of the other views already open.

Equally important are users' preferences. If a user moves a view, your application should maintain that position.

### Position of a Main View

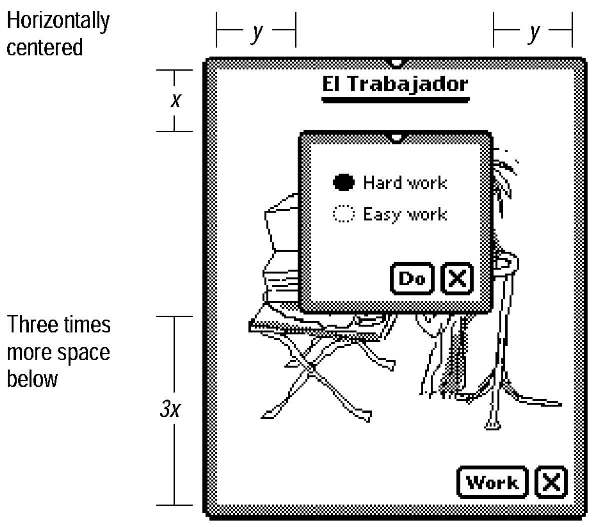
The initial position of a main view that fills the screen is obvious. Most smaller main views are centered horizontally but are off-center vertically. Usually there is about three times as much uncovered screen space below the view as above it. Some main views, such as the built-in Find and Assist views, are centered at the bottom edge of the visible screen area.

If the main view is movable, your application should save its position before closing it, and should reopen it in the position at which the user left it. Keep users in control.

## Position of Auxiliary Views

When a user opens a slip, palette, or other auxiliary view, your application should initially position it directly over the view to which it relates. This arrangement reinforces the relationship between the auxiliary view and its related view, and also puts the auxiliary view near the user's focus. In general you should horizontally center a small auxiliary view over its related view, and place it near the top of the related view, leaving about one-fourth of the uncovered portion of the related view visible above the auxiliary view. Figure 2-25 shows the best position for a small auxiliary view.

**Figure 2-25** Where to position a small auxiliary view



Keep an auxiliary view within the bounds of its related main view (its parent view). If an auxiliary view hangs outside its parent view, the Newton system draws and refreshes the auxiliary view unpredictably. Moreover, the auxiliary view does not get any pen input from outside the parent's bounds. These restrictions have no practical effect on an auxiliary view that is attached to the root view instead of an application's base view.

## Closing a View

Closing a container view makes it go away. Most views close in response to user actions. If a view has a Close box (and most views do), a user can close the view by tapping the Close box. A view may also have other controls that close it. In addition, an application should close a status slip on its own when it finishes the operation that occasioned the status slip.

An application determines what happens visually, audibly, and logically when a user closes the application's views. The user may see a visual effect and hear a sound effect, or the view may seem simply to disappear. The Newton system determines the visual and sound effects when a user closes a notification alert, confirmation alert, or routing slip.

### Closing a Main View

Tapping the Close box in an application's main view closes the application. The main view goes away, together with any of the application's auxiliary views that are also open.

Because a Newton device is personal, most applications should maintain their state while closed, even if the Newton device goes to sleep or a user turns it off. An application that involves an ongoing task should save its state before closing, and it should return to that saved state the next time it opens. State information to be saved and restored includes newly and partially entered data, the positions of all movable container views (including the main view, if it is movable), and anything else the application will need to recreate what the user sees at the time the application closes.

An application doesn't need to save and restore its state if the application involves discrete, short-term tasks—a dictionary, e-mail, and so forth.

## Closing a Slip

A user can close any slip except a confirmation alert by tapping the Close box at the slip's lower right corner. The slip goes away, and the application accepts any changes a user made in the slip unless the slip has a take-action button next to the Close box (as described in "Slips" on page 2-15). If a user taps a take-action button, such as Do or Find, the slip goes away and the application initiates the named action with the settings the user made in the slip. If a user taps a Close box that is next to a take-action button, the slip goes away and the application does not initiate the named action.

To close a confirmation alert a user taps the OK button (or other affirmative button) to authorize the pending action, or taps Cancel (or equivalent) to cancel the action.

## Closing a Drawer

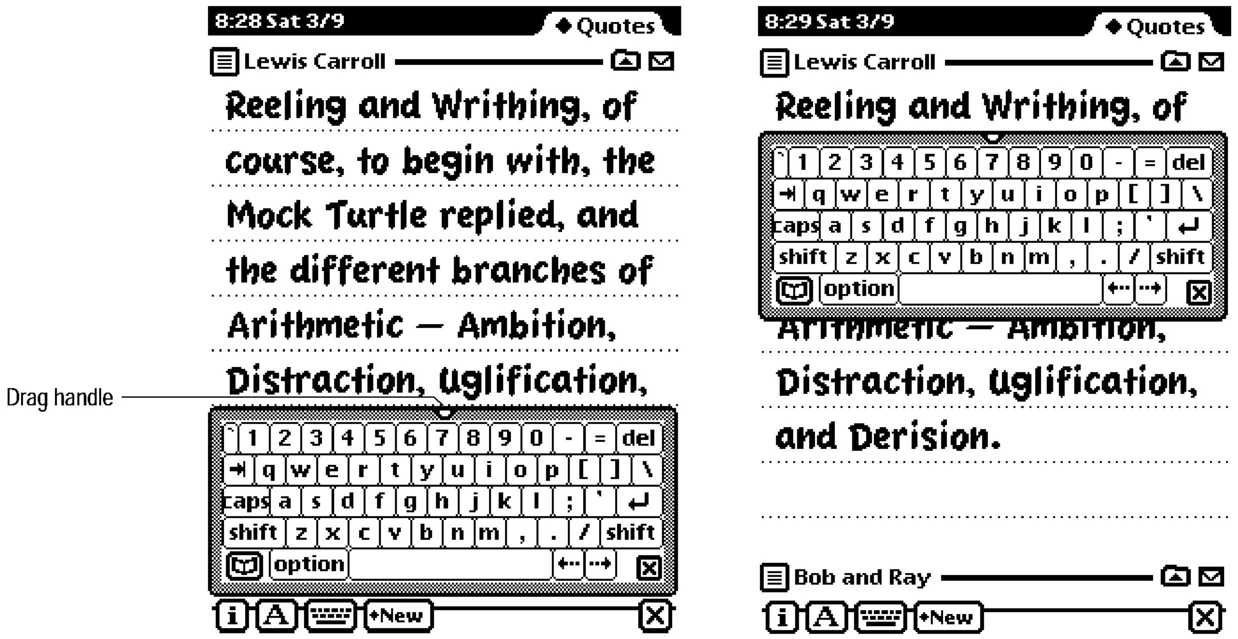
A user can close a drawer by tapping its Close box. Alternatively, a user can close a drawer that has no other views open in front of it by tapping the same button that opened the drawer. If a drawer is open but another view is in front of it, tapping the drawer's button twice closes the drawer. (The first tap brings the drawer to the front.)

## Moving a View

Users expect to be able to move views that have matte borders. To move a view, a user puts the pen on the drag handle and drags to a new location. Figure 2-26 points out a drag handle.

Moving a view doesn't affect the appearance of its contents. A main view can't be moved off the screen, and an auxiliary view can't be moved outside the bounds of the main view (unless the auxiliary view is a child of the root view).

**Figure 2-26** Dragging a view's drag handle moves the view



Reeling and Writhing, of course, to begin with, the Mock Turtle replied, and the different branches of Arithmetic — Ambition, Distraction, Uglification,

Reeling and Writhing, of Arithmetic — Ambition, Distraction, Uglification, and Derision.

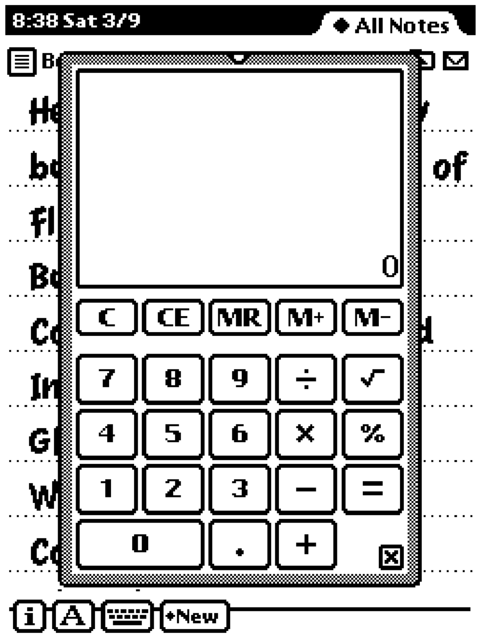
1. User drags the keyboard view's drag handle up

2. Keyboard view moves up

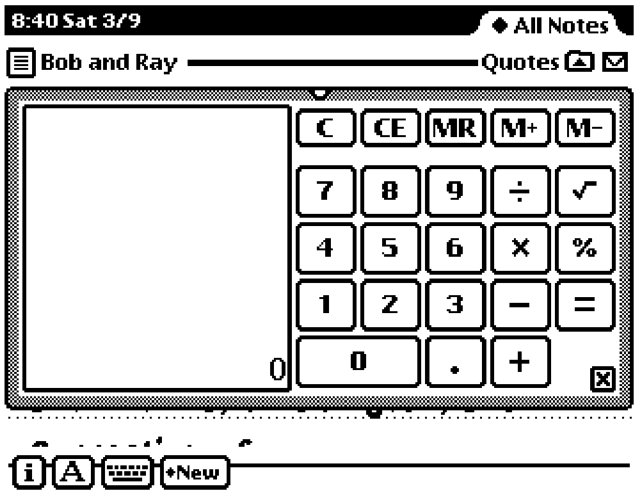
## Changing a View's Size

Your application determines the size of its views. It should base its view sizes on the screen size of the Newton device on which it is running, since Newton screens can come in a wide range of sizes. For example, a container view that fills the screen on an Apple MessagePad 120 (which measures 240× 320 pixels) will not fill the screen on a MessagePad 100 (which measures 240× 336 pixels). The same image would be too tall to fit on a screen of a MessagePad 120 that a user has rotated to the wide orientation (320 × 240 pixels). An application can dynamically determine the screen size, and based on that information can calculate appropriate view sizes. An application may also need to adjust view layouts according to the aspect ratio of the screen. Figure 2-27 shows how the built-in Calculator adjusts its size and layout when a user rotates the display.

**Figure 2-27** Dynamically adjust a view's position, size, and layout to fit the screen



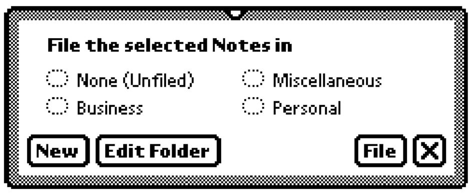
Regular orientation on a MessagePad 120



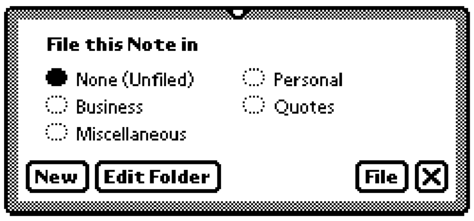
Sideways orientation on a MessagePad 120

An application may grow or shrink one of its views in response to user actions, but users should not be allowed to change view size directly. Do not allow users to resize a view by dragging a corner of it. Figure 2-28 shows how the Filing slip changes size after a user creates a new folder.

**Figure 2-28** A view may change size in response to user actions



1. Before creating a new folder

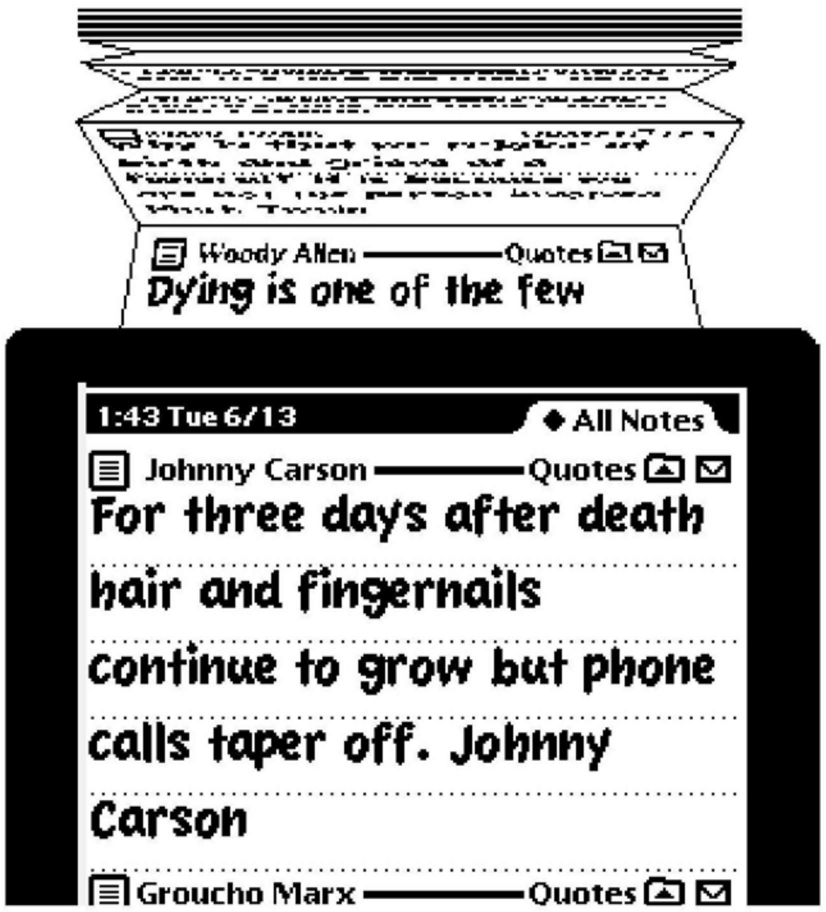


2. After a user creates the Quotes folder

## Scrolling

An application that deals with multiple instances of similar information—multiple notes in the Notepad, multiple names in the Name File, multiple days in the Date Book, and so on—can't display all the instances at once in a single view. People **scroll** the information to move currently displayed information out of view and bring other information into view. The information appears to roll out at one edge of the view and roll in at the opposite edge. Figure 2-29 shows a conceptual view of notes ready to be scrolled in the Notepad's main view.

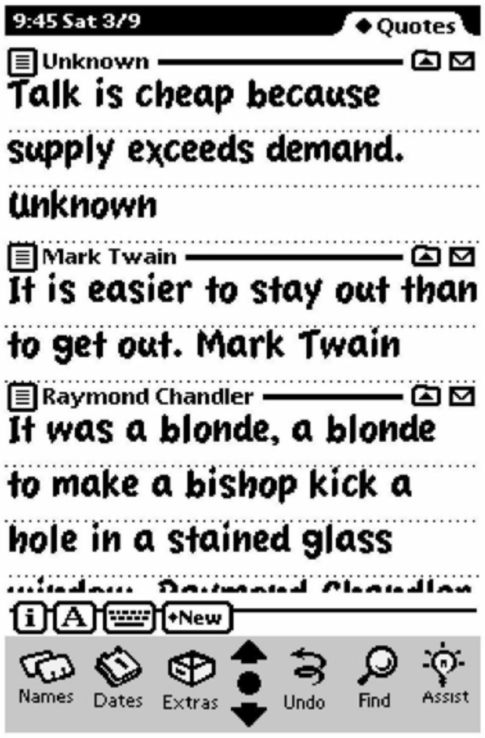
**Figure 2-29** Ready to scroll Notepad notes into view from above or below



## Scrolling With Scroll Arrows

A user scrolls information in a view by tapping scroll arrows on a Newton device. Scroll arrows always come in pairs, each arrow pointing away from the other and toward information that is currently hidden. Tapping an arrow means "Show me more of the information that's hidden in this direction." For example, when a user taps a scroll arrow that points down, the information moves up, bringing up what was just below the view. Pressing and holding the pen on a scroll arrow causes continuous movement in the appropriate direction. Figure 2-30 shows the change when a user scrolls the Notepad by tapping a down arrow.

**Figure 2-30** Scrolling by tapping a down arrow



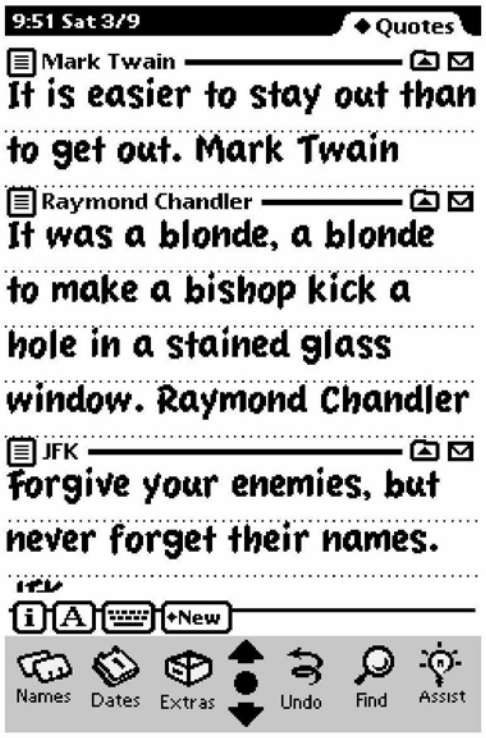
9:45 Sat 3/9 ◆Quotes

Unknown
Talk is cheap because
supply exceeds demand.
unknown

Mark Twain
It is easier to stay out than
to get out. Mark Twain

Raymond Chandler
It was a blonde, a blonde
to make a bishop kick a
hole in a stained glass

1. Before tapping the down arrow



9:51 Sat 3/9 ◆Quotes

Mark Twain
It is easier to stay out than
to get out. Mark Twain

Raymond Chandler
It was a blonde, a blonde
to make a bishop kick a
hole in a stained glass
window. Raymond Chandler

JFK
Forgive your enemies, but
never forget their names.

2. After tapping the down arrow

Each tap on a scroll arrow moves one unit in the chosen direction. Your application determines how much one unit is. For example, the Notepad moves one note for each tap on the arrow; for a note longer than the view, each tap scrolls the number of displayed lines minus one. The Names File application moves one "card" for each tap. The Date Book's day-at-a-time view moves one day for each tap, and the week-at-glance view moves a week per tap. Time Zones moves from city to city alphabetically. If your application's information falls naturally into sections, each tap on a scroll arrow should scroll one section. If not, scroll a screenful minus one line at a time (a "page").

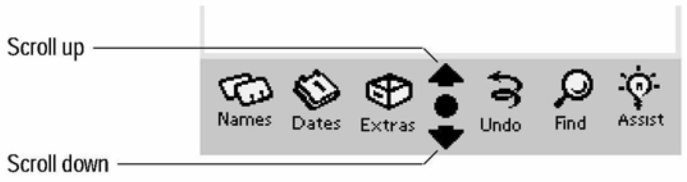
Whether your application should scroll smoothly or unevenly depends on the type of information being scrolled. With smooth scrolling, each tap on a scroll arrow moves the same amount. That is how the Date Book, Names File, Calculator, and Time Zones applications work, for example. In some cases, uneven scrolling is better than smooth scrolling. The Notepad scrolls by uneven increments—note by note—to take advantage of a user's visual memory of where he or she wrote things.

While scrolling up by uneven increments, an application may encounter an item that is too large to display all at once. Since the application can't show the whole item, it must either show the bottom of the item or the top of the item. The appropriate response depends on whether the view scrolls page-by-page or continuously like a roll of paper. A view that scrolls continuously should scroll up to the bottom of an item that is too large to show all at once. A view that scrolls page-by page should scroll up to the top of an item that is too large to show all at once. For instance, the Notepad (which scrolls like a roll of paper) scrolls up to the bottom of a note that is taller than the height of the Notepad main view. In contrast, the Out Box (which scrolls detail items page-by-page) would scroll up to the top of the same note.

## Universal Scroll Arrows

Newton devices have two universal scroll arrows for user control of scrolling. The universal scroll arrows are part of the Newton system; they are not attached to one view. On an Apple MessagePad 120, they are located in the center of the screen, below the display area. Figure 2-31 shows the universal scroll arrows.

**Figure 2-31** The universal scroll arrows at the bottom of a MessagePad screen



Any view can have its scrolling controlled by user taps on the universal scroll arrows, but they only affect one of the open views. To be affected, a view must meet two requirements. First, the view must be set up during application development to receive taps on the universal scroll arrows. Second, it must be in front of all other open views that have also been set up to receive those taps. It is entirely possible for the view that is affected by the universal scroll arrows to be partially or completely covered by other open views that were not set up to receive scroll-arrow taps. (A view that receives scroll-arrow taps also receives taps on the Overview button, which is described in “Overview” on page 2-44).

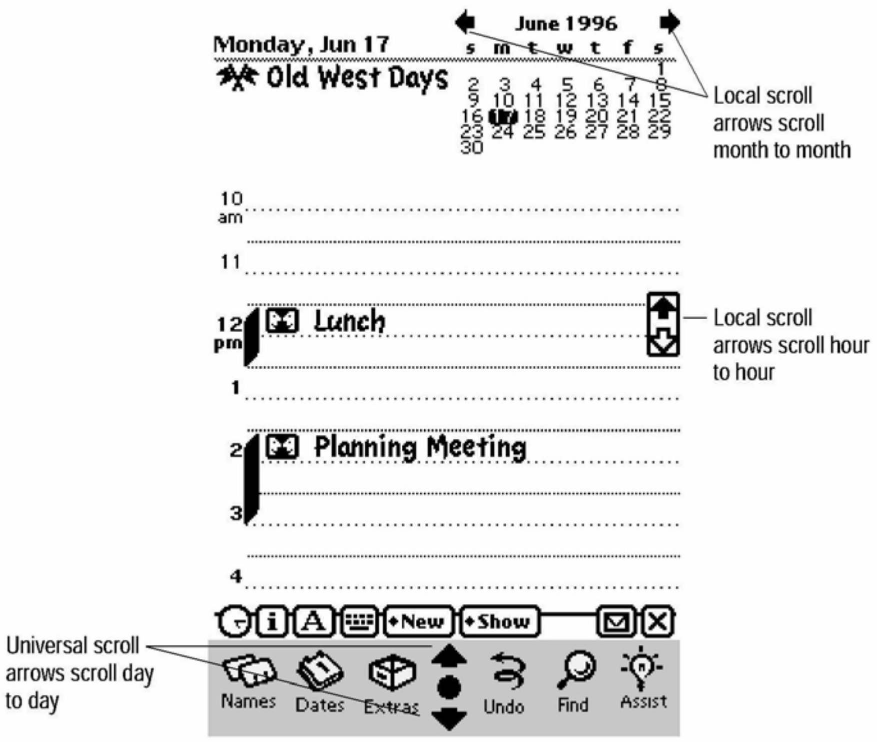
There is no convention for indicating what will scroll when a user taps a universal scroll arrow. Users must learn by experience what will scroll when they tap the universal scroll arrows.

Generally, the universal scroll arrows should scroll most of the information in a view. An application should not use the universal scroll arrows to scroll part of the information embedded in a view. For instance, the built-in Date Book application uses the universal scroll arrows for scrolling from day to day, not for scrolling from hour to hour or month to month.

## Local Scroll Arrows

For scrolling part of the information embedded in a view, an application should use local scroll arrows. For instance, the Date Book has a set of local scroll arrows for scrolling from hour to hour and another set for scrolling from month to month. Figure 2-32 illustrates the Date Book’s use of scroll arrows.

**Figure 2-32** How scroll arrows work in the Date Book's Day view

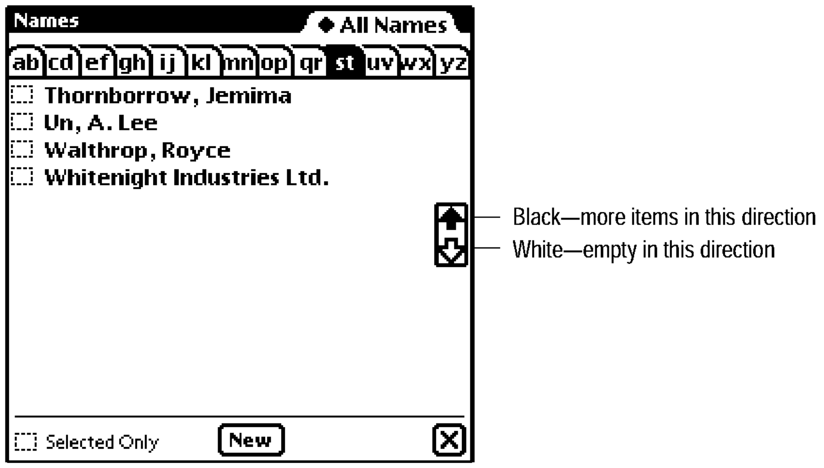

June 1996
Monday, Jun 17
Old West Days
Local scroll arrows scroll month to month
10 am
11
12 pm
Lunch
Local scroll arrows scroll hour to hour
1
2
Planning Meeting
3
4
New
Show
Universal scroll arrows scroll day to day
Names
Dates
Extras
Undo
Find
Assist

Usually each tap on a local scroll arrow scrolls one item of information. If a user presses and holds the pen on a local scroll arrow, items scroll by continuously. After five items have scrolled by, the application can begin scrolling page by page. For scrolling purposes, the size of a page is one less than the number of items that fit in the view. As a shortcut, a user can double-tap a local scroll arrow to scroll a page. These two forms of accelerated scrolling are optional, but an application that features accelerated scrolling should behave as described here.

A view can contain local scroll arrows even if it does not respond to the universal scroll arrows. The local scroll arrows shouldn't scroll the whole view; they should only scroll some part of the view.

Local scroll arrows can use color—white or black—to indicate whether scrolling will bring more items or any more empty space into view. An arrow is black if tapping it will bring more items into view. An arrow is white if tapping it will not bring more items into view. Figure 2-33 illustrates the use of color in local scroll arrows.

**Figure 2-33** Scroll arrow color may indicate what scrolling will reveal

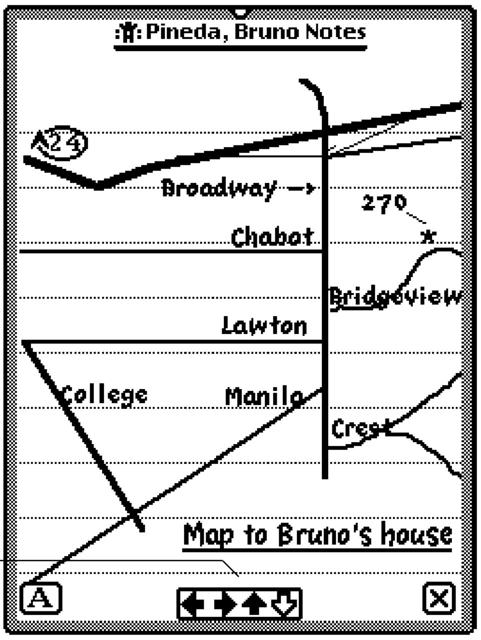

Names
◆ All Names
ab cd ef gh ij kl mn op qr st uv wx yz
Thornborrow, Jemima
Un, A. Lee
Walthrop, Royce
Whitenight Industries Ltd.
Black—more items in this direction
White—empty in this direction
Selected Only
New

If you implement local scrolling in your application and want users to easily recognize your local scroll arrows, make them look like the arrows in the built-in applications. Do not design your own scroll arrows or make them look like scroll arrows on personal computers.

## Four-way Scrolling

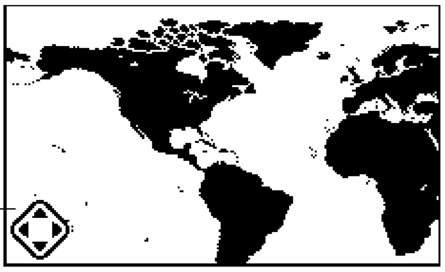
A view that allows the user to pan up, down, left, and right across a map, drawing, or other large document must provide a four-way (two-dimension) scrolling control. The four-way scroller should be centered at the bottom of the view. Figure 2-34 shows how the standard four-way scroller looks.

**Figure 2-34** A control for scrolling in four directions



There's an alternate four-way scroller that may be better in some situations. The alternate scroller is more compact than the standard scroller, but users find the standard scroller easier to target. Figure 2-35 shows the alternate four-way scroller.

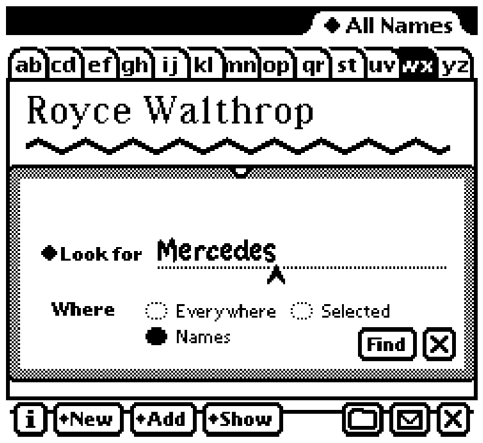
**Figure 2-35** An alternate control for scrolling in four directions



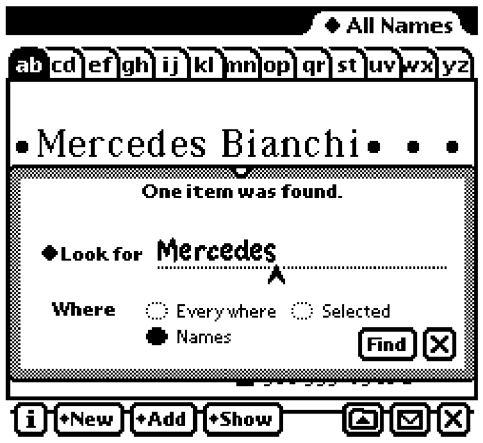
## Automatic Scrolling

In the discussions of scrolling behavior and appearance in the previous sections, the user controls scrolling by deciding which scroll arrow to use and how long to use it. Most of the time the user should be in control, but sometimes an application should scroll a view automatically. When your application performs an operation and the effect is to select something that's not currently visible, your application must scroll to show the new selection. For example, when the user searches for some text, your application locates the desired text. If this text appears in a part of the information that isn't currently visible, your application should scroll the information to show the found text. Figure 2-36 shows the effect of automatic scrolling in the Names File.

**Figure 2-36** Automatic scrolling



1. Before the search for *Mercedes*



2. After the search for *Mercedes*

An application should not scroll automatically any more than is necessary to bring a selection into view. Users want to control what shows in a scrollable view. If part of a selection is showing in the view after the user performs an operation, don't scroll at all. For example, if the user increases the text size of a lengthy selection so that the bottom part extends out of view, your application should not automatically scroll to the end of the selection.

## Scrolling Performance

Scrolling the contents of a view can sometimes seem slow. Here are some techniques you can use to improve scrolling speed:

- Implement the accelerated scrolling behavior described in "Local Scroll Arrows" on page 2-39.
- Scroll multiple lines at a time, rather than just a single line at a time, when the user taps a scroll arrow.
- Reduce the number of child views that need to be redrawn, if possible. For example, make a list that is implemented as several paragraphs (each a separate view) into a single paragraph.
- Set the view fill to white. Many views need no fill color, so you may be inclined to set the fill color to none when you create such a view. However, it's best to fill the view with white if you don't need a transparent view. This can increase the performance of your application because when the system is redrawing the screen, it doesn't have to update views behind those filled with a solid color such as white.

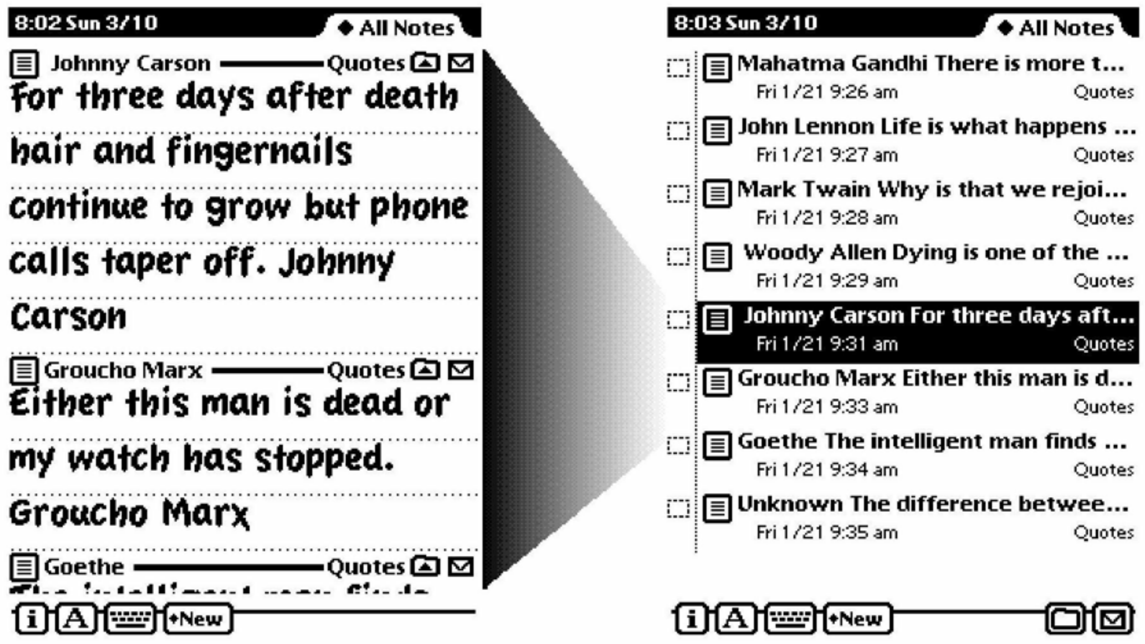
## Overview

In addition to scrolling, an application that deals with multiple instances of similar information can show an overview of the information—a view of the forest instead of the trees. From the overview a user can select a part of the information to see in detail. Figure 2-37 shows a conceptual view of the Notepad's overview.

### Overview Contents

An overview is not another view displayed on top of the current view; an overview is the same view in a different format. Instead of showing individual data items in full detail, an overview presents a summary list of multiple data items.

**Figure 2-37** How an overview relates to a detail view



8:02 Sun 3/10 ◆ All Notes

Johnny Carson Quotes
for three days after death
hair and fingernails
continue to grow but phone
calls taper off. Johnny
Carson

Groucho Marx Quotes
Either this man is dead or
my watch has stopped.
Groucho Marx

Goethe Quotes

8:03 Sun 3/10 ◆ All Notes

| | | |
|---|---|---|
| Mahatma Gandhi There is more t... | Fri 1/21 9:26 am | Quotes |
| John Lennon Life is what happens ... | Fri 1/21 9:27 am | Quotes |
| Mark Twain Why is that we rejoi... | Fri 1/21 9:28 am | Quotes |
| Woody Allen Dying is one of the ... | Fri 1/21 9:29 am | Quotes |
| Johnny Carson For three days aft... | Fri 1/21 9:31 am | Quotes |
| Groucho Marx Either this man is d... | Fri 1/21 9:33 am | Quotes |
| Goethe The intelligent man finds ... | Fri 1/21 9:34 am | Quotes |
| Unknown The difference betwee... | Fri 1/21 9:35 am | Quotes |

An overview commonly takes the form of a table of contents. It lists the titles or names of items that can be viewed in more detail. Together with the name or title of each item, the overview lists a a key bit of information about the item. For instance, the Notepad lists the first part of each note next to its title. The Names File lists a phone number with each name. The Call Log lists the date and time of each call. And the Extras Drawer lists each item's size and where it is stored.

If an item's title or supplemental information is too long for the space available in an overview, the application can use an ellipsis to indicate there is more information.

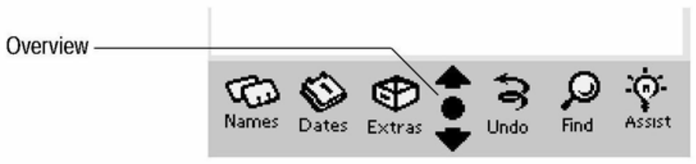
Next to each item in an overview there may be a small icon that symbolizes the type of item—note, checklist, or outline in the Notepad; person, company, group, or place in the Names File; and so forth. The overview may also have a checkbox next to each item so a user can select one or more items and act on the selected items with controls in the status bar, such as a Filing button or Action button (see "Primary Controls and Status Bar" on page 2-11). A gray line separates checkboxes from data items.

If an overview lists items that users may have filed in folders, the overview should have a folder tab at the top so users can determine which folder's items are displayed (see "Folder Tab" on page 8-19). If listed items cannot be filed in folders, the application can organize the overview by some other criterion, such as by date. For example, the Date Book groups meetings and events by date, starting with the displayed date.

## Overview Button

The control for seeing an overview is the round black button located between the universal scroll arrows. On an Apple MessagePad the Overview button is at the center of the screen below the display area. Figure 2-38 shows the Overview button.

**Figure 2-38** The Overview button at the bottom of a MessagePad screen

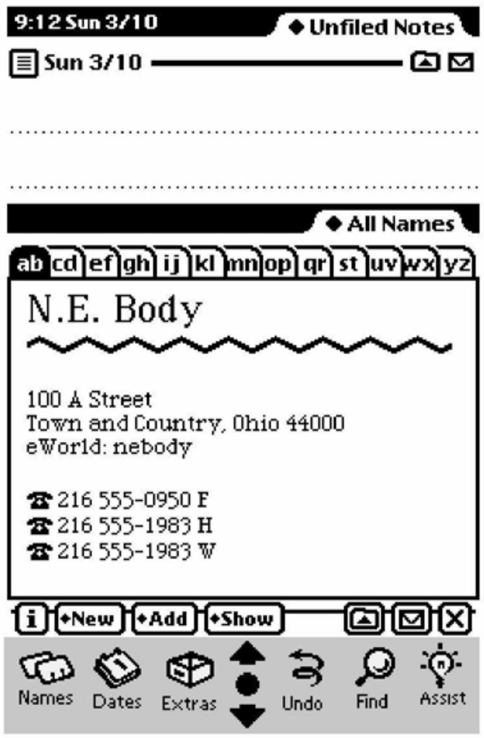


Like the universal scroll arrows, the Overview button is a function of the Newton system and is not attached to one view. The Overview button affects one open view. To be affected a view must be set up during application development to receive taps on the Overview button. In addition, the view must be in front of all other open views that have also been set up to receive those taps. The view that is affected by the Overview button may be partially or completely covered by other open views that were not set up to receive Overview-button taps. (A view that receives taps on the Overview button also receives taps on the universal scroll arrows, which are described in "Universal Scroll Arrows" on page 2-38).

## Switching to and from an Overview

To see an overview, a user taps the Newton device's Overview button. The detail item that was displayed should either be centered in the overview or at the top of the overview. Figure 2-39 shows the change when a Names File user taps the Overview button.

**Figure 2-39** Getting an overview

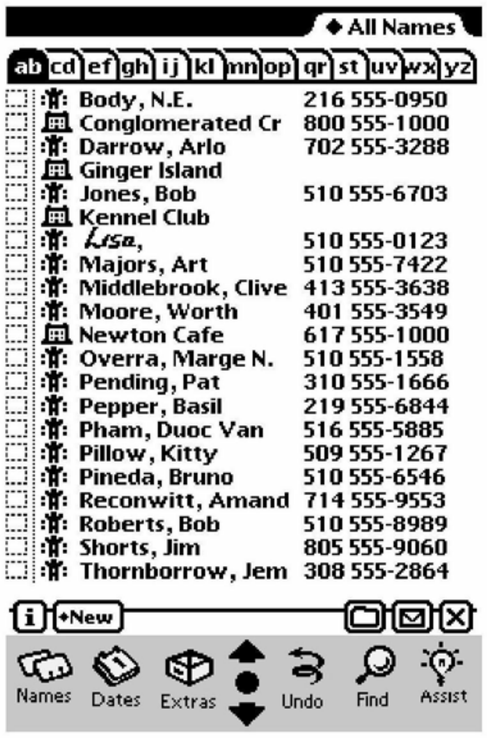


N.E. Body

100 A Street
Town and Country, Ohio 44000
eWorld: nebody

☎ 216 555-0950 F
☎ 216 555-1983 H
☎ 216 555-1983 W

1. Before tapping the Overview button



| | |
|---|---|
| Body, N.E. | 216 555-0950 |
| Conglomerated Cr | 800 555-1000 |
| Darrow, Arlo | 702 555-3288 |
| Ginger Island | |
| Jones, Bob | 510 555-6703 |
| Kennel Club | |
| Lisa, | 510 555-0123 |
| Majors, Art | 510 555-7422 |
| Middlebrook, Clive | 413 555-3638 |
| Moore, Worth | 401 555-3549 |
| Newton Cafe | 617 555-1000 |
| Overra, Marge N. | 510 555-1558 |
| Pending, Pat | 310 555-1666 |
| Pepper, Basil | 219 555-6844 |
| Pham, Duoc Van | 516 555-5885 |
| Pillow, Kitty | 509 555-1267 |
| Pineda, Bruno | 510 555-6546 |
| Reconwitt, Amand | 714 555-9553 |
| Roberts, Bob | 510 555-8989 |
| Shorts, Jim | 805 555-9060 |
| Thornborrow, Jem | 308 555-2864 |

2. After tapping the Overview button

To see an item in detail, a user taps its title in the overview. Tapping an item listed in the overview closes the overview and displays the detail view for the item that was tapped. Your application could also take different actions depending on where the user taps an entry in the overview. For example, in a Names File overview, tapping the left part of a line, where the name is displayed, causes the normal view of the tapped name to appear; but tapping the right part of the line, where the telephone number is displayed, initiates a phone call.

If an application spends more than a few seconds preparing an overview, it should display a status slip with a message such as "Preparing overview..." (see "Status Slips" on page 2-20).

## Scroll and Overview in an Overview

An overview should respond to the universal scroll arrows and the Overview button. When a user taps a scroll arrow, your application should scroll all the currently visible items out of view except the last item. After scrolling, there should always be one item still visible from before scrolling. When scrolling the last item into view, the application can leave more than one item still visible from before scrolling so that the overview remains full of items; users can't do anything with empty space in an overview.

When a user taps the Overview button, the overview goes away and the detail view reappears, showing the same item as before the overview appeared. Thus, repeatedly tapping the Overview button switches back and forth between the overview and the detail view. Avoid any temptation to use the Overview button to zoom up or down multiple levels of detail or to step through a hierarchy of views, because the user can easily become confused about whether the next tap will go up or down.

An application that doesn't have a list-style overview can use the Overview button to toggle a view between two levels of detail or magnification. If there are more than two levels of zooming, the Overview button must always toggle between the same two levels, preferably the biggest and smallest levels. For access to other levels, use another type of control described in Chapter 3, "Controls." For example, you could use a button on the status bar—the Show button or a special Zoom button or zoom-looking picture button. For completeness, the magnification control should provide access to all levels, including the two levels controlled by the Overview button.

## Closing an Overview

Tapping the Close box has the same effect whether a view is displaying item detail or an overview—the application closes. Tapping a Close box in an overview does not switch to item detail.

## Nonfunctional Scroll and Overview Controls

Some views scroll or display an overview, but not both. Rather than doing nothing when a user taps a nonfunctional scroll or overview control, a view should provide feedback. If a view that scrolls but doesn't have an overview is frontmost and a user taps the Overview button, the view should notify the user with a message such as "This view does not have an overview." Likewise, if a user taps a scroll arrow when a view that doesn't scroll is frontmost, the view should notify the user with a message such as "This view does not scroll." To avoid confusing users, don't begin messages about not scrolling or not having an overview with "This application" unless the message applies to every one of the views in your application. Furthermore, better messages state the title of the view in place of the generic "This view" or "This application.

If the frontmost view has local scroll arrows but doesn't respond to the universal scroll arrows and doesn't let views behind it receive universal scroll arrow events, then the view should display a message that explains why the universal arrows don't work or what the user must do to make them work. Under these circumstances a message such as "You must close this view to use the universal scroll arrows" is clearly more accurate than "This view does not scroll."

No view has to receive taps on the Overview button and universal scroll arrows. If the frontmost view was not designated during application development to receive taps on those controls, the taps go to the next lower view that was so designated. For example, a slip or other auxiliary view can simply let its main view respond to the Overview button and the universal scroll arrows.

C H A P T E R 3

# Controls

Controls are graphic objects that cause instant actions or audible results when the user manipulates them with the pen. Some controls change settings that modify future actions. Other controls allow users to make choices or to assign parameters in a range. Controls display existing choices so that they are visible to users. Because of their appearance and behavior, controls enhance the user's sense of direct manipulation.

This chapter describes the appearance and behavior of the following basic Newton controls:

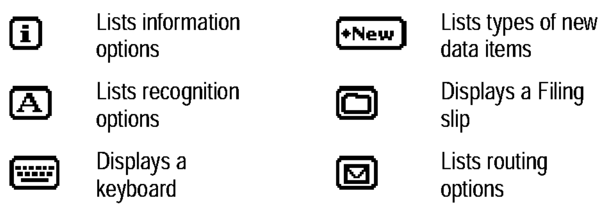
- Text and picture buttons
- Close boxes (regular and large)
- Radio buttons
- Checkboxes
- Sliders
- Hot spots

This chapter ends with brief descriptions of standard Newton buttons that appear in the status bars of many applications: the Analog Clock, Info, Recognizer, Keyboard, New, Show, Filing, Action, Item Info, and Rotate buttons.

## Buttons

A **button** is a small graphic object that performs an action when tapped. The action that the button performs is described by text or a picture on the button, as shown in Figure 3-1.

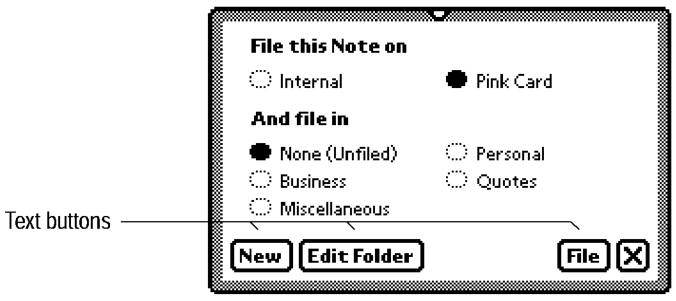
**Figure 3-1** Tapping a button initiates an action


Lists information options
Lists types of new data items
Lists recognition options
Displays a Filing slip
Displays a keyboard
Lists routing options

### Text Buttons

A **text button** is a rounded rectangle containing text that names the button's function. Figure 3-2 shows some typical text buttons in a slip.

**Figure 3-2** A text button's name states what the button does



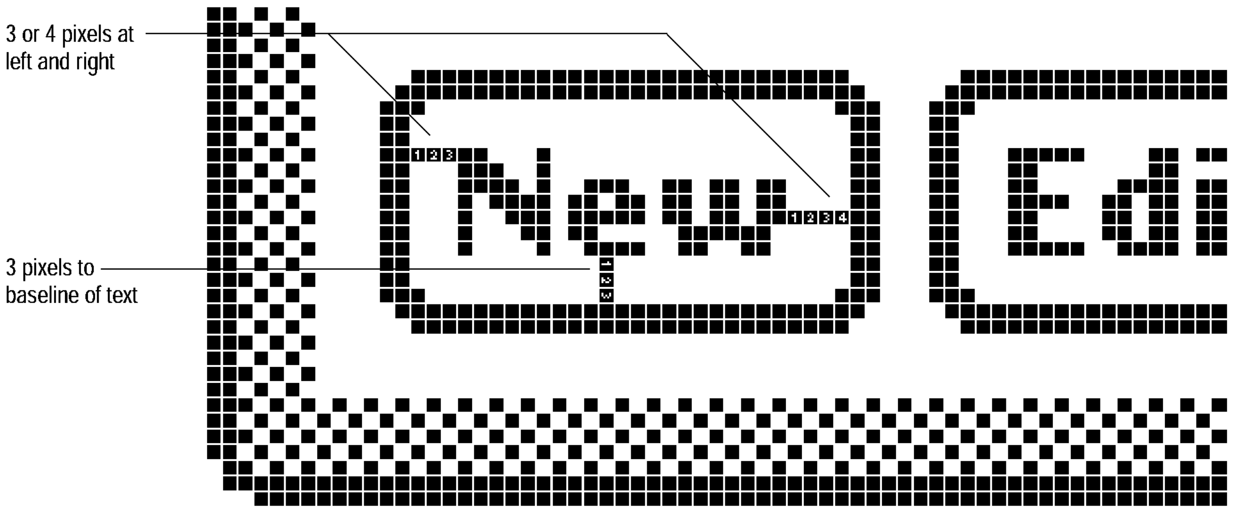
File this Note on
○ Internal ● Pink Card
And file in
● None (Unfiled) ○ Personal
○ Business ○ Quotes
○ Miscellaneous

## Text Button Sizes

A text button should be the same height as the large Close box (described under "Close Boxes" on page 3-14) and wide enough for its name to fit centered on one line in the bold style of the system font. Make the button wide enough to leave as much space at the sides of the text as there is space above the text.

On an Apple MessagePad, make text buttons 13 pixels tall (not counting the 2-pixel black border). Use 9-point text in the bold style of the system font for the button name. Leave three pixels between the button's bottom border and the baseline of the text, and make the button wide enough to leave three or four pixels between the name and the button's left and right borders. In a group of buttons, you can make all buttons uniformly narrower if you must, but always leave at least two pixels of white space at the right and left ends of the text. Figure 3-3 shows the preferred spacing between the name and border of a text button on an Apple MessagePad.

**Figure 3-3** Leave standard margins between a button's name and its borders



If your application has buttons whose names change during the operation of the application, the application must resize the button when its name changes so that the spacing always conforms to the guidelines.

## Naming Text Buttons

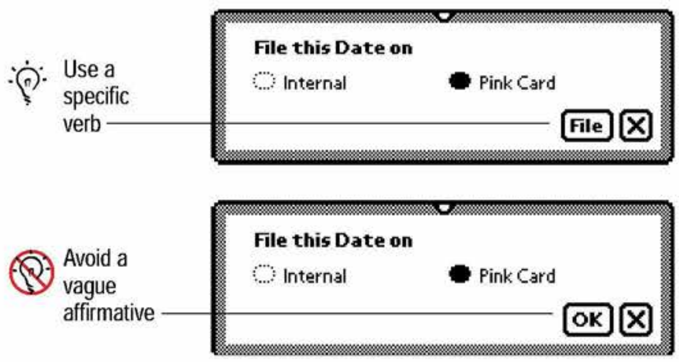
Keep button names short. Never use more than three words for a button name, and try to limit button names to one word. Capitalize button names like book titles. That is, always capitalize the first and last words of the name, and capitalize all other words except articles (*a, an, the*), coordinating conjunctions (for example, *and, or*), and prepositions of three or fewer letters. Since button names should seldom be more than two words, almost all words in button names should be capitalized.

Avoid punctuation and symbols in button names. Except for very common symbols such as an ampersand (&), users find symbols ambiguous. Do not use ellipses (…) in the button name even if tapping the button displays another slip. However, a button name should begin with a diamond symbol (◆) if the button pops up a picker. (For complete information on pickers, see Chapter 4, “Pickers”).

## Naming Take-Action Buttons

A user typically reads the text in a slip until it becomes familiar, and then relies on visual cues, such as button names or positions, to respond. To assist users in quickly spotting which button in the slip initiates an action, name the take-action button with a specific verb such as Print, Fax, or File. These words are self-sufficient, whereas vaguely affirmative names such as OK and Yes require the user to scan other parts of the slip to verify what action the button initiates. Figure 3-4 compares a specific button name to a generic button name in the same context.

**Figure 3-4** Name buttons distinctively wherever possible

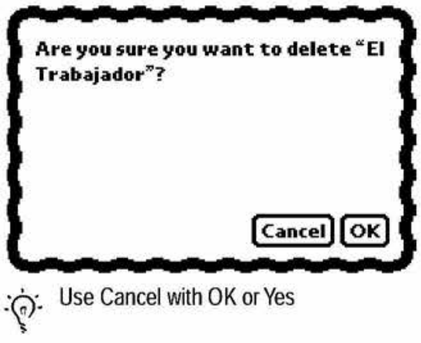


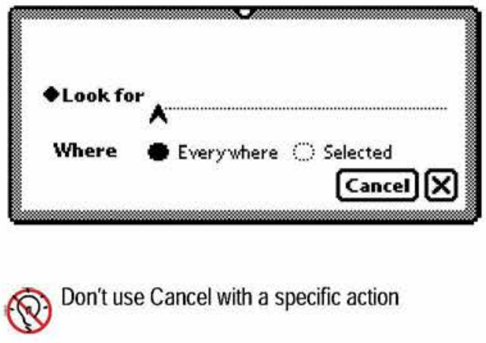
There are cases where a button named OK or Yes serves best. You may want to name a button with a vague affirmative to encourage the user to look elsewhere in the slip for a complete description of a pending action with far-reaching consequences. Another place to use OK or Yes is in a slip where you simply can't name the action to be taken with one or two words. If you name buttons with generic words, be sure other text in the slip makes clear what action the button initiates.

## Naming Cancel- and Stop-Action Buttons

A slip with a button that initiates an action also needs a means of canceling the pending action. On a Newton device you ordinarily use a large Close box (described under "Close Boxes" on page 3-14), not a button named Cancel, to dismiss a slip and take no action. However, you can use a button named Cancel to complement an OK or Yes button. Figure 3-5 shows where to use a Cancel button and where not to use one.

**Figure 3-5** Where to use a button named Cancel

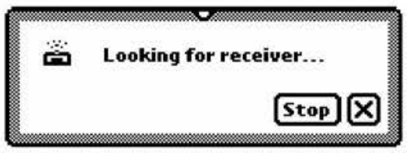

Are you sure you want to delete "El Trabajador"?
Cancel
OK
Use Cancel with OK or Yes



A button named Cancel should close the view it's in and return the application to the state it was in before the view appeared. Cancel means "Dismiss the operation I started, with no other effects." A Cancel button should not be the only button that can close a view; in such a view, use a Close box instead.

Rather than Cancel, use a text button named Stop to allow a user to halt an operation that's underway, especially if this button is next to a large Close box. In that context a Cancel button may look to some users like a duplicate of the Close box. Figure 3-6 illustrates the use of a Stop button.

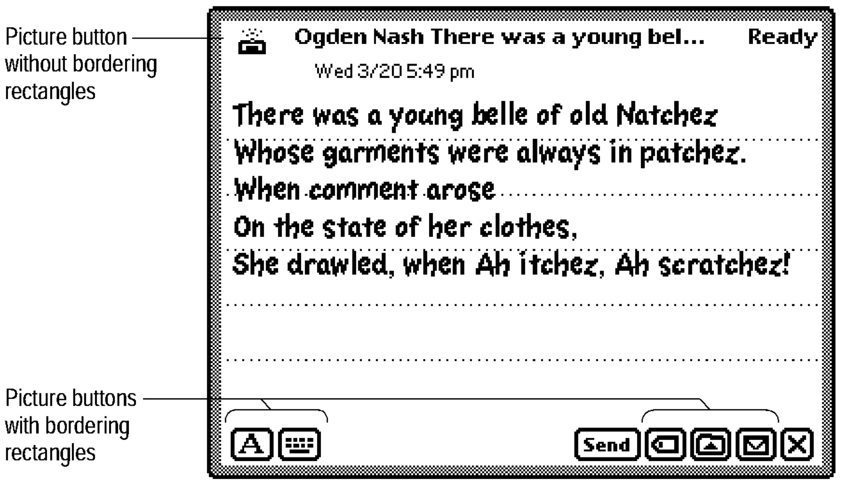
**Figure 3-6** A Stop button lets a user halt an operation



## Picture Buttons

A **picture button** is a small picture (an icon) that represents the button's function. The picture is usually bordered by a rounded rectangle, like a text button with a picture instead of a text name. Figure 3-7 shows several picture buttons.

**Figure 3-7** A picture button depicts what the button does

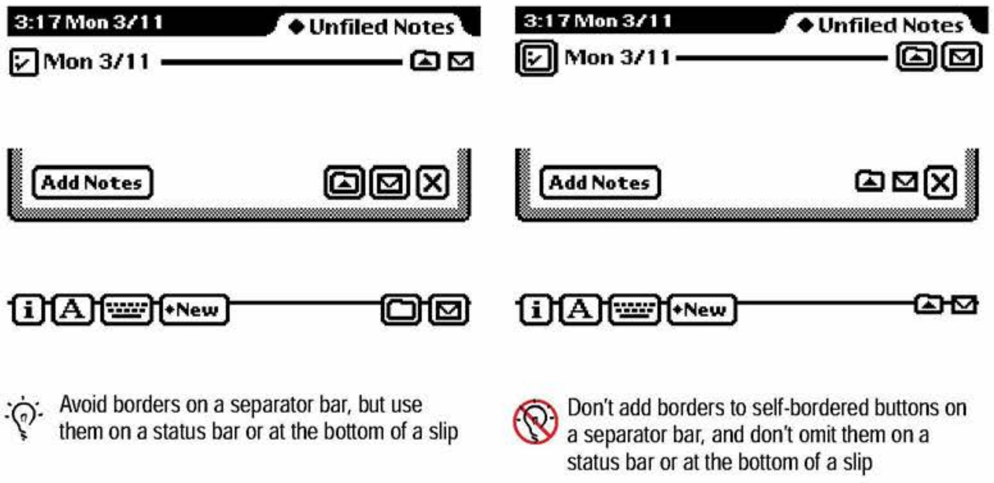

Picture button without bordering rectangles
Picture buttons with bordering rectangles
Ogden Nash There was a young bel...
Ready
Wed 3/20 5:49 pm
Send

There was a young belle of old Natchez
Whose garments were always in patchez.
When comment arose
On the state of her clothes,
She drawled, when Ah itchez, Ah scratchez!

A picture button should be as tall as the large Close box (described under "Close Boxes" beginning on page 3-14) and wide enough to enclose the picture. Leave white space between the picture and the interior edge of the button border. On an Apple MessagePad, make picture buttons with borders 13 pixels tall (not counting the 2-pixel black border), and leave at least one pixel of white space between the picture and the border.

Picture buttons on a status bar definitely need bordering rectangles to match the other items there and to isolate the pictures from the bar. Conversely, picture buttons on separator bars do not touch the bar and often look better without bordering rectangles. For example, the Filing button and Action button must have a border when they are on a status bar, but look better without a border on a separator bar. A button looks good without a border if its picture has an unbroken line around it—a sort of self-border. Figure 3-8 shows where you should omit picture button borders and where you should keep the borders.

**Figure 3-8** Where to use borders with small, self-bordered picture buttons


3:17 Mon 3/11
◆ Unfiled Notes
Mon 3/11
Add Notes
New
3:17 Mon 3/11
◆ Unfiled Notes
Mon 3/11
Add Notes
New
Avoid borders on a separator bar, but use them on a status bar or at the bottom of a slip
Don't add borders to self-bordered buttons on a separator bar, and don't omit them on a status bar or at the bottom of a slip

## Designing Picture Buttons

Picture buttons are tempting to use because they are more compact than text buttons. But designing an unambiguous picture small enough for a picture button is very hard. A button's function may be more evident if you label it with a short name instead of a picture.

Avoid pictures that look like the buttons of other applications, such as Names and Dates. Users may think your look-alike button will open the corresponding built-in application, and may be confused if your buttons do something else.

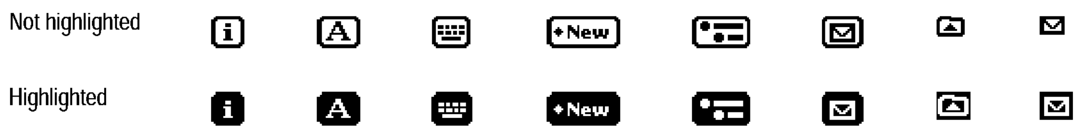
## Button Behavior

Although text buttons and picture buttons look different, their basic behavior is the same. Both types of buttons provide similar feedback to the user, and an application disables both types the same way.

### Button Feedback

When a user taps a text button or a picture button, the button highlights (inverts) to give visual feedback to the user that the item has been tapped. Figure 3-9 shows how several buttons look when highlighted.

**Figure 3-9** Tapping a button highlights it



A button stays highlighted as long as the user continues to press the pen on that button. When the user lifts the pen from the highlighted button, the action associated with the button takes place. Your application must continue to highlight the button until the action is complete. In the case of a button that displays an ordinary slip (not a status slip), the button stays highlighted only until the slip appears. In the case of a button that pops up a picker (described in Chapter 4, “Pickers”), the button stays highlighted as long as any action initiated by the picker is in progress.

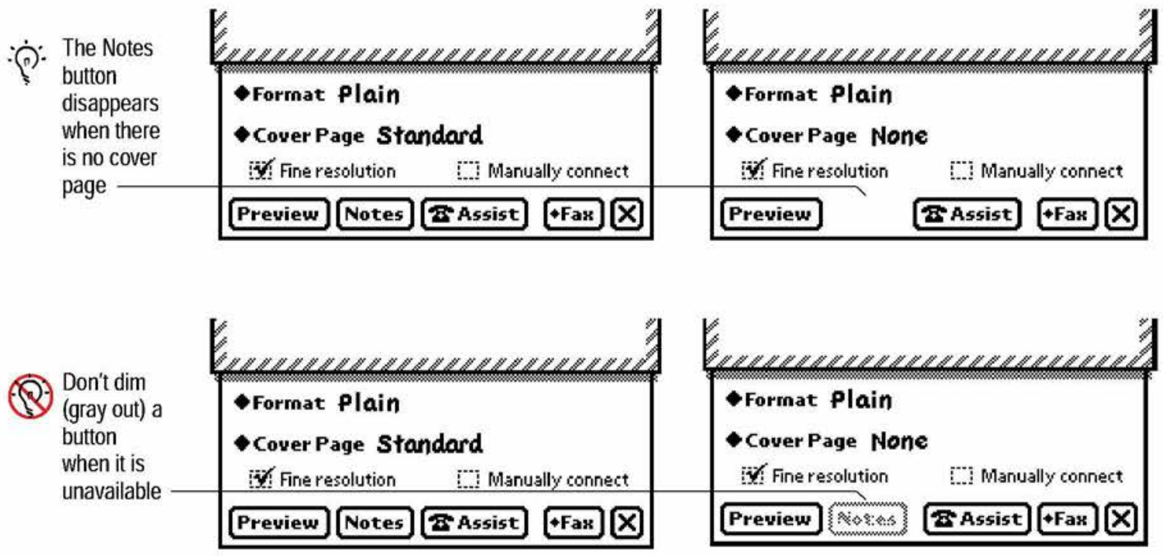
Keeping the button highlighted provides the minimal feedback to the user that Newton is still working. When an action begun by a button takes more than a few seconds, your application should provide more feedback by displaying a status slip that names the action underway (as described in “Status Slips” on page 2-20).

If your application uses buttons made from system protos, the system automatically adjusts button highlighting in response to a user’s pen movements. When a user slides the pen away from a highlighted button while still pressing the pen on the screen, the button becomes unhighlighted. The button tracks the pen movement as long as the user keeps pressing the pen. If the user slides the pressed pen back over the button, it is highlighted again. If the user lifts the pen while the pointer is not over the button, nothing happens. The display of electronic ink is turned off while the pen is tracked. A button is not highlighted if a user initially presses the pen outside the button and slides the pressed pen over the button.

## Button States

Text buttons and picture buttons are never dimmed (displayed in gray instead of black) in a Newton application. If a button is not available, your application should make it disappear. If you want it to be available, your application should make it reappear. Figure 3-10 shows how to disable a Newton button.

**Figure 3-10** A button disappears when it isn't available


The Notes button disappears when there is no cover page
Don't dim (gray out) a button when it is unavailable
Preview
Notes
Assist
Fax

◆Format Plain
◆Cover Page Standard
☑ Fine resolution ☐ Manually connect

◆Format Plain
◆Cover Page None
☑ Fine resolution ☐ Manually connect

A button can disappear and reappear with no visual effect or with a subtle visual effect such as zoom closed and zoom open. Generally, buttons should not flash as they appear. Visual effects that attract the eye virtually compel immediate action, as if they were shouting, "Tap me now!"

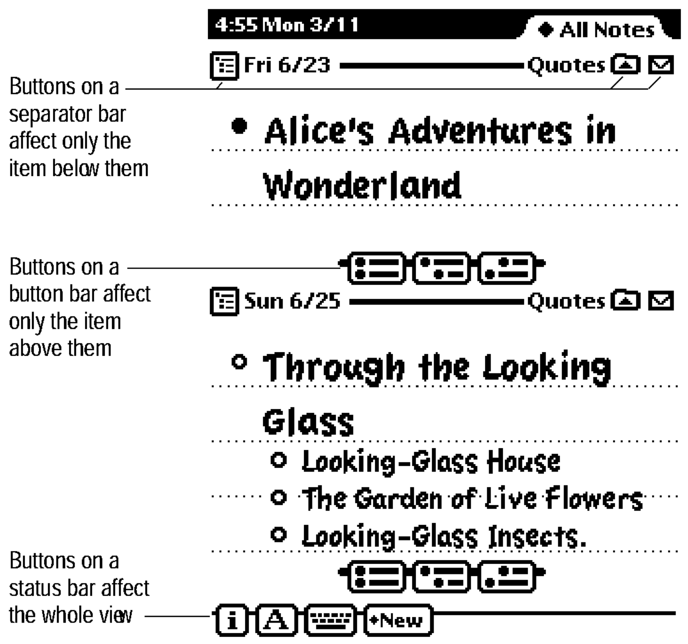
## Button Placement

Text buttons and picture buttons are easiest to use at the bottom of the view that contains them. In that position a user's hand won't cover the view while tapping a button. Buttons that affect all items in a main view should go on a status bar at the bottom of the main view (see "Primary Controls and Status Bar" on page 2-11). The status bar is optional if the main view is small, like the main views of the built-in Connection, Card, and Time Zones applications. Buttons at the bottom of a slip or other auxiliary view are generally not on a status bar.

In a view that displays multiple items, each item is headed by a separator bar (see "Separator Bars" on page 2-11). Buttons that apply to only one item do not go at the bottom of the view. Instead, the buttons are attached to each item's separator bar, and they scroll along with the item when a user scrolls the view. For example, a Filing button and an Action button go at the right end of each separator bar (see "Filing Button" on page 3-27 and "Action Button" on page 3-28). Other buttons can go on a button bar that floats near the end of an item. Figure 3-11 shows buttons on a separator bar, buttons on a floating button bar, and buttons on a status bar.

It's possible for your application to add buttons to another application's status bar. For example, your application could add a button to the backdrop application's status bar so users could bring your application to the front without going through the Extras Drawer. This could be a boon or a nightmare depending on how crowded the status bar is in the backdrop application. If you want to add a button to another application, make sure users can disable the feature.

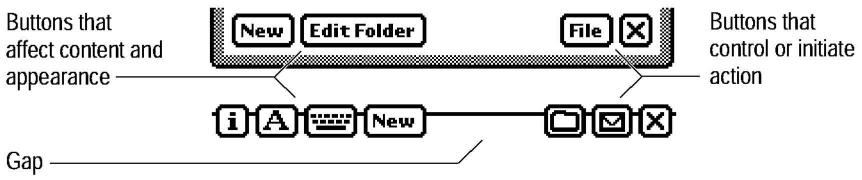
**Figure 3-11** Where to put buttons in a view


4:55 Mon 3/11
◆ All Notes
Fri 6/23
Quotes
Alice's Adventures in Wonderland
Sun 6/25
Quotes
Through the Looking Glass
Looking-Glass House
The Garden of Live Flowers
Looking-Glass Insects.
New
Buttons on a separator bar affect only the item below them
Buttons on a button bar affect only the item above them
Buttons on a status bar affect the whole view

## Button Spacing

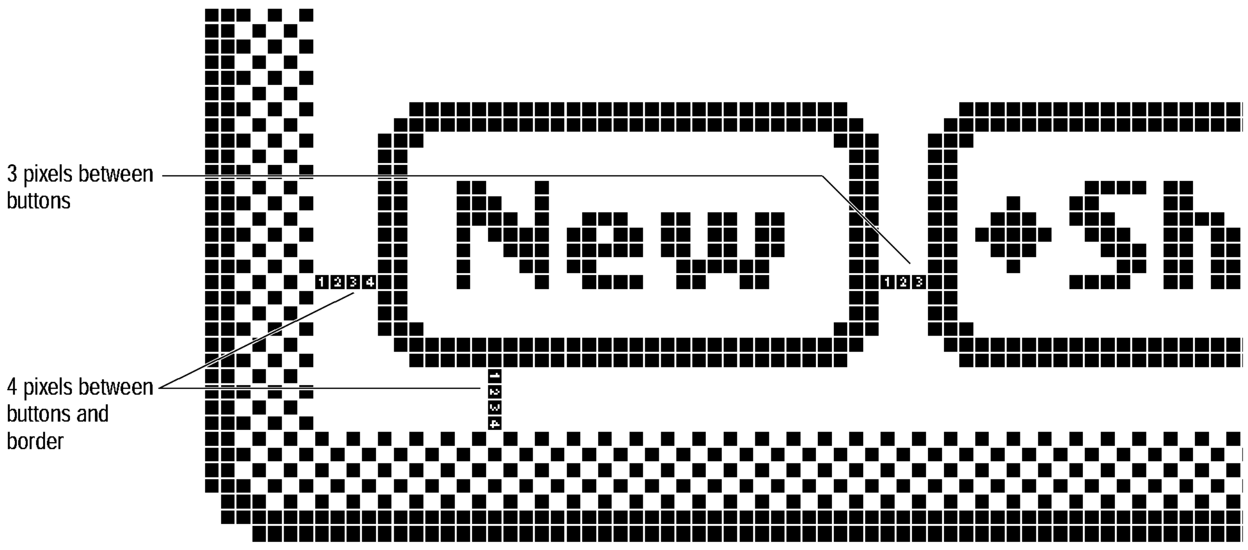
Group text and picture buttons with similar functions together. Users assume buttons near each other are related. Generally, buttons that directly control or take action are on the right, and buttons that affect content or appearance are on the left. Separate the left and right groups with blank space, if the number and sizes of buttons permit. Figure 3-12 shows buttons groups at the right and left sides of a container view, with a gap separating the two groups.

**Figure 3-12** Group buttons by function


New
Edit Folder
File
New
Buttons that affect content and appearance
Buttons that control or initiate action
Gap

Avoid spacing consecutive buttons so close together that they look cramped. On an Apple MessagePad, space consecutive buttons in a group three pixels apart, and leave four pixels between buttons and the view's border. If you must, you can reduce the space between consecutive buttons to two pixels, but no less. Figure 3-13 illustrates the MessagePad guidelines.

**Figure 3-13** Regular spacing between buttons on a MessagePad



If your application changes a button's name, it should dynamically resize the button and maintain the correct spacing between buttons. Your application should maintain the correct spacing between buttons when a user rotates the display (with the Extras Drawer). Accommodate the changing width of a view by adding or taking away space between groups of buttons. Your application also needs to maintain the spacing between buttons if their names change during operation.

### Large Buttons

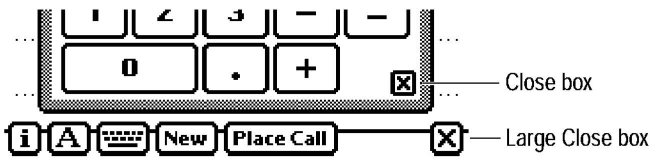
If a user needs to be able to tap some text buttons or picture buttons in your application with a finger instead of a pen, you can use large buttons. If your large buttons won't fit at the bottom of a view, it's OK to put them along one side of the view. Put them only on one side, but be sure the user can choose whether they appear on the right or left. A user's left hand would block the whole screen while tapping large buttons along the right edge of the screen, so left-handed users need to be able to set an option that shifts large buttons from the right side to the left side of the screen. If your application includes large buttons and you want them to work when a user rotates the display (with the Extras Drawer), your application needs to be able to adjust the position of the large buttons for regular or sideways orientation of the display.

## Close Boxes

The Close box and large Close box are both picture buttons that contain the picture of a cross shaped like an X. Both buttons work in the same way. Tapping a Close box or a large Close box closes the container view in which the button appears.

The differences between the Close box and large Close box are purely cosmetic. The large Close box is the same height as standard text and picture buttons. The frame around the X is not part of the large Close box's picture. The Close box is slightly smaller than the large Close box, and its picture includes the frame around the X. Figure 3-14 compares the Close box and the large Close box.

**Figure 3-14** A Close box compared to a large Close box

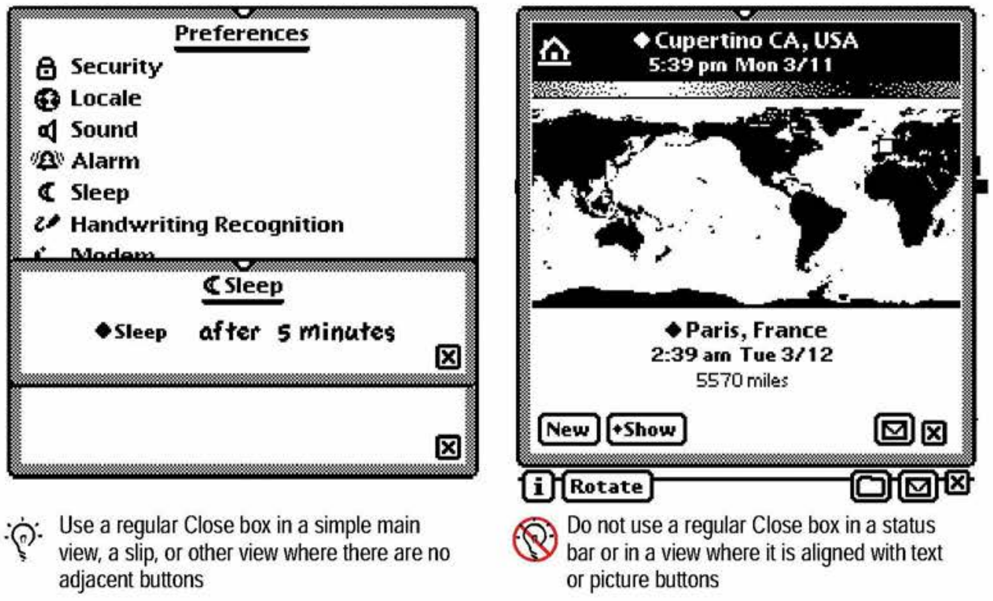


Always put the Close box or large Close box in the bottom right corner of the container view it closes.

## Where to Use a Regular Close Box

The contents of a container view determine whether it should have a Close box or large Close box. A large Close box looks best alongside text buttons or picture buttons. A regular Close box does not look good next to text or picture buttons because it is smaller than they are. Figure 3-15 shows where and where not to use a regular Close box.

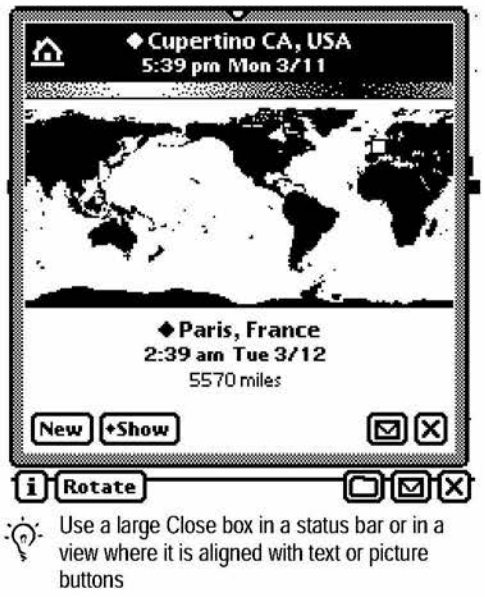
**Figure 3-15** Where to use a regular Close box


Preferences
Security
Locale
Sound
Alarm
Sleep
Handwriting Recognition
Modem
Sleep
◆Sleep after 5 minutes
◆Cupertino CA, USA
5:39 pm Mon 3/11
◆Paris, France
2:39 am Tue 3/12
5570 miles
New
◆Show
Rotate
Use a regular Close box in a simple main view, a slip, or other view where there are no adjacent buttons
Do not use a regular Close box in a status bar or in a view where it is aligned with text or picture buttons

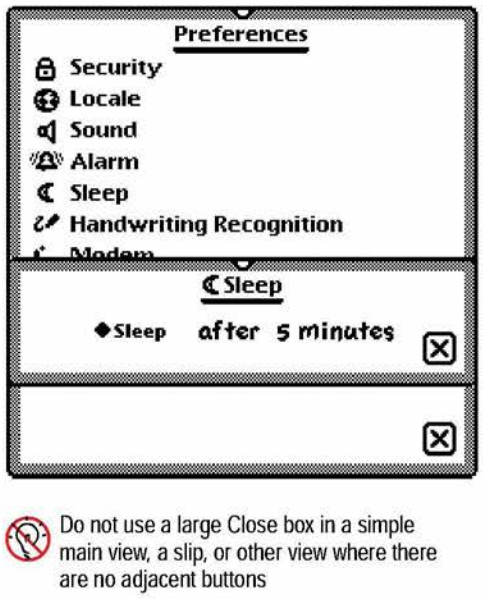
## Where to Use a Large Close Box

Always include a large Close box at the right end of a main view's status bar so the user can close your application. In a view without a status bar, the large Close box looks better than a regular Close box alongside text or picture buttons, but do not use a large Close box in a slip with an OK or Yes button. Instead, use a Cancel button (see "Naming Cancel- and Stop-Action Buttons" on page 3-5). Figure 3-16 shows where to use a large Close box and where not to use one.

**Figure 3-16** Where to use a large Close box



Use a large Close box in a status bar or in a view where it is aligned with text or picture buttons



Do not use a large Close box in a simple main view, a slip, or other view where there are no adjacent buttons

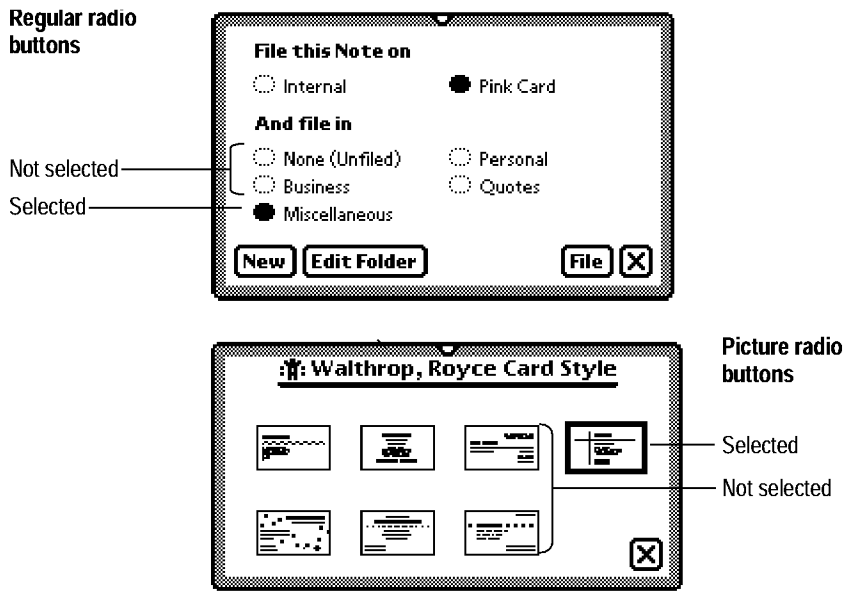
## Radio Buttons

A **radio button** is a control that displays a setting that can be either on or off. Each radio button is part of a group in which only one radio button can be on at a time. (The inverse arrangement, in which only one button is off, is also possible, but is not described separately in this book because it is logically equivalent to the normal arrangement.)

There are two types of radio buttons. One is a small oval that is empty if it is not selected, or is filled with solid black if it is selected. The oval radio button is labeled to the right with a word or phrase.

The second type of radio button is a small picture with a border (unless the picture itself has a continuous edge). Typically, several of these picture radio buttons are placed next to each other, and the one that is selected is indicated by a thick border. Figure 3-17 shows some regular radio buttons and some picture radio buttons.

**Figure 3-17** Only one radio button in a cluster can be selected



An application can use radio buttons to control options, such as the order in which to sort information. An application can also use radio buttons to change the attributes of a selected object, such as the style of a view. In contrast, an application should use a text button or a picture button—never use a radio button—to bring up a slip; to confirm, authorize, cancel, or stop an action; or to initiate a process.

To operate a radio button the user can tap any part of it, including the text or picture that identifies it. Tapping one button in a cluster turns off whichever button was on before.

A cluster of radio buttons must contain at least two items. Instead of using a single radio button, use a checkbox (see the next section, "Checkboxes"). At the opposite extreme, a cluster shouldn't contain more than seven radio buttons. A large cluster of radio buttons simply takes up too much space on the screen. Rather than a lot of radio buttons, use a picker (see Chapter 4, "Pickers").

A cluster of radio buttons always has the same set of choices. It never changes contents depending on the context. If more than one group of radio buttons is visible at one time, the groups need to be visually separate from each other. Each cluster may have a heading to identify it.

Radio buttons and cluster headings are usually capitalized like book titles. However, in some contexts it makes more sense to capitalize them like sentences. If the radio button text completes a sentence begun by the label of the radio button cluster, the heading should be capitalized like a sentence and the radio buttons should be all lowercase (except for proper nouns).

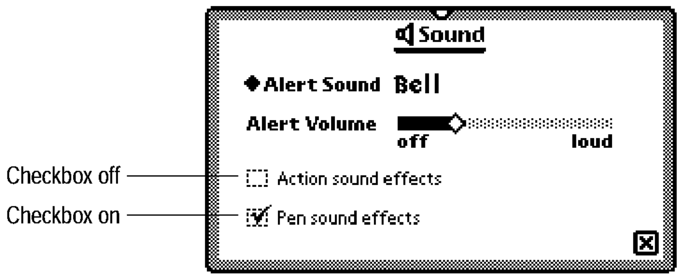
Avoid punctuation in radio buttons and cluster headings. In particular, do not end a cluster heading with a colon. The heading's meaning is clear without the colon.

## Checkboxes

**Checkboxes,** like radio buttons, provide alternative choices for users. A checkbox is a small dotted box that may include a check mark (✓). It is labeled to the right or left with a word or phrase. Use a checkbox to indicate an option that must be either off or on.

The user selects or deselects the checkbox by tapping it or its label. When the option is selected, a check mark appears in the box. When the option is not selected, the box is empty. Checkboxes act like toggle switches, which can be on or off. A checkbox is on when it is selected and off when it is not selected. Figure 3-18 shows some typical checkboxes.

**Figure 3-18** Each checkbox can be on or off



You can have one checkbox or as many as you need. Checkboxes are independent of one another, even when they offer related options. Any number of checkboxes can be on or off at the same time. It's a good idea to group sets of checkboxes that are related, and to separate the groups from other groups of checkboxes and radio buttons. Each group may have a heading to identify it.

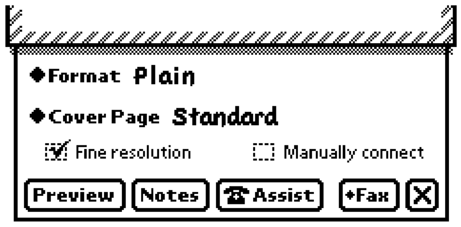
Labeling a checkbox unambiguously can be difficult. The label should imply two clearly opposite states. For example, the Sound preferences slip (Figure 3-18) shows a checkbox labeled "Action sound effects" that controls whether or not sound effects are played for actions. If the checkbox is selected, the action sound effects are played. The clearly opposite state, when the checkbox is not selected, is to not play action sound effects.

Checkboxes are usually capitalized like sentences. However, in some contexts it makes more sense to capitalize them like book titles. Checkbox labels should be all lowercase (except for proper nouns) if the checkboxes are part of a group and each label completes a sentence begun by the label of the group.

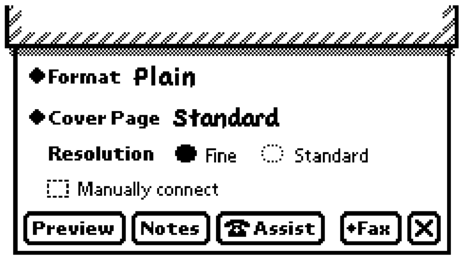
If you can't think of a label for a checkbox that clearly implies its opposite state, you might be better off using radio buttons. With radio buttons, you can use two labels, thereby clarifying the states. It's sometimes tempting to use a checkbox because one item takes up less space than two. However, the resulting item may be ambiguous and thus difficult to understand.

Sometimes using one checkbox instead of two radio buttons can make users focus more carefully on a choice between two states. One state is clearly labeled, but a user must think to figure out the unlabeled state. While thinking, the user may briefly ponder the significance of changing the checkbox's state. For example, a checkbox in a fax routing slip lets a user select fine resolution or not. This option could be implemented with two radio buttons, perhaps labeled "Fine" and "Standard." A user is more likely to think about all the ramifications of the choice with a checkbox than with two radio buttons. Figure 3-19 illustrates the two approaches.

**Figure 3-19** One checkbox vs. two radio buttons



Checkbox takes less space and may make a user think about unstated consequences, but the unlabeled state may be ambiguous



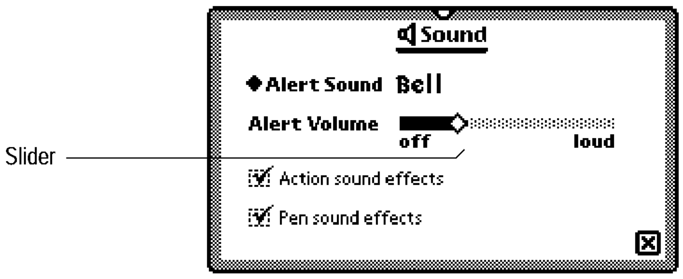
Radio buttons explicitly label choices, but take more space and may obscure unstated consequences

You can use checkboxes to control options and set attributes in your application. But use a text button or picture button, not a checkbox, to bring up a slip; to confirm, authorize, cancel, or stop an action; or to initiate a process.

## Sliders

The Newton interface also includes a **slider,** with which users can set a magnitude, position, probability, or other value in a range. Users set a value by dragging a diamond-shaped knob. The knob indicates the current value relative to the maximum and minimum values. A slider should have labels that identify the range and direction of the slider. Figure 3-20 shows an example of a slider.

**Figure 3-20** A slider used for data input



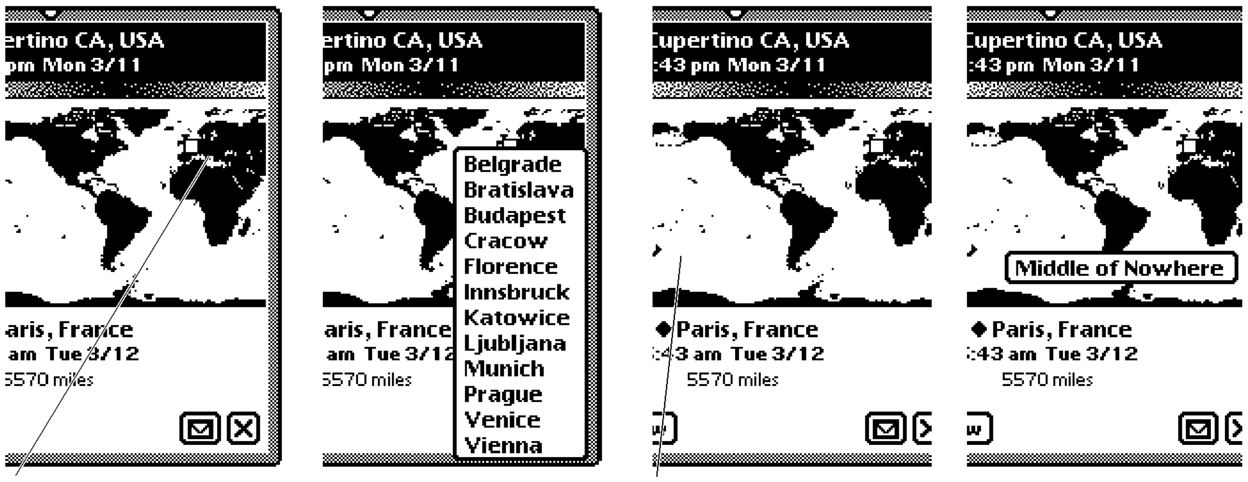
## Hot Spots

Some views need to have many small, unnamed controls that respond like buttons when tapped. For example, a view that contains a map might respond to a user tapping a place on the map by displaying information about the place tapped. These **hot spots** may be visible or transparent.

Make it clear what elements of a view are hot spots unless you want to hide them from users. If space and design considerations permit, give hot spots a distinctive look that users can learn means “tap here.” Simple geometric shapes—such as circles, squares, and triangles—are possibilities. When a user taps a visible hot spot, highlight it to provide feedback. Users need feedback to reassure them that they have really tapped the hot spot.

Hot spots that are very small and close together may have to be invisible to avoid cluttering the view they’re in. Feedback is very important with small invisible hot spots, because a user can’t be sure which hot spot he or she is tapping. One possibility is to display a list of all the hot spots near where the user tapped, and let the user select one of the listed spots by tapping it in the list. If the user taps a place where there aren’t any nearby hot spots (a “cold” spot), the application can provide feedback by listing one item that explains the situation, such as “Nothing here.” Figure 3-21 shows how the Time Zones application responds when a user taps one of its invisible hot spots.

**Figure 3-21** Providing feedback for small, transparent hot spots



1. A user taps a hot spot

2. A list of nearby cities appears, and the user can tap one to select it

1. A user taps where there isn't a hot spot

2. The application provides feedback nevertheless

Of course, sometimes the whole point of hot spots is to make users guess where to tap. Secret hot spots would be fine in maps meant to teach geography by exploration, for example.

In addition to graphical hot spots, there can be hot spots in text. For example, double-tapping a word pops up a picker for editing the word (see "Correcting Misrecognized Text" on page 6-29).

## Standard Newton Buttons

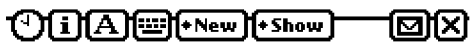
A typical application includes some of the following standard Newton buttons, either on a status bar, on a separator bar, or in a slip: the Analog Clock, Info, Recognizer, Keyboard, New, Show, Filing, Action, Item Info, and Rotate. This section describes where each of those standard buttons belongs and what it does.

Other specific controls defined by the Newton system are described elsewhere. For descriptions of scroll arrows and the overview button, see "Scrolling" on page 2-36and "Overview" on page 2-44. The Undo button is described in "Error Correction" on page 6-37. The Notify Icon is described in "Notify Button and Picker" on page 8-2. The Action button is described in "Action Button and Picker" on page 7-8. The New, Show, Filing, Find, and Assist buttons are described in Chapter 8, "Newton Services."

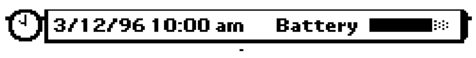
## Analog Clock Button

Space permitting, you can include an **Analog Clock button** at the left end of your application's status bar. The clock continuously displays the time of day. If a user taps the clock, a small panel bearing a digital clock and battery gauge slides out. The panel goes away automatically after three seconds unless the user taps it sooner. Figure 3-22 shows how the Analog Clock button works.

**Figure 3-22** How the Analog Clock button works



The Analog Clock button displays the time



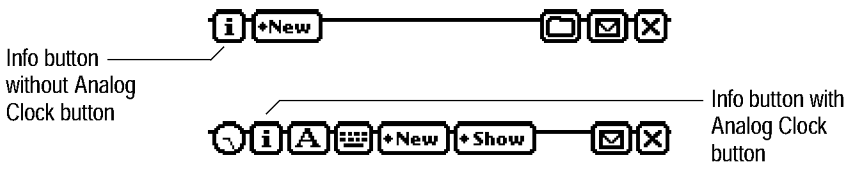
Tapping the Analog Clock button briefly displays a digital clock and battery gauge

## Info Button

An Info button lets users access your application's About box, preference settings, and on-screen help. Put the Info button at the left end of the status bar, next to the Analog Clock button if it is present. Figure 3-23 shows examples of Info button placement.

Tapping the Info button pops up the Info picker, which is described on page 4-24.

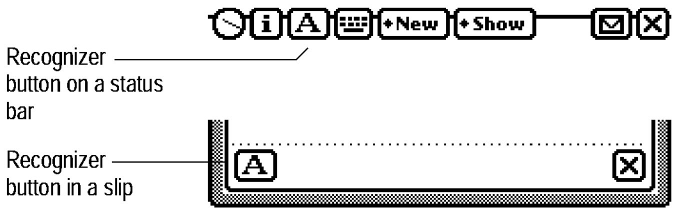
**Figure 3-23** Where an Info button goes


+New
+Show
Info button without Analog Clock button
Info button with Analog Clock button

## Recognizer Button

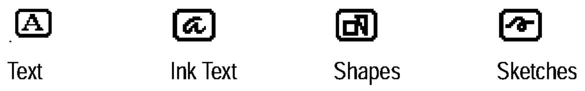
A Recognizer button lets users control the system's recognition of handwriting and drawing. An application's main view should have a Recognizer button to the right of the Info button on its status bar if users can write or draw in the main view. Additionally, a Recognizer button can go in the lower left corner of each slip in which users can write or draw. Figure 3-24 shows a Recognizer button on a status bar and one in a slip.

**Figure 3-24** Where a Recognizer button goes


+New
+Show
Recognizer button on a status bar
Recognizer button in a slip

The Recognizer button is a picture button, and the picture on the button indicates the type of recognition currently in effect. Figure 3-25 shows how the Recognizer button looks for each type of recognition.

**Figure 3-25** The Recognizer button indicates the type of recognition in effect

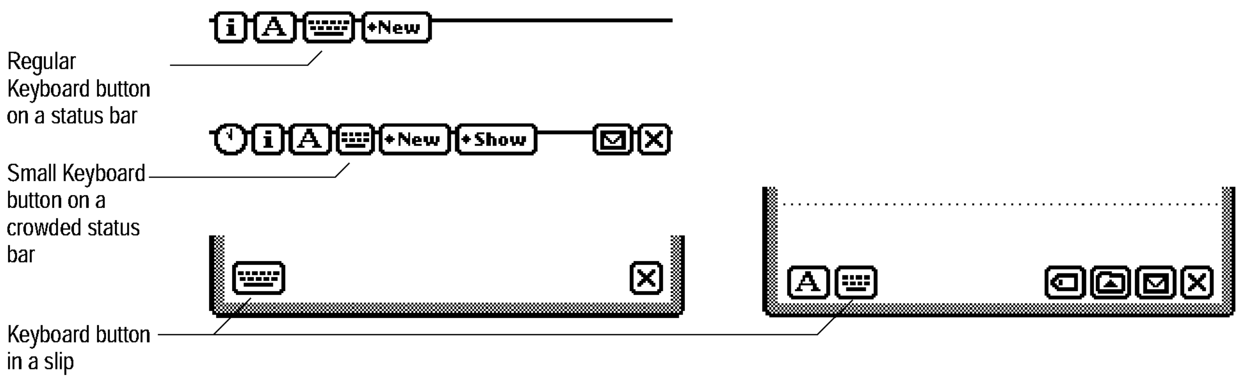


Tapping a Recognizer button pops up the Recognizer picker, which is described in "User Control of Recognition" on page 6-16. For more information on recognition of handwriting and drawing, see "Recognition" on page 6-15.

## Keyboard Button

A Keyboard button lets users bring up an on-screen keyboard. Users can also use the Keyboard button to switch between the available styles of on-screen keyboards. If users can enter text in your application, it should have a Keyboard button to the right of the Recognizer button in the status bar. Alternatively, a Keyboard button can go in the lower left corner of each slip in which users can enter text. The Keyboard button comes in two sizes, regular and small. Use the small size only if space doesn't permit using the regular size. Figure 3-26 shows both sizes of Keyboard button on status bars and in slips.

**Figure 3-26** Where a Keyboard buttons goes

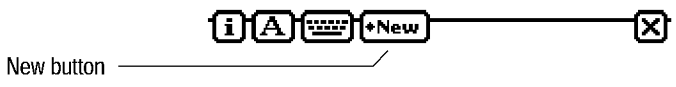

Regular Keyboard button on a status bar
Small Keyboard button on a crowded status bar
Keyboard button in a slip
New
New
Show

Tapping the Keyboard button brings up an on-screen keyboard. Tapping the Keyboard button while a keyboard is displayed pops up the Keyboard picker. Keyboards and the Keyboard picker are described in "Typing" on page 6-32.

## New Button

A New button lets users create a new data item and to specify the format of the item, such as a new note, checklist, or outline in the built-in Notepad application. If users can create new data items in your application, it should have a New button to the right of the Keyboard button on the status bar. Figure 3-27 shows a New button on a status bar.

**Figure 3-27** Where a New button goes

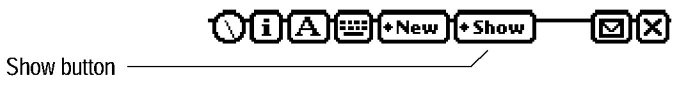


Tapping a New button brings up a New picker, which is described on page 4-25.

## Show Button

A Show button lets users change views for displaying information, such as the Card view or the All Info view in the built-in Names File application. An application should have a Show button to the right of the New button if users can pick different views in the application. Figure 3-28 shows a sample Show button.

**Figure 3-28** Where a Show button goes

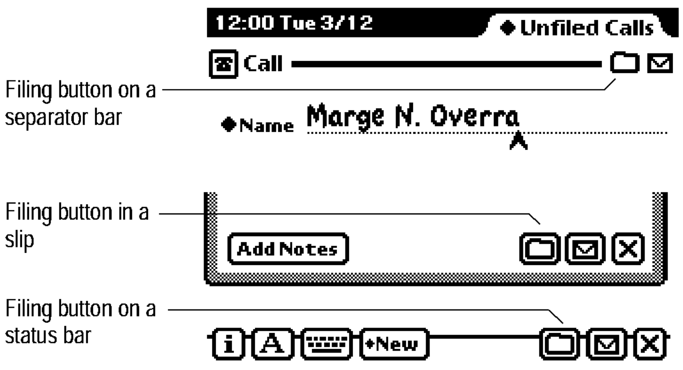


Tapping a Show button brings up a Show picker, which is described on page 4-26.

## Filing Button

A Filing button lets users designate a folder and a storage location (if more than one is available) for data that's currently displayed. How much data is affected depends on where the Filing button is located. If the Filing button is on a status bar, it affects all the data in the main view or all the selected items in an overview. If the Filing button is on a separator bar, it affects the information between that separator bar and the next one. If the Filing button is in a slip, it affects all the information in the slip. A Filing button goes to the left of an Action button near the right end of a status bar; at the right end of a separator bar; or in the lower right corner of a slip. Figure 3-29 shows Filing buttons on a status bar, on a separator bar, and in a slip.

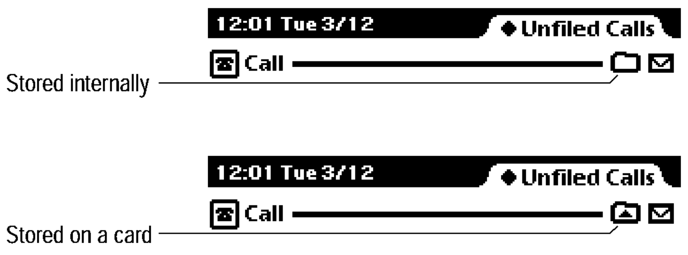
**Figure 3-29** Where a Filing button goes



Tapping a Filing button brings up a Filing slip. The slip lists the storage locations currently available, such as the internal store and a named card store. The slip also lists the folders defined for the current application, and has buttons for editing the existing folder names or creating new folders. Tapping a Filing button brings up a Filing slip, which is described in "Filing Button and Slip" on page 8-14. For general information about filing, see "Filing" on page 8-13.

By looking at a Filing button for a particular data item, users can tell whether the current item is stored internally or on a card. If the item is stored on a card, the Filing button contains a small black triangle. If the item is stored internally, the Filing button contains nothing. Figure 3-30 compares the two states of the Filing button.

**Figure 3-30** A Filing button reports where a data item is stored

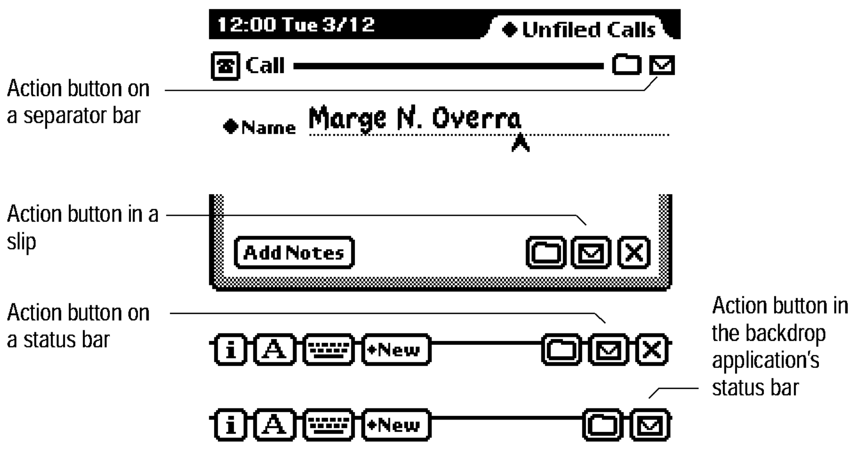


## Action Button

An Action button lets users send data through various means, such as print, fax, beam, and e-mail. In many applications, users can also use the Action button to duplicate and delete data that’s currently displayed. The location of the Action button determines how much data is affected. If the Action button is on a separator bar, it affects the information between that separator bar and the next one. If the Action button is in a slip, it affects all the information in the slip.

In a slip or on a status bar, the Action button goes next to the Close box (except that in the backdrop application, which has no Close box, it goes at the right end of the status bar). An Action button goes at the right end of a separator bar. Figure 3-31 shows examples of Action buttons on a status bar, on a separator bar, and in a slip.

**Figure 3-31** Where an Action button goes

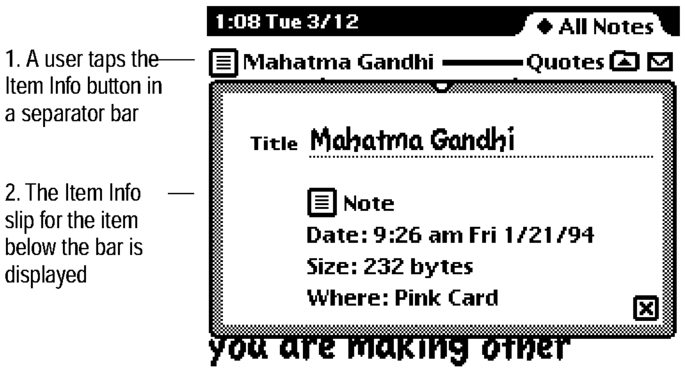

12:00 Tue 3/12
◆Unfiled Calls
Call
◆Name
Marge N. Overra
Add Notes
New
New
Action button on a separator bar
Action button in a slip
Action button on a status bar
Action button in the backdrop application's status bar

Tapping an Action button pops up the Action picker, which is described on page 4-26. For general information about sending and receiving data, see Chapter 7, "Routing and Communications."

## Item Info Button

The picture button at the left end of a separator bar, which is called an Item Info button, depicts the type of item below the separator bar. For example, the Notepad application includes three types of items: plain note, checklist, and outline. Tapping an Item Info button brings up an Item Info slip, in which a user can change the title of the item below the separator bar. The Item Info slip also reports statistics about the item, such as its type, when it was created, where it is stored, and how much storage space it occupies. Figure 3-32 shows a sample Item Info button and slip.

**Figure 3-32** Seeing an Item Info slip

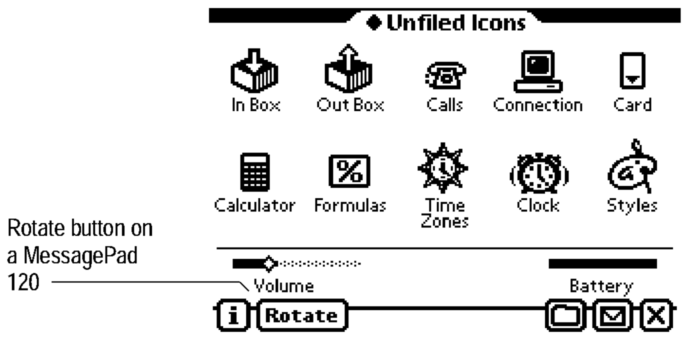

1. A user taps the Item Info button in a separator bar
2. The Item Info slip for the item below the bar is displayed
1:08 Tue 3/12
◆ All Notes
Mahatma Gandhi
Quotes
Title Mahatma Gandhi
Note
Date: 9:26 am Fri 1/21/94
Size: 232 bytes
Where: Pink Card
you are making other

If a user scrolls an item's separator bar out of view while its Item Info slip is displayed, the Item Info slip closes automatically and does not reopen automatically if the user scrolls the separator bar back into view. To see the Item Info slip again, the user must tap the Item Info button.

## Rotate Button

On Newton devices that can change the orientation of the display, the Rotate button enables users to control which way the display faces. The effect of tapping a Rotate button depends on the number of available orientations. If a device has two orientations—regular and sideways like a MessagePad 120—then tapping a Rotate button changes between the two orientations. If a device has more than two orientations, then tapping a Rotate button presents a list of possible orientations. Figure 3-33 illustrates a Rotate button on a MessagePad 120.

**Figure 3-33** A Rotate button lets users change the screen orientation



C H A P T E R 4

# Pickers

A **picker** is a black-bordered, unmovable view that pops up in response to a user action, such as tapping a button, label, or hot spot. A picker contains a set of items such as commands, attributes, states, or application data. The items may be presented as a simple list, a table with rows and columns, a numerical counter, or a calendar, and those formats can be mixed in a single picker. Users can choose an item or merely browse the picker. Most pickers close immediately when a user chooses an item, but the more elaborate pickers have a Close box that a user taps after choosing an item. Also, most types of pickers close if a user taps outside them.

This chapter details the appearance and behavior of the following types of Newton pickers:

- List pickers
- Number pickers
- Date and time pickers
- Data overview pickers

This chapter ends with brief descriptions of prefabricated Newton pickers that pop up in many applications: the Info, New, Show, Action, and People pickers.

## List Pickers

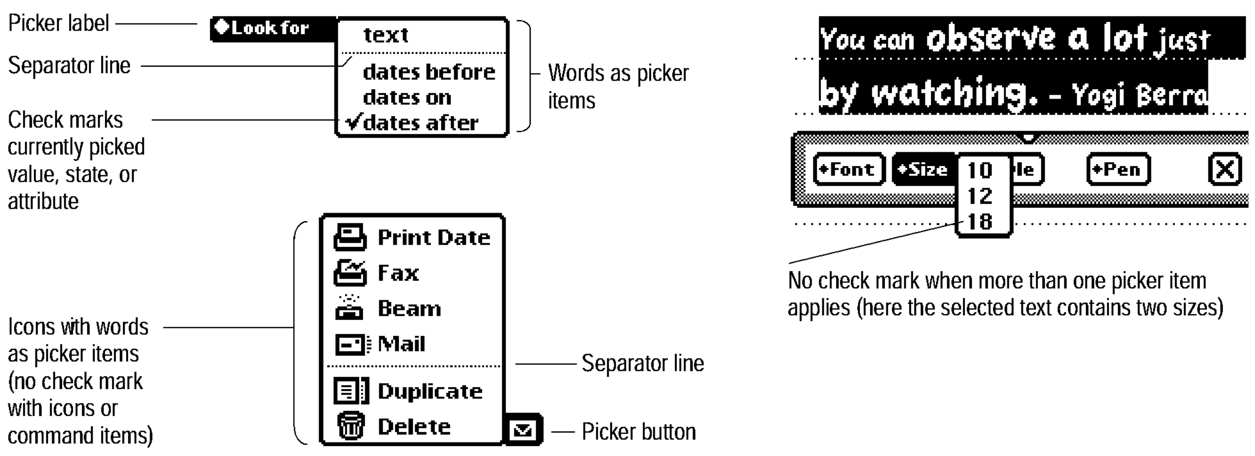
As its name suggests, a list picker presents users with a list of items from which to choose. This section describes the following aspects of list pickers:

- what list pickers can contain
- how the items can be organized
- where list pickers can pop up
- how people use list pickers

### Elements of List Pickers

A list picker includes words or icons (bitmap pictures) that name picker items. The list may contain marks next to picker items, and separator lines (which are not pickable) between groups of items. Additionally, some of the items can be disabled (not pickable). Figure 4-1 points out features of list pickers.

**Figure 4-1** The parts of list pickers


Picker label
Separator line
Check marks currently picked value, state, or attribute
Look for
text
dates before
dates on
✓dates after
Words as picker items
You can observe a lot just by watching. – Yogi Berra
Font
Size
10
12
18
le
Pen
No check mark when more than one picker item applies (here the selected text contains two sizes)
Icons with words as picker items (no check mark with icons or command items)
Print Date
Fax
Beam
Mail
Duplicate
Delete
Separator line
Picker button

A list picker does not include a title because the picker's context should make its purpose clear. The picker may contain scroll arrows, a Close box, and other controls as described in "Using a List Picker" on page 4-9.

## Check Marks

A check mark (✓) has special meaning in a list picker. In a picker that lists attributes, values, or states, a check mark indicates which picker item is in effect. A check mark is not used in a list of commands because no command is in effect until the user picks one. Also, check marks are not shown for picker items with an icon.

## Icons

Icons are completely optional in list pickers; in fact, there are several reasons to avoid them. Use icons to clarify and distinguish the wording of picker items, but not purely for decoration. For a discussion of the pros and cons of icons in list pickers, see "Icons in a Picker" on page 5-12.

## Item Names

In a list picker, use one word for item names when possible, and capitalize it. When you must use more than one word, you should generally capitalize just the first word. You can capitalize differently if it makes sense in a particular context. For example, the items in the Date Book's Show picker are titles of views and are capitalized like book titles. If the text of a picker item ends a sentence begun by a picker label, you should capitalize the label like a sentence and not capitalize the picker items at all (except for proper nouns). The Find slip's Look For picker is an example of sentence capitalization.

Avoid punctuation and symbols in list pickers. Except for very common symbols such as an ampersand (&), users find symbols ambiguous. In particular, do not put an ellipsis (...) at the end of a picker item that will bring up a slip. Unlike menus on personal computers, Newton pickers do not use an ellipsis or any other symbol to indicate a command that will need more information from the user.

You use different parts of speech to name items in a list picker, depending on what effect they have when the user picks one. For picker items that act as commands, use verbs (or verb phrases) that declare the action that will occur when the user picks the item. For example, Duplicate means "Duplicate the current data item," and Fax means "Fax the current data item." Your picker command names should fit into a similar sentence.

In a list picker that lists several actions (in the form of verbs), include an object of the action (in the form of a noun) with the first item. Subsequent items that refer to the same object need only list the action; they don't need to repeat the object. For example, start with Print Note and follow up with Fax, Beam, and Mail (where "Note" is understood in all but the first item).

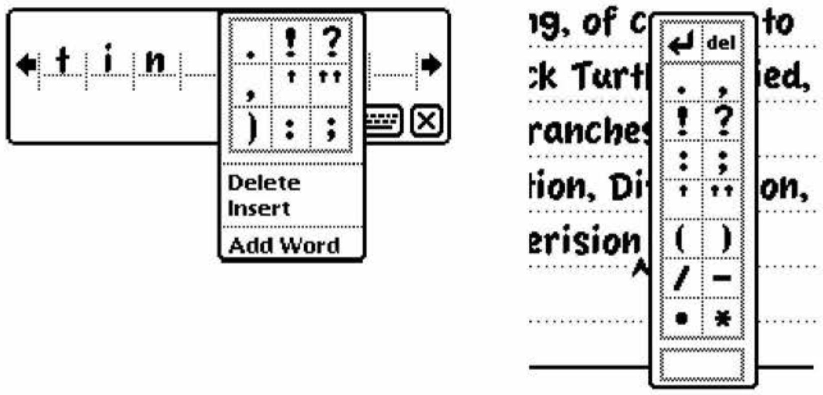
If a picker item changes an attribute or a state, use a word or phrase that describes the change. Descriptive words (nouns and adjectives) in pickers imply an action. They should fit into the sentence "Change to . . . " or "Make this . . . ". For instance, while picking a label for a phone number in the Names File, a user might think, "Make this phone number the Home number." A user who is about to change the view in the Names File might think, "Change to the All Info view."

List pickers display items in the bold style of the system font. On an Apple MessagePad, the item names are 10-point text.

## Table of Items

A list picker can include a two-dimensional table of items with any number of rows and columns. The table can contain anything that can be represented by a bitmap picture. (The entire table is actually implemented as one bitmap picture, complete with the border between cells and around the table.) Figure 4-2 shows a list picker that contains a table of items.

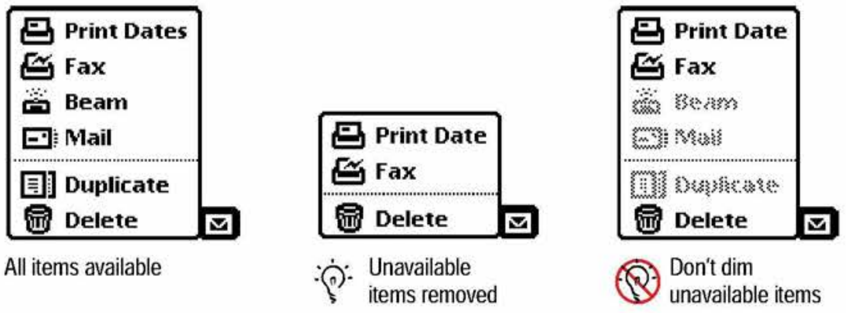
**Figure 4-2** A list picker can contain a two-dimensional table of items



## Unavailable Items

An application may need to make some of a list picker's items available only in certain contexts. To make items unavailable, an application should remove them from the picker. For example, the Date Book application removes Beam and Mail from its Action picker under some circumstances. Figure 4-3 shows how to make picker items unavailable.

**Figure 4-3** Remove unavailable items from a list picker



Applications should not attempt to imitate the interface of personal computers by dimming unavailable picker items. Although applications can designate picker items as unselectable, the system does not display them in gray text or otherwise make them visibly different from selectable items. Newton picker items should simply disappear when they are unavailable.

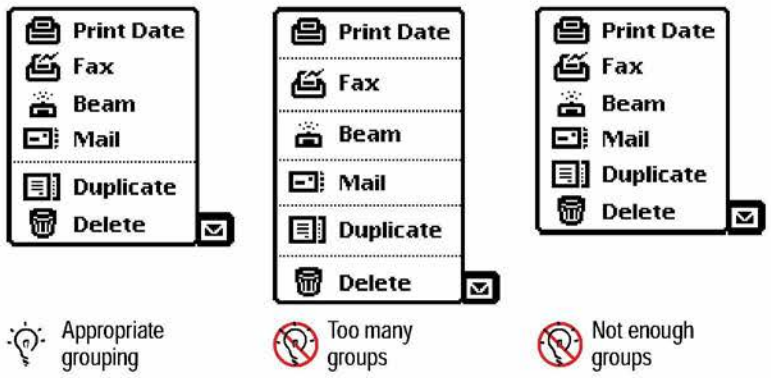
## Organization of List Pickers

The items in a list picker should be logically related to each other and to the label, button, or whatever else controls the picker. Arrange the items in a list picker in an order that makes sense and is most convenient to users. In general, start the list with the items most likely to be picked and end the list with the items least likely to be picked. If all items are equally likely to be picked, or if you want to arrange them without prejudice or bias, list them alphabetically.

You can organize a list picker visually, making it easier to locate an item by grouping related items. In a picker that contains several types of items—actions, attributes, values, and states—group the items according to type. In a picker that contains a single type of item, look for another criterion. For instance, the Printer picker in a Print routing slip lists specific printers in one group and commands that access other printers in a second group. The Look For picker in the Find slip has two groups: one for finding text and another for dates.

You put a separator line between groups of items in a list picker. How many separator lines to use is partially an aesthetic decision. Remember that separator lines take up screen space and that the Newton interface relies on aesthetic integrity as a means of good communication. Figure 4-4 contrasts a good balance of grouping with too little grouping and too much grouping.

**Figure 4-4** Grouping items in list pickers

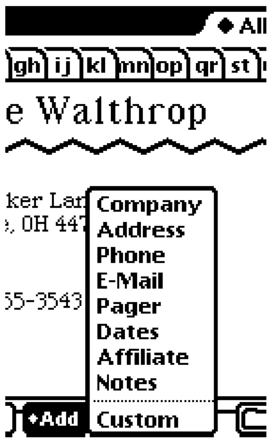


For general grouping of items in a picker, you should only use a dotted separator line, never a solid separator line. The solid separator line is reserved for setting apart choices related to storage, such as the names of available card stores and the internal store, at the bottom of a picker (see "Folder Tab" on page 8-19).

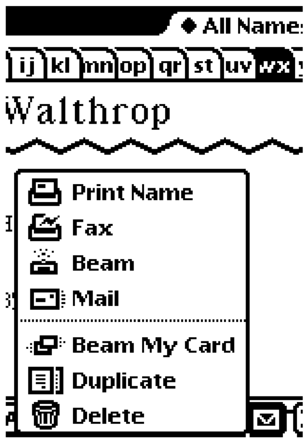
## Sources of List Pickers

A list picker can pop up from a text button or text label marked with a black diamond (◆). In addition, a picker can pop up from a picture button or a hot spot, but for aesthetic reasons picture buttons and hot spots are not marked with black diamonds. Figure 4-5 shows pickers from those four sources.

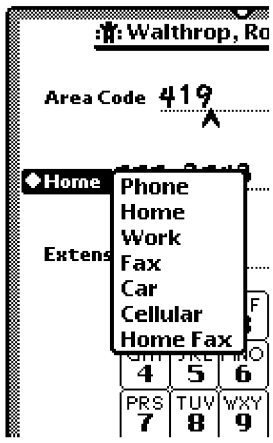
**Figure 4-5** Pickers can pop up from buttons, labels, and hot spots



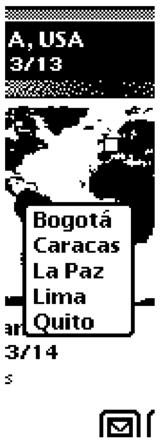
Picker from a text button



Picker from a picture button



Picker from a label



Picker from a hot spot

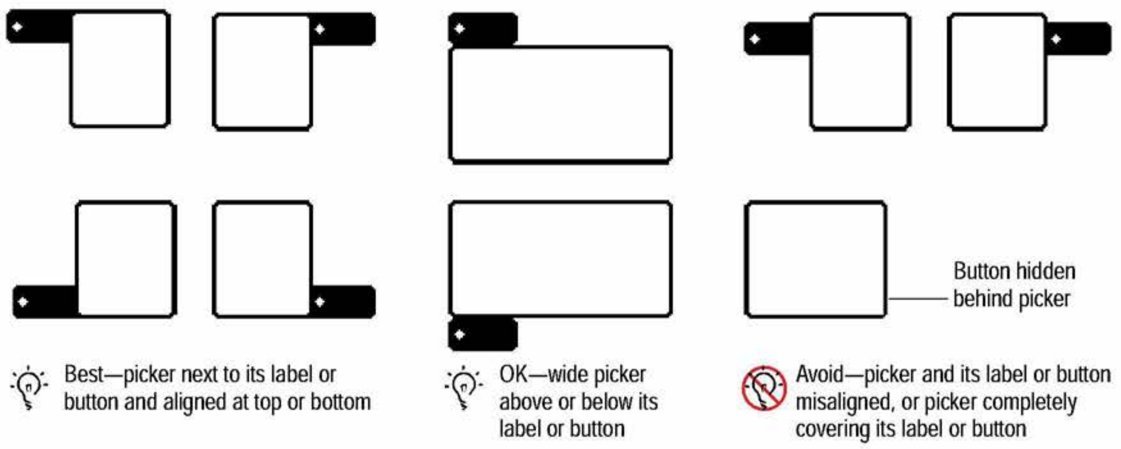
For picker control at the bottom of a view or on the status bar, use text or picture buttons. Elsewhere in a view, label pickers usually look best.

## Position of List Pickers

A list picker that pops up from a button or text label should appear next to the label or button but should not completely cover the label or button, so users can see where the picker comes from. If a picker is too wide to fit in the space next to the picker label or button, the picker should appear below (or above) and flush with the label or button.

If possible, a list picker should line up with the top edge of the label or button that makes it appear. If top alignment would cause the picker to extend past the bottom of the screen, then the picker should line up with the bottom of the label or button that makes it appear. A long picker (or a short screen) may require aligning the top or bottom of the picker to the top or bottom of the screen. Figure 4-6 illustrates proper and improper alignment of list pickers and the labels or buttons that make them appear.

**Figure 4-6** How a list picker should align with its label or button


Button hidden behind picker
Best—picker next to its label or button and aligned at top or bottom
OK—wide picker above or below its label or button
Avoid—picker and its label or button misaligned, or picker completely covering its label or button

If you want your application to work when a user rotates the display (with the Extras Drawer's Rotate button), your application may need to make picker alignment dependent on screen size.

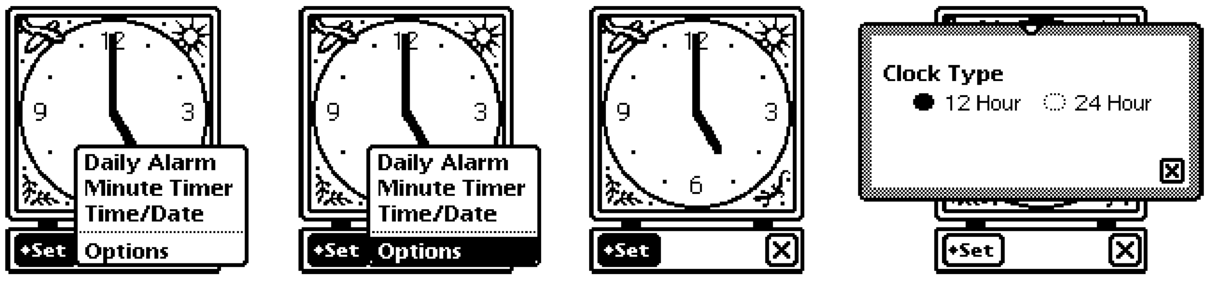
## Using a List Picker

A user makes a list picker pop up by tapping the button or label that controls it. While the picker is open, the application highlights the picker's controlling button or label. Nothing else happens until the user touches the screen again. The user does not have to press and hold the pen on the button or label to keep the picker open; the picker stays open until the user touches the screen again. If the user touches the screen outside the list, the picker goes away.

### Picking an Item

A user picks an item listed in a list picker by tapping the item. The picked item blinks briefly, the picker disappears, and the action or effect associated with the picked item happens. Figure 4-7 shows the sequence of events for the Set button in the built-in Clock application.

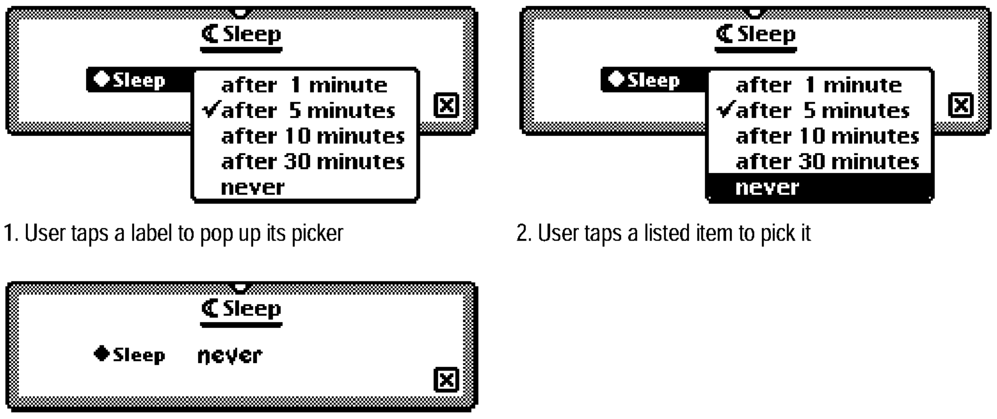
**Figure 4-7** Using a list picker from a button



1. User taps button to pop up its picker
2. User taps a listed item to pick it
3. Picker disappears but button stays highlighted until…
4. Picked item takes effect

In the case of a list picker that pops up next to a text label, the current value of the picker (the most recently picked item) is usually displayed next to the picker label. The label and the value are customarily aligned at their baselines. The value should be displayed in the casual font, not in the system font like the picker label and picker items. Figure 4-8 shows the sequence of events with the label picker in the Sleep preferences slip.

**Figure 4-8** Using a list picker from a label



1. User taps a label to pop up its picker
2. User taps a listed item to pick it
3. Picker disappears and picked item appears next to picker label

If a user touches a picker list and slides the pen instead of lifting it, the picker tracks the pen movement. As the pen appears over an item in the list, the item is highlighted. When the user lifts the pen within the list, the currently highlighted item blinks briefly, the picker disappears, and the effect associated with the picked item happens. If the user slides the pen outside the list, so that no item is highlighted, and then lifts the pen, the picker simply goes away. The display of electronic ink is turned off while the pen is tracked.

An item that can't be selected, such as a separator line, is not highlighted when a user taps it or slides the pen over it. Tapping an unselectable item or lifting the pen while it is over an unselectable item makes the picker go away.

After the user picks an item and the picker disappears, your application must continue to highlight the label or button that controls the picker until the action or effect associated with the picked item is complete. In the case of a picker item that displays an ordinary slip (not a status slip), the picker's controlling label or button stays highlighted only until the slip appears. Keeping the label or button highlighted provides the minimal feedback to the user that the application is still working. When a process begun by a picker item takes more than a few seconds, your application should provide more feedback by displaying a status slip that names the process underway (as described in "Status Slips" on page 2-20).

## User Editing of Pickers

If you want to allow users to be able to add, remove, and change items in a list picker, include an item "Edit" or "Edit Picker" at the bottom of the picker, with a separator line above it. Choosing this item should bring up a slip in which users can edit the picker. The slip should clearly indicate how many items the picker can contain. Title the slip to make its purpose clear, for example "Edit Category List."

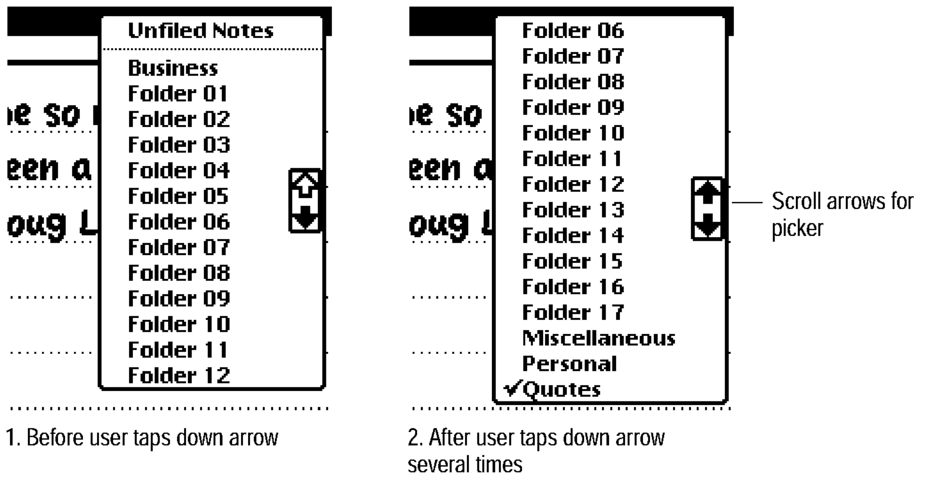
Instead of allowing users to edit picker items, an application could automatically include recent inputs for an input field in that field's picker. For information on input fields, see Chapter 6, "Data Input."

## Scrolling

A list picker may contain too many items to display at once on some Newton devices. This can happen when a user rotates the display (by tapping the Rotate button in the Extras Drawer). It can also happen if a user adds many items to a customizable picker, such as a folder picker.

When a list picker becomes too long to fit on the screen, the Newton operating system automatically reduces the picker's length and adds ordinary local scroll arrows to the picker. A user can tap the scroll arrows to bring more picker items into view. Figure 4-9 shows a scrolling folder picker.

**Figure 4-9** List pickers that are too long to display all at once have scroll arrows



As usual, the color of the scroll arrow indicates whether tapping it will bring more items into view. An arrow is black if tapping it will bring more items into view. An arrow is white if tapping it will not bring more items into view.

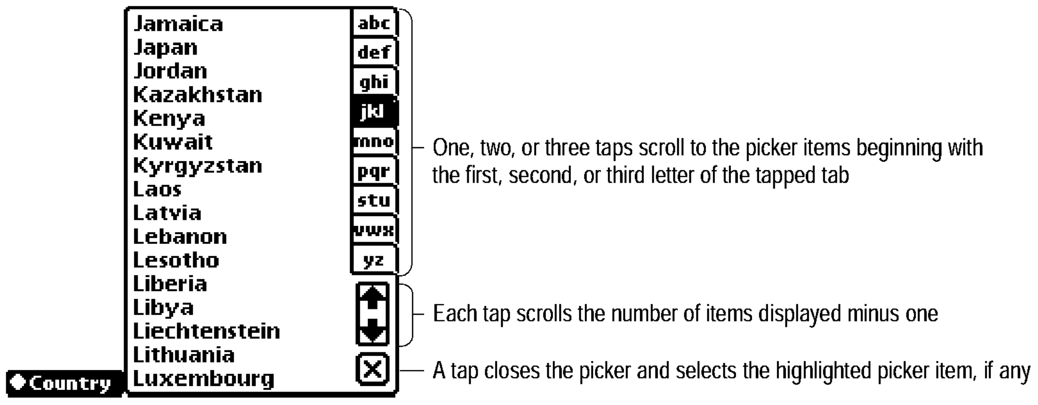
Users can also scroll list pickers by dragging from the middle of the picker past the top or bottom of the picker. Users cannot scroll a list picker with the universal scroll arrows. Tapping a universal scroll arrow or anywhere else outside a list picker makes the picker go away.

Scrolling pickers are harder to use than pickers that don't scroll, because users have to remember the picker items that aren't currently visible. You should keep your pickers short and avoid scrolling pickers in your applications.

## Index Tabs

If the list of picker items is very long, scrolling from one end to the other can be tedious. To give users a quicker method of traveling through a long list of picker items, you can augment scroll arrows with a series of small labeled index tabs displayed along the right edge of the picker. The tabs essentially divide the picker items into sections alphabetically, and a user taps a tab to see the section it identifies. Figure 4-10 illustrates picker scroll arrows and index tabs.

**Figure 4-10** A lengthy picker can include scroll arrows and index tabs


Jamaica
Japan
Jordan
Kazakhstan
Kenya
Kuwait
Kyrgyzstan
Laos
Latvia
Lebanon
Lesotho
Liberia
Libya
Liechtenstein
Lithuania
Luxembourg
Country
abc
def
ghi
jkl
mno
pqr
stu
vwx
yz
One, two, or three taps scroll to the picker items beginning with the first, second, or third letter of the tapped tab
Each tap scrolls the number of items displayed minus one
A tap closes the picker and selects the highlighted picker item, if any

Tapping an index tab scrolls to the picker items that begin with the first letter of the tapped tab. Double-tapping an index tab scrolls to the picker items that begin with the second letter of the tapped tab. Triple-tapping an index tab scrolls to the picker items that begin with the third letter of the tapped tab.

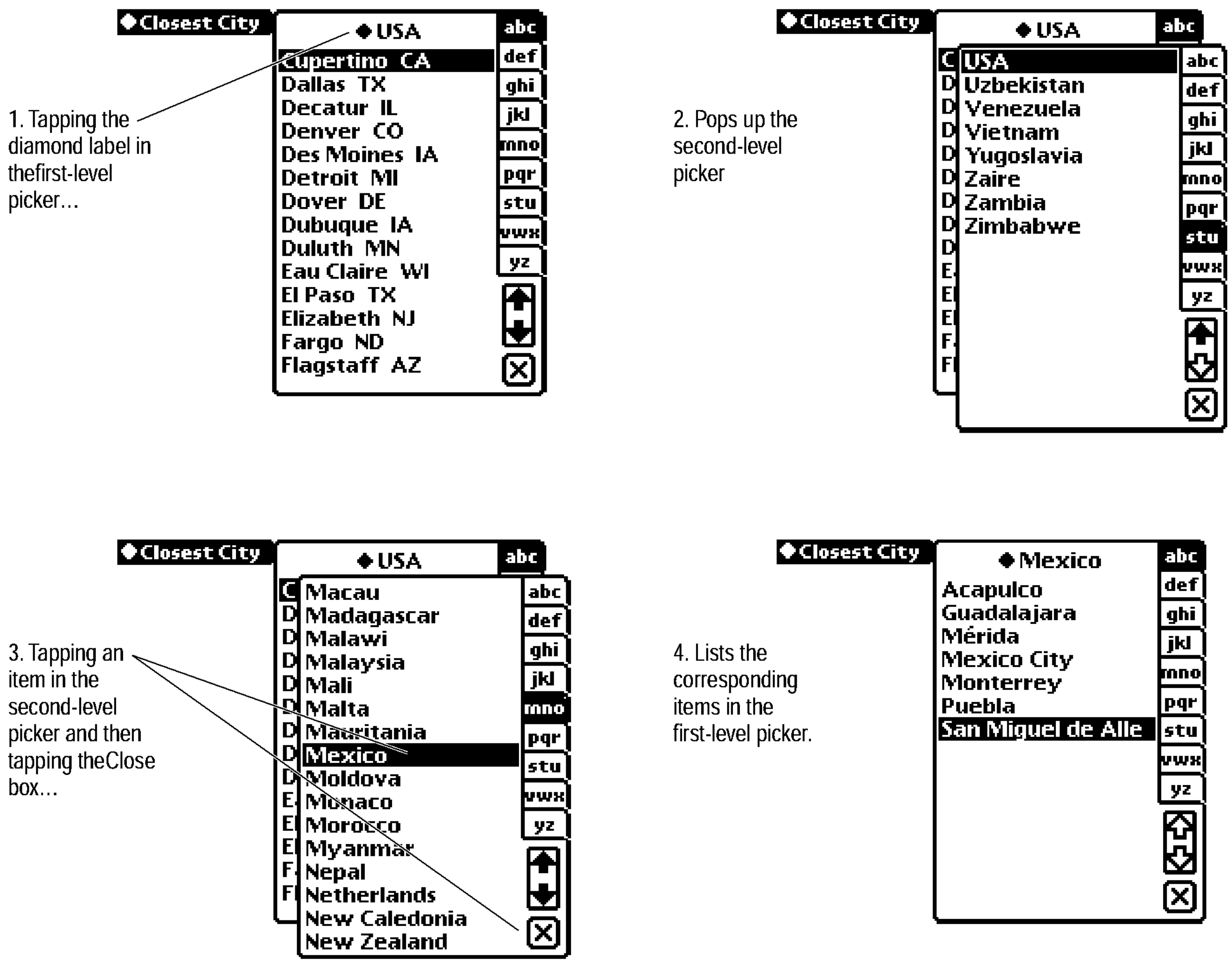
In a list picker with an index tab, tapping a listed item selects the item but doesn't close the picker. To close the picker and confirm the selection, a user taps the picker's Close box. To cancel the selection and close the picker, a user taps outside the picker.

## Hierarchical List Pickers

If a list of picker items is extremely long, index tabs won't be enough to prevent interminable scrolling. What happens is a user taps a tab and immediately sees the beginning of the corresponding section of picker items, but the user still must scroll several times to find an item that's not near the beginning of that section. For example, imagine one picker that lists several hundred cities from around the world.

The solution to this situation is to create a two-level hierarchy of list pickers. The first-level picker contains a set of picker items and a diamond label. Tapping the diamond label pops up a second-level picker that lists different sets of picker items for the first-level picker. Picking an item in the second-level picker determines which set of items appear in the first-level picker. For example, compared to a single picker that lists hundreds of cities, a two-level picker hierarchy simplifies picking a city anywhere in the world. The first-level picker lists cities for just one country and contains the name of the country as a diamond label. A user can pick one of the listed cities or tap the diamond label to pop up a second-level picker that lists other countries. If the user picks a country, then the first-level picker changes to list that country's cities. Figure 4-11 shows how the city and country hierarchical pickers work.

**Figure 4-11** How a two-level hierarchy of list pickers works



1. Tapping the diamond label in thefirst-level picker...

2. Pops up the second-level picker

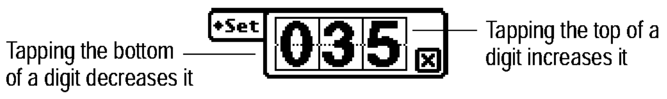
3. Tapping an item in the second-level picker and then tapping theClose box...

4. Lists the corresponding items in the first-level picker.

## Number Picker

A number picker displays a number that a user can change by tapping the digits of the number itself. The digits are large and are split into top and bottom halves to make them easy for users to target. Tapping the top half of a digit increases it, and tapping the bottom half of a digit decreases it. Designed initially to replicate the old "mechanical digital" alarm clocks, the look of the number picker evolved so that only the flipping digits remain. Figure 4-12 shows an example number picker.

**Figure 4-12** A number picker simplifies specifying a numerical value


Tapping the bottom of a digit decreases it
◆Set
035
Tapping the top of a digit increases it

Like a list picker, a number picker pops up when a user taps its label, which begins with a diamond. Because a user may have to tap several times to specify a number, a number picker has a Close box. Tapping the Close box confirms the specified number and makes the picker go away. A user can cancel and close the picker by tapping anywhere outside it.

Specialized number pickers for specifying dates and times are used in other pickers described in the next two sections.

## Date and Time Pickers

The system includes pickers for specifying a time, a date, a date and time, a start and stop time, a start and stop date, or a time offset. Each of these pickers pops up when a user taps its label, which begins with a diamond. Then the user can specify a date or time by tapping in the picker. The picker does not go away automatically when a user selects a date or time in it. Date and time pickers also contain a Close box, which the user must tap to confirm the selected time or date. The user can cancel the selection and close the picker by tapping anywhere outside the picker. Figure 4-13 and Figure 4-14 show several time and date pickers.

**Figure 4-13** Time pickers specify a time, a time range, or a time offset

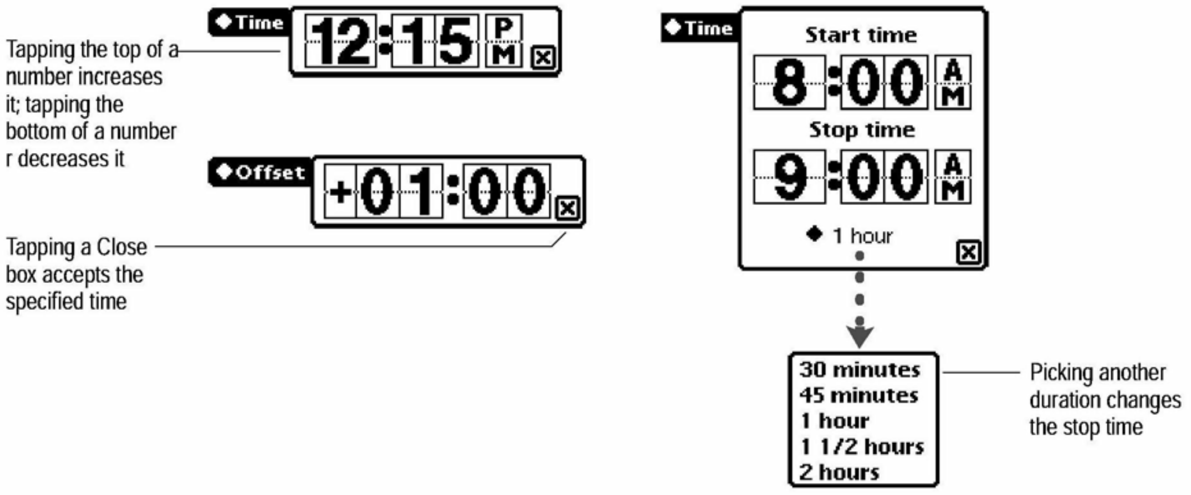
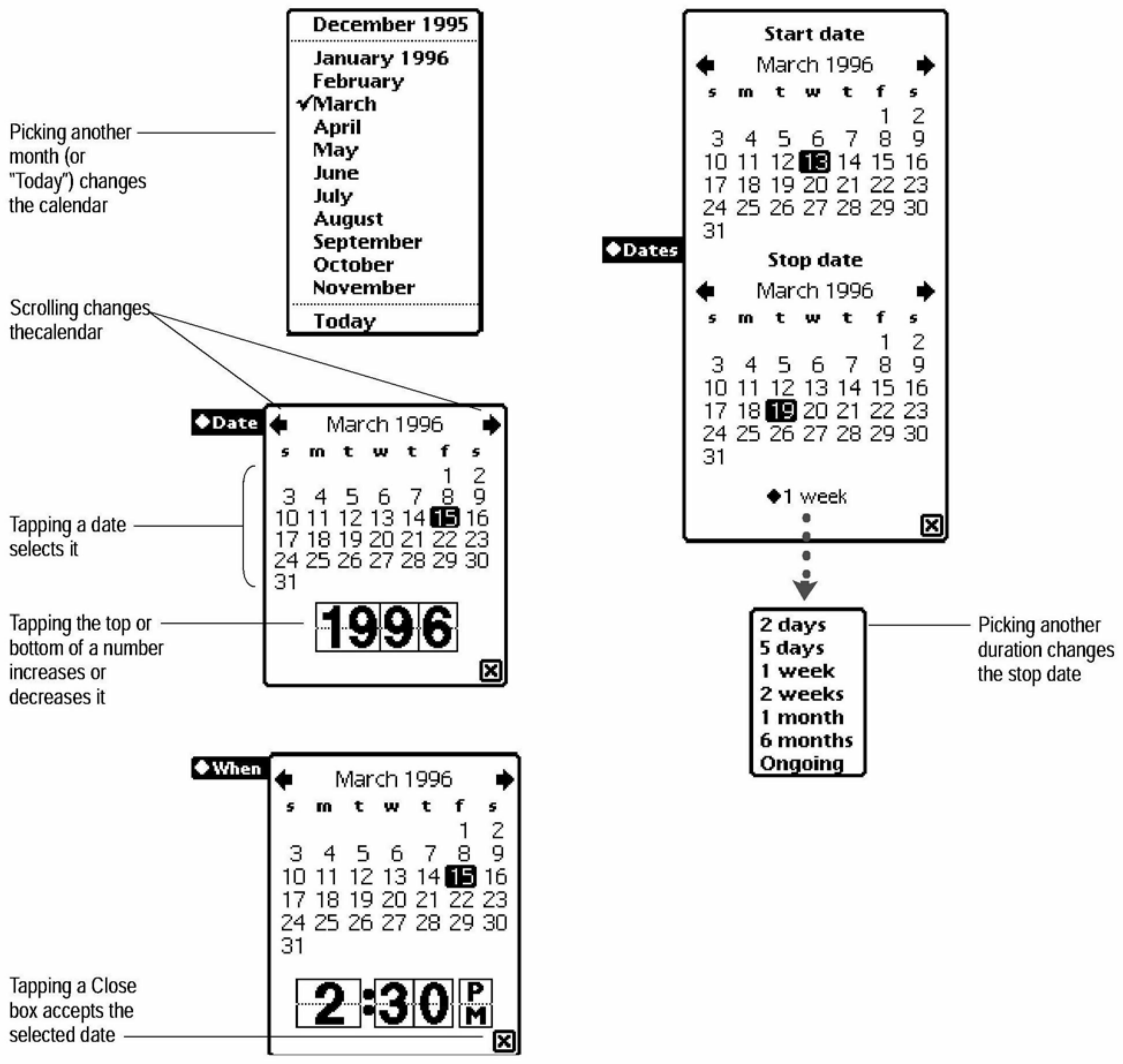

Tapping the top of a number increases it; tapping the bottom of a number r decreases it
Tapping a Close box accepts the specified time
◆Time
12:15 PM
◆Offset
+01:00
◆Time
Start time
8:00 AM
Stop time
9:00 AM
◆ 1 hour
30 minutes
45 minutes
1 hour
1 1/2 hours
2 hours
Picking another duration changes the stop time

**Figure 4-14** Date pickers specify one date or a date range


December 1995
January 1996
February
✓March
April
May
June
July
August
September
October
November
Today
Picking another month (or "Today") changes the calendar
Scrolling changes thecalendar
◆Date
March 1996
Tapping a date selects it
Tapping the top or bottom of a number increases or decreases it
1996
◆When
March 1996
2:30 PM
Tapping a Close box accepts the selected date
Start date
March 1996
◆Dates
Stop date
March 1996
◆1 week
2 days
5 days
1 week
2 weeks
1 month
6 months
Ongoing
Picking another duration changes the stop date

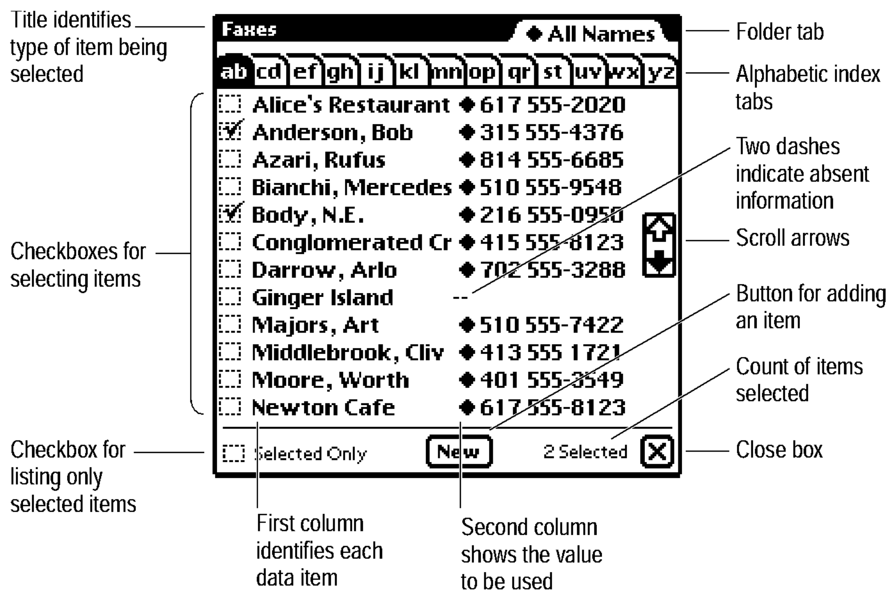
## Overview Pickers

Like list pickers, overview pickers can pop up in response to a user tapping a text label or button marked with a black diamond, a picture button, or a hot spot. And overview pickers, like list pickers, are used to present a user with items from which to choose. That's about where the resemblance between the two types of pickers ends. For example, list pickers allow a user to select only one item, but overview pickers generally allow a user to select one or many items. The following sections describe overview pickers in detail.

### Contents of Overview Pickers

An overview picker presents a one- or two-column extract of stored data, such as names and phone numbers from the Names File data. The first column lists the identities of the individual data items that a user can select, and the second column provides additional information that a user can edit in place. A title in the upper left corner identifies the type of data in the picker. An overview picker can include a number of controls for finding specific entries, including a folder tab, alphabetic index tabs, and scroll arrows. At the bottom an overview picker can have a checkbox for restricting listed items to those currently selected, a New button for adding items to the list, a counter that reports the number of items selected, and a large Close box. Figure 4-15 points out features of overview pickers.

**Figure 4-15** The parts of overview pickers


Faxes ◆ All Names

ab cd ef gh ij kl mn op qr st uv wx yz

| | | |
|---|---|---|
| ☐ | Alice's Restaurant | ◆617 555-2020 |
| ✓ | Anderson, Bob | ◆315 555-4376 |
| ☐ | Azari, Rufus | ◆814 555-6685 |
| ☐ | Bianchi, Mercedes | ◆510 555-9548 |
| ✓ | Body, N.E. | ◆216 555-0950 |
| ☐ | Conglomerated Cr | ◆415 555-8123 |
| ☐ | Darrow, Arlo | ◆702 555-3288 |
| ☐ | Ginger Island | -- |
| ☐ | Majors, Art | ◆510 555-7422 |
| ☐ | Middlebrook, Cliv | ◆413 555 1721 |
| ☐ | Moore, Worth | ◆401 555-3549 |
| ☐ | Newton Cafe | ◆617 555-8123 |

☐ Selected Only New 2 Selected ☒

Title identifies type of item being selected
Checkboxes for selecting items
Checkbox for listing only selected items
First column identifies each data item
Second column shows the value to be used
Folder tab
Alphabetic index tabs
Two dashes indicate absent information
Scroll arrows
Button for adding an item
Count of items selected
Close box

In most cases, your application is not responsible for the wording, punctuation, or capitalization of items in either column of an overview picker. Nor is your application responsible for the order of the items. The items are generally taken directly from stored data.

Overview pickers display items in the bold style of the system font. On an Apple MessagePad, the item names are 10-point text.

## Position of Overview Pickers

The conventional position for an overview picker is centered at the bottom of the screen. Overview pickers are large, so your application should not attempt to position one near the diamond label or other control that makes it appear. The usual size of an overview picker is 234 × 231 pixels.

## Using an Overview Picker

A user makes an overview picker appear by tapping the appropriate label. The picker stays open until the user taps its Close box. The user does not have to press and hold the pen on the button or label to keep the picker open. An overview picker stays open even if a user taps, writes, or draws outside the picker.

### Picking Items

A user selects items listed in an overview picker by tapping them. A selected item has a check mark in its checkbox. Tapping a selected item deselects it. For selecting or deselecting, tapping an item's name has the same effect as tapping its checkbox.

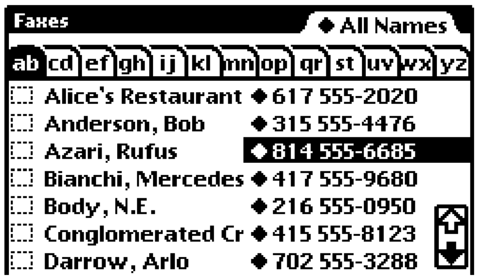
The picker reports the number of selected items at its bottom right corner, and the user can preview the set of selected items by tapping the Selected Only checkbox at the bottom left of the picker. An application can suppress the count of selected items, the Selected Only checkbox, or both in any of its overview pickers.

If a value in the right column of an overview picker begins with a black diamond, tapping it pops up a list picker that contains alternate values from the stored data plus a command for entering a new value. Figure 4-16 shows how this works in an overview picker that lists names and phone numbers.

If a user selects an item for which there is no value displayed in the right column of an overview picker, a slip appears in which the user can enter the needed information.

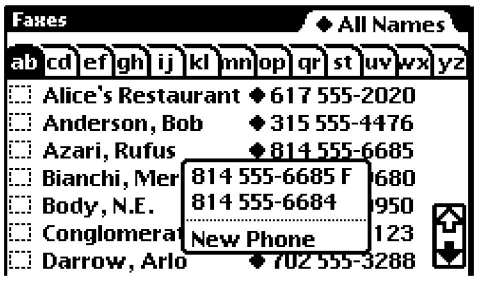
An overview picker can be set up to only allow selecting one item. In this case selecting an item automatically deselects the previously selected item. Setting up an overview picker for only one selection does not change the way the picker looks (the checkbox next to each item does not change to a radio button).

**Figure 4-16** Entering a new value in an overview picker

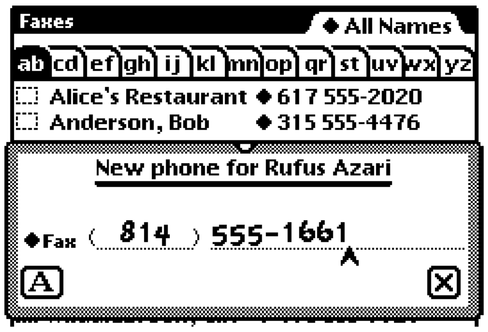


| | |
|---|---|
| Alice's Restaurant | ◆617 555-2020 |
| Anderson, Bob | ◆315 555-4476 |
| Azari, Rufus | ◆814 555-6685 |
| Bianchi, Mercedes | ◆417 555-9680 |
| Body, N.E. | ◆216 555-0950 |
| Conglomerated Cr | ◆415 555-8123 |
| Darrow, Arlo | ◆702 555-3288 |

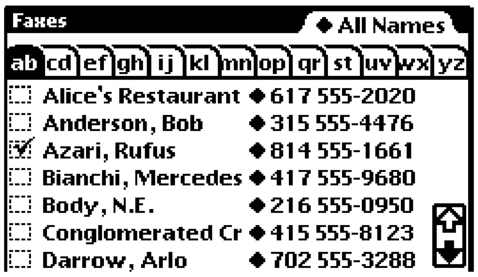
1. A user taps in the second column where a diamond indicates a list picker will pop up


Faxes
◆All Names
ab cd ef gh ij kl mn op qr st uv wx yz
Alice's Restaurant ◆617 555-2020
Anderson, Bob ◆315 555-4476
Azari, Rufus ◆814 555-6685
Bianchi, Mer
Body, N.E.
Conglomerat
Darrow, Arlo ◆702 555-3288
814 555-6685 F
814 555-6684
New Phone

2. The user picks the New command


Faxes
◆All Names
ab cd ef gh ij kl mn op qr st uv wx yz
Alice's Restaurant ◆617 555-2020
Anderson, Bob ◆315 555-4476
New phone for Rufus Azari
◆Fax ( 814 ) 555-1661

3. The user enters a new value in a slip and then taps the slip's Close box



| | |
|---|---|
| Alice's Restaurant | ◆617 555-2020 |
| Anderson, Bob | ◆315 555-4476 |
| ✓ Azari, Rufus | ◆814 555-1661 |
| Bianchi, Mercedes | ◆417 555-9680 |
| Body, N.E. | ◆216 555-0950 |
| Conglomerated Cr | ◆415 555-8123 |
| Darrow, Arlo | ◆702 555-3288 |

4. The new value appears in the overview picker and the item is selected

When a user closes an overview picker, the selected item or items are customarily displayed next to the picker label. The label and the value are customarily aligned at their baselines. The value should be displayed in the casual font, not in the system font like the picker label and picker items.

## Scrolling Items

Most overview pickers list more items than can be displayed at once. Users can see more items by using the scroll arrows in the picker (unless the application has suppressed them). The color of the scroll arrow indicates whether tapping it will bring more items into view. An arrow is black if tapping it will bring more items into view. An arrow is white if tapping it will not bring more items into view.

Users can also scroll overview pickers with the universal scroll arrows. In addition, users can scroll overview pickers by dragging from the middle of the picker past the top or bottom of the picker.

### Creating New Items

When the item a user wants is not included in an overview picker, the user doesn't have to close the picker and go to another application to create the item. Users can create entirely new items without leaving an overview picker that has a New button. (If you don't want users to be able to create new items from within an overview picker, you can suppress the picker's New button.)

To create a new item, a user taps the New button at the bottom of the overview picker. A slip appears in which the user enters just the information needed for the picker. The new item is added to the picker and to the other information in Newton storage. For example, tapping the New button in an overview picker that lists names and fax numbers would bring up a slip in which the user enters a first name, last name, and fax number. The name and fax number would be added to the overview picker and to the Names File data. Later the user could use the Names File application to fill in additional information for the new person.

## Standard Newton Pickers

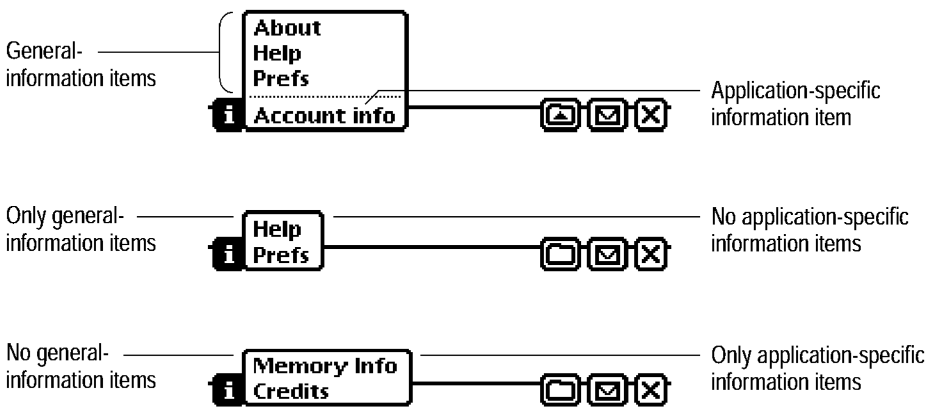
A typical application has some of the following standard Newton pickers pop up from buttons on its status bar or on separator bars: the Info picker, New picker, Show picker, Action picker, and People picker. This section describes all of those standard Newton pickers.

Additional pickers defined by the Newton system are described in other parts of this book. The Keyboard, Recognition, and Alpha Sorter pickers are described in Chapter 6, "Data Input." The Action picker is described in Chapter 7, "Routing and Communications." The Filing, Folder, Find, and Assist pickers are described in Chapter 8, "Newton Services."

## Info Picker

The Info picker pops up from the standard Info button at the left end of the status bar and gives users access to preference settings for the application, general information about the application, or on-screen help for the application. The Info picker contain any of these general-information items: About, Help, and Prefs. The Info picker may also contain additional items unique to the application. Figure 4-17 shows several example Info pickers.

**Figure 4-17** An Info picker lists information items


General-information items
About
Help
Prefs
Account info
Application-specific information item
Only general-information items
Help
Prefs
No application-specific information items
No general-information items
Memory Info
Credits
Only application-specific information items

If you do not have all three general-information items—About, Help and Prefs—keep the ordering listed, but leave out the unused items. List any application-specific items below a separator line. If your application does not have any unique items in its Info picker, or if it has only unique items (not About, Help, or Prefs), then do not include the separator line. Any unique items you include in the Info picker should specify a state or initiate an action that affects the whole application.

An About box should include your application's name, version number, and copyright information. You can include additional information in the About box if you wish, such as a logo, credits, and acknowledgments. Include a Close box so users can put the About box away after reading it. Use a conventional matte border.

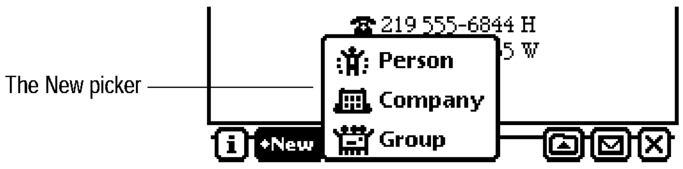
Choosing Help from an Info picker displays online help for the application. For more information, see "Help" on page 8-28.

Choosing Prefs from an Info picker displays a slip containing application-specific preference settings. For more information, see "Application Preferences" on page 8-31.

## New Picker

The New picker lists the types of data items that can be created in the currently active application, such as a new note, checklist, or outline in the built-in Notepad application. In an application that supports Newton stationery, the New picker lists all the data structures defined for the application. For example, the built-in Names File's New picker lists Person, Company, and Group data structures. The New picker pops up from the standard New button on the left side of the status bar, next to the keyboard button. Figure 4-18 shows the New picker for the Names File application.

**Figure 4-18** The New picker lists types of data items that users can create

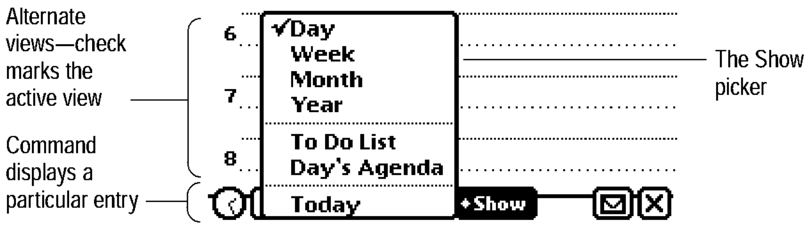


Picking an item from a New picker creates a new data item of the type picked. If the currently active application displays multiple data items one after another in a paper roll fashion like the built-in Notepad, then the New picker creates an item after the last item in the current view and automatically scrolls it into view.

## Show Picker

The Show picker lists alternative views for displaying data in an application, such as the Card view and All Info view in the built-in Names File application. In an application that supports Newton stationery, the Show picker lists all the available views for types of data that the application uses. The active view has a check mark next to its name in the Show picker. In addition to alternate views, a Show picker can list commands that display a particular data item in the currently selected view. For example, the built-in Date Book application's Show picker lists Today, which is not an alternate view but is a command to display the calendar data for today. Figure 4-19 illustrates the Show picker for the Date Book.

**Figure 4-19** The Show picker lists alternate ways to see an application's data

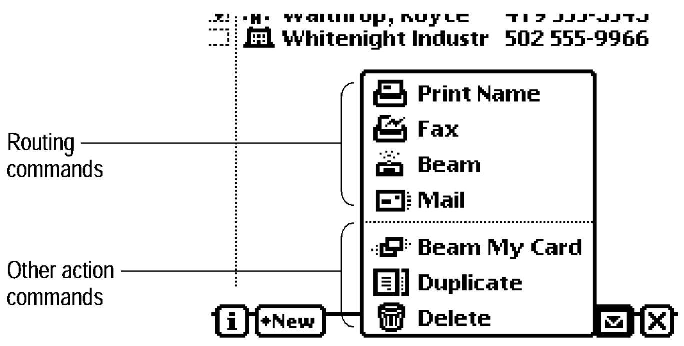

Alternate views—check marks the active view
Command displays a particular entry
6
7
8
✓Day
Week
Month
Year
To Do List
Day's Agenda
Today
◆Show
The Show picker

Picking a view or a command from a Show picker changes the view.

## Action Picker

The Action picker lists commands for sending and receiving data by communications methods, such as printing, faxing, beaming, and e-mailing. In many applications the Action picker includes additional commands for acting on data that's currently displayed. A separator line divides the routing commands from the other action commands. Figure 4-20 shows an example Action picker.

**Figure 4-20** The Action picker lists commands for acting on data

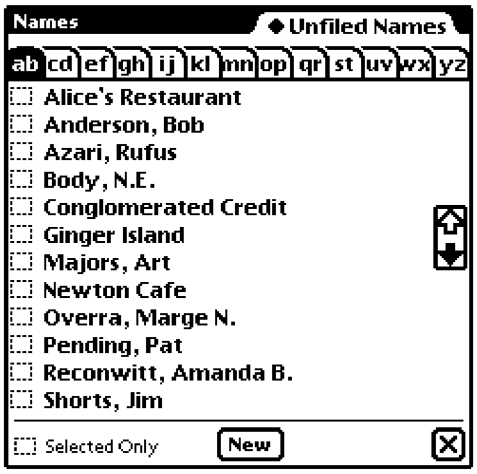


Picking a routing command from an Action picker starts the routing process, which is detailed in Chapter 7, “Routing and Communications.” Picking the Duplicate command, if the Action picker includes one, makes an exact copy of the data item or items affected by the Action picker. Picking the Delete command, if the Action picker includes one, brings up a confirmation alert asking the user to authorize deleting the data item or items affected by the Action picker. The function of other commands listed below the separator line in an Action picker are determined by the application that adds them there.

## People Picker

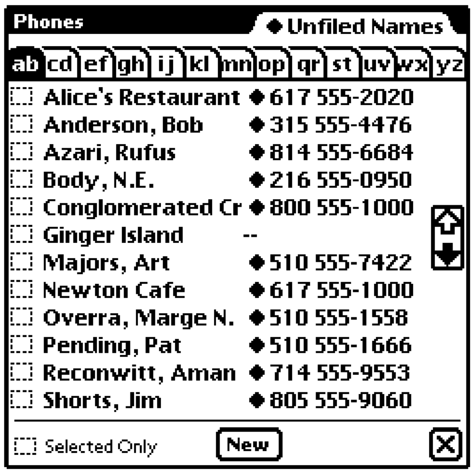
The People picker is an overview picker that contains names of people and companies from the Names File and Owner Info applications. For each name, a People picker can also include a phone number, fax number, or e-mail address. The controls and behavior of a People picker are the same as any ordinary overview picker (see “Overview Pickers” on page 4-19). Figure 4-21 shows a sample People picker.

**Figure 4-21** A People picker excerpts items from the Names File and Owner Info applications



Alice's Restaurant
Anderson, Bob
Azari, Rufus
Body, N.E.
Conglomerated Credit
Ginger Island
Majors, Art
Newton Cafe
Overra, Marge N.
Pending, Pat
Reconwitt, Amanda B.
Shorts, Jim

Names only



| | |
|---|---|
| Alice's Restaurant | ◆617 555-2020 |
| Anderson, Bob | ◆315 555-4476 |
| Azari, Rufus | ◆814 555-6684 |
| Body, N.E. | ◆216 555-0950 |
| Conglomerated Cr | ◆800 555-1000 |
| Ginger Island | -- |
| Majors, Art | ◆510 555-7422 |
| Newton Cafe | ◆617 555-1000 |
| Overra, Marge N. | ◆510 555-1558 |
| Pending, Pat | ◆510 555-1666 |
| Reconwitt, Aman | ◆714 555-9553 |
| Shorts, Jim | ◆805 555-9060 |

Names and associated data