Addison-Wesley Publishing Company

CAROL KAEHLER

HyperCard™ Power

Techniques and Scripts

Foreword by Bill Atkinson



Computers
H
HIHA
1795
902

HyperCard™ Power:

Techniques and Scripts

HyperCard™ Power: Techniques and Scripts

Carol Kaehler

Foreword by Bill Atkinson

Illustrated by Kristee Kreitman and Marge Boots

Addison-Wesley Publishing Company, Inc.

Reading, Massachusetts Menlo Park, California New York
Don Mills, Ontario Wokingham, England Amsterdam Bonn
Sydney Singapore Tokyo Madrid Bogotá Santiago San Juan

Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this book and Addison-Wesley was aware of a trademark claim, the designations have been printed in initial capital letters (i.e., MacPaint).

HyperCard is a trademark of Apple Computer, Inc.

The representations, opinions and views stated herein are solely those of the author in her individual capacity and do not represent and are not intended to represent the opinions and views of Apple Computer, Inc.

Library of Congress Cataloging-in-Publication Data

Kaehler, Carol.
HyperCard power.

On t.p. the registered trademark symbol "TM" is superscript following "HyperCard" in the title.
Includes index.
1. Macintosh (Computer)—Programming. 2. HyperCard (Computer program) I. Title.
QA76.8.M3K34 1988 005.36'5 87-30664
ISBN 0-201-06701-3

Copyright © 1988 by Addison-Wesley Publishing Company, Inc.
All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior written permission of the publisher.
Printed in the United States of America. Published simultaneously in Canada.

Technical Reviewer: Louella Pizzuti
Cover design by Doliber Skeffington
Text design by Christy Rosso
Set in 12-point Sabon by Compset, Inc.

ISBN 0-201-06701-3
ABCDEFGHIJ-HA-898
First printing, March, 1988

for Ted

Contents

	Foreword	xiii
	Preface	xx
	Acknowledgments	xxvi
Introduction	**HyperCard: A Visual Information Organizer**	1
	Creativity and Sharing Information	
	A Common Currency	
	Who This Book Is For	
	From User to Author: The Continuum	
	How to Use This Book	
	How This Book is Organized	
	Before You Start	
Part One	**Using HyperCard**	11
Chapter 1	Browsing HyperCard	13
	Meet HyperCard!	
	Stacks: Ordered Collections of Cards	
	Using the Go Menu to Get Around	
	Using the Keyboard to Navigate in HyperCard	
	Clicking Buttons	
	HyperCard Buttons	
	Finding Text	

Adding Your Own Text
Editing Text
Summary

Chapter 2 Customizing Stacks 36

Using the Paint Tools to Add Pictures
Adding Pictures to the Weekly Calendar
Borrowing Pictures that Already Exist
Card or Background Picture?
Making the Card Picture Opaque
Using the Temporary Layer
Giving Yourself More White Space
Using the Field Tool
Enlarging the Text Field
Adding Text to the Field
Customizing the Way the Text Looks
Using the Button Tool
Navigating with Buttons
Creating a New Button
Using a Button Icon
Linking the Button to Another Card
Summary

Part Two Becoming a HyperCard Author 85

Chapter 3 Building Stacks 87

Borrowing
Collecting
Making Information Your Own
Starting a New Stack
A Copy or a Shell
Starting with a Stack Idea
Building a Personal Information Base
Adding a New Text Field
Setting Properties for a Text Field
Adding Cards to the Stack
Deleting the Fields You Don't Need
Adding More Cards
Adding a Background Button that's Linked to a Specific Card
Building a New Monthly Calendar
Trying the Buttons

Copying a Button from the Button Collection
Summary

Chapter 4 Designing New Stacks 123

Thinking About What the New Stack Should Do
Creating a New Address Stack
Why Create a New Stack?
Getting Information about Stacks
Adding Clients' Names
Deleting Cards
Ordering and Sorting New Stacks
Designing the Way the Bill Looks
Changing the Background Picture
Designing the Buttons
Getting Rid of Buttons You Don't Need
Moving Buttons on the Card
Using Paint Text
Identifying the Fields
Adding Your Business Logo
Adjusting the Text Fields
Creating a New Field
Filling in Amounts Due
Moving among Fields
Adding Information for Other Clients
Adding the New Stack to the Home Card
Adding a Picture to Represent Your New Stack
Adding a Button that Goes to the New Stack
Printing the Bill
Summary

Part Three Scripting 181

Chapter 5 Gaining Power with Scripts 183

HyperTalk Scripts
Using the Message Box
Asking What Time or Date It Is
Asking Where the Mouse Is
Asking What the Current Tool Is

Asking the Name of Objects
The Card Is the Most Basic Object
Calculating
Training Buttons to Issue Commands for You
The Button Dialog Box
Message Handlers: Who Has Control?
Naming Objects
Naming Hierarchies
Absolute vs. Logical References
Looking at a Script HyperCard Wrote for You
Modifying Existing Scripts
A Script that Sorts Cards
If . . . Then . . . Else Control Structure
Editing the Script to Sort by Date
The Objects Menu
Any Object Can Have a Script
Where to Put a Script
Messages the System Sends
Putting a Message Handler Lower in the Hierarchy
Going to Another Stack Within a Script
Summary

Chapter 6 Expanding a Stack's Capabilities 228

Making the Client Stack More Powerful
Creating a Button that Totals a Bill
Editing a Button's Script
Creating a Button that Subtracts Payments
Sending a Message to Another Object
Creating a Button that Moves Current Charges into Past Due
Keeping Permanent Records
Creating a New Stack Using the Current Background
Moving Old Records into the Client History Stack
Printing your Client Histories
Summary

Chapter 7 Practical HyperCard Tricks 264

Text and Graphics: A Good Marriage
Restraint and Other Design Guidelines

Starting Simple
Layers and Backgrounds
    Background and Card Pictures
    Background and Card Buttons
    Bring Closer and Send Farther
    The Order of Objects on a Card
    Using More than One Background
    Adding a New Background to a Stack
    Adding Backgrounds by Copying a Card
    More than One Stack
Real Text and Paint Text
    Fancy Fonts and Freedom from the System File
    Shared Text
    Lining Things Up
    Lining up Paint Text
    Aligning Pictures You Cut and Paste
Visual Effects
Animation
Keeping Track of Buttons
    Buttons under Construction
    Card or Background Buttons
    Creating Big Background Buttons
Controlling Headers
Keeping Your Stacks Private
Using Pop-up Fields
Sorting Tricks
    Creating a Hidden Field to Sort By
Summary

Chapter 8 Hooked on Scripts 314

Where to Put a Script
Inheritance
System Messages and What Triggers Them
A Moment in the Life of HyperTalk
Scripts You Can Use
    Choosing a Command
    Choosing a Tool
    Searching a Script for Text
    Finding Backgrounds
    Merging Stacks
    Displaying Windows on a Large Screen
    A Spreadsheet
    The Target
    Push and Pop Card

Finding Where a Message is Defined
Adding a Card
Plots
Playing a Tune
Limiting the Scope of the Find Command
Generating Bar Codes
Automatic Indexing
A Pictorial Index
Summary

Chapter 9 Talking to the Rest of the World 353

The Scrapbook and Notepad
Opening Other Applications
Importing and Exporting Text
Importing and Exporting Pictures
Networks: AppleTalk, Phone Lines
Sound
Summary

Afterword 365
HyperCard and Hypermedia
Education
Form and the Shape of Content
Gathering
Becoming a Minimalist
Annotation

Appendix 1: HyperCard Windows and Commands 373
Appendix 2: HyperTalk Commands 396
Appendix 3: Moving Resources 414
Appendix 4: System Requirements and Comforts 418
Index of Hints 420
Index 427

Foreword

HyperCard—A Software Erector Set

The Macintosh dream really centers around putting personal computing power in individuals' hands. The Macintosh has done a lot of that by using graphics, a mouse, a consistent user interface and direct manipulation metaphors to make the computer more usable by a non-computer nut. In so doing, it unfortunately has made the job of writing programs for Macintosh much harder.

Writing a Macintosh application requires a professional programmer with a year's commitment to learning the Macintosh toolbox. That has brought us farther from our goal. In

some ways the Apple II was closer to allowing individual people to have control of the machine, because they could type in a few lines of Basic.

What I hope HyperCard will do is complete that Macintosh dream of putting the power in individual people's hands. HyperCard is a software "erector set," where you can draw things the way you do in MacPaint and use copy and paste buttons, where you can put together information organization systems that allow individual people to get control of their Macintosh.

You will start by using the stacks that come with HyperCard, by modifying them in small ways, then in larger ways. Then you will learn about the painting tools and add graphic embellishments and eventually add your own buttons and make your own stacks. Some of you will go on to create professionally produced stacks for sale. A lot of you will customize your environment the way you like it and will make individual applications that you will use for yourself.

Text, Graphics, and Interaction

Information on computers has moved from being just ASCII text to information that includes text and graphics. That step has made the information much more accessible and has also fostered the desktop publishing industry. The next step in making information more accessible is to include text, graphics, and interaction. That's one of the ways that HyperCard hopes to lead the industry and pull all the computer companies toward a new expectation, a new level of what information is on personal com-

puters. For instance, a 1040 form with the lines and boxes is more understandable than if you just had the text there. The graphics gives it more understandability. But if you extend that to a 1040 form with interactivity that helps you fill out the form, of course it's even more digestable; the information is more accessible.

So one of the things that HyperCard is trying to do is raise the level of art—raise it to where people expect information to be interactive. The second thing it's trying to do is raise the level of expectation about customizability. When you get an application, if it isn't quite what you want, you're not stuck with it. People now take a kind of fascist approach to software, not because they want to, but because they have to write one program that will solve 100,000 people's needs (in order to pay back the development costs). Instead, there should be one program that would be useful to 100,000 people, but is customizable so that each of those people with their slightly different needs can tweak it and bend it to be their own.

As I mentioned, HyperCard is an "erector set" for software. When you buy a regular erector set, it has boxes of parts, nuts, bolts and a screwdriver, and also, importantly, it has a brochure of things that were built with this erector set. It has a picture of a little truck and a windmill. You start by building the truck and the windmill, exactly as in the pictures. Then you customize a little bit. You make a long bed truck, and then you branch out into making your own things from scratch. The stacks that come with HyperCard are the brochure. They're ready-to-work samples that teach you how to use HyperCard. They are working things that you start modifying a little

bit, and eventually you branch out into making your own stacks with your own ideas. They teach you how to use HyperCard.

The erector set analogy goes further. You get boxes of parts in HyperCard. You get a box of art parts that you can copy and paste; you get a box of Stack Ideas that you can go and say, "Yeah, I want one like that;" you get a box of buttons that you can go through and say, "Yeah, I want the dial button." When you copy and paste one of those buttons, it comes complete with its "brains" and is ready to work. It knows how to dial. It's as though the parts in the parts box came with motors and batteries included. This enables people—without any programming experience at all—to put together things that are re-active and interactive. You make stacks that have "life" to them. They're beautiful because you have tools with which to draw and make original illustrations, and they are interactive because you can copy and paste buttons that *do* something.

There is another way that HyperCard is an improvement over the original erector set. First, you don't run out of parts. One of the first things that happens when you use an erector set is that you run out of a piece of a particular size. Because you are always making copies in HyperCard, you never run out of that piece. In an erector set, if the right shaped piece didn't come in the set, you have no way to create it. HyperCard's pieces are themselves built out of a general language that allows you to make and customize your own pieces. You can take any button that's almost what you want, go inside it with the scripting language, and modify its behavior a little bit to be exactly what you want it to be. You start by modifying the buttons. You're looking inside them, learning how they work, and how you can customize them

a little bit. The script says, "Visual effect scroll right" and "go to previous card." You can see you might change that to "visual effect dissolve, go to next card" or something like that. With the HyperTalk language, the scripting language, there is really an unlimited range of growth within HyperCard. How much a button can do is open-ended.

The button library that comes with HyperCard has a hundred or so buttons in it. After HyperCard has been out for a year, a typical button library that a person has will contain thousands of buttons. They will have collected these neat buttons that somebody wrote. It doesn't cost anything to share your buttons, because you can peel off infinite copies for free. Making a copy of your button doesn't diminish your own library.

I envision that the erector set approach will allow a lot of people who have no control over their computer now, who are just using applications the way they came, to be able to get the feel that they are working their computer, rather than that the computer is working them.

The HyperCard Project

The first thing Carol Kaehler and I worked on together was the MacPaint manual. We had a great time working together on that, and that's why I chose to have her do the Help System for HyperCard. She's good.

While I was developing early versions of HyperCard, Carol was my heavy user. I remember very clearly sorting out some of the rough sketch ideas for the Help system. We actually had some paper cards that

we spread out on the floor, subdividing what was to be taught. There was a lot less to be taught then, and the structure had to be revised a lot as we went through. But the basic shape, the structure, is still there.

Carol was a prototypical user. If she couldn't use something, I changed it! She was a guinea pig of the first magnitude. She was sensitive to what things she didn't get and why they were confusing. We'd talk about them, I'd come back with another try, and she'd make suggestions. Some of them would be right on, and I'd say, "Yeah, that's great." I'd put it in and try it and we all liked it. But with some things she'd ask for I'd say, "That's beyond the scope of what we're going to do in this project." However, as the project grew and grew in scope, a lot of her suggestions got included. For example, it was her pleading for multiple backgrounds per stack that contributed to that happening.

The HyperCard team has been very close. We've spent a lot of time together, and a lot of time talking about what-ifs. And a lot of time imagining what our customer is going to be feeling. We've been spinning a lot of dreams for people who'll be using HyperCard.

At one point we were going to try to switch over from a card metaphor to a book metaphor. We were going to use the library card catalog, shelves, books, chapters, etc. At a certain point, however, we just realized that, really, a stack of cards was a much better metaphor. People just don't feel good tearing out pages of books and sorting the pages of a book. The stack of things was a much more general metaphor. You could think of a stack of books, you could think of a stack of slides, you could think of a stack of records.

The card as the basic entity just seemed a lot more maleable and flexible to handle different information situations.

Carol knows the tips and tricks inside HyperCard, and she shares them with you in this book. Spend an afternoon with Carol in HyperCard—and you will learn a lot!

—Bill Atkinson

Preface

I met Bill Atkinson when I joined Apple's Macintosh team in September, 1982. Chris Espinosa, who had been around Apple since childhood (both his and Apple's), had hired me to write the Macintosh owner's guide—the book that would teach new users the basics of using the mouse, creating and editing documents, and managing the desktop.

Macintosh Retreat, September 1982

My arrival at Apple coincided with a retreat where the 70 members of the new Macintosh division were to spend two intense days sharing their progress, pointing to areas of po-

tential problems, and getting agreement from the rest of the groups about what we were going to do in the next year. The groups represented hardware, software, marketing, publications, industrial design, manufacturing, finance, and human resources.

Blue and white baseball caps were passed out on the bus. "Macintosh division" printed across the front announced that Macintosh had been made official. We drove south along the Pacific Coast to Pajaro Dunes, California, the "traditional" Silicon Valley conference site, and settled into salt-air weathered condominiums. Later that afternoon, the first of the groups began their presentations. First came the Marketing folks—with the standard outline of goals and strategies, and a comparison of the Macintosh to the Cuisinart as an affordable, powerful appliance "for the rest of us" based on familiar metaphors such as the desktop. As the first day's sessions concluded I braced myself for the second day—what I thought would be a dry set of technical presentations from the Engineering groups.

But this group of engineers seemed very different from those I'd encountered before. The mood was intense, but it had nothing of the flavor I expected from traditional engineering types. Each speaker outdid the one before in irreverance, flamboyance, or wryness. It was as if they had pondered the thought of presenting the dry facts and had cast the plan aside to present, instead, something totally unexpected from a hardware or software engineer.

When Burrell Smith, the designer of the main logic board (the hardware that holds the main processor and memory), came up for his turn, instead of presenting the specifications for the logic board, he

played his guitar. Jerry Mannock, the industrial designer of the boldly different Macintosh case, appeared in a Scottish tam, complete with tassel, and "hammed" his way through a skit that was reminiscent of an initiation into Camp Fire Girls. To a backdrop of recorded bagpipe music, he concluded, "Repeat after me: you will not change the design, you will not change the design. . . . " Bill Fernandez, lab manager and another Apple old-timer, joined Burrell in an impromptu blues harmonica and guitar duo.

Debi Coleman, then financial controller of the Macintosh division, announced the results of the "Mac Man" contest, tallied from votes for the sexiest man in the division. (The official winner was the handsome industrial engineer Bill Bull, although it was later rumored that Steve Jobs had actually won by a landslide.)

When I was introduced as the newest writer in User Education—having been there just one day—Randy Wiggenton, the author of MacWrite, unhesitatingly hooted, "What's she written so far?" The entire group was like that—boisterous and irreverent, full of one-liners and quick comebacks. I was getting slightly nervous. These people were very smart and very cocky and many of them didn't think the Macintosh even needed to have a user's manual. What was I doing there anyway? Bewildered by RAM and ROM and logic boards, I sank into my seat, looked around me, and longed for a friendly face.

Much later I understood why Chris had hired me to write the owner's guide, though I'd never used a personal computer and had written only one small book that explained to nonprogrammers how to slog through JCL (an unforgiving, unfriendly language used to control IBM mainframe computers). The

Macintosh wasn't a traditional computer, and the Macintosh team didn't want it described in traditional computerese. The Macintosh was the computer "for the rest of us" and part of the rest of us were people who didn't care to deal with computers that made us learn their language.

During a break in the sessions I sought out Bill Atkinson. My husband, Ted Kaehler, knew Bill from his work on computer recognition of speech with George White at Auricle. Ted, who was then a member of the Smalltalk group at Xerox Palo Alto Research Center (PARC), had told me that Bill was a great programmer and a thoughtful, intense man worth seeking out.

MacPaint

After talking to Bill briefly that day, I didn't see much of him for several months. He was buried in the innards of QuickDraw, and I was immersed in writing the owner's guide. About six months later, I found him at the back of the Macintosh building on Bandley Drive in Cupertino. He was demonstrating an early version of the MacPaint that he was writing. I was waiting for the Finder to be complete enough to write about; Chris Espinosa suggested I spend some time with MacPaint.

Chris had designed an overall plan for all the Macintosh manuals. They would be slim, straightforward volumes divided into three parts—a brief tutorial, a "cookbook" with various tasks laid out step by step, and a standard reference section. I set about fitting the MacPaint manual into this design.

A few weeks later I nervously handed Bill a draft of the manual. It led the reader through designing a let-

terhead. It explained each drawing tool, selecting and modifying, copying, pasting. Bill arrived back at my office about a week later, looking not too happy. He insisted the book design wasn't right for MacPaint. MacPaint, he said, was an application that demanded exploring, with just a few hints for not getting lost in the exploration. After I took a few hours to recover from the total change in direction, I created a startup screen that said

Think Graphic!

Whenever I switched my Macintosh on to work on the book that message greeted me and drew me back to the simple principles of helping the exploration process without getting in the user's way.

Beyond MacPaint

With MacPaint, Bill had begun to move outward from the inner mysteries of QuickDraw, which does the most basic job of putting bits on the Macintosh screen. With MacPaint he had begun to share the power of the Macintosh by handing users the paintbrush. What would be the next step in Bill's personal evolution and its expression in Macintosh?

Keeping it simple, allowing learning by exploration, and presenting a highly graphic interface are the Bill Atkinson traditions that are self-evident in HyperCard. But beyond that, with HyperCard, Bill has unleashed a lot more of Macintosh's power, and he has shared it with all of us. Now it's possible for everyone to enter the "priesthood" of programmers by following a gentle path of ascent. Now we can all mold and shape how the Macintosh presents and organizes information.

Complexity and Simplicity

On one side of Bill's computer sits a six-inch-thick notebook filled with a listing of the HyperCard program. On the other side is a sheet of paper with just a few items written on it—things for him to think about while he's finishing up the product.

Speed it up
Keep it simple
Ship it on time.

Bill's trademarks are a balance of complexity and simplicity, and a willingness to do whatever needs to be done to get the job done right. Bill has very high standards for himself and the people he works with, and he commits his seemingly boundless energy to maintaining those standards. He's willing to hang in there doing work that many people would consider too menial, and he stays with it until it's as good as his standards demand.

I used to wonder, when people recalled years after their experience with great men and women, if they had known at the time what they were experiencing. Did they savor the encounters, did they really "get it?" Or did they invent the exhilaration years later in retrospect? With my eyes wide open these past two years, I've known without a doubt that every minute of working with Bill has been a gift.

It's not unusual to find programmers with the technical skill to control a computer. What is much rarer is the good judgment this magician brings to the task of knowing what the computer can do to empower people. HyperCard is the pinnacle of his skill and good taste, and it's a gift to everyone who uses the Macintosh.

Acknowledgments

This book rose out of exuberance for HyperCard and for Bill Atkinson. He has been both an unequaled teacher and a rambunctious cohort. And this book wouldn't have been written without Ted Kaehler, whose intelligence, calm, and support have pervaded the entire HyperCard project. Ted has inspired and taught me and many others along the path to programming. His ability to look at things without presupposition and to propose preposterous solutions causes great things to happen. He is my hero.

Credit goes also to everyone else on the HyperCard team—Dan Winkler for writing a language nonprogrammers could love, Kristee Kreitman and Marge Boots for the graphics that make HyperCard come alive, Adam Paal for the beautiful

printed page, Chris Espinosa for making all of it really happen, and Bill Fernandez for his uplifting spirit. Thanks also to Scot Kamins, James Redfern, David Leffler, Sioux Lacy, Steve All, Mike Farr, Bob Eddings, Robin Shank, Gary Bond, Phil Wyman, Karyn Gray, Cliff Guren, Mike Holm, Ken Rosen, Ken Doyle, Mary Sinclitico, and Laura Atkinson (for discovering katyatutu).

Chris Espinosa taught me a lot about writing for the Macintosh. Joanna Hoffman persuaded Chris to read my resume, and Caroline Rose, Lynnea Johnson, and Pam Stanton-Wyman shared those early Macintosh years with strong constitutions and good humor. Thanks for that.

Thanks also to Martha Steffen, who steered me in the right direction, and to Sue Espinosa, Mike McGrath, and the rest of Apple Customer Publications and Computer-Based Training for making it possible for me to pursue this adventure.

Connie and Gordon Nasby always thought I could do anything. That came in handy. Mark and Sarah Embury put up with take-out food and parents whose default conversation subject has been HyperCard for the past two years. Mark read and tested early drafts of this book. Sarah lent her artistic flourish.

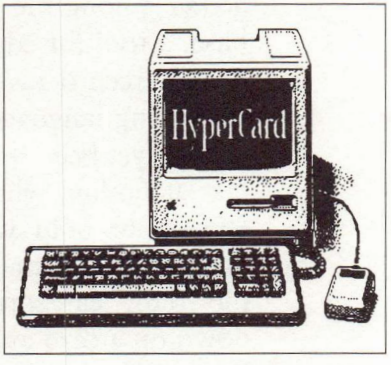
Louella Pizzuti provided comic relief as well as a technical review of the manuscript; Molly Tyson mentioned me in her book; analogies were inspired by Roy Ginsburg; Lani Blazer stuck it out with us.

Finally, thanks to Steve Stansel, Carole Alden, Linda O'Brien, Alexandra McDowell, and Maeve Cullinane of Addison-Wesley. Their professional sensibilities pointed me in the right direction when I veered.

Introduction

"If you're going to look at all of human knowledge, you might want to have a way to organize it."
Bill Atkinson

HyperCard: A Visual Information Organizer



Imagine a bulletin board strewn with cards—3″ x 5″ index cards, business cards, scraps torn from a notebook, meeting announcements, and real estate listings. "House for Rent" is neatly printed on one card, "Violin for Sale" is scrawled on another, a third card an-

nounces in delicate calligraphy a meeting of amateur photographers.

So far it sounds like an ordinary bulletin board. But as you approach this bulletin board you notice pieces of string leading from card to card, connecting one piece of information to another. A string tacked to the Violin for Sale card leads to another card with a violin for sale at a lower price. Two different strings from the second card go to a card with other musical instruments for sale and to an orchestra soliciting new recruits. The card with the house for rent leads to another house for rent in the same part of town and to a house in the same price range. So as you follow the strings, you move easily to just the information you need. And when you don't feel like following the strings anymore, you just browse through cards that catch your eye, discovering things you didn't even know you were looking for.

Now imagine this magic on your Macintosh. HyperCard is an organizer for any kind of information. It provides the string that ties information together among the various cards. It's a freeform data base, a tool for education, illustration, and presentation, a record keeper, an authoring system, a programming language, and probably many things that haven't yet been thought of. HyperCard is amazingly fast at finding what you're looking for and flexible enough to hold whatever kinds of cards you can dream up. Information doesn't have to be sorted into the "right" categories. It's more like jotting something down on a card and sticking it on the bulletin board. Except that you can find it again!

Most of us are already deluged with more information than we want. But HyperCard puts us in control

of this mountain of information. It lets us get at and organize just what's important to us—information not only on hard disks and networks but on videodisks and CD ROMs as well.

With HyperCard, you don't have to choose between text and pictures. For the first time, combining text and pictures in the same application is not only possible, it feels perfectly natural.

Creativity and Sharing Information

To get an idea of how HyperCard can spark imaginations and create forms that are more than a sum of their parts, imagine a group of three- and four-year-old children sitting around a table with masses of clay in front of them. You can learn something about creativity and sharing information by observing these young artists. One child fashions a pitcher out of clay. Her neighbor looks over and says to herself, "Maybe I could do that!" But her piece comes out a little differently—she can't get the pitcher's spout to stand up like that so she bends it around, and the effect is a graceful variation of the original. By now the other kids have started their own variations, observing, imitating, and modifying their pieces of art to suit themselves. (They haven't yet heard about the problem of people "owning" their ideas.) All the artists copy, improve, try wild departures, alternately coming together to share and then going off alone to pursue their own aesthetic senses.

Information thrives in a setting where it isn't limited by standards of form and traditions of presentation.

A database comes alive when it doesn't have to lock each piece of information into a previously defined field that has a strict definition of the information it contains. HyperCard lets all kinds of information emerge in its new forms, modeled by HyperCard authors, shared with other users, and remodeled by budding authors into still newer and perhaps better forms.

A Common Currency

HyperCard's architecture and interface make it easy to exchange, organize, and present information. Because HyperCard stacks (the equivalents of documents in other applications) share a common data format, you can trade or sell them to anyone who has HyperCard. This combination of an extremely broad user community and consistent (yet mightily flexible) format promises to bring about a new commodity called *stackware*™—information that is accessible to anyone who has a Macintosh with HyperCard.

This stackware has a number of advantages over traditional forms of information exchange. Because HyperCard allows fast and freeform searching—finding text in whatever form it takes (rather than just a certain exact pattern of letters following letters)—it's much better than using a book when you want to look something up. Because there's no preset structure you have to follow for designing new stacks, you can create stacks that function exactly how you want them to, and because of HyperCard's built-in painting tools, you can make stacks be as beautiful as they are useful.

Who This Book Is For

This book is for anyone who wants to know more about HyperCard—either to customize stacks for your own use or to learn how to design your own stacks from scratch. It covers all the basics and includes many shortcuts and hints gleaned from a year and a half's intensive work with HyperCard in designing the Help system. Although this book is not primarily about the HyperTalk language, it gently introduces scripting so that nonprogrammers can easily learn to incorporate HyperTalk into their own scripts. Complete command function and system message descriptions are also included in an appendix.

From User to Author: The Continuum

You might start by using HyperCard to browse through information that someone else has prepared and presented. But it's easy to get hooked on the power of HyperCard. Here's how it can happen: You add your own friends' names to an existing stack of name and address cards. Then you decide to customize the cards by adding pictures to some of them. You decide it might be handy to check your appointments against your husband's appointments, so you use the HyperCard "string" to link the two appointment books together. Then you redesign the appointment book to make it show two weeks at a glance, because that's how you visualize time. . . . Pretty soon you'll wonder if HyperCard can help you keep track of your Christmas lists or your financial records, or if

you could use it to run your small catering business, and by then it's too late. You're hooked on the power of controlling the information in your computer.

HyperCard is preset to let you browse or type. As you become more familiar with the way HyperCard works, you can change your user level to match the power you want. Figure I.1 shows the User Preferences card and what each of the levels include. When you're ready for more power, go to the Preferences card by choosing Home from the Go menu. Then choose Last from the Go menu to go to the last card in the Home stack—the Preferences card, where you can click the level you want to be at.

How to Use This Book

HyperCard itself allows freeform exploration, but it also has a built-in progression from browsing information to modifying it, and finally, to writing scripts. This book follows that same progression. You don't have to read it strictly in the order presented, but the chapters are laid out to gradually expose you to HyperCard's power.

You can also find topics easily by using the table of contents or the index. Hints are scattered throughout the book and summarized at the back of the book for quick reference.

How This Book Is Organized

This book unfolds HyperCard's gentle path from browsing through information to customizing it by

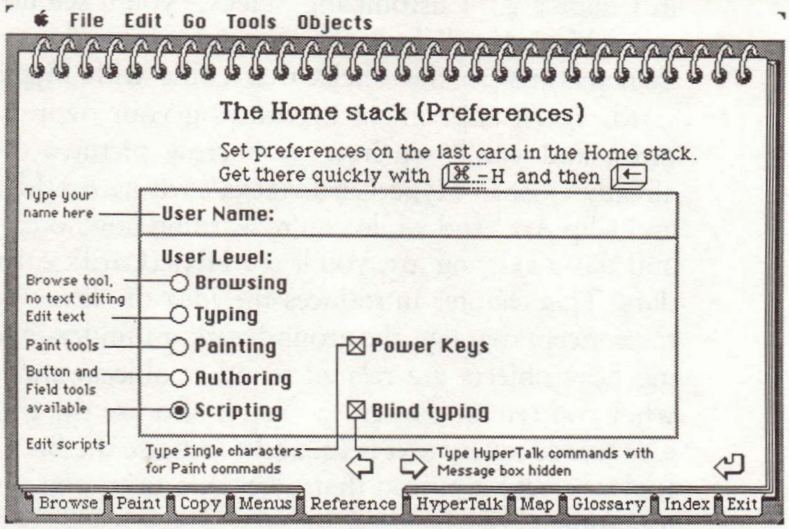


FIGURE I–1 *The User Preferences card is the last card in the Home stack. Get there by choosing Home from the Go menu and then choosing Last from the Go menu. The five user levels let you add capabilities as you become increasingly familiar with HyperCard.*

adding and editing text and graphics, to copying and modifying buttons and fields, to creating new stacks, and finally, to writing scripts using HyperTalk—HyperCard's built-in programming language.

Chapter 1, "Browsing HyperCard," is a short guided tour of HyperCard. In this chapter you'll see HyperCard's basic elements—stacks, cards, buttons, fields, and the tools you use to manipulate them. You'll discover how to navigate among cards and stacks using the Go menu and clicking buttons. You'll find out about searching for text using the Find command. This chapter also includes a summary of the standard Macintosh techniques HyperCard uses for adding new text and editing what's already there.

In Chapter 2, "Customizing Stacks," you'll see how to use HyperCard's built-in painting tools to add your own pictures to the stacks that come with HyperCard. You'll experiment by drawing your own pictures, and you'll see how to borrow pictures that already exist in HyperCard stacks such as Art Ideas and Clip Art. And while you're learning how to copy and paste existing art, you'll see HyperCard's calendars. This chapter introduces the idea of "layers"—the concept that lays the groundwork for understanding how objects are related to other objects and to what you see on the screen. You'll also use the Field tool to add a new text field, and you'll use the Button tool to create a button that takes you from one card to another.

Chapter 3, "Building Stacks," shows you how to use the collections of stack "seedlings" included in special collection stacks. You can use these seedlings to build your own custom stack. This chapter leads you through building a personal information base out of one of these seedlings to building a new calendar out of another.

In Chapter 4, "Designing New Stacks," you start a new stack from scratch. Using the example of a small business, this chapter shows how you can use HyperCard to keep track of people and money. While Chapter 3 shows you how to borrow an idea and change it, this chapter uses the "build-it-from-the-ground-up" approach.

Chapter 5, "Gaining Power with Scripts," introduces HyperTalk, HyperCard's language. You start by using the Message box to send simple messages that take you to other cards and stacks or that ask the time or date. And then you look under the hood of an exist-

ing button to see the script that tells HyperCard what to do when you click this button. This chapter explains what Message Handlers and other control structures are, introduces a few basic commands, and shows you how HyperCard uses names to designate the various objects that can have scripts—buttons, fields, cards, backgrounds, and stacks.

In Chapter 6, "Expanding a Stack's Capabilities," you see how a design can grow along with your needs and expertise. You make the stack you started in Chapter 4 more powerful, using more HyperTalk commands in new scripts that you add to the stack's objects. You also find out how to send a message to another object. (Easy!)

Although hints and tricks are scattered throughout this book, Chapter 7, "Practical HyperCard Tricks," is a compendium of not-so-obvious techniques that can save you time and help you produce more elegant stacks.

Chapter 8, "Hooked on Scripts," is a collection of annotated scripts, some included with HyperCard and some contributed by early HyperCard hackers. Examining and modifying other authors' scripts can be one of the best ways to learn how to write your own scripts.

Chapter 9, "Talking to the Rest of the World," tells you how to get information into and out of HyperCard.

The Afterword looks at some of the implications of using hypermedia.

The Appendixes include HyperCard Windows and Commands, HyperTalk Command descriptions

and examples, Moving Resources, and System Requirements and comforts.

HyperCard requires 1 megabyte of memory. This means at least a Macintosh Plus or a Macintosh upgraded to 1 megabyte and equivalent ROMs. Although a hard disk isn't strictly required, you'll probably find yourself wanting one as soon as you begin using more than the Address and Appointment stacks.

Before You Start

HyperCard relies on the Macintosh clock being set correctly. (It uses the clock setting to determine the correct time and date for some of its fields and for the calendars to work right.) Check that your Alarm clock is set correctly by choosing it from the Apple menu and clicking numbers to reset it if necessary. (See your owner's guide for details.)

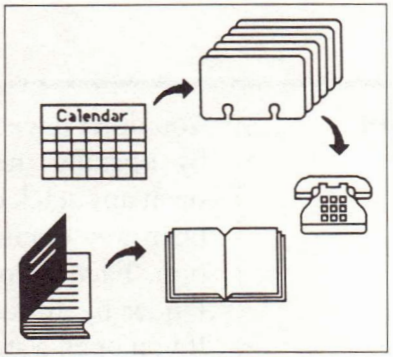
Because HyperCard automatically saves any changes you make to stacks as you're working (and without asking), you need to make a copy of any stack you don't want to accidentally alter. You can copy stacks in the Finder in the usual way to make a backup, or you can use the Save a Copy command within HyperCard to make a backup copy. Save a Copy makes a copy of the entire current stack but does not automatically switch you to that stack. (This is different from the Save As command in many applications.) Rather, you continue to work on the original stack. If you want to work on the backup copy, use the Open File command to open the backup copy.

Part One:
Using HyperCard

Chapter 1

Browsing HyperCard

"Can I help you?"
"No, I'm just browsing."
Anonymous shopper



Meet HyperCard!

Your first experience with HyperCard can change your mind forever about what a Macintosh application should be. You see familiar elements everywhere—text, pictures, commands, palettes, and all the Macintosh conveniences you already know and love. But something's

different. With HyperCard, text and pictures are together in one place, and you can move through information of any kind with the ease and editability you never had before, searching for exactly the information you want or clicking buttons that can take you to all kinds of interesting information you didn't even know was there.

When you start HyperCard by double-clicking the application in the Finder, you don't open a blank document as you do when you open other Macintosh applications. Instead you step into an inviting visual directory called the Home card—the first card in the Home stack. (A stack is the HyperCard equivalent of a document.) The Home card is shown in Figure 1.1. The Home stack is a good place to start exploring HyperCard and it's also an easy-to-get-to home base whenever you're using HyperCard.

Hint

You don't have to start HyperCard from the Finder by opening the application itself. You can also open any stack directly if you want to, just as you'd open any document you create with any application. Each HyperCard stack is represented in the Finder by an icon that looks like a stack of cards. If you open a stack directly, HyperCard will get the global information it needs from the Home stack and then take you directly to the stack you opened.

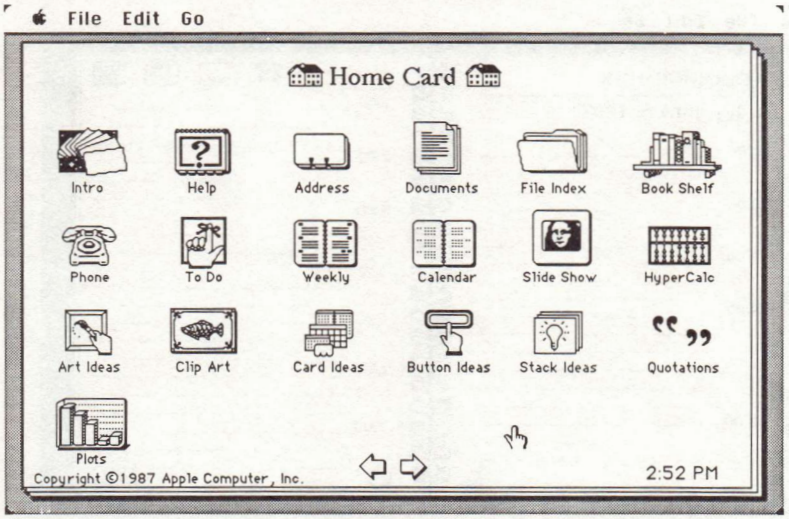


FIGURE 1–1 *The Home Stack is a visual directory of the stacks you use most often. Clicking any picture takes you to the stack the picture represents.*

Stacks: Ordered Collections of Cards

You can use HyperCard to organize any information you want to browse through and keep track of. Each stack is an ordered collection of cards. Usually the cards in a stack have something in common with each other; they belong together for one reason or another. A stack might be any of the following: a collection of name and address cards; an appointment calendar with cards for the various days, weeks, or months; a

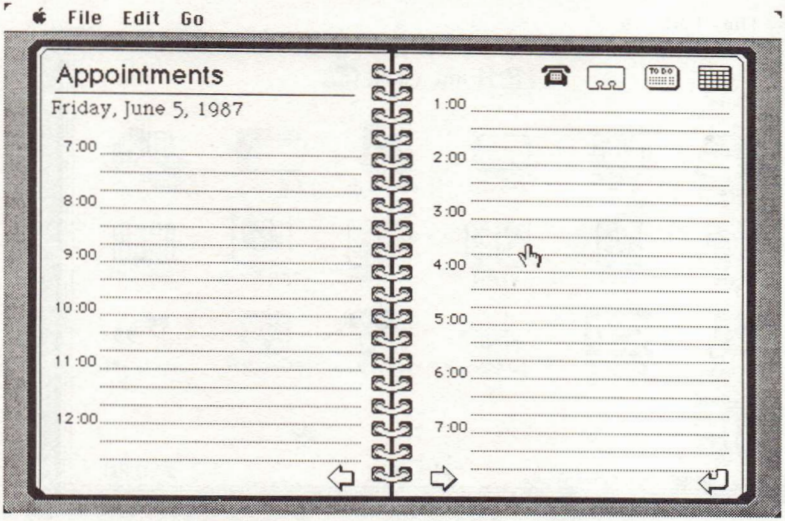


FIGURE 1–2 *Cards within a background share the same look because they use a common background picture. For example, each card in this appointment stack looks like a page in an appointment book.*

photo collection; an atlas; or even a library card catalog.

The cards in a single stack generally have the same look. They get this look by sharing a common background picture. Each address card looks like other address cards, and the pages in a appointment book all look as if they belong together. Figure 1.2 shows a card from an Appointment Book stack.

Each card in a stack can have its own picture as well as a background picture shared with the other cards. Of course, cards in a stack can have text as well as pictures. Figure 1.3 shows an Appointment Book stack. Text has been added to the blank card to customize it with appointments, and the HyperCard

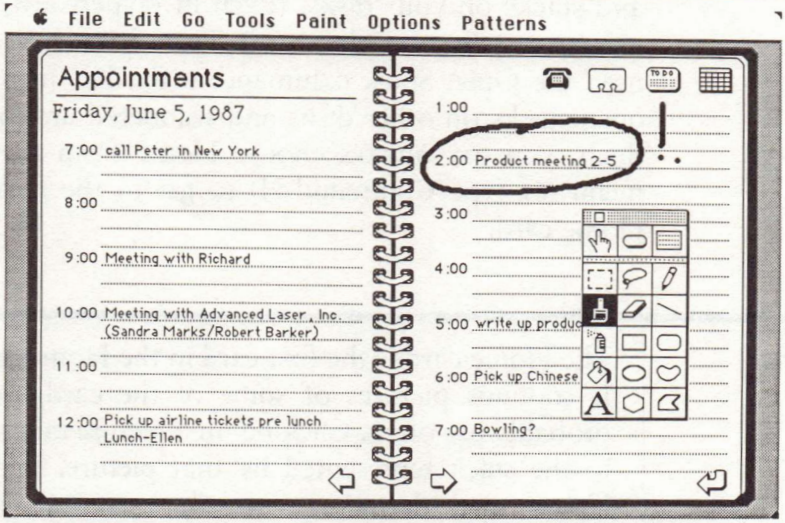


FIGURE 1–3 *Individual cards can also have a picture of their own. You can use any of the Paint tools to create new pictures or copy and paste pictures from other stacks or MacPaint documents. On this card, the paintbrush was used to circle an important meeting.*

built-in Paintbrush tool was used to circle an important meeting.

Using the Go Menu to Get Around

There are plenty of ways to get around in HyperCard. You may have read the Introduction stack and the Help stack to learn how to get around by clicking buttons and arrows. The Go menu also helps you navigate among stacks and within the current stack.

The Go menu is just for navigating among the cards

and stacks on your disks. (Even in HyperCard, you can be in only one place at a time.) Even if you've used the Open Stack command in the File menu to open stacks on other disks and wandered around in them, you can always choose Home from the Go menu (or type Command-H) to get to the familiar Home card.

Hint

The Home card is the first card in the Home stack. It contains pictures of some of the cards you'll probably use often. Clicking on a picture takes you to the stack represented by that picture. But the Home card doesn't necessarily represent every stack you have created on your own disks. To see the actual stacks on your disks, use the Open Stack command in the File menu.

The Go menu takes you to other places as well as Home. The Back command takes you back to the card you most recently looked at. Help takes you to the first card in the Help system. (If the disk that contains the Help stacks isn't currently inserted, it will present the standard file dialog box and ask you to point to Help.) Recent shows you the 42 cards you've seen most recently. First and Last take you to the first or last cards within the current stack.

Home, Help, First, Next, Previous, and Last are stationary points within the HyperCard stacks. Where you go when you choose one of them depends on the pre-existing structure of the stacks and where you are within them. But the Back and Recent commands refer you to different cards, depending on where you've been. Back and Recent move you through time rather

than structure position. They're a little like the crumbs Hansel and Gretel dropped to find their way back again (except that they never get eaten by the birds.)

Hint

Pressing the key at the top left corner of the keyboard (the tilde/accent grave key or the Escape key) retraces your steps exactly—going back through each card you've seen. (When you're using a Paint tool, you press the Command key as well as the tilde key to do this.) So if you've gone back and forth between two cards several times, retracing your steps with this Go Back key will take you to each card as many times as you were there originally. However, the Recent dialog box displays each recent card just once. So if you've flipped repeatedly between a couple of cards, choose Recent from the Go menu to save time in retracing your path.

Using the Keyboard to Navigate in HyperCard

Although HyperCard stacks can be huge and varied, you'll never feel lost because you can always get back to the last card you were looking at (or the one before that, and before that, up to the most recent 42 cards you've seen). And if you've been exploring for awhile and want to get back Home, there are easy ways to do that also. Here's an overview:

- Use the key on the top left corner of keyboard (the tilde/accent grave key or the Escape key if you have a newer keyboard) to move backwards through the cards you've seen before. It's the same as choosing Back from the Go menu, and is quicker. This key is sometimes referred to as the Go Back key.
- Use the left and right arrow keys (sometimes called cursor keys) to move through cards in the current stack. Use the right arrow to go to the next card, the left arrow to go to the previous card (in their stack order).
- Use the up and down arrow keys, if you have them, to go back (the down arrow) or forward again after you've gone back (the up arrow).
- Use Command-down arrow to push a card. Pushing a card tags it as a card you want to come back to later. Use Command-up arrow to pop back to the tagged card.
- To get back to the Home card, type Command-H, or type Go Home in the Message box and press Return.
- To get to Help, type Command-?, or type Go Help in the Message box and press Return.

Clicking Buttons

You can also use the Browse tool (it looks like a hand) to click buttons. Clicking a button tells

HyperCard to carry out any command built into the button. One of the most common commands for a button is to take you to another card—in either the same stack or a different stack. For example, Home buttons abound in HyperCard. Whenever you see a small picture of a house, it's a safe bet that clicking it will take you to the Home card. Navigating among cards and stacks by clicking buttons is one of the main ways to get to the information you need. Figure 1.4 shows a button in the HyperCard Introduction stack. Clicking the top button takes you to a card, shown in Figure 1.5.

You can also click buttons to open documents created with applications such as word processors, spreadsheets, data bases, or graphic applications, or to open "mini-applications" you create with HyperCard itself. Figures 1.6, 1.7, and 1.8 show you some examples of the types of applications you can open from HyperCard.

HyperCard Buttons

Along with pictures and text, buttons are one of the three basic elements of HyperCard cards. Every Macintosh application has buttons you click to OK or Cancel an action, to Save a document, or to Open another. But as you just saw, HyperCard buttons can do a lot more. And they can look any way you want them to look (see Figure 1.9).

Some have text that hints at what clicking the button might do. Some look like ordinary Macintosh but-

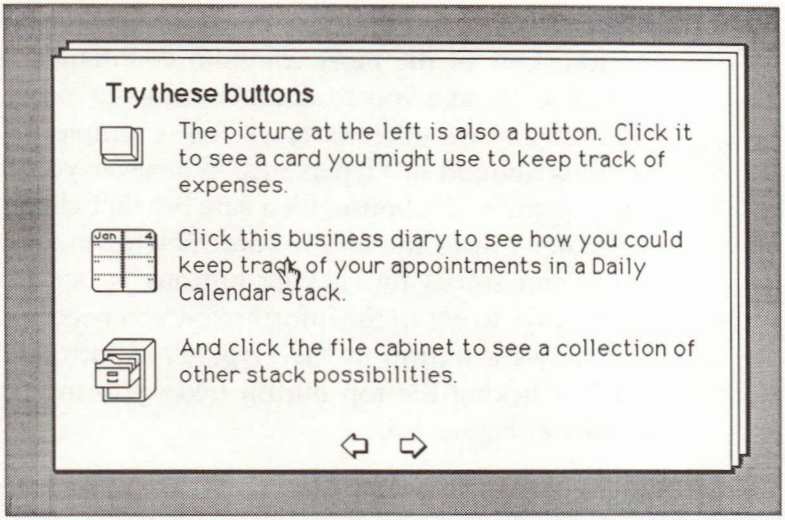


FIGURE 1–4 *This card has several buttons. The arrows at the bottom of the card take you to the previous or next card. The pictures at the left are also buttons that take you to other cards.*

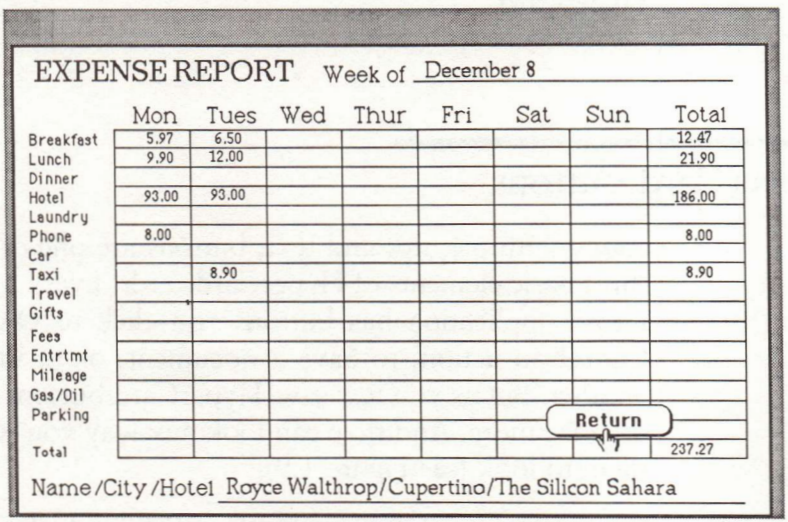


FIGURE 1–5 *Clicking the button shown at the top left of Figure 1.4 takes you to this card—an expense report for keeping track of a business trip's expenses.*

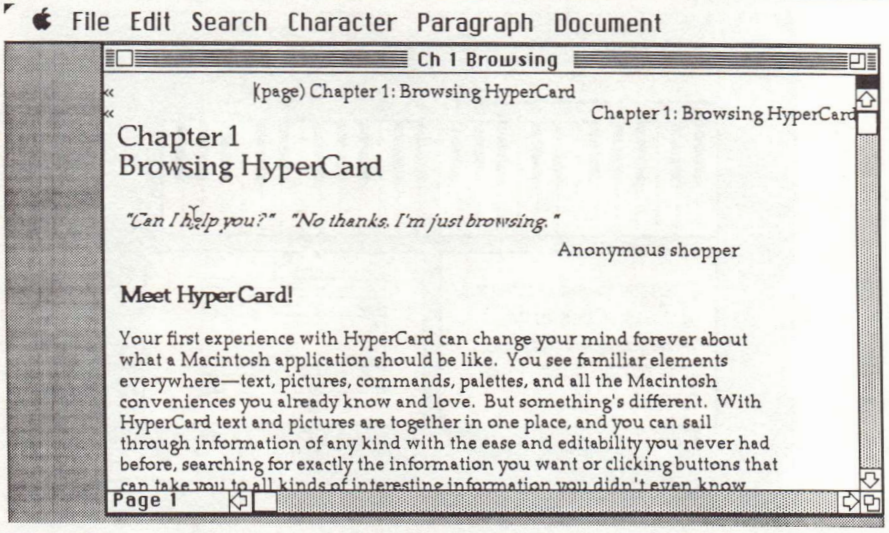


FIGURE 1–6 *A HyperCard button can open a word processor or any other application. When you quit the other application you come back to HyperCard.*

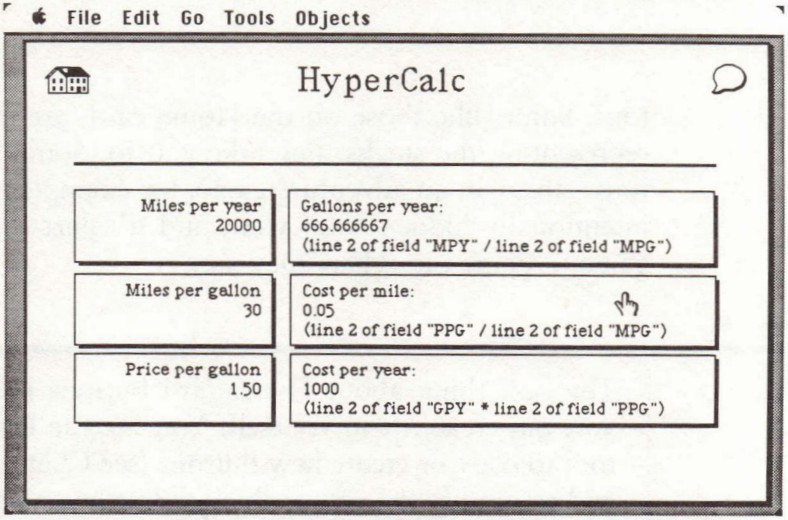


FIGURE 1–7 *You can use HyperCard to balance your books. This is a small but powerful spreadsheet built in HyperCard.*

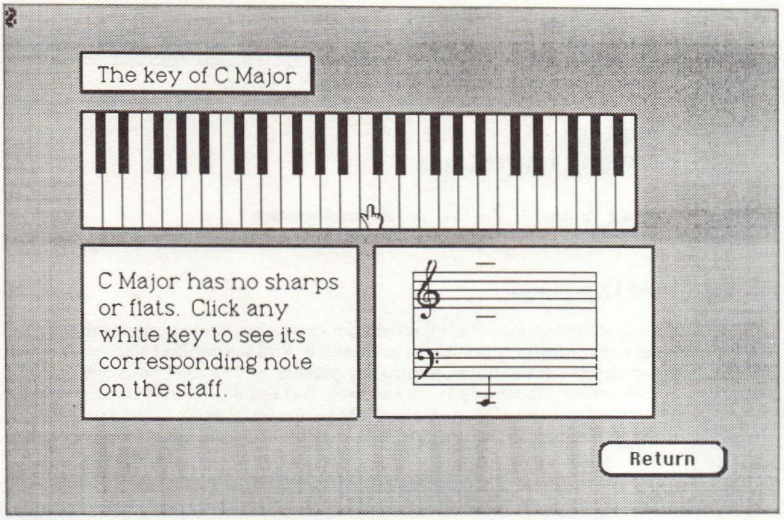


FIGURE 1–8 *A HyperCard stack that teaches basic ear-training skills in music. Clicking a note on the keyboard shows the note on the staff, so you can learn to correlate notes on the staff with their counterpart on the keyboard. Clicking the note also plays it, so you can train you ear to recognize pitches correctly.*

tons. Some, like those on the Home card, are icons representing the stacks they take you to. Some buttons—those in an adventure game, for example—are intentionally hidden from view, and it's part of the game to figure out where they are.

Hint

The best thing about HyperCard buttons is that you can create them yourself. You use the Button tool to copy or create new buttons (see Chapter 2) and to modify the scripts that tell buttons what to do (see Chapter 5).

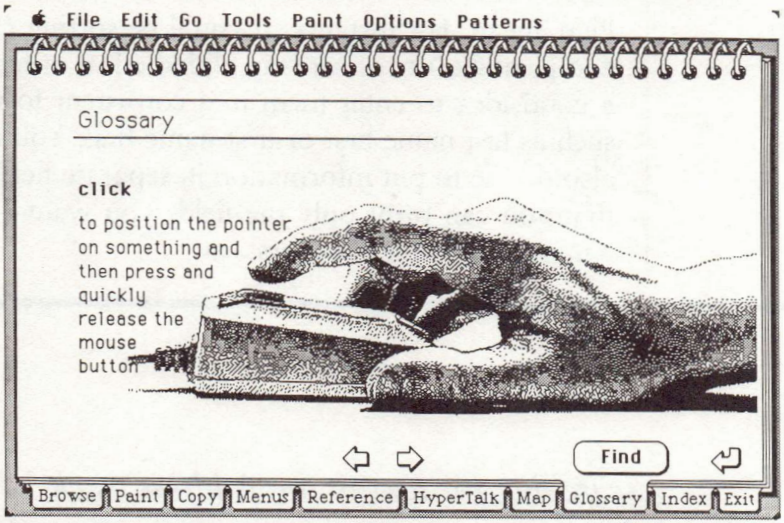


FIGURE 1–9 *Each of the tabs in the notebook are actually buttons that take you to various parts of the Help system.*

Finding Text

After you've browsed through cards by clicking buttons and choosing commands from the Go menu, you'll probably want to start looking for something specific. For example, rather than moving around in the Address stack by clicking forward to the next card or backward to the previous card, you might want to look up someone's name or find a phone number. It doesn't matter if you like to keep your Address stack by last name and your spouse likes it listed by first name. HyperCard can find what you're looking for no matter what order it's in. The information doesn't even have to be in the same order on all the cards—HyperCard will still find it.

Hint

Plan ahead. For instance, if you'll be sorting cards and printing lists of cards in alphabetical order, it's a good idea to enter them in a consistent form—such as last name first or first name first. You may also decide to put information in separate fields so that you can print only the fields you want—just names and phone numbers, say.

Searching for text in the Address stack

You can search for text while you're using any of the HyperCard tools by choosing Find from the Go menu. Here's how you might use this command to find Steve's name in the Address stack.

1. Open the Address stack by clicking its icon on the Home card.

The Address stack is represented by a small picture of an Address card.

2. Choose Find from the Go menu.

HyperCard automatically switches to the Browse tool (even if you don't have the Tool window open) and the pointer becomes the Browse hand. The Message box appears automatically whenever you go to the Address stack, so you can easily look up someone's phone number or address (see Figure 1.10). But even if the Message box weren't already visible, choosing Find from the Go menu would display the box.

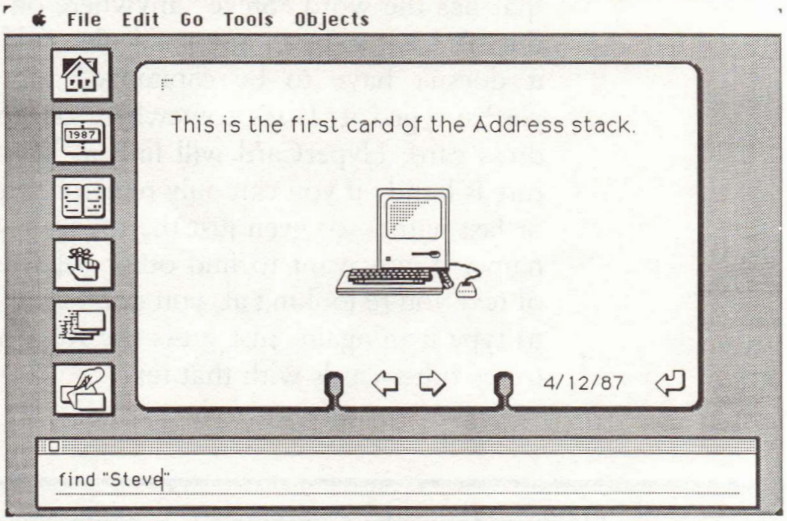


FIGURE 1–10 *You can use the Message box to specify text you want HyperCard to find in the current stack. The text can be anywhere on any card in the stack. Pressing Return finds the specified text.*

Hint

Whenever the Browse tool's pointer is a hand, HyperCard knows you're just browsing—clicking buttons or searching for text—rather than editing information that's already there or adding new information. The hand becomes an I-beam only when it's positioned over a text field that you can edit.

3. Type: Steve and press the Return key.

There is the card with your friend's name and address. HyperCard takes you to any card

that has the word "Steve" anywhere on it. It doesn't have to be in any particular field and it doesn't have to be capitalized the way you've typed it. If it's somewhere on an address card, HyperCard will find it. This feature is handy if you can only remember a first or last name—or even just the town or street name. If you want to find other occurrences of text you're looking at, you don't even have to type it in again. Just press the Return key to see other cards with that text.

Hint

Holding down the Command key while you click or drag across text on a card using the Browse tool specifies that text and then places it in the Message box, replacing any other selection there. This is a quick way to find other occurrences of the text in the stack.

Now suppose Steve's address is on Oak and you want to find other Oak addresses in your file.

4. Select the word "Steve" by dragging across it.

5. Hold down the Command key while you click the word "Oak."

HyperCard replaces the word "Steve" with the word "Oak" in the Message box. This technique replaces selected text if there is any; otherwise it replaces all text in the Message box, including the Find command. If this happens, choose Find from the Edit menu again, and then Command-Click.

6. Press the Return key.

HyperCard finds the next occurrence of the word "Oak" within the Address stack.

When HyperCard finds the text you're looking for, it highlights it by enclosing it in a rectangle. If HyperCard can't find the text, it beeps. (You might be tempted to say, "C'mon. You didn't really look." It's that fast.)

Hint

HyperCard highlights found text by drawing a box around it. The highlighting you're used to in other Macintosh applications—black and white inverted—is reserved for selecting text or graphics for further action such as copying or editing.

If you want to search another stack for "Oak," you can open another stack and press the Return key again. Even when you move from stack to stack, HyperCard keeps the Message box open and anything in it (such as the Find command) remains there.

HyperCard search rules

Here are the rules HyperCard follows when searching for text:

- When you choose Find from the Edit menu, HyperCard presents the Message box with the message: Find "|" There's an insertion point between the quotes so that you can type in the text you want to find.

Hint

HyperCard keeps "hint bits" to narrow down its search. (Hint bits store information about patterns of text that include three or more characters.) So you can specify just the first three letters of the word or words you're searching for. HyperCard is much faster if you give it at least three letters at the beginning of a word you're looking for. And the more words or word beginnings you give it, the faster and more selective it is.

- Type a few characters of one or more words you're looking for. Then press either the Return key or the Enter key to search the current stack for the text you specify.
- HyperCard finds the first card that contains the text you specified and highlights the first occurrence of the first word you specified. (It doesn't matter what order you specified. As long as all the specified text is contained on a single card somewhere in the stack, HyperCard finds it.)

Hint

Sometimes the Message box appears automatically because a button, card, or stack's script tells HyperCard to display it.

- If you press Return again, HyperCard finds the next occurrence of the text, if there is one, even though the Message box may no longer be visible.

- If you choose Find again, HyperCard displays the Message box with the text highlighted that you previously specified.
- Clear the Message box by selecting and backspacing. If there's no insertion point in a field, just Backspace or press the Clear key.

Hint

You can type any command, such as Go Home, in the Message box. An easy way to get to other stacks, even those that don't appear on the Home card, is to type Go followed by the stack name. HyperCard will take you directly to that stack.

Adding Your Own Text

After you've browsed awhile through the cards and stacks that come with HyperCard, you'll want to add your own information. For example, you'll probably want to include your friends' names or your favorite restaurants in the Address stack, and your own appointments in the calendars.

Hint

HyperCard automatically saves any change you make. If you want to be able to go back to earlier versions of stacks you've changed, be sure to make a copy of the stack first by choosing Save a Copy from the File menu before you change it. Then use the Open Stack command to open the copy with the new name you gave it.

Editing Text

Unlike many applications, HyperCard doesn't have a rigid agenda for editing text. You don't have to start by creating a new document, or specifying text fields, or building documents to get going. You can simply start by editing the information that's already there.

Hint

HyperCard is preset to keep you from accidentally changing anything. There's a card in the Home stack called User Preferences that lets you set the level of power. Go to the User Preferences card—it's the last card in the Home stack—by choosing Home from the Go menu. Then press the left arrow key to go immediately to the last card. Set your user level to Typing if it's not already set there, to allow you to add and edit text in text fields. Later you'll change this level to match the experience you've gained.

You can edit text in HyperCard the same way you edit text in other Macintosh applications. Type, select text, cut or copy it, paste it, delete it, or replace it. When you've cut or copied text, the Paste command in the Edit menu includes the word "text" to show that you've cut or copied text rather than a picture, a field, a button, or a whole card.

Text fields (and pictures) can be inherited by every card in the stack, or they can belong to individual cards. The text you type or paste into the fields belongs to that specific card. For example, in a stack of name and address cards, each card in the stack has

Hint

You can edit text wherever the pointer is an I-beam. When you move the Browse hand over a field that's not protected, it becomes an I-beam. When you click in an unprotected field you get an insertion point. To get the Browse hand back again, just move the mouse out of the field. HyperCard shows the I-beam over the field and the Browse hand anywhere outside the field.

three fields (name and address, phone number, and date) that are in the background and appear on every card in the stack. But the text contained in each field—the specific names and addresses—appears on and belongs to one particular card.

Here's a summary of the basic Macintosh editing techniques:

To copy text from one card to another (or to a different place on the same card):

- Select text by dragging across it
- Choose Copy Text from the Edit menu
- Go to another card
- Click where you want the copied text to go
- Choose Paste Text from the Edit menu

To move text:

- Select it by dragging across it
- Choose Cut Text from the Edit menu
- Go to the card you want to move it to
- Click where you want to insert it
- Choose Paste Text from the Edit menu

To delete text:

- Select it by dragging across it
- Choose Clear Text from the Edit menu or press the Backspace key

To replace text:

- Select it by dragging across it
- Type the text you want to replace it with or choose Paste Text from the Edit menu to replace it with the text that you last cut or copied

Press the Backspace key to erase a character you just typed if you made a mistake. Add new cards as you need them by choosing New Card from the Edit menu. Delete cards when you no longer want them by choosing Delete Card from the Edit menu. Don't delete pre-existing cards yet—you'll need them for the exercises in the upcoming chapters of this book.

Summary

In this chapter, you've learned how to move around in HyperCard using buttons, the Go menu, and keyboard keys. You've learned how to search for text using the Message box, and how to do some simple editing tasks such as copying, moving, deleting, and replacing text. You now may be itching to expand the things you can do in HyperCard. How do you get at the other tools? What about making some graphic changes to the existing stacks, or creating your own new stacks? You're ready to move from browsing information to customizing stacks and cards. From there, it's only a short step to creating your own stacks. With HyperCard there's no line dividing users

of information from creators of information. There's just a seductive path on which HyperCard's power unfolds. And as you begin to uncover more of HyperCard's power, you'll also begin to see your own.

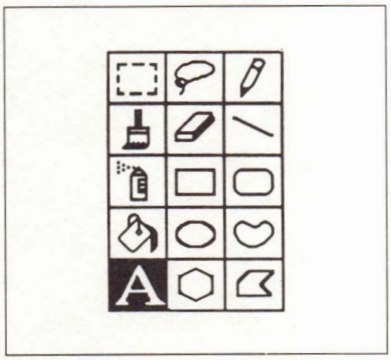
There's a phrase you often hear among HyperCard users who are discovering their own power in controlling the information in their computers. It starts something like this: "I'll bet I could . . ."

The next chapter tells you how to make changes to the existing stacks with graphics, how to change text fonts, and how to create and use buttons.

Chapter 2

Customizing Stacks

"The people who make the changes are the people who have the knowledge."
James Burke,
Connections



HyperCard's power unfolds in a gradual and gentle way, so you can grow into the application at your own pace. You've started by exploring, browsing through information in stacks that are already there. Next you'll probably want to add some of your own information,

modifying or adding cards to existing stacks. No doubt you won't be able to resist the familiar Paint tools to doodle on some cards. You might try the Field tool next, adding a new text field to a card in the Address stack or an appointment book. And sooner or later you'll want to try your hand with HyperCard's power tool—the Button tool. This chapter guides you through these tools.

HyperCard stacks aren't ever finished. Stacks aren't just created by creators to be just used by users. In fact, with HyperCard there is no real distinction between users and creators, readers or authors. There's just a steady progression (with an occasional leap) from browsing through existing stacks to adding your own information, to copying pictures and fields and buttons, to thinking "I'll bet I could . . . " With HyperCard you can customize stacks to do exactly what you want.

Using the Paint Tools to Add Pictures

You notice the Paint tools the first time you look in the Tools menu. Most of them look like the tools you're used to in MacPaint—for example, lines, rectangles, paint brush, spray paint. But there are a few new tools and a few enhancements to the way the familiar MacPaint tools work.

The differences between pictures in HyperCard and pictures in MacPaint can make experiments a little confusing until you understand how pictures work in HyperCard. There are both background and card pictures, and card pictures can be either transparent or opaque. When drawing a picture on a card, you'll

find the tools work just as you expect them to, and you can learn the intricacies later.

Hint

For the most part, the Paint tools work like the MacPaint tools you're probably used to. There are a few new tools and also a couple of new concepts—the idea of both a card and a background picture on the screen at the same time, and the ability to make a picture transparent or opaque.

Adding Pictures to the Weekly Calendar

You might start your artistic control of HyperCard by using the Paint tools to add a picture to the weekly calendar. For example, here's how you can circle an important date on your calendar so that it's easy to notice. You can consider the circle a picture because you use the Paint tools to create it as a graphic image.

1. Go to the Home card (if you're not already there) by choosing Home from the Go menu or pressing Command-H.

This always takes you Home no matter where you are in HyperCard.

2. Choose Last from the Go menu (or press Command-4) to go to the last card in the Home stack where you can set your User Level Preference.

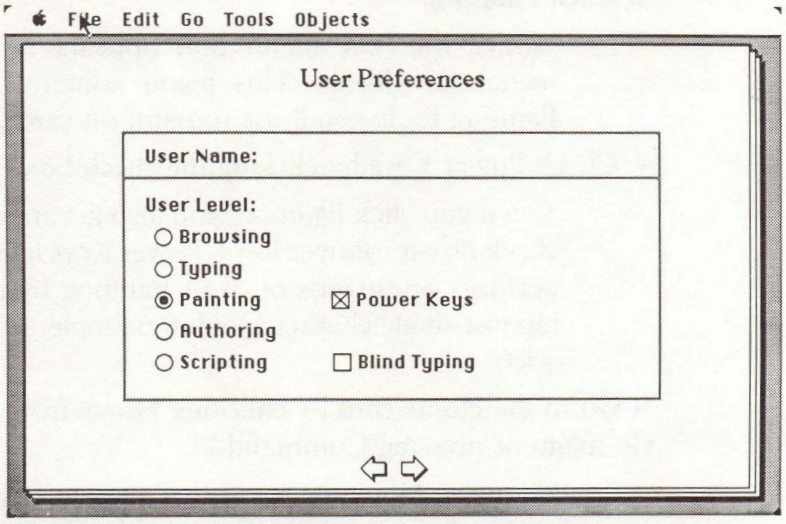


FIGURE 2–1 *The User Preferences card is where you tell HyperCard how much power you want. You can start gradually by just typing text into existing fields, then add the Paint tools when you want to add or change pictures. The Authoring level gives you access to the Field and Button tools. Scripting is for editing scripts using HyperTalk.*

The Preference Card (shown in Figure 2.1) includes the five User Levels—Browsing, Typing, Painting, Authoring, and Scripting. Each level adds capabilities to what you can do with HyperCard.

Hint

You can also get to the User Preferences card—the last card of the Home stack—by pressing the left arrow key while you're viewing the Home card (the first card in the Home stack). And whenever you're at the last card in a stack you can get to the first card by pressing the right arrow icon. Cards in a stack are arranged in a circle.

3. Click Painting.

Notice the new menu that appears in the menu bar—Tools. This menu contains the Painting tools you'll use to paint on cards.

4. Check Power Keys by clicking the check box.

When you click Painting, you also get another check box for Power Keys. Power Keys let you perform commands on your painting by typing just single characters—for example, "s" to select.

5. Go to the Home card by choosing Home from the Go menu or pressing Command-H.

Choosing Home or pressing Command-H always takes you to the first card in the Home stack. The Preferences card is the last card in the Home stack.

6. Go to the weekly calendar by clicking its icon on the Home card.

The calendar opens to the current week. (HyperCard is smart enough to check the current date and then take you to the appropriate week.) As you move the pointer over various parts of the card, it becomes an I-beam whenever it's positioned over a text field. This gives you a clue that you can click in any of those places and start typing text.

Hint

If the calendar doesn't open to the correct week, reset your Macintosh clock by choosing Alarm Clock from the Apple menu and clicking on the numbers to change them.

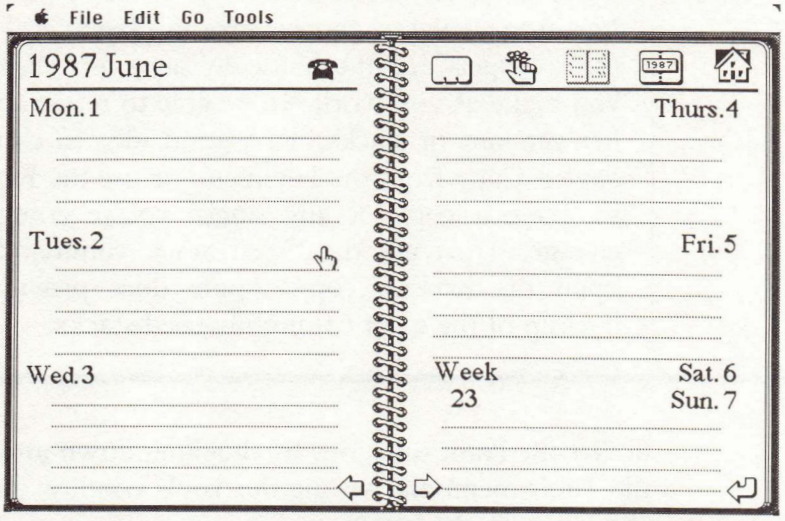


FIGURE 2–2 *The weekly calendar is represented by a small picture on the Home card that says "Weekly." Clicking the picture, which is actually a button, takes you to a Weekly Calendar, where you can keep track of appointments.*

Figure 2.2 shows the weekly calendar with the Browse hand. Moving that hand into any of the text fields changes it to the I-beam pointer.

7. Flip to any card in the weekly calendar you'd like to add a picture to.

You can move to other cards by using the arrow keys on the keyboard (cursor keys) to move to previous weeks (left) or forward to coming weeks (right). Or use the Browse tool to click the arrow icons on the screen to move in either direction. (If you keep clicking on the right arrow icon, you will eventually get to the To Do cards and the six-month calendar cards—they are all part of the Datebook stack.)

This chapter's example shows circling a date to remember, but you could add any picture to any day you'd like to have a picture on.

Hint

You never have to choose Save in HyperCard because HyperCard automatically saves any changes you make as you work. To be able to retrieve earlier versions of stacks, it's a good idea to choose Save a Copy from the File menu or use the Finder to make a copy of any stacks you're going to change. Then use the Open Stack command to open the backup copy. Apple does provide a backup of the most commonly used stacks.

8. Get the Paint tools out by dragging down and off the Tools menu and setting the Tools window down anywhere on the screen.

In Fig. 2.3 the Paintbrush tool is highlighted to show it's the selected tool. Checking any tool in the Tools window selects the tool.

Hint

HyperCard has "tear-off" windows. That means you can select Tools in the usual way, or you can drag the window to another place on the page (tear it off) and then select. To get rid of the Tools menu to keep it out of the way, click on the small box in the top left corner of the menu.

Hint

While you're making design changes to stacks—whether you're editing their pictures, fields, or buttons—keep the Tools window handy by leaving it out on the desktop. Tear it off and drag it out of the way, then select a tool. The palette will stay where you put it.

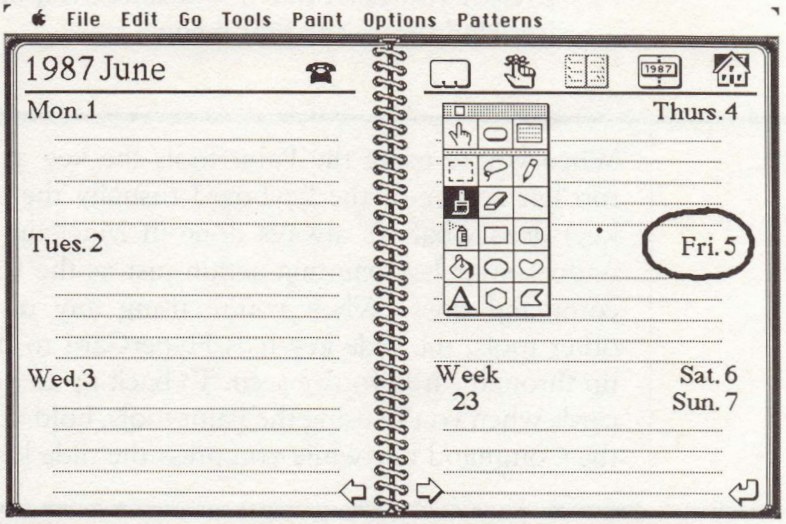


FIGURE 2–3 *You can use any of the Paint tools to add your own pictures to cards in the Weekly calendar. Dragging down and off the Tools menu "tears off" the menu so you can move it around on the screen.*

If you have a large-screen Macintosh, set the Tools palette out of the way so you can see and work on the entire card. Of course you could continue to choose the tools you need from the Tools menu, but it's more convenient to tear the menu off, set the window down, and move it around the screen as you work. That way a single click gives you the tool you need.

9. Experiment with the Paint tools by clicking the Paintbrush tool.

Notice that Paint, Options, and Patterns appear on the menu bar when you click on the Paintbrush.

10. Circle a date on the card as shown in Figure 2.3.

Press the tilde key (or whatever key is at the top left corner of your keyboard) to undo the

circle if you don't like it and start over, or select Undo from the Edit menu.

Hint

When you're using the Paint tools the key at the top left corner of the keyboard (usually the tilde key) does what it's always done in MacPaint—it undoes your last painting action just as the Undo command does. When you're using any of the other tools, the tilde key tells HyperCard to back up through cards you've seen. To back up to other cards when you're using the Paint tools, hold down the Command key while you press the tilde key.

More than Undo

With MacPaint you can always undo your last painting action, and you can always Revert to the last version you saved on a disk. But if you want to undo several changes, or if you want to undo just one change but you've clicked somewhere else on the document, you're stuck.

With HyperCard it doesn't matter if you forget to undo an action until you've already made more than one change to your drawing. The Revert command gives you an additional level of Undo. Just choose Revert from the Paint menu. You can recover from mistakes or simply change your mind as long as you stay with the Paint tools and don't leave the card you're working on. HyperCard remembers the way the card was when you came to it, and choosing Revert takes the card back to that state. If you make some changes and want to remember those changes but continue to work on the card, choose Keep from the Paint menu. It remembers the current state as the one

to revert to if you subsequently choose Revert. This gives you a lot of control over work in progress.

Hint

Whenever you're about to embark on major changes or do something you might want to back out of, choose Keep from the Paint menu to save what you've done so far. Choosing Revert from the Paint menu goes back to the way the card looked when you last chose Keep or when you last came to that card.

Borrowing Pictures that Already Exist

If you're the artistic type you can start from scratch and create your own graphics—with all the power of the Paint tools right within HyperCard. But if you're not an artist, you can still have great graphics in your stacks. HyperCard has a collection of images (the Clip Art and Art Ideas stacks) to copy and use in your own stacks. You can also copy any existing MacPaint documents. And you can purchase other collections as well. Chapter 9 gives instructions on how to move things into and out of HyperCard.

You can copy pictures from existing HyperCard stacks such as Art Ideas, Card Ideas, Stack Ideas, Help, or any of the other stacks. You can copy images and use them as they are, or you can copy them and then use the Paint tools to add your own touches to them or just rearrange them on a card.

Practice borrowing art from the Art Ideas stack by placing the picture of a birthday cake on your weekly calendar as a reminder of an upcoming birthday.

1. Select the Browse tool again.

HyperCard knows what you want to work on by the tool you've selected. When one of the Paint tools is selected, it knows you're working on pictures. And when the Browse tool is selected, it knows you want to move around among cards and stacks—clicking buttons or searching for text or editing text.

2. Choose Message from the Go menu.

The Message box has several functions in HyperCard. One is to provide a place for you to specify text you want to find using the Find command. But the Message box is also a place for you to specify actions for HyperCard to perform. For example, you can go to other stacks without using the Home card or the Open Stack command by typing "Go" followed by the name of the stack you want to open and then pressing the Return key.

3. Type: Go Art Ideas

4. Press the Return key.

Pressing the Return key tells HyperCard to carry out your command—Go to the first card in the stack named "Art Ideas," in this case. Figure 2.4 shows the first card in the Art Ideas stack.

This art stack includes a directory of art topics. You can click a topic to go to a card filled with related art. Or you can browse through the Art Ideas cards by clicking the arrows at the top right of the card.

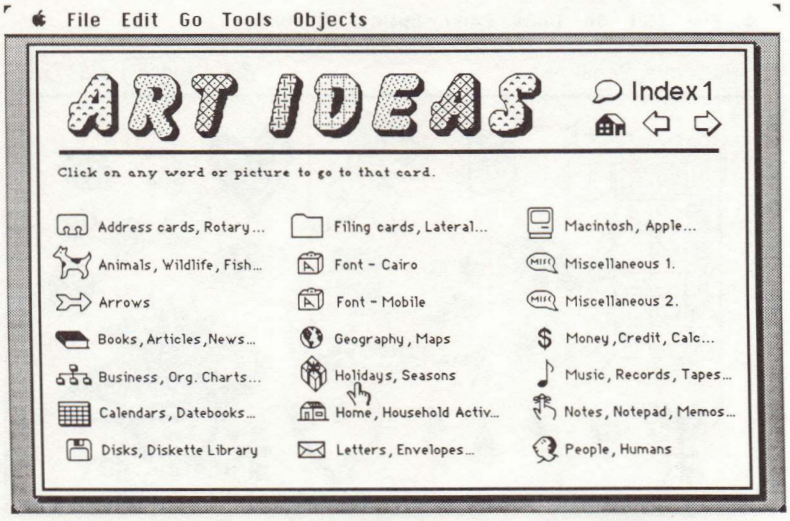


FIGURE 2–4 *The first card in the Art Ideas stack, a treasure box full of snippets of art for you to copy and use in your stacks.*

5. Click the picture of the gift. It says "Holidays, Seasons" next to it.

HyperCard takes you to a card filled with presents and a birthday cake.

6. Click the selection rectangle in the Tools window.

(First tear the Tools menu off again if you've put it away.) You can use the selection rectangle or the lasso to capture any image you want to use, but the selection rectangle is still the easiest way to select an image. And with HyperCard you can use the selection rectangle to select something and then lasso the selection so you can easily move it around and into tight spaces.

7. Select the birthday cake by dragging diagonally across it.

Figure 2.5 shows the birthday cake selected with the selection rectangle. Be sure to start

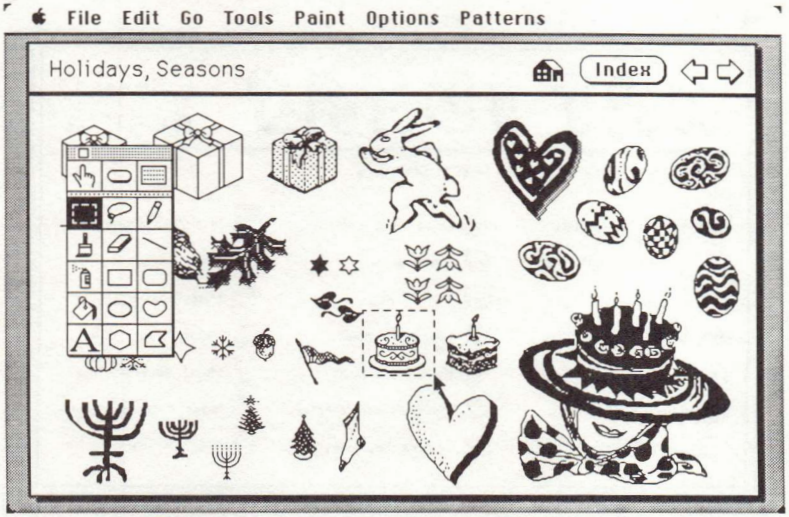


FIGURE 2–5 *You can use either the selection rectangle or the lasso to select a picture. The selection rectangle is quicker, and the lasso lets you move about in tight spaces. Combine the best of both tools by selecting with the selection rectangle and then changing to the lasso by typing Command-S.*

dragging well outside of the limits of the birthday cake. If the selection isn't large enough and part of the cake is excluded, just start selecting again. If you accidentally move the selection, choose Undo from the Edit menu.

Hint

Whenever you want to copy a picture in HyperCard, first choose one of the Paint tools from the Tools window. HyperCard needs to know whether you want to copy text, a text field, a button, an entire card, or a picture. When any Paint tool is selected, the Cut and Copy commands in the Edit menu apply to pictures.

8. Type Command-S (for Select) to lasso the birthday cake.

The selection rectangle suddenly hugs the cake. With the selection rectangle you select the space enclosed by the rectangle, even if there's no image. But the lasso selection snuggles up alongside the exact image, excluding any white area.

Hint

In HyperCard you can change a rectangle selection into a lassoed selection by typing "Command-S." You can tell HyperCard to remember the Power Keys, so that it's automatically checked each time you start HyperCard. Do this by checking the Power Keys preference on the Preference card—the last card in the Home stack. Then you don't even have to press the Command key to change a rectangle selection into a lassoed selection; just type "S."

It's generally easier to select using the selection rectangle, but lassoing an image lets you move it into a tight spot. And lassoing also lets you select just the object, without any area that surrounds it. With all the new Paint commands available in HyperCard's Paint menu, you'll often want just the object selected, so you can choose a subsequent command for it without affecting any surrounding picture.

9. Try a few commands in the Paint menu or, if you checked Power Keys on the Preferences card, just type single characters to transform the selected birthday cake.

Try Lighten and Darken. Choose Trace Edges for an interesting effect; flip it or rotate it. The

Power key for Lighten is "L." For Darken it's "D." Trace Edges is "E."

10. Choose Revert from the Paint menu.

This reverts the selection (or the entire card if nothing is selected) back to the way it originally was or the way it was when you last chose Keep.

Hint

As long as you don't select a tool other than a Painting tool and you don't leave the current card, you can choose Revert. This will either revert your last Painting action (or the current selection if there is one) or revert the entire card to the way it was when you came to it last.

---

11. Use the selection rectangle to reselect the cake and choose Copy Picture from the Edit menu.

The selected image is copied onto the Clipboard—not visible but there nevertheless. You'll see the contents of the Clipboard the next time you choose Paste.

Hint

It's a good idea to get into the habit of copying rather than cutting images. That way you'll still have the original to copy again.

Pasting borrowed art

A general rule with Macintosh is to select some information and then tell the Macintosh what you want to do with that information. You just copied a

picture by selecting it and then choosing Copy. With HyperCard you can copy not only pictures but text, buttons, fields, or entire cards. Depending on the tool you're using when you select, HyperCard copies the appropriate item. HyperCard remembers what you last copied and switches you to the appropriate tool, if necessary, when you choose Paste.

Now paste the birthday cake picture that's on the Clipboard into the weekly calendar.

1. Choose the Browse tool.

You've finished Painting, so you need to give HyperCard a hint about what you want to do next.

2. Choose Recent from the Go menu.

You see miniatures of the 42 most recent cards you've seen. Clicking any one of the miniature pictures takes you to that card.

3. Click the minature of the Weekly Calendar card.

This weekly calendar is set up so that you can see the week at a glance, or see the larger view of the current six months by clicking on the small calendar icon at the top of the weekly calendar. (Clicking a corresponding small calendar icon at the top of the Yearly calendar likewise brings you to the Weekly calendar.)

4. Click the Yearly calendar icon near the top of the calendar to go to a card that represents the current six months.

This takes you to one of the last two cards in the Weekly Calendar stack, depending on the current month, as shown in Figure 2.6. These cards each show six months of the year. From these two cards you can click on any day to go back to the Weekly calendar that represents

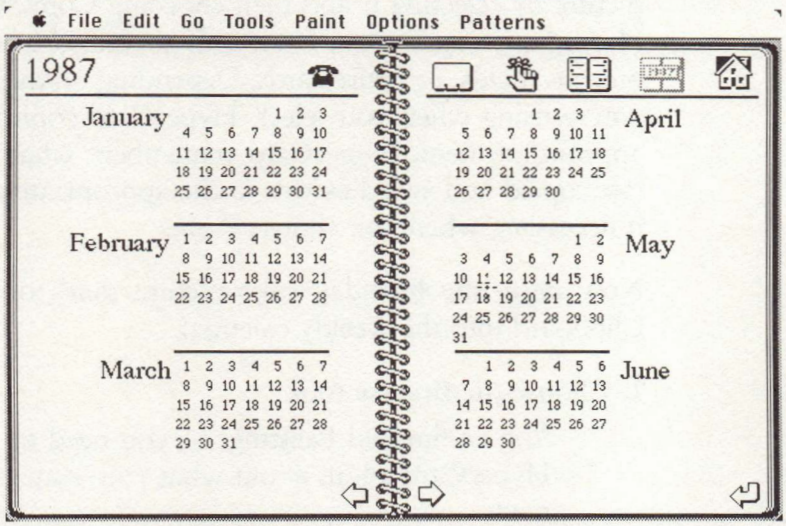


FIGURE 2–6 *The Yearly calendar lets you see the year at a couple of glances. Each card represents six months of the year. Clicking a date takes you to that day in the Weekly calendar.*

that day. Use the right or left arrow to go to the other six months of the year.

5. Click the day you want to celebrate.

Let's say your sister's birthday is on Oct. 5. you would click there to tell HyperCard to remember that place on the calendar card.

Hint

Both of these calendars, plus the To Do list, are actually in the same stack—called Datebook. Three buttons on the Home card take you to various parts of the Datebook stack—the To Do button, the Weekly button, and the Calendar button.

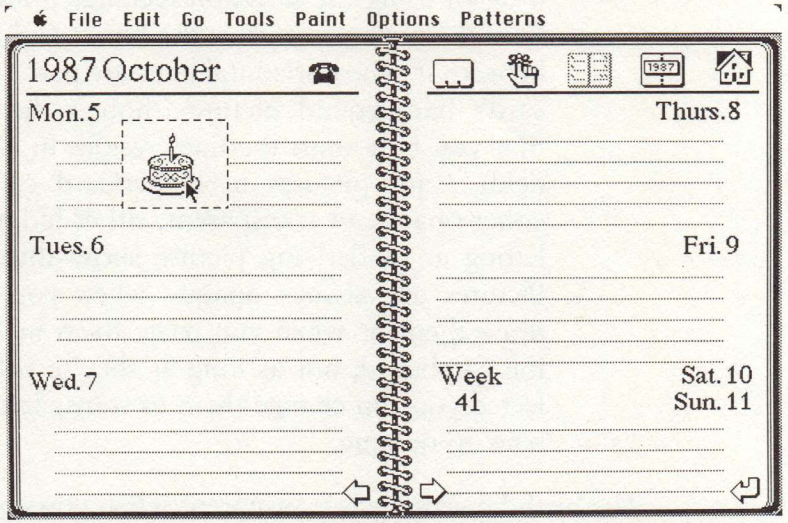


FIGURE 2–7 *A picture is opaque when you first draw it or paste it onto a card. But you can make any picture transparent so that you can see through to the background picture.*

6. Choose Paste Picture from the Edit menu.

HyperCard switches you back to the selection rectangle tool (because you're pasting a picture) and pastes the image of the picture you copied onto the current card as shown in Figure 2.7. Don't click outside the image of the cake yet. There's another concept to explore first.

Opaque or transparent

1. Without clicking outside the selection, move the picture around on the screen by dragging from anywhere inside it.

(If you have already clicked somewhere and the birthday cake is no longer selected, select

it again using the selection rectangle.) Notice how the picture obscures the horizontal lines beneath it. The horizontal lines are part of the card's background picture (though the text that you type onto the lines resides in a text field). Card pictures in HyperCard can be either opaque or transparent, either hiding or letting an underlying picture show through. Pictures are always opaque when you first draw them or when you paste them in from the Clipboard, but as long as they're still selected you can change them to transparent or back to opaque.

The birthday cake was transparent when you copied it (though you probably didn't notice it because there was nothing behind it to show through), but it became opaque while it was on the Clipboard. When you pasted it into the calendar, the opaque image obscured the underlying picture—the horizontal lines.

Hint

For consistency with MacPaint and other pictures in the Clipboard, pictures in HyperCard are always opaque in the Clipboard. After you paste the pictures, you can change them back to transparent if you want.

2. With the pasted picture still selected, choose Transparent from the Paint menu.

If you've checked Power Keys on the Preferences card on the Home stack (or if you choose Power Keys now from the Options menu), you can just type a "T" to do the same thing.

Hint

Checking Power Keys in the Options menu turns Power Keys on for this session only. HyperCard remembers the Power Keys setting on the Preferences card even when you quit HyperCard and start it again.

3. Move the selection around the screen again by dragging it.

Now the underlying calendar picture shows through—lines, spirals, and icons. As long as a recently pasted picture is still selected, you can move it around and change it from opaque to transparent and back again. If you click anywhere outside the picture and then you decide to move it or make it transparent, just select it again.

Card or Background Picture?

Every stack has at least one background picture that cards can share. For example, every Address card has a picture that looks like a card in a rotary file. But a card can have an individual picture as well as the background picture it shares with the other cards in that stack.

While you've been using the Paint tools to edit pictures, you've been working with card pictures. You can make changes also to the cards' background picture to change the way each of the cards that share that background will look. Background pictures give

you several advantages. They save space on the disk by storing just one picture for many cards, and they give you the ability to edit every card's picture by editing just the background.

Hint

Whenever you want to use the same picture for more than half the cards in a stack, you should put the picture in the background.

Adding a picture to the background

You just added a picture to one of the Weekly calendar cards, but you could also add a picture to every card that shares the same background by editing the stack's background picture.

Hint

The Weekly calendar, the Yearly calendar, and the To Do list are all part of one stack called "Datebook." This is an example of using more than one background within the same stack. The Weekly calendar has one background; the Yearly calendar has another; the To Do list has yet another background.

In this exercise, you'll learn how to copy a picture of a moneybag and paste it to the background of each Weekly calendar card to remind yourself to go to the bank.

1. Choose the Browse tool.

2. Choose Message from the Go menu.

If "Go Art Ideas" is still in the Message box from the last time you used it, just press Return. Otherwise, type it in and press Return. HyperCard takes you to the first card in the Art Ideas stack.

Hint

When there's no insertion point in the Message box or anywhere else, HyperCard replaces any existing message in the Message box with any text you type.

---

3. Click the dollar sign.

It says "Money, Credit, Calc" next to it. HyperCard shows you a card with images relating to money and banking.

4. Use the selection rectangle to select a money bag.

Just as you did for the birthday cake, tear off the Tools menu and click on the selection rectangle.

5. Choose Copy Picture from the Edit menu.

The money picture is copied. Although you can't see it there, it replaces the picture of the birthday cake on the Clipboard.

6. Choose the Browse tool.

If you stay in a Paint tool, pressing the Undo key (the top left key on the keyboard) will undo your last painting action. If you use the Browse tool, the top left key will take you back quickly to the last card you were working on.

Hint	HyperCard maintains as much consistency with MacPaint as possible (while adding some useful new features). One important consistency is that the top left key still "Undoes" painting actions anytime you're using a Paint tool. You can go back to another card while you're in a Paint tool by pressing the Command key as well as the Undo key.

Or use any of the other ways you know about to get back to the calendar:

- Choose Back from the Go menu repeatedly until you're back to the calendar.
- Go Home and then click Weekly.
- Choose Open Stack from the File menu and then open Datebook and click buttons to move around.
- Choose Recent from the Go menu and click the card you want.
- Choose Message from the Go menu and type "Go Datebook."

HyperCard always offers many alternative ways to get places.

7. Choose Background from the Edit menu.

Any text on the card temporarily disappears, as does the picture you added to the card. You see a blank Weekly calendar card. And the menu bar has a hem-stitched look.

You can add any picture to either single cards one at a time (as you did with the birthday cake) or every card that shares the same background. When Back-

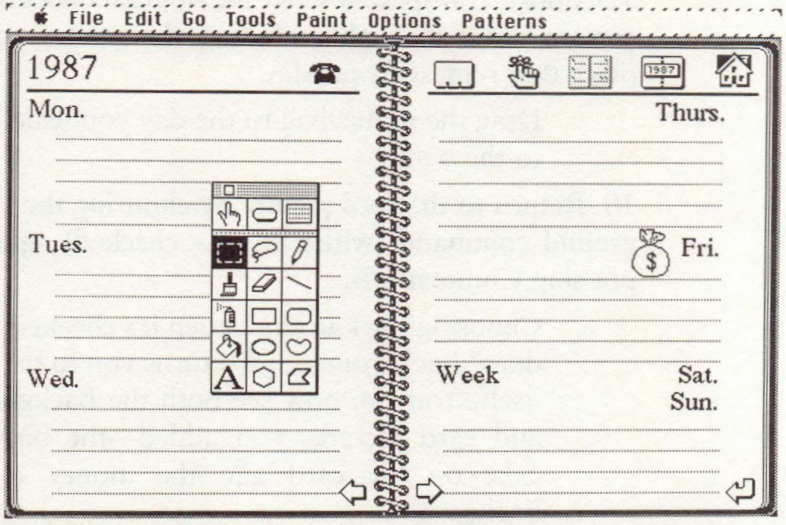


FIGURE 2–8 *You can also paste pictures into a stack's background so that each card that shares the background will share the picture. The diagonal lines at the top remind you you're working in the background.*

ground is checked in the Edit menu, anything you do to the picture affects every card with that background. (By the way, a stack can have as many backgrounds as you need. See Chapter 7.)

Hint | You always know when you're in the Background, because slanted lines appear in the menu bar.

8. Paste the money bag in by choosing Paste Picture.

Figure 2.8 shows the money pasted into the Background.

9. Choose Transparent to let the original background picture show through and drag the money to the place that represents payday.

Drag the moneybag to the day you usually go to the bank.

10. Return to the card picture by choosing the Background command (which is now checked) again or pressing Command-B.

Choosing Background when it's checked "Undoes" Background and returns you to the card itself. You can now see both the Background and card pictures you added—the birthday cake on the card and the money in the Background.

Hint

When you're looking at a card's background, you see only the background picture. But when you're looking at the card, you see both the card picture and any background picture underlying it (unless you've made part of the card picture opaque).

11. Choose the Browse tool.

If you're still using a Paint tool, clicking on arrows or icons will leave Paint marks but won't take you anywhere. Only the Browse tool lets you click on buttons to move around in HyperCard.

12. Flip through the cards by clicking the left or right arrows.

Notice that although the birthday cake is on just one card, the money bag appears on every card in the Weekly calendar stack.

Making the Card Picture Opaque

Even though the money picture appears in the Background, you can hide it on individual cards. You make an opaque layer on top of the background image you want to hide. For example, if you're not going to the bank this week, you can "erase" the reminder for this week only.

1. Use the selection rectangle to select the moneybag picture on this week's calendar.

Even though you see and seem to be selecting the moneybag, that picture actually appears in the background, and right now you're editing the card picture rather than the background picture. Rather than selecting the picture itself, you're selecting an area of blank space on the card picture.

2. Try moving the selection.

Notice that you don't move the moneybag, which is in the background, but only the blank space you've selected on the card.

Hint

In HyperCard the "marching ants" that surround a selection seem to disappear when you drag the selection. (This is different from MacPaint.) The marching ants make it easier to see exactly what's around the image and to move it into just the place you want it.

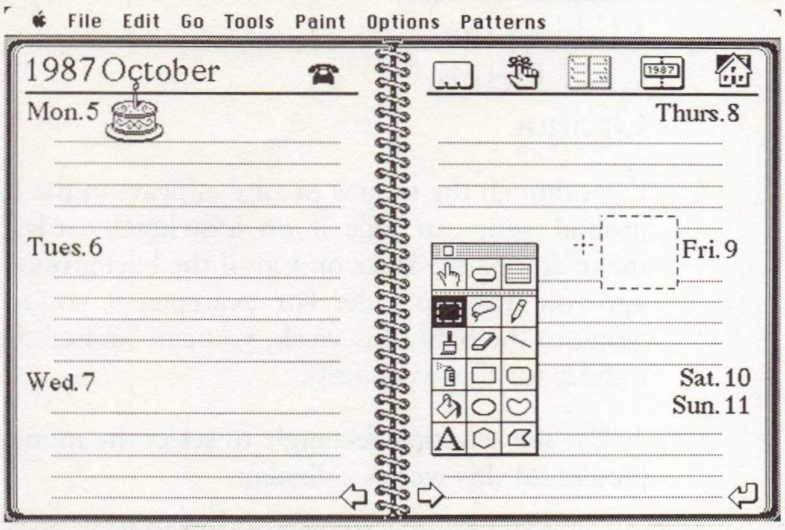


FIGURE 2–9 *You can cover up areas of the background by making that area of the card opaque. This card's background includes the picture of money, but because that area is obscured on the card it doesn't show through.*

3. Reselect the area over the money bag picture.

Drag from the top left to the bottom right to select the area.

4. Choose Opaque from the Paint menu or press "O" if you have Power keys checked.

Now you see one of the reasons for layers and for opacity. The selected area becomes opaque, obscuring the image under it—in this case, the moneybag in the background picture is hidden. You can see this in Figure 2.9, where the moneybag is no longer visible.

Shrinking a picture in the background

When a picture is part of the background, you have to edit it in only one place to change it on every card.

1. Choose Background again to edit the background picture.

HyperCard brings you back to the card's background picture. The money is still there because you hid it on one card but did not remove it from the background.

2. Make a small change by selecting the moneybag and then pressing the Command key while you drag the selection rectangle inward to shrink it.

If you also hold down the Shift key while you drag, the selection will keep its original proportions. To enlarge the image, just drag outward while pressing the Command key.

3. Return again to the card picture by choosing the checked Background command again or pressing Command-B.

This card still obscures the picture with an opaque area.

4. Use the Browse tool to move to other cards in the stack and notice that the picture is now smaller on every card, even though you changed it in just one place.

That's because you changed it in the Background of the stack.

Using the Temporary Layer

Besides the background and the card pictures, there is one more, temporary layer whenever you paste a picture in. While it's selected, you can move and manipulate it as you wish. To see this layer, paste in another copy of the moneybag on the calendar.

1. Choose Paste Picture from the Edit menu or press Command-V.

There's another moneybag. Like other Macintosh applications, HyperCard keeps any picture (or anything you copy or cut) in the Clipboard until you switch off the Macintosh or until you cut or copy something else. Pasting the pictue doesn't remove it from the Clipboard.

The picture floats above the calendar. Just as in MacPaint, pictures you paste float around in a layer that belongs just to them until you click somewhere else on the card.

2. Make the picture transparent by choosing Transparent from the Paint menu and move the picture around on the screen.

Now you have another layer to play with. The existing card picture will show through or remain hidden until you click to finalize your choice.

3. Press the Backspace key to get rid of the selected picture.

Now the second moneybag is gone.

Giving Yourself More White Space

You may decide that the Address cards that come with HyperCard are too small to keep track of everything you want to keep there. Maybe you'll want to include people's birthdays on their cards. Or when you get an idea of what to give someone for Christ-

mas, you might want to jot it down. Address cards are a good place for shirt sizes or a favorite perfume or wine. When you're searching for text, HyperCard isn't a bit fussy about where you put text, as long as it's in a text field—so you can find notes no matter where you put them.

Hint

Each card in HyperCard is the full screen size of the Macintosh Plus. Many cards appear to be smaller because they include a gray background around their edges or they don't look like "cards" in the way we're used to thinking of cards. So even though the Address cards look as if they extend to just the white area on the screen, the actual card size is the full Macintosh Plus screen size.

Make an Address card look larger by extending the white area, as follows:

1. Choose Message from the Go menu. Type "Go Address" and press the Return key.

HyperCard takes you to the first card in the stack named "Address."

2. Choose Background from the Edit menu or press Command-B.

HyperCard takes you to the card's background.

3. Click the selection rectangle in the Tools palette and select the right portion of the Address card as shown in Figure 2.10.

You don't have to select a large area. You're just going to extend the card by copying an edge.

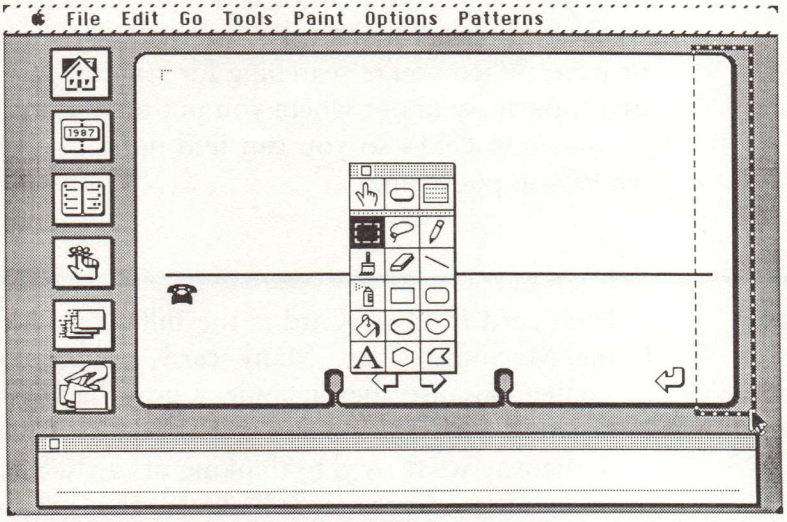


FIGURE 2–10 *One side of the Address stack's background picture is selected and being extended to the right so that there's more space for information on the card.*

Hint

Choosing Grid from the Options menu before you select avoids gray "seams," messy lines that show the edges between the original card and the copied portion of the card.

4. Hold down the Option key and the Shift key while you drag the selected area to the right.

Select an Option-Shift drag to do the same thing along the top of the card. Extend it just a little so that the card sits squarely in the center of the screen and the angled bracket that defines the text field is nudged up a bit.

Hint

Holding down the Option key copies the selected area. Holding down the Shift key ensures that the extended card picture stays lined up with the original.

Now you'll have more space to work with on each card. You can make these artistic adjustments to any background picture anywhere. You're the boss.

5. Choose the checked Background command from the Edit menu to return to the card picture.

Now your address cards are a little larger than the original.

Using the Field Tool

Even though HyperCard lets you intermingle text and graphics freely, it does need to know what areas of a card have text in them so that it can search for text and let you change such characteristics as font, size, and style. So even though the Address card looks larger now, the areas where you can add text—the fields—are still the same size. To be able to add text to the entire area of your enlarged card, you need to enlarge the text fields as well as the background picture. Making the text field larger increases the amount of text you can have on the card. It also changes the way the text is positioned—where line breaks fall, whether the shape is more a horizontal or a vertical box.

Enlarging the Text Field

A text field's size is never frozen. You can use the Field tool any time to change the size and shape of any field, whether it's a background field or a card field.

1. Set your User Level to Authoring by going to the Home stack, pressing the left arrow key, and clicking Author on the References card. Then use the Go Back key to return to the Address stack.

The Author User Level makes the Field and Button tools available. If you click either of these tools without at least the Authoring User Level, HyperCard presents a dialog box telling you that your User Level isn't high enough.

2. Select the Field tool from the Tool palette.

When you're using the Field tool, HyperCard displays and lets you edit any existing fields in the stack. The Address stack has three fields—one for the name and address, one for the phone number, and one for the date last modified. You can see these three fields enclosed with a black line in Figure 2.11.

You use the Field tool (choose it from the Tools window) whenever you want to add fields to a stack or make changes to existing fields. You might want to add fields so that you can separate information for printing labels without phone numbers, for example, or for performing calculations on numbers in a field.

3. Drag from a corner of the first field upward and outward toward the corner of the card so that the field looks like the one in Figure 2.12.

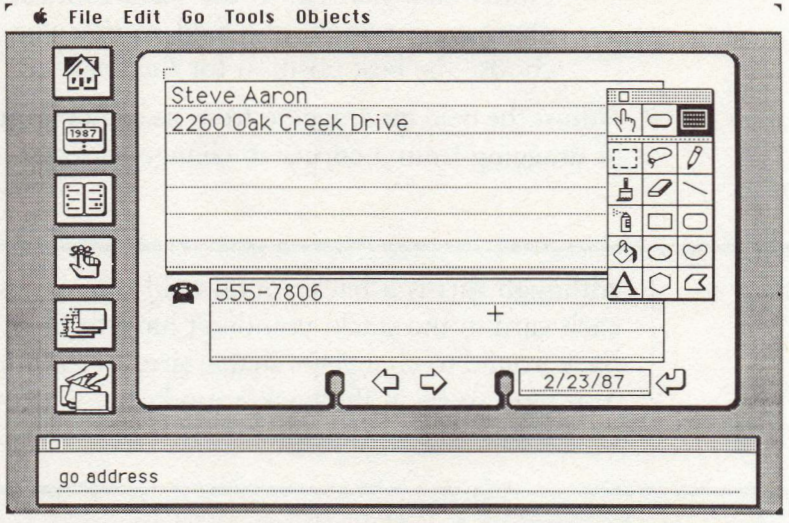


FIGURE 2–11 *When you use the Field tool all the fields are outlined so you can drag them to move them or adjust their size and shape.*

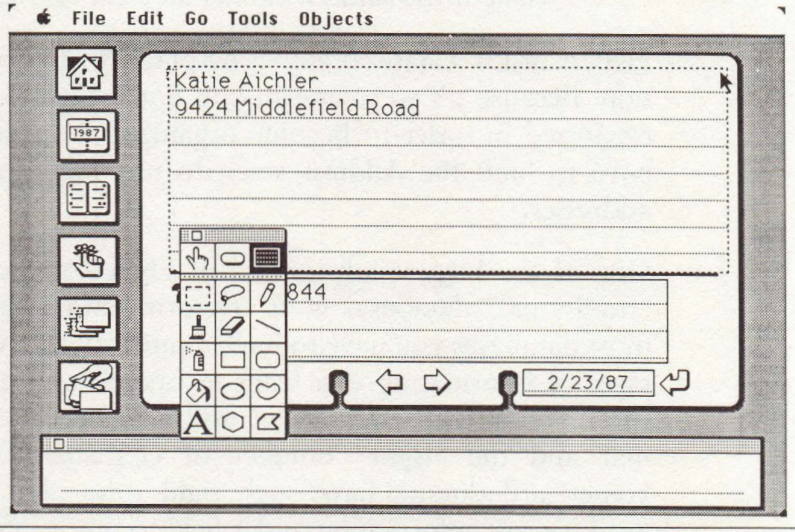


FIGURE 2–12 *You use the Field tool to drag fields to change their size and shape. Making this field larger by dragging from a corner lets you use more space on the card for text.*

Notice that the text in the Field tool follows along as you move the field so that you can choose the best position for the field.

4. Adjust the field by dragging from its center to move it or dragging from a corner to change its size.

Hint	Although this is a background field that exists on each card in the stack, you don't have to be in the background to change its shape, size, style, or font. You need to be in the background only when you create a background field.

Now you'll have more space to add text to the card.

5. Make the phone number field larger as well, by dragging its right edge more to the right.

Make it the same width as the field above it.

HyperCard is a natural gathering place for information. Because it's so freeform, you don't have to get organized in order to become organized. You don't have to limit the Address stack to just names and addresses.

One of the biggest advantages of HyperCard over "traditional" databases is its freeform nature. With most databases you have to have a category for every piece of information you want to organize. The information in each category has to be the "right" format and the "right" number of characters. But HyperCard doesn't have such rigid rules. All you need is some information and a field to put it into.

Any miscellaneous information you might otherwise put on paper can go right on to an address card. For

instance, if a friend is coming to visit, and calls you with the flight numbers and times, you might jot them down temporarily on the Address card. You might also use Address cards to record anniversaries and birthdays.

6. Use the Browse tool and choose Find from the Go menu.

You don't really need the Browse tool to use the Find command, but you're finished with editing fields for now, so leave the Field tool behind. Find a friend's card by typing their name and pressing the Return key. You'll see that the card is larger now, with increased text field sizes.

Adding Text to the Field

Here is how to use the Browse tool to add text to a field. You'll make a note to yourself of your friend's favorite flower right on the address card.

Some fields are "locked" so that they can't be edited. For example, most of the text in the Help stacks is locked. You can unlock a field by double-clicking it using the Field tool and unchecking Lock Text.

1. Click in the first field—the one that contains your friend's name and address.

When you click in a field's white space, HyperCard puts the insertion point at the nearest left margin to where you click, adding Return characters as necessary.

2. Type: yellow roses.

Now you'll remember your friend's favorite flower whenever you see the card.

Hint

Make entries in your Address stack for names or phone numbers you'd otherwise forget. Create a card that says "Flowers," for example, and list the florists in your mom's home town. Now you can send her flowers without remembering the florist's name. Or make a card with "Windows" on it and list the window-cleaning service someone recommended. (If windows means software to you instead of the kind you have to clean, put Dan Ingall's phone number there. He invented the windows environment.)

Customizing the Way the Text Looks

Text in a text field can be any font, size, or style available. You can change the text in a field to any font you have installed in the System file of the startup disk you're currently using. (Your Macintosh owner's guide or Utilities Guide—if you have a newer Macintosh—explains how to use the Font/DA mover to move fonts into and out of the System file.) All text within a field shares the same font, size, and style.

Hint

Create different styles of text by creating separate fields for the text you want to look different. For example, you might want headings to be boldface type and the rest of the text to be plain. Creating a new field for the heading lets you display it in a different style from the body of text.

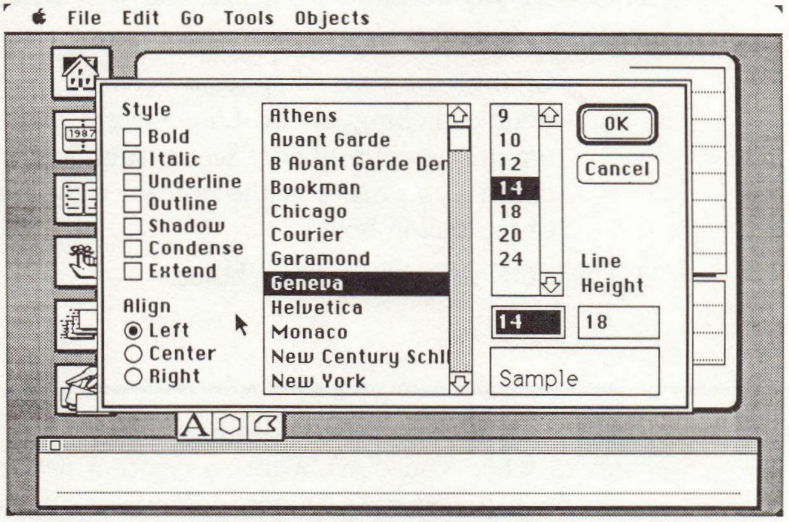


FIGURE 2–13 *The text style dialog box shows the fonts installed in the System file on the current startup disk. All text in a field shares the same font choices. You can have multiple fonts in a stack by creating separarte fields for each text style you want to use.*

You can also create text using the Paint text tool. You can't search for text you created with the Paint text tool, but you can rotate it, flip it, and manipulate it the way you manipulate any other Paint image. You'll learn how in Chapter 4.

1. Use the Field tool to click one of the fields and choose Text Style from the Edit menu.

It doesn't matter what card you're on. Each card shares the same background fields.

When you choose the Text Style command using the Field tool, the command affects only the currently selected field.

Figure 2.13 shows the dialog box that appears.

2. Choose any available font, size, and style by clicking your choices.

Change the vertical spacing between lines by double-clicking in the Line Height box and typing new numbers. See a sample of the choices you make in the bottom right corner of the dialog box.

3. Click OK to confirm your choice.

Hint

Use Paint text to embed bold words or change text styles when you don't want to create a new field for the new text style. Do this only when you won't need to do much editing of the text later—for text you're using as a display, for example.

4. Select another field with the Field tool and then choose Text Style from the Edit menu.

Now you can choose different styles and fonts for this field.

Using the Button Tool

Buttons are the heart of HyperCard. They provide the action that allows you to move to other cards, to automatically carry out commands in menus, and to provide visual and sound effects. They also allow HyperCard programmers to write new actions, thereby extending HyperCard's power. But you don't have to be a programmer to love the power that buttons give you.

Navigating with Buttons

You encountered buttons the first time you used HyperCard, clicking your way through the Introduction to HyperCard, exploring some of the stacks on the Home card, or flipping through names in the Address cards or through the calendar. Buttons can look like the buttons you're used to in Macintosh applications—OK or Cancel, for example—or they can look like arrows, dog-eared pages, or the stacks they represent (on the Home card, for example). In fact, buttons can look like anything you want them to, because you can use the Paint tools to draw pictures that accompany the buttons you create.

Hint

There are several ways to see whether a button belongs to a single card or the card's background. You can use the Button tool to select a button and choose Button Info from the Objects menu to see whether it's a card or background button. If it's a background button, you'll see the words "Background button number" followed by the button number. Or you could choose Background from the Edit menu. All the card buttons will disappear. If you can still see a button, you'll know it's a background button. A third way to tell if a button is in the background is to hold down the Command and Option keys. Background buttons have a double border.

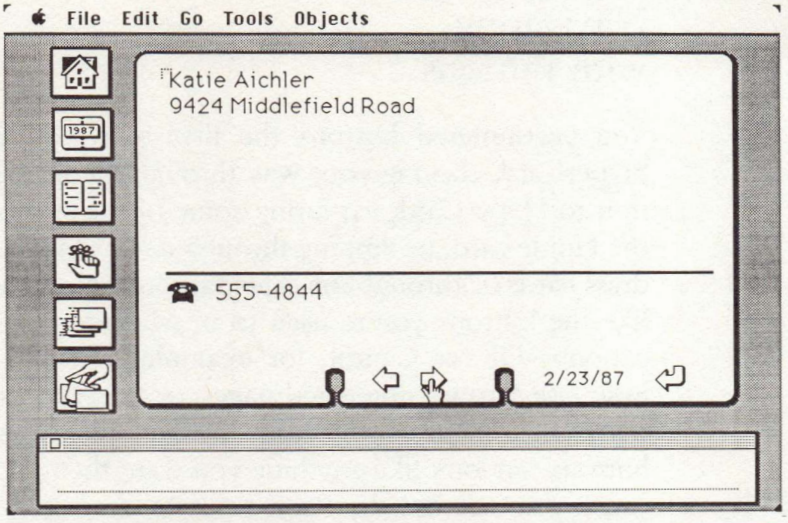


FIGURE 2–14 *An address card's buttons take you to the previous or next card or to the card you were at when you came to the address stack. They also take you Home, to the weekly or yearly calendar, to a To Do list, flip through the stack, sort, or dial a phone.*

You can be content for quite a while using the buttons that are provided for you in existing HyperCard stacks, but before too long you'll want some buttons to call your own.

You're already familiar with the buttons on the Address stack. Figure 2.14 shows a card in the Address stack. This stack has an array of background buttons along the left side of the card, as well as several arrows and a telephone button.

The forward and backward arrows always take you to the next or previous card; likewise, each of the other buttons that appear on the cards always provides a consistent action. The Return arrow goes back to the card you were looking at when you came to the Address stack.

Like pictures and fields, buttons can be on the card itself or in the card's background, where the button is shared among all the cards that use that background. All of the existing buttons in the Address stack are *background buttons*. They appear automatically on each new card in the stack.

Creating a New Button

Creating a button is as easy as choosing the Button tool and then choosing New Button or Command-dragging in the place you want the button to be.

In the next two lessons you'll create new buttons and then link a button to a specific card.

The Objects menu appears when you have your User Level set to Authoring or above. Here's how to create a new button that will take you to another card in the Address stack—the card of a business contact that a friend referred you to.

1. Use the Find command to go to Katie Aichler's card and then choose New Button from the Objects menu.

HyperCard automatically creates a new button for you and plops it into the middle of the screen. It also switches you to the Button tool if you're not already using it.

2. Use the Button tool to drag the button to the bottom left corner of the white area of the card.

Be sure to keep the button in the white area of the card. If you put it in the same area as the background buttons, you might be confused later.

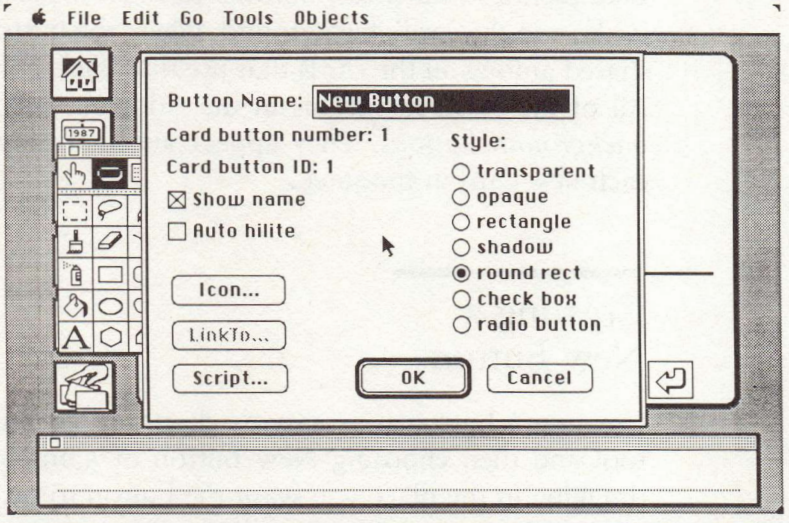


FIGURE 2–15 *The button dialog box is where you start discovering HyperCard's power. You can choose a button style, name the button, choose an icon to represent it, and link the button to another card. The Script button is dimmed unless you've changed your user level to Scripting on the Preferences card in the Home Stack.*

There are a couple of ways to find information about a button—its name, ID, etc. You can select the button and choose Button Info from the Objects menu, or you can just double-click the button using the Button tool.

3. Choose Button Info from the Objects menu.

Because the button you just created is already selected, you don't have to select it first. You do need to have a button selected to choose this command. If there's no selected button, the command is dimmed, as the Field Info command is currently.

The dialog box shown in Figure 2.15 appears with a place for the button name, its number within its

background or card (button numbers are assigned by order of creation—lower numbers appear toward the back, and buttons you add are placed closer and have a higher number); its ID (a unique, permanent name HyperCard assigns when you create the button); its Style (transparent, opaque, rectangle, shadow, round rect, check box, or radio button); and buttons that say "Icon...,""Link to," "Script," OK, and Cancel. Because you created this button using the New Button command, its name is preset to "New Button." Buttons you create by Command-dragging aren't named.

The "Show name" check box is already checked, and the "round rect" button style is checked. Show name will display the button's name in the button when you put the dialog box away; round rect is the shape of the button you're used to seeing on the Macintosh.

You can produce a wide range of button looks by choosing various combinations of button styles, icons, and whether or not the button's name is visible. You can change the button's look anytime, so experiment, now or later, with this new button.

4. Click OK to see how the button looks and then double-click the button using the Button tool to see the dialog box again.

5. Type: Contact to rename the "New Button" "Contact."

Hint

You don't have to follow any rules about using uppercase and lowercase—just type button names the way you want them to look on the screen and when you print them.

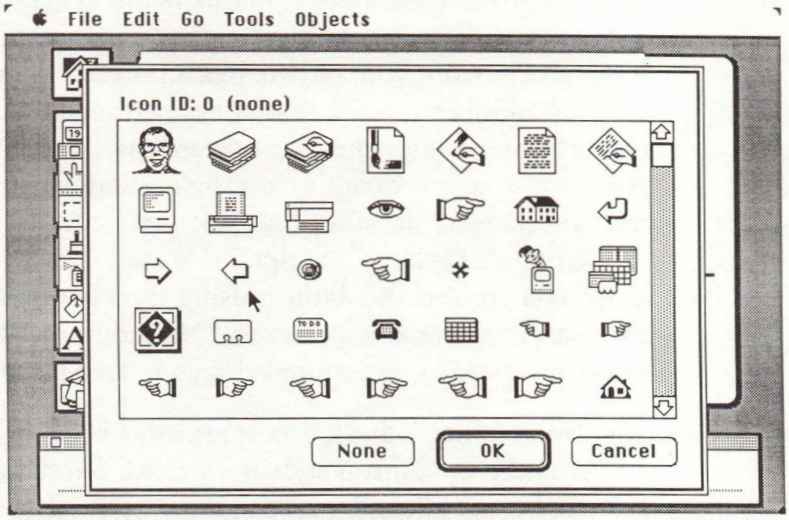


FIGURE 2–16 *The icon dialog box displays the available icons that can represent buttons in your stacks.*

6. Click OK to see how the button looks.

The button displays its name—Contact—in the round rect style.

Using a Button Icon

Besides the button styles listed in the dialog box, a button can also be represented by an icon—like the icons you see in the Finder. You can use any icon resources that are available in the current stack or in HyperCard itself.

1. Use the button tool to double-click your new button to see its dialog box again.

Remember not to use the Browse tool when you want to double-click to see the dialog box.

2. Click Icon to see the available icons.

Figure 2.16 shows some of the icons that are available. Use the scroll bar to scroll through the directory of icons.

3. Click the icon you want to use for the button that goes to the contact's card. Click OK to close the icon dialog box.

You might have to adjust the button's size by dragging one of its corners or move it by dragging from its center.

Hint

The icon dialog box shows you the icon resources in the current stack, the Home card, the current System file, and in HyperCard itself. You don't see icons that exist only in another stack. But you can use those icons as well by going to the stack that contains the one you want and then copying and pasting a button that uses the icon.

See Appendix 3, "Moving Resources," for information on how to move icon resources among stacks.

Linking the Button to Another Card

Buttons can do an endless array of actions, but by far the most common action for a button is to link you to another card. So HyperCard includes a LinkTo button that automatically takes you from the card you're looking at to another card in the same stack

or another stack. The LinkTo button gives you a great deal of control by linking cards and stacks together without having to write programs. HyperCard takes care of those technical details for you by writing a script that is performed whenever you click the button using the Browse tool.

You've created the button on your friend's address card. Now it's time to link that card to the business contact your friend told you about.

1. Double-click the button to see its dialog box again.

2. Click LinkTo.

This tells HyperCard that you want the selected button to take you to a specific card whenever you click it using the Browse tool. Now tell HyperCard where that card is.

Up pops the Destination window, with two choices—This Card and This Stack—and a Cancel button, as shown in Figure 2.17. This Card and This Stack are the two most common places to link a button to, so HyperCard makes it easy for you by offering them as choices.

Notice that even though you're still working with the Button tool, HyperCard temporarily switches you to the Browse tool so that you can browse around to find the place you want to link this button to. You can use any of the usual methods to browse through cards and stacks: You can use the Go menu, or choose Open Stack from the File menu. You can go Home and click buttons, and you can use the tilde key to back up.

3. Use the Browse tool to go to the card of the person your friend recommended to you.

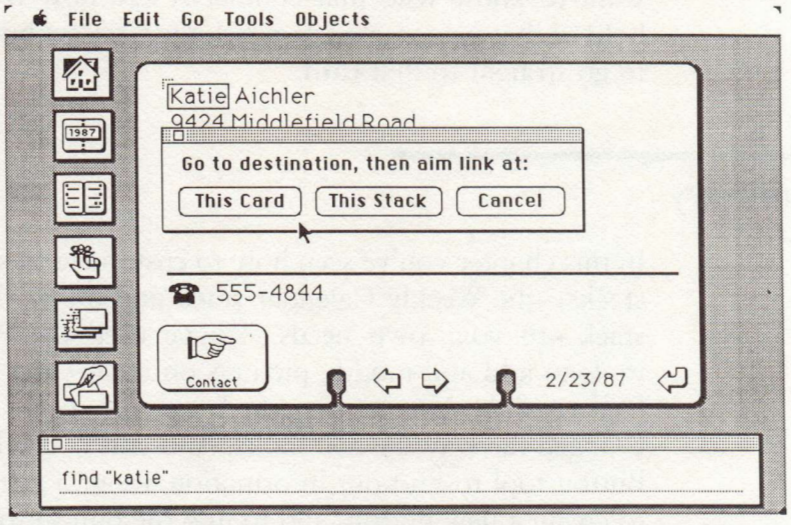


FIGURE 2–17 *The LinkTo Destination window comes up when you click the LinkTo button. HyperCard switches you to the Browse tool so that you can choose a destination for the button. When you complete the link, HyperCard puts you back into the button tool so that you can make other modifications to the button.*

For this exercise, go to any card in the address stack. You could also go to a new card that you add by choosing New Card from the Edit Menu. This new button will take you there instantly from your friend's card—tying the two cards together.

4. Click This Card on the Destination window.

You've just told HyperCard which card to link the button to.

5. HyperCard switches you back to the Button tool and takes you back to the original card.

Use the Browse tool to try out the new button.

Now whenever you see the name on the first card, you'll always see a button that says "Contact." If you

want to know who that contact is and how to get hold of that person, you can click the Contact button to go straight to that card.

Summary

In this chapter you've seen how to customize existing stacks—the Weekly Calendar stack and the Address stack—to your own needs. You've used the Paint tools to add and modify pictures on a card and in a background. You've modified a text field and changed the way the text looks. And you've used the Button tool to find out information about a button, to create a new button, and to link the button to another card.

Maybe you've seen just enough to wonder about the other possibilities. In the next chapter, you'll see how you can use the stack "seedlings" that HyperCard provides to start building some stacks of your own.

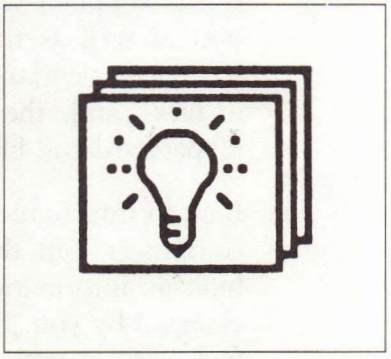
Part Two:

Becoming a HyperCard Author

Chapter 3

Building Stacks

"I can't remember what you put at the beginning of a program. I always copy it from somewhere else."
Bill Atkinson



By now you know how to personalize any of the existing HyperCard stacks by adding your own text and pictures and by adding and modifying text fields and buttons. But while you're exploring every nook and cranny of existing stacks, you'll probably think of an idea for a

stack that doesn't yet exist. And you might want to build that stack yourself.

Borrowing

With most applications you create documents from scratch. With a word processor you have to supply the words. With a graphics application you draw the shapes you need using various painting tools. With a spreadsheet, you enter the numbers. Sometimes you can use templates to get started in a spreadsheet or you can paste in ready-made art to add a professional touch to documents, but for the most part you usually build a document yourself from the ground up.

With HyperCard, your Macintosh already has some information built in. HyperCard supplies not only the form but some of the content as well. Each of the stacks supplied with HyperCard contains information as well as the stack structure. The datebooks have real calendars, the Slide Show is full of beautiful images, and the Help system tells you about HyperCard and HyperTalk.

Built-in information is a new concept with personal computers. But the extraordinary thing about this built-in information is that it is designed to be changed by you. It's meant to be pilfered and modified to fit in with what you want the stacks to do. The most obvious examples of this are the Address cards and the Datebook. But there is nothing in HyperCard that can't be altered.

Collecting

While I was working on HyperCard's Help system, I realized part way through that I wasn't keeping my usual supplies of "raw materials" (art and text) hidden away in various applications and in the Scrapbook. This was because I could put something—a picture, text, or even a button—anywhere in HyperCard and retrieve it again later to copy or move somewhere else. HyperCard is a living, mutable library of resources.

Stack Ideas, Card Ideas, and Art Ideas are special "collection" stacks of backgrounds, cards, and snippets of art meant to be copied and built into new, complete stacks. But you're not limited to just these stacks for finding your raw material. You can copy pictures, text, fields, buttons, or entire cards from anywhere in the existing HyperCard stacks. This is an application that invites you to find something you like, copy it, and then modify it to do exactly what you want it to do.

Making Information Your Own

The real power of HyperCard won't emerge until you make cards and stacks your own by adding your own information to them. This chapter shows you how to borrow pictures, fields, and buttons that already exist to put together a stack of your own. You'll start with a copy of a card in the Stack Ideas stack and then modify it to keep track of personal information.

You'll modify fields, use Paint text, and add cards and buttons. You'll see the relationship between card pictures and the background pictures. You'll also see how to create a new calendar from one of the other stack seedlings.

Starting a New Stack

Whenever you create a new stack, HyperCard gives you the choice of starting the new stack from scratch or using any existing stack as a model for the new stack. If you use an existing stack as the template, the new stack has the same background picture, the same background buttons, and the same background fields as the card it's modeled on. All that's missing is the text and any pictures, buttons, or fields you want to add to individual cards.

A Copy or a Shell

When you duplicate documents of any kind in the Finder, you get an exact copy of the original document. You can duplicate HyperCard stacks in the same way to produce an identical copy of what you started out with. (You can also copy a stack without quitting HyperCard by using the Save a Copy command in the File menu.) But there's another way to copy existing stacks in HyperCard. Choosing New Stack from the File menu starts you off with a copy of the stack you're looking at when you choose the command. Specifically, the new stack is a copy of the

current background. The new stack includes everything in the original except text in fields and card-specific fields, buttons, and pictures.

Hint

A stack can have many backgrounds, as we saw with the Datebook stack. Cards in one background don't necessarily have to be together in one place. A card inherits a background when it is created—it gets whatever background belongs to the card you're at when you create the new card.

Hint

Use the Finder or the Save a Copy command to duplicate a stack if you want the new stack to come complete with text and card-specific pictures, fields, and buttons; use the New Stack command if you just want to copy the background structure of an existing stack.

The first step in creating a new stack is to decide what you want the stack to do and to look like, and then see if there's already a stack that comes close. Don't worry if the stack you're copying doesn't exactly match what you have in mind. HyperCard tries to save you some work by offering existing stacks as templates to use for the new stack; however, you can later modify anything—the picture, the fields, or the buttons.

Starting with a Stack Idea

You could start with any working stack that already does something similar to what you want the new stack to do. But HyperCard also includes multiple-background stacks that are meant to be copied and fleshed out by you. The Stack Ideas stack on the Home card is a collection of stack "seedlings" that you can use to build new stacks. Each seedling consists of one card with a background—including a background picture, background fields, and background buttons. Each card in this stack has a separate background, which can either form the basis for a whole new stack or be an additional background in an existing stack.

Building a Personal Information Base

You'll use the Address stack to keep track of friends or clients, and you'll use appointment books to keep track of your time. What about all of those other details of life that can prevent you from being organized?

You can build a stack that tells you where your personal documents are: bank accounts, insurance policies, credit cards. Of course, you'll keep the actual policies and account books somewhere else—at the bottom of a box in the garage, perhaps? HyperCard will just organize the information so that you can get to the real thing more easily.

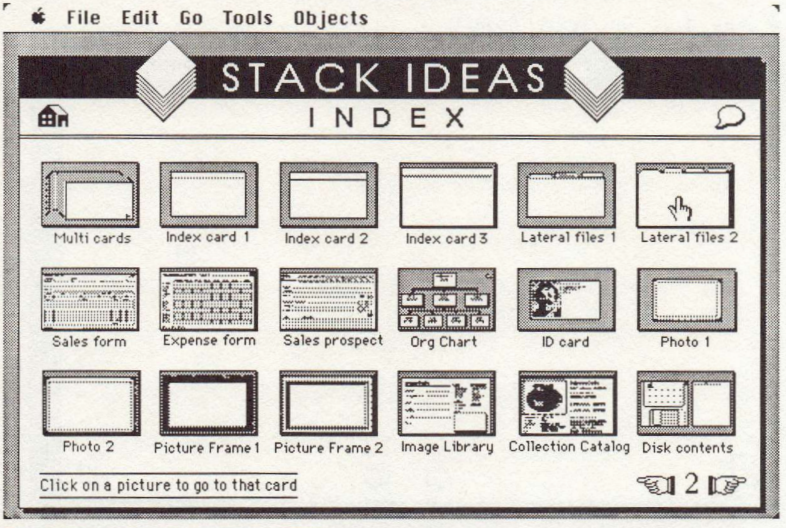


FIGURE 3–1 *Stack Ideas is full of stack "seedlings" you can use to start new stacks. Each card in the Stack Ideas stack has its own background. You plant the seedling by starting a new stack or copying the card to another stack.*

Copying a stack seedling

To learn how to use one of the cards in the Stack Ideas stack as a seedling for your new stack, try creating a personal finances stack. Here's how.

1. Go Home and click the Stack Ideas icon.

You'll see an index of the cards and backgrounds included in this stack. At the top of the first index card are buttons that go to Home and to Help for using this stack. At the bottom right is a hand that takes you to other cards in the index.

2. Click the hand that points to the right to go to the second card of the index as shown in Figure 3.1, and then click the miniature of the Lateral Files 2 card.

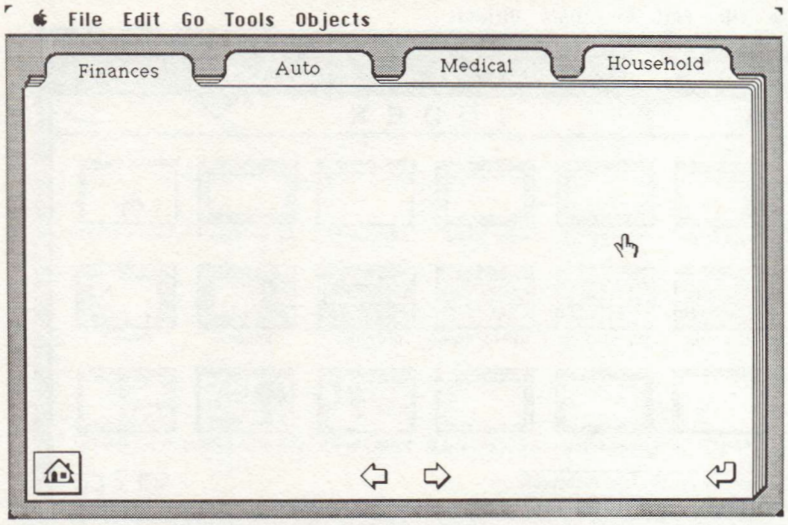


FIGURE 3–2 *The file folder seedling's background has a picture, buttons, and text fields. When you create a new stack from this seedling, you get the picture and the text fields on the folder's tabs, but you add your own custom text.*

The background picture on this card looks like a set of lateral files. Each tab represents a subject you might want to keep a file for—Finances, Auto, Medical, and Household. You can see this card in Figure 3.2.

Making a new stack while you're viewing this card will create a new stack with the same background as this card.

Creating the new stack

Whenever you choose New Stack, you have the choice of starting the new stack from the one you're looking at when you choose the command. It's the easiest way to get going on a new stack. Here's how you create a new stack.

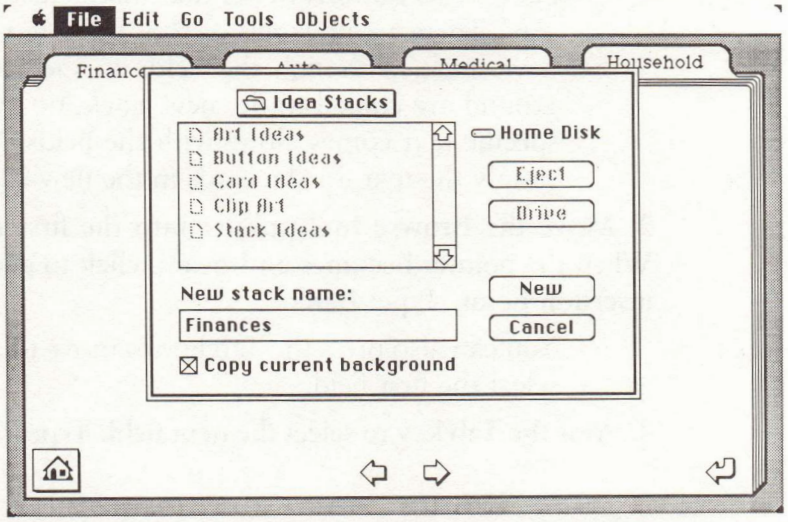


FIGURE 3–3 *You use the standard file dialog box to name new stacks the same way you name new documents in other Macintosh applications. You use the "Look for Stacks In" card in the Home stack to tell HyperCard where the stacks are.*

1. Choose New Stack from the File menu.

You see the familiar standard file dialog box, with a couple of differences. (See Figure 3.3.) At the bottom of the dialog box is a checkbox (which is preset to checked) for whether to copy the current background or not. If this checkbox isn't checked, you'll start with a blank new background. Leave it checked to start off your new stack with the lateral files background.

2. Name the new stack "Finances" and click the New button.

When the dialog box disappears, you see your new stack—with one card in it and the background this card uses. Try the forward and

backwards buttons to see that there's just one card. There are no labels on the "tabs" in your new stack. Although the fields in the background are copied to the new stack, no card-specific text comes along with the fields. You supply the text for the cards in the new stack.

3. Move the Browse tool pointer into the first tab. When the pointer becomes an I-beam, click to get an insertion point. Type: Bank.

You can also press the Tab key to move to and select the first field.

4. Press the Tab key to select the next field. Type: Car.

Hint

Pressing the Tab key lets you type in the next field without having to use the mouse to click in it. It's a shortcut for people who don't want to take their hands off the keyboard while they're typing information into the Macintosh.

5. Press the Tab key and type: Credit Cards.

Notice that each time you press the Tab key, the insertion point appears in the center of the Tab. One of the properties you can set for a field is that text appear centered in the field rather than starting at the left edge. The fields on these tabs are centered.

6. Press the Tab key and type: IRAs.

Of course, you can type any categories that match the kinds of records you keep. You can add more subjects later; use these to get started.

Now you'll add another field to contain information about each of the subjects.

Adding a New Text Field

You can add a new text field whenever you want to by choosing New Field from the Objects menu or by choosing the Field tool and Command-dragging on the card or background where you want to add the field. Here is the procedure.

1. Choose the Field tool from the Tools menu.

When you're using the Field tool, all of the existing fields are outlined. This stack has fields for each of the four folder tabs.

2. Choose Background from the Edit menu to work in the stack's background.

Now you can create a text field in the background that will be shared by each card in the stack.

3. Command-drag diagonally to create a field that covers most of the card.

Drag the field from its center to move it; drag from an edge to make it larger or smaller. Aim for a field that looks like the one in Figure 3.4.

Setting Properties for a Text Field

Every field has text properties such as the font, size, and style of the text. Follow these steps to set your

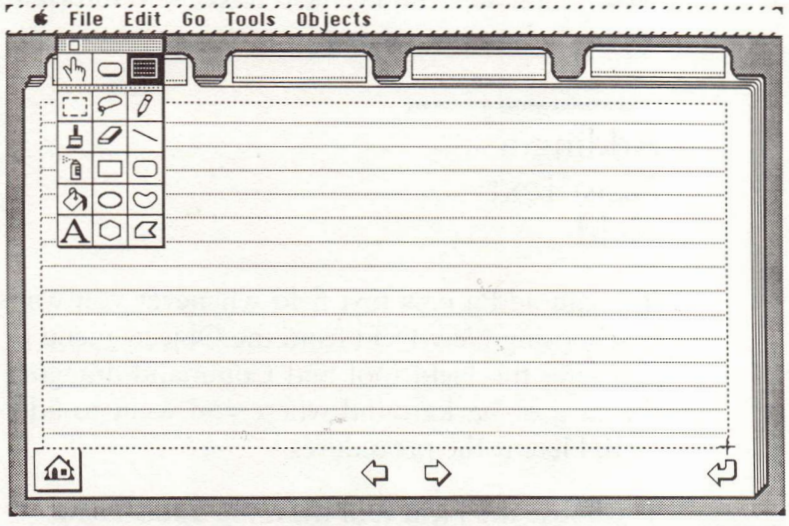


FIGURE 3–4 *You add new text fields by dragging with the Field tool while holding down the Command key. You can also choose New Field from the Objects menu to create a new field even if you aren't using the Field tool.*

new field's properties. You can set field properties whenever the field is selected using the Field tool, or whenever there's a selection such as an insertion point within the text in the field.

1. Click anywhere in the large field to select the field.

2. Choose Text Style from the Edit menu to set a font and style for the field.

Make sure you're using the Field tool to select the field. The choices you make next will affect all of the text in this field.

The Text Style command is dimmed unless there's a selected field or a text selection.

Hint

You can change a field's style when the field is selected using the Field tool or when its text is selected using the Browse tool.

HyperCard presents a Text dialog box where you can make choices for the entire field.

Hint

Although all text in a field shares the same font and style choices, you can add new text styles by creating a new field wherever you want a different style. Use one field for titles and another for the main body of text, for example.

3. Click to select a style (with nothing checked the style is plain), an alignment (left is preset), a font, and a size. Click OK to confirm the setting.

Hint

Double-clicking either the font or size choice also confirms text style changes and puts the text dialog box away.

You don't have to be in the background to make changes to the field. To make any change to a background picture you need to be in the background. But you can change the size or shape of background buttons or fields or set other properties for them without first choosing Background from the Edit menu. You do need to be in the background to create a background button or field.

Hint

Choose Background from the Edit menu to make any changes to the background picture, but modify background fields and buttons from either the card or the background.

4. Switch to the Browse tool to click in the large field and then type the information about your banks: the names, your various accounts in each, their telephone numbers, their working hours, their phone numbers and addresses, the name of the helpful teller, etc.

Figure 3.5 shows what a sample card looks like. The structure and text style of a background field is in the background and is shared by every card that uses that background. But the text itself belongs to each individual card that shares the background.

Adding Cards to the Stack

So far, there's just this one card in the new Finances stack. The tabs for your car and IRA information do not have cards yet. The arrows along the bottom of the card are preset to take you to the next card in the stack. After you add some new cards to represent the other subjects on the tabs, clicking these arrows will take you to the next or the previous card in the stack.

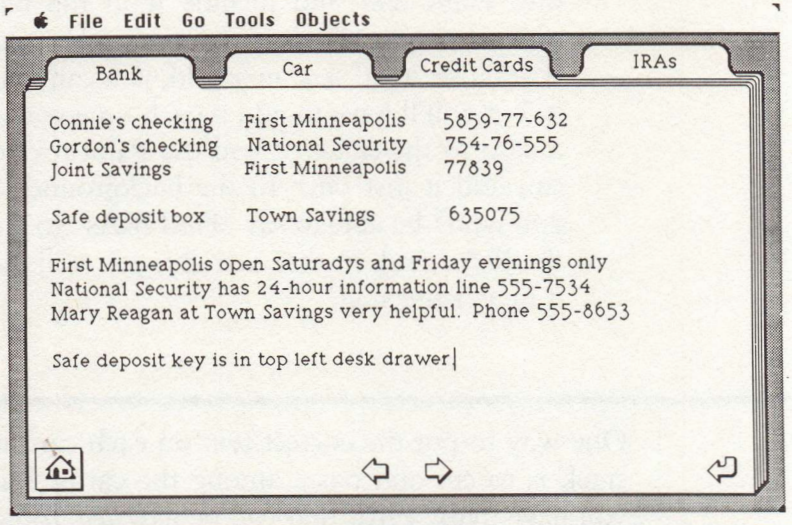


FIGURE 3–5 *Filling in the text. The text you type into a field belongs to a specific card, even though the field is a background field that is shared by every card that uses that background.*

Now that your first card, Bank, is complete, add a second card to your new stack. The second card will contain information about your car.

1. Choose New Card from the Edit menu or type Command-N.

The new card is added after the one you were looking at when you chose the command, and HyperCard takes you to the new card.

The text on the tabs disappears because it was specific to the first card, the one you typed it on. There are a couple of ways to include it on all cards. You could copy and paste text from each tab on each new card you add to the stack, or you could transform each tab

into Paint text and include it in the background picture. Each way has its advantages. If you use "real" text in a field, you can search it, but you'll have to add it each time you add a card to the stack. If you use Paint text you can add it just once to the background, but you won't be able to say "Find IRAs" to go to the IRAs card. In this exercise, you will paste it in as Paint text.

Hint

One way to put the correct text on each card in a stack is to cut and paste among the cards. But if you have many cards that can be tedious. You can also use Paint text in the background so that the text is shared by each card that uses the background.

2. Click the left arrow near the bottom of the card to go to the previous card.

You added a new card in the previous step, so clicking the arrow will return you to the Bank card.

3. Select the text in the first field—"Bank"—by dragging across it.

4. Choose Cut Text from the Edit menu.

The text is removed and placed on the Clipboard. With HyperCard the object you last placed on the Clipboard is the object you subsequently paste. If it's text, then you paste text; if it's a picture, you paste a picture. Likewise with fields and buttons.

When you have text in the Clipboard you can either paste it as text (by choosing Paste Text from the Edit menu) or you can choose any Paint tool before you choose Paste Text. This will paste the text in as Paint text rather than the regular text that goes in fields.

5. Choose Background from the Edit menu.

This will let you paste the word "Bank" in the background so that it will appear on each card in the stack.

6. Select any Paint tool to let HyperCard know you want to paste Paint text rather than real text.

7. Choose Paste Text from the Edit menu.

The text is pasted into the center of the screen.

Now the text is like any other picture. You can move it around.

Hint

Normally pictures are pasted into the same relative spot they're cut or copied from. But when you paste text in as Paint text, HyperCard pastes it into the center of the screen.

8. Drag the Paint text to the first folder tab. Position the Paint text on the tab.

9. Do the same for the other three tabs.

Choose the checked Edit Background command to go to the card; use the Browse tool to select the text on one of the tabs; choose Cut Text from the Edit menu; choose Background from the Edit menu; select a Paint

tool; choose Paste Text; drag the Paint text to one of the tabs.

Deleting the Fields You Don't Need

Now that you've made the text into Paint text, you no longer need the fields that held the regular text. You don't need to be in the Background to delete Background fields. All you need is the Field tool.

1. Click the Field tool. Select each tab's field in turn and press the Backspace key.

You could also press the Clear key if you have one or choose Cut or Clear Field from the Edit menu.

2. Click Delete when HyperCard asks for confirmation.

This dialog box is a safeguard against accidentally deleting a field. Such a field might contain a huge amount of information that's unrecoverable once deleted.

Adding More Cards

Use the New Card command to type information about your car on the second card. You might include the dealer's name, the name, address, and phone number of the place you take it for repairs, its last tuneup date, when the registration is due, when you last paid the insurance, the color code and key num-

bers so that you can match paint or get new keys with just a phone call.

Add two more cards for your credit cards and IRAs. Now you have four cards total. Fill in a card with your credit card information—where you have accounts and the account numbers. Lastly, fill in a card with a list of IRAs: include the year and where the account is.

Move among the cards using the left and right arrows near the bottom of the cards. Notice that when you're at the last card in the stack—the IRAs card—pressing the right arrow takes you back to the first card. And likewise the left arrow on the first card takes you to the last card. The cards in HyperCard stacks are arranged in a circle.

Adding a Background Button that's Linked to a Specific Card

You could get around in the new stack by just using the arrows at the bottom of the cards or using the Go menu to go to the First, Next, Previous, or Last card in the stack. But you could also add buttons that link you to specific cards from any other card in the stack. Each card will have these buttons in the background, so you can move to any one card from any other card in the stack. Here's how.

1. Choose Background from the Edit menu.

HyperCard takes you to the background shared by all the cards in the Finances stack.

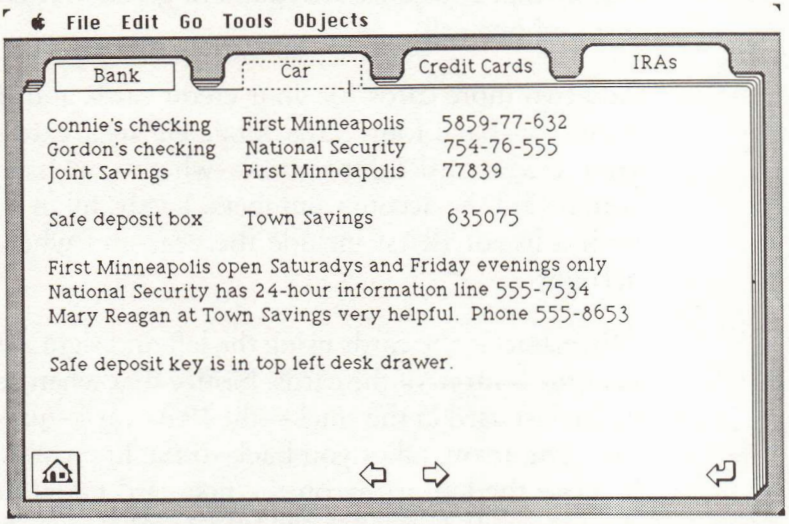


FIGURE 3–6 *Holding down the Option key while you drag a button makes a copy of it, just as Option-dragging a picture in MacPaint copies it. The copied button comes complete with any script the original had.*

2. Click the Button tool and hold down the Command key while you drag diagonally to create a new button on the "Bank" tab.

3. Double-click the button. When the dialog box appears, click LinkTo.

4. Go to the "Bank" card if you're not there already and then click This Card.

Now this background button is linked to this specific card. Whenever you click this button using the Browse tool, no matter which card you're at in the stack, you'll go to the Bank card.

5. Hold down the Option and the Shift key while you drag the button to the right, placing the copy of the button on the "Car" tab as shown in Figure 3.6.

The Option key makes a copy; the Shift key keeps it lined up with the original. The copy is identical to the original button, including being linked to the "Bank" card. Even though you aren't in the Background, it's a background button because you option-copied a background button. But you can re-aim the copy using LinkTo.

6. Double-click the copy and then click LinkTo.

7. Go to the card that has your car information on it.

8. Click This Card in the Destination dialog box.

You don't have to be at the destination card before you click LinkTo. You can use the Browse tool to move around after the Destination palette comes up.

9. Do the same for the other two cards: go to the card, Option-Shift drag a copy of one of the other buttons, double-click it, click LinkTo, and click This card.

Using card pictures to differentiate cards

You can use pictures to differentiate cards. Here's one example of how to do this:

1. Go to the first card in the stack by choosing First from the Go menu or pressing Command-1.

This card's picture doesn't need to change because the Background picture for the stack shows the first tab frontmost.

2. Go to the next card—the "Car" card.

Now try your hand at using the Paint tools to make the "Car" tab look like it's frontmost in the stack.

1. Use the line tool to extend the horizontal line across the "Bank" tab.

Hint	Hold down the Shift key while you draw with the line tool to draw a straight line.

2. Click the pencil tool and choose FatBits from the Options menu to do the detail work.

The pencil is the best tool for FatBits because you can click or drag individual dots on or off. Hold down the Shift key to draw a straight line.

Hint	Command-click at an exact spot with the pencil to go into FatBits at that very spot or to leave FatBits.

3. Return to the normal view by clicking the small window, choosing the checked FatBits command, or Command-clicking with the pencil.

Now you see how your changes really look.

4. Hold down the Command key while you use the eraser to erase the line under the "Car" tab as shown in Figure 3.7.

Hold down the Command key while you erase, because the picture is in the background rather than on the card. Holding down the Command key while you erase actually paints over the image with opaque white rather than erasing.

5. Command-click with the pencil at the corners of the tab to get to FatBits, where you can do the detail work.

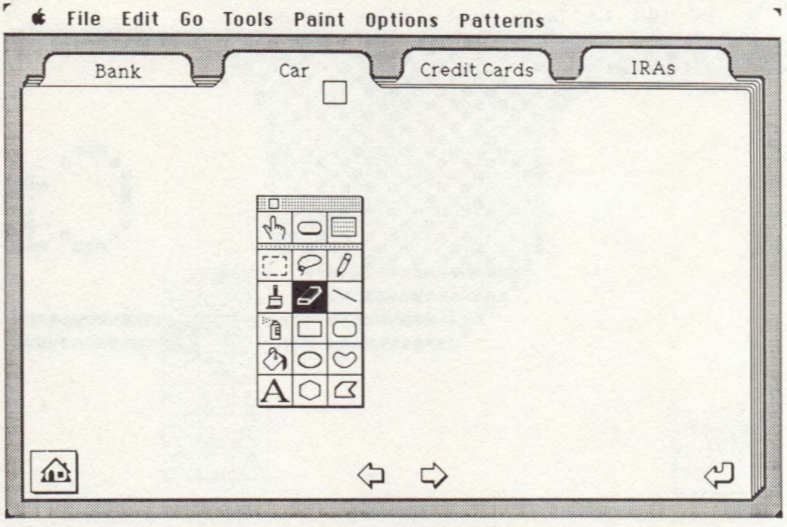


FIGURE 3–7 *Modifying a tab on the "Car" card. You can use any of the Paint tools to modify the picture on this card. If you erased normally on this card, you would really erase nothing because this picture is in the background. But when you hold down the Command key while you erase, you're actually painting with white rather than erasing, and the effect is that you no longer see through to the background picture.*

Click pixels on the screen to create a diagonal line to separate the tab on the frontmost card from the rest of the cards as shown in Figure 3.8.

6. Command-click again to return to the normal view.

Review your detail work.

7. Hold down the Option key while you press "D" to see the "data."

All you see are the changes you made to the picture. Holding down the Option key and

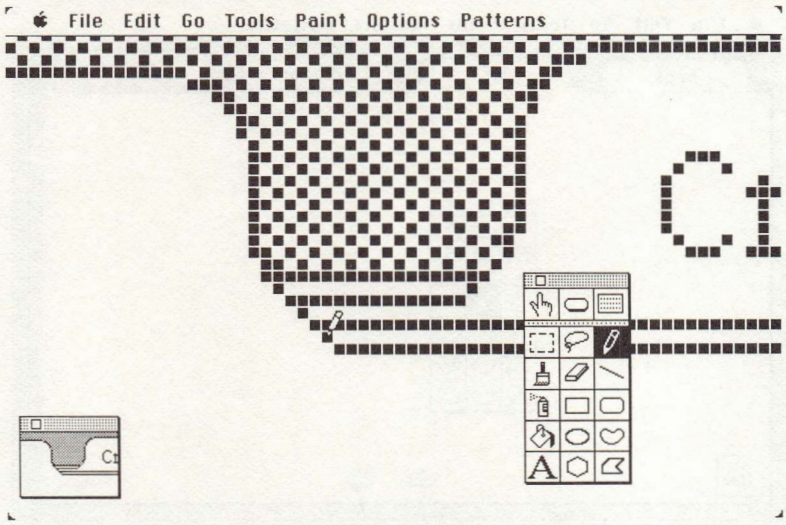


FIGURE 3–8 *FatBits makes detailed work easier to control. Each square represents a pixel on the screen. You can click the squares off or on to make them white or black.*

pressing "D" shows you the data in the card picture, but not the background picture.

8. Hold down the Option key while you type "O" for Opaque.

HyperCard highlights the opaque portions—everything you pasted on the card picture.

All painting you do is opaque when you draw it or paste it in from the Clipboard. HyperCard tries to clean up after you (and save some space on the disk) by removing any opacity that's not actually obscuring a background picture underneath it. HyperCard checks the opacity whenever you leave a card and removes any that's not covering something. If there's nothing underneath, there's no reason for opacity, and HyperCard automatically makes it transparent.

Alter the card picture of each of the other cards in the same way, to show each card's tab at the front of the stack. Extend the horizontal line across the "Bank" tab and then expose the individual cards' tabs.

Building a New Monthly Calendar

The Home card includes buttons that go to the Weekly and Yearly calendars and the To Do list, which are all part of the Datebook stack. The Calendar button takes you to a card with a six-month view, and the Weekly button takes you to a card that shows the current week. But most people feel strongly about how they visualize time, and you may not see your time divided up into weeks. The Stack Ideas stack has a number of calendar alternatives you can use to build new calendar stacks. Here's how you can create a calendar stack that matches your picture of time.

1. **Go Home.**
2. **Go to Stack Ideas.**

   You see the first card in a garden of stack seedlings you can use to start new stacks. Figure 3.9 shows this card.

3. **Click the "Monthly Cal 1" picture.**

   This takes you to the card you can use as a template for a new monthly calendar stack. This card is shown in Figure 3.10. The Stack Ideas stack is just a collection of individual cards—each with its own background. This card's background has a background picture, fields, and buttons.

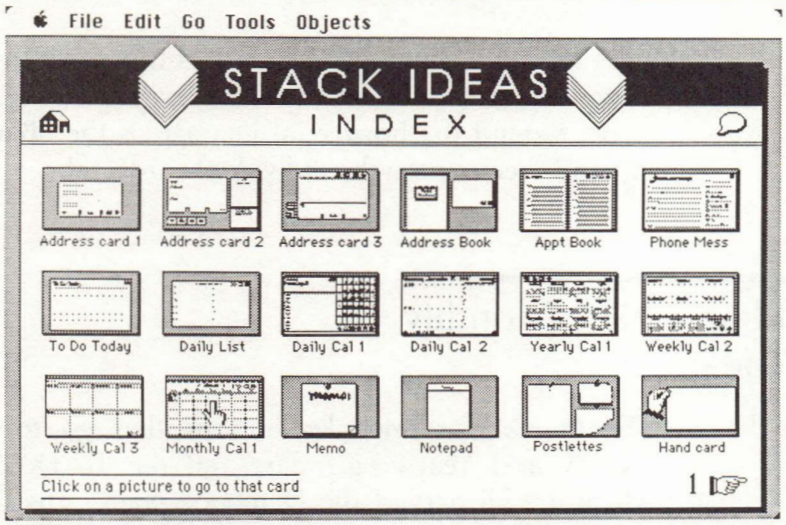


FIGURE 3–9 *A card of stack "seedlings" you can plant in new or existing stacks. A seedling comes complete with its background, and new cards you add after it share the same background.*

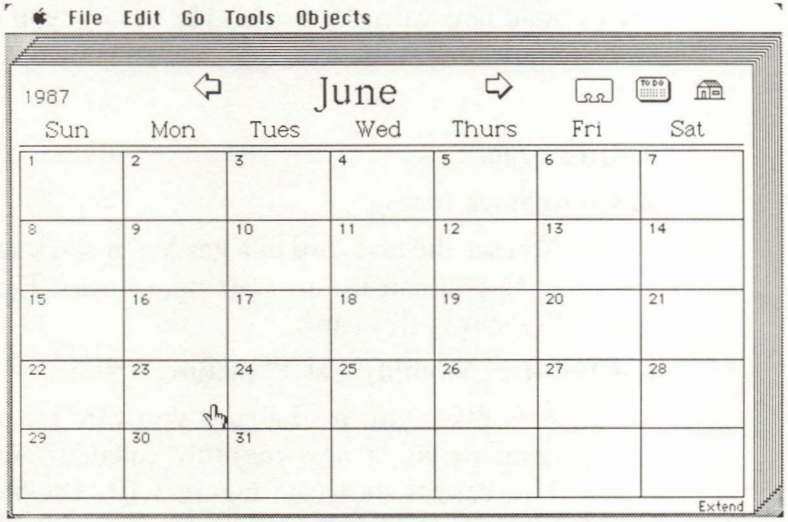


FIGURE 3–10 *The Monthly calendar seedling is a card with a background that includes a picture of an entire month. The background also includes an Extend button that propagates an entire year of months, each on a new card. The button automatically calculates the correct months based on the current date.*

4. Choose New Stack from the File menu.

This is the same thing you did when you created the lateral files stack for your finances.

Hint

Where you are when you choose New Stack is important, because HyperCard makes a copy of the current background to use as a template for the new stack. If you should ever want to create a new stack without copying from an existing stack, you can do that by clicking Copy Current Background to uncheck it and create a blank new stack with a blank background.

5. Name the new calendar stack.

Call it any name you'll remember or recognize easily in a list of stacks. Click the Ideas Stacks title at the top of the list to get to the other folders, if any, on the disk. Click the Drive button to see hard disks or other disks you insert in other disk drives. If you have a hard disk, you probably have copied each of the HyperCard disks to it, and getting at any of your stacks is no problem. If you're using an 800K external disk drive with your Macintosh, you just have to be a little cagier in planning where to keep your work.

Put the new stack on any disk where you have the space. (The New button is dimmed if there's not enough space on the current disk. Use the Drive and Eject buttons to move among other disks you can use to create your calendar.) If you place the new stack in a

folder, remember that the entire folder—including your new stack—will be replaced anytime you drag another folder with the same name to this disk. You might want to put your own stacks in "My Stacks."

Hint

If ever HyperCard can't find a stack it needs, it asks you to point to the right folder and never again forgets where you told it to look. It adds the correct pathname to its list on the "Look for Stacks in" card in the Home stack.

6. When you have the new calendar named and in the folder you want it, click New.

Voila! You've created another new stack. This new stack has just one background with one card. But with this calendar you don't have to manually add each month's cards and dates. Whenever you create a new stack from this stack seedling, HyperCard asks you if you'd like to make a full year's worth of calendars.

7. Click OK to have HyperCard create more cards for the stack, one for each of the remaining months in the year.

HyperCard uses the current month as the starting point, first checking the calendar to see what date it is. It adds a card for the next month, puts the name of the month in the Month field, and the days of the month in the correct box on the calendar. It knows how many days each month has, and even keeps track of leap years.

Hint

Check the Alarm Clock setting by choosing Alarm clock from the Apple menu, and reset it if necessary. HyperCard uses the current time and date in many of its actions. If your Macintosh clock is set wrong, the calendars won't work correctly.

8. Use the left and right arrow keys on the keyboard to move among the cards in the calendar.

As you can guess, the left arrow takes you to the previous month, the right arrow takes you to the next month. The Command key plus left arrow takes you to the first month (the first card in the stack) and the Command key plus right arrow takes you to the last month in the calendar—the last card in the stack.

Hint

If you have an older Macintosh keyboard, it doesn't have arrow keys. But you can do the same thing by pressing Command-2 for Previous and Command-3 for Next.

Trying the Buttons

The calendar you created by starting with the Stack Ideas template also includes several buttons. You can keep those buttons the way they are or link them to new destinations.

Here's how you might use the existing buttons. Say you have a lunch date you want to confirm.

Your calendar says "Lunch with Andy at Fontanas." Choose Find from the Edit menu or press Command-F to display the Message box with the Find command and an insertion point. Command-click the word "Andy" on your calendar to place the text at the insertion point; click the small Address card to take you to your Address stack; and press the Return key to find Andy's card in the Address stack. You could also check the restaurant's phone number while you're in the Address stack. Finally, press the Go Back key at the top left of the keyboard a couple of times to return to the Monthly calendar.

Let's say Andy wasn't at his desk when you called him. Click the "To Do" picture. This takes you to a list of things to do, which is actually part of the DateBook stack that includes the Weekly and six-month calendar. Make a note to call him back. Use the Go Back key to return to the calendar.

Copying a Button from the Button Collection

Just as you can copy pictures from anywhere in existing HyperCard stacks, you can copy any button that already exists in HyperCard. The Button Ideas stack has a great assortment of buttons in one place, but you could also select any button anywhere and copy it to a new place.

The button you'll copy, the speeding cards button, will quickly flip through the pages of your calendar; it would be convenient to have the button on each

card in the new Calendar stack so you can get a birds' eye view of what's coming up.

Hint

Remember that most buttons have scripts that define actions such as going to a certain card or sorting cards in a stack or choosing menu commands for you. If you don't yet know how to edit these scripts, experiment only with buttons that have obvious actions you can use—a left or right arrow, for example, that takes you to the previous or the next card.

1. Go to the button collection by selecting the Browse tool, choosing Message from the Go menu, typing: Go Button Ideas and pressing the Return key.

This takes you to the first card in the Button Ideas stack as shown in Figure 3.11. Like the Stack Ideas stack, this stack is a collection, rather than a working stack. It has an interesting assortment of ready-made buttons you can copy.

Hint

The Message box isn't just for the Find command. You can also type the Go command, followed by the name of the stack you want to go to.

2. Click any of the buttons named "Home, Business" to see a collection of commonly used buttons.

You'll see some of the same icons that appear on the Home card and other HyperCard stacks.

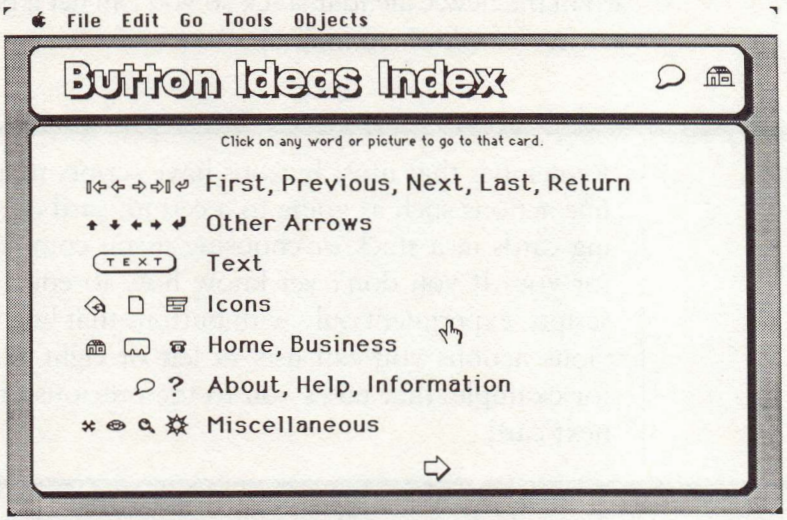


FIGURE 3–11 *The button collection is full of buttons you can copy to your own stacks. Some buttons use text to describe what they do; others come complete with icons. When you copy a button you also get its script, which provides the button's action.*

3. Click the Button tool.

Whenever you want to create, copy, or modify a button you use the Button tool.

4. Click to select the button that looks like multiple cards speeding by.

The button says "View" beneath it. When you're using the Button tool, all buttons on the card are outlined. All the buttons on the card or in the card's background are outlined in black. You can see which buttons are in the background by holding down both the Command and Option keys. Background buttons have a double border. Clicking any one of the

outlined buttons selects it so that you can modify or copy it.

5. Choose Copy Button from the Edit menu.

Just as HyperCard copies a picture when you have a picture selected, and text when text is selected, whenever you've selected a button, HyperCard copies the button and any script that button has.

Hint

If you're copying a button that includes an icon, it will be copied along with the button. You can tell whether a picture is an icon because it's limited to 32 by 32 pixels in size and it moves along with the button when you drag it with the Button tool. Use icons where you can to save yourself some work in creating a picture. But if the button is separate from the picture, you must use the Paint tools to copy the picture separately, thus going through the copy and paste routine twice.

6. Press the tilde key to retrace your steps back to the Calendar stack.

You want to paste the button into the calendar.

7. Choose Background from the Edit menu to edit the card's background.

Just as with pictures, buttons can be in the background, where they're shared by every card that has that background. Or they can be on the card itself.

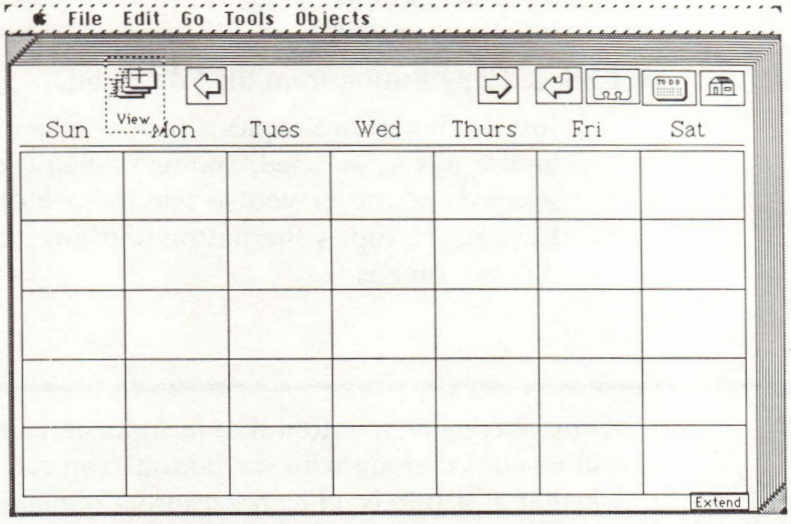


FIGURE 3–12 *Using the Button tool to move a button*

8. Choose Paste Button from the Edit menu.

The button is pasted onto the Calendar stack's background, in the same position you copied from. You will quickly grow to love the way HyperCard remembers objects' positions and dumps them into the same relative spot. It saves you lots of time in adjusting objects to line up properly.

However, in this case the button's old position isn't the best place for the button in the new stack.

9. Use the Button tool to drag the button to the top left corner of the calendar.

You can see where to place this button in Figure 3.12.

Hint

As with fields, you don't have to be in the background to move this background button. The only time it matters whether or not you're in the background is when you create a button or field.

10. Try the speeding cards button by clicking it using the Browse tool.

When you choose the Browse tool, HyperCard takes you out of the background. Clicking the button flips you through the pages of your calendar so that you can quickly see what's coming up.

11. Use the Button tool to double-click the button.

Double-clicking the button with the Button tool brings up the Button Info dialog box for that button.

Standard Macintosh buttons are highlighted whenever you click them. Autohighlighting is also one of the properties you can set for HyperCard buttons.

12. Click the Auto hilite check box and then click OK.

Clicking Auto hilite sets the button to be inverted (black becomes white and vice-versa) whenever you click the button using the Browse tool.

13. Use the Browse tool to try the button again.

Now the button is highlighted whenever you click it.

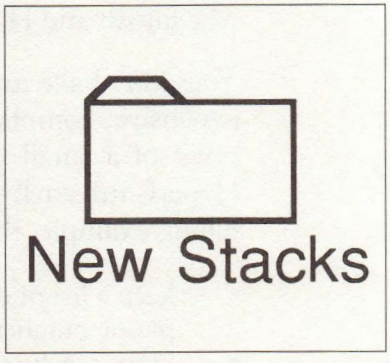
Summary

In this chapter, you've had a taste of building new stacks by starting with a stack seedling and modifying it to create a stack. You can also create a stack from the ground up, designing it from spare parts you find around HyperCard and some new parts you create yourself. The next chapter tells you how to design a stack from scratch.

Chapter 4

Designing New Stacks

"I can see that a certain amount of restraint is needed."
Kevin Kelly, Editor,
The Whole Earth Review



In the last chapter you saw how you can use the stack seedlings in the Stack Ideas stack to create and customize new stacks. You built an organizing stack for your financial information and you built a calendar where you could see a month at a time. Building stacks out of

stack seedlings is the easiest way to get started in building your own stacks.

This chapter shows you how to design and create a stack to keep track of a small business—your customers and the services you provide for them. Although this example shows a housekeeping business, you could use the stack for any small business. And you can use the techniques you learn in this chapter to build any custom stack.

Thinking About What the New Stack Should Do

Maybe you've always kept track of your small business on paper, using a calculator to total up costs, keeping records in the top drawer of your desk, wondering from time to time if you missed a payment from one of your clients or if you charged them accurately for last month's service. But now with a Macintosh and HyperCard, it's time to get organized.

You don't have to be a mathematical whiz or buy an expensive, complicated accounting program to keep track of a small business. You can do it all with a HyperCard stack you design yourself. For this client billing example, the new stack should:

- keep a list of clients, with name, address, phone number
- keep track of services rendered and amounts due, including amounts past due
- provide a simple bill that you can print and send to the client

Creating a New Address Stack

The Address stack that comes with HyperCard is already set up to keep track of people's names, addresses, and phone numbers. You could just put your client list there. But sometimes it makes sense to have a separate list for a specific purpose—so that you can print labels to send bills or advertising to your clients, for example. And in the case of this small business, it would be convenient to combine this address list with a simple bill you can send to your customers, as shown in this chapter.

Why Create a New Stack?

You already have an Address stack. You could add your clients' names to it and modify the new cards to hold additional information that the original Address stack doesn't allow for. One reason to start a new stack is just for mental organization and comfort. It might just feel better to have all of your business contacts together in one stack. Another reason is that you might decide to make major changes to the stack—changes that wouldn't work for cards in the original Address stack.

You could later add a new background to the original stack to meet the needs of your client list, but it's cumbersome to move existing cards into a new background later. It's also generally a good idea to minimize the number of different backgrounds within a stack because a stack is at its best when it looks and

feels like a cohesive unit—designed that way rather than grown into a multibackgrounded hodgepodge. It's best to anticipate early on that you want some separation and to start a new stack right then and there.

Hint

Try to anticipate what you want a stack to do and plan it from the start. It's much easier than reorganizing a stack that already has many cards in it.

Here's how to create a new stack starting with the existing Address stack:

1. Go to any card in the Address stack so that you can use its background as a model for your new stack.

Remember that most of what you do in HyperCard is based on copying what you need from somewhere else—a button from the Button Ideas stack, a card from the Card Ideas stack, or a background from the Stack Ideas stack, for example. In this case you're really doing the same thing you did when you made the calendar from the calendar seedling, except that you're copying a background from an existing, working stack instead of from a collection stack.

2. Choose New Stack from the File menu.

HyperCard presents a list of stacks that already exist in the current folder (the folder the Address stack is in). There's a blank for the new stack's name, and a checkbox for copying the current stack's background, as shown in

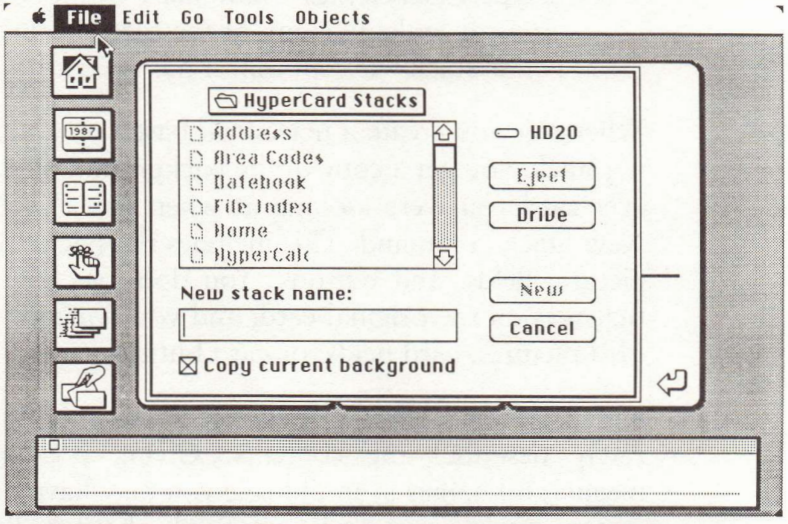


FIGURE 4–1 *You're not limited to the Stack Ideas and Card Ideas stacks for starting new stacks. You can use any existing stack as a model. Starting a new stack from the Address stack gives you all the functionality of the original stack without its text.*

Figure 4.1. If you unchecked this box, your new stack would start out with a blank background, in which you could create or paste in new background pictures, buttons, and fields. But for this client list you'll copy the background used by the Address stack. Therefore, leave the Copy Current Background box checked.

3. Name the stack by typing "My Clients" in the blank space, and then click New.

When you click New, the stack is saved in the folder named at the top of the list. You could also use the Drive and Eject buttons to move among folders and disks to place the new stack anywhere you want it. The title at the top of the list names the current folder, where the new stack will be placed.

HyperCard creates a new stack with one card in it. It looks like any of the cards in the Address stack—except that it has no text.

Whenever you create a new stack based on an existing stack, you get a copy of the background of whatever card you were looking at when you chose the New Stack command. This includes the background picture, fields, and buttons. You don't get the text that was on the original card, and you don't get any card pictures, card fields, or card buttons.

Each time you create a stack, use a name that accurately describes the contents. Giving documents meaningful names is an old rule you may have heard so many times you no longer think about it. But if you're not yet doing it, start now. You'll be amazed at how much more organized you'll feel if you can scan stack names in the Finder and know exactly which stack you want. And if you keep dated versions of stacks—"Expenses August 1987," for example—you'll be able to get to the right stack without having to check its last modified date.

Getting Information About Stacks

You can always find out information about the current stack using the Stack Info command in the Objects menu.

The Objects menu is devoted to information about objects. "Objects" is the general word for things like stacks, backgrounds, cards, fields, and buttons. You don't see the Objects menu unless you've checked User Level Author or above on the Preferences card

in the Home stack. But once you've become an author (as you have when you've used the Button or Field tools), you can use the Objects menu to find out about any object in HyperCard, including stacks you yourself create.

Hint

If you don't see the Objects menu in the menu bar, go Home and press the left arrow key to go to the User Preferences card. Click Author.

1. Choose Stack Info from the Objects menu.

The dialog box that appears (shown in Figure 4.2) shows the name you just gave this stack—My Clients. It also tells you where the stack is, including the disk it's on followed by the names of any folders it's in. Colons separate each level of this pathname. Right now this stack contains just one card (HyperCard starts you off with one) and one background—the background you copied from the Address stack.

Also shown here is the current size of the stack—including the current card and the one background—and the amount of space free—currently 0. The amount of space free is the minimum amount you would save if you chose Compact Stack from the File menu. It's nothing now, but as you work on a stack—adding and moving fields, buttons, and text—space gets used on the disk that you can recover by compacting the stack. The stack's performance also gets slower as the disk space is used less efficiently. Compacting a stack reduces it to its minimum size and maximum efficiency.

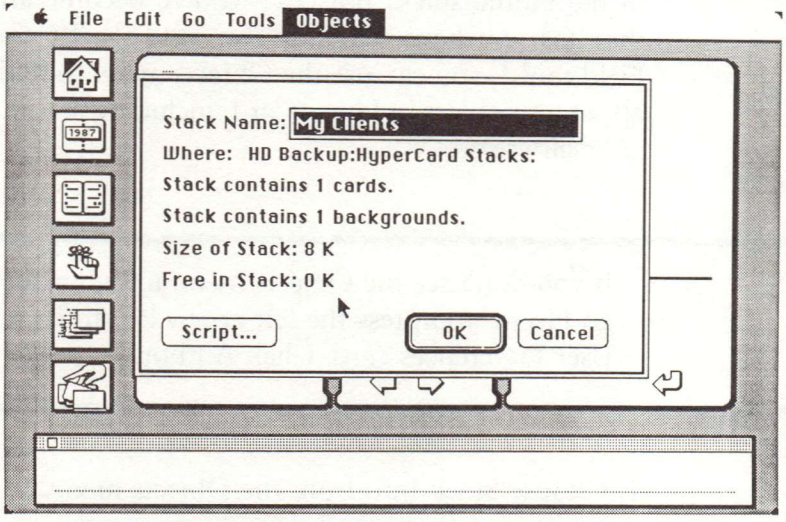


FIGURE 4–2 *The Stack Info dialog box gives you information about the current stack—its name, which you can change here if you want, where the stack lives (colons indicate a hierarchical level such as a folder, disk, or file server), the number of cards in the stack, the number of backgrounds in the stack, and the amount of space that could be recovered on the disk by using the Compact Stack command in the File menu.*

Hint | Compacting stacks keeps them as small and as fast as possible.

2. Click Cancel to go back to the card.

You can have more than one stack with the same name, as long as the stacks are in different folders. The "Look for Stacks In" card (see Figure 4.3) in the Home stack tells HyperCard which folders to look for stacks in.

If you have two stacks with the same name in two different folders, HyperCard will normally open the

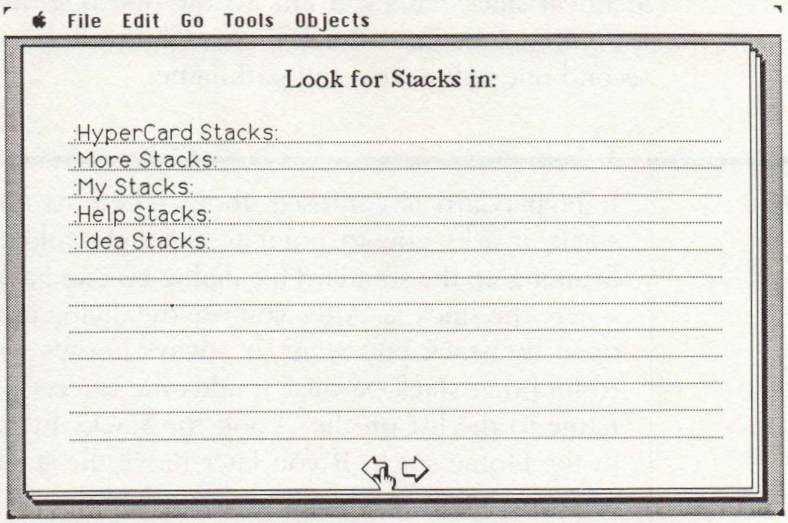


FIGURE 4–3 *The "Look for Stacks In" card in the Home stack lists the places HyperCard knows to look for stacks. When you add a stack to any folder not listed on the card, you add the folder's entire pathname to the list.*

Hint

Always choose Cancel in a dialog box when you want to leave something the way it is. In some cases, clicking OK may record some information you really don't want. For example, when you open a button's script, HyperCard puts some information into it—a control structure you'll learn about in the next chapter. If you aren't going to add something else, you don't really want HyperCard's additions either. Better to Cancel.

one in the first folder named in the list of pathnames on the "Look for Stacks In" card. Anytime you use LinkTo in the button dialog box to make a link, the link will go to the correct stack. In the case of two

identical stack names, a link to the first one on the list will include just a stack name, while a link to the second one will have a full pathname.

Hint

If HyperCard is confused about where to find a stack, it asks you to point to the right folder by bringing up the standard file dialog box and asking where the stack is. After you use the dialog box to open the stack you want, it always knows where to find that stack because it adds the correct pathname to the list on the "Look for Stacks In" card in the Home stack. If you later move the stack, it asks you again and again updates the list.

Many stacks include more than one background—to have different background buttons, fields, or pictures within the same stack. The Address stack has just one background, however. Therefore, it doesn't matter which of the Address cards you're at when you choose New Stack. You'll get the same background.

Hint

You can find out how many backgrounds exist in the current stack by choosing Stack Info from the Options menu.

Adding Clients' Names

Start to develop your new stack by adding cards for a few clients. Here's how to do it:

1. Use the Browse tool to click with the I-beam pointer near the top left corner of the card—in the name and address field. When you see an insertion point, type a client's name and address.

As you move the pointer around in the top left corner of the card, notice that the pointer changes shape, depending on where you've positioned it. When the pointer is over a text field it becomes an I-beam so that you can add text. This is true anywhere in HyperCard. The background picture of your new stack includes an angle bracket to give you a clue where to click, but many stacks don't indicate text fields with a picture. The I-beam pointer is your clue you can click to edit text.

Hint

Be consistent in ordering all of the names in the same way (last name first, usually). Later when you sort the stack you'll be glad you did.

With most word processors, clicking anywhere beyond the text causes the insertion point to jump to the first available space following the existing text. But HyperCard adds extra Return characters when necessary, so that clicking in any spot places the insertion point at the nearest left margin.

Hint

You can also press the Tab key to select the next field. If there's nothing in the field, HyperCard places an insertion point there. If there's already text in the field, HyperCard selects the existing text and anything you type will replace that text.

There's a second field on this card just for phone numbers.

2. Click in the second field or press the Tab key and type the client's phone number or numbers.

Keeping the clients' names in a separate field from their phone numbers means that later, if you want to print just their names and addresses (for mailing labels, for example) you can select and print just the name and address field. Or if you want to sort the list by telephone area code, you can do that by sorting by the phone field.

Hint

If you get in the habit of typing phone numbers in a consistent way—area code surrounded by parentheses, prefix and number separated by a hyphen—dialing a number you place in the Message box by Command-dragging over it and then pressing a dial button will work consistently.

3. When you've finished adding the first client's information, add a card for another client by choosing New Card from the Edit menu or pressing Command-N.

A new, blank card appears. The text you type into a field belongs to the specific card you type it on, even if the field is a background field. Text you type onto this new card will belong specifically to it. Type in the information for another client on the next card.

Press the Tab key to get an insertion point or click in the first field.

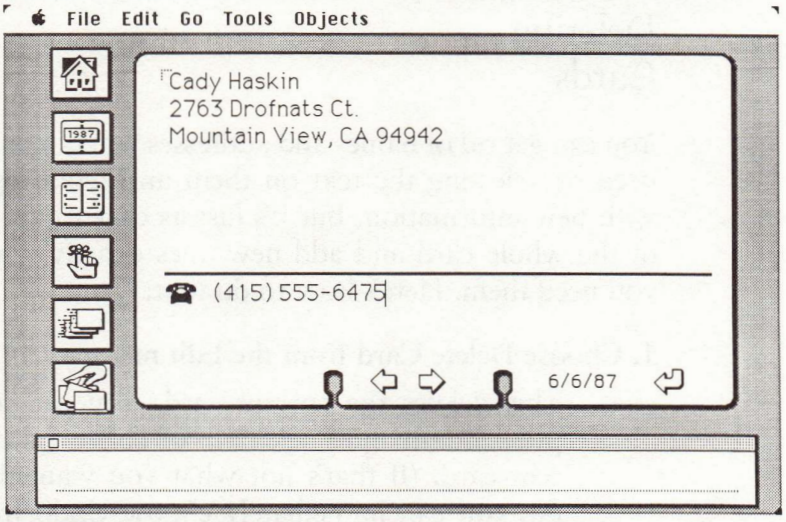


FIGURE 4–4 *Typing into an address card. You can type whenever the Browse tool pointer is an I-beam. Click to get an insertion point.*

Figure 4.4 shows a new card with its information filled in.

4. Add the rest of your clients' names to new cards in the stack, choosing New Card, tabbing, and typing for each client you want to add.

Of course, you don't have to add all of the clients now. Whenever you want to add a new name, choose New Card from the Edit menu, click with the I-beam pointer, and type the new information. New cards are always added after the card you're looking at when you choose New Card. You can alphabetize as you go, or add cards anywhere and use the Sort button to rearrange them, as shown in the next section.

Deleting Cards

You can get rid of names and addresses you no longer need by selecting the text on them and replacing it with new information. But it's just as easy to get rid of the whole card and add new ones exactly where you need them. Here's how to do that:

1. Choose Delete Card from the Edit menu.

This deletes the current card. You can also press Command-Backspace to delete the current card. (If that's not what you wanted to do, you can immediately choose Undo from the Edit menu.) When you delete a card, HyperCard takes you to the next card in the stack.

Hint	Undo can recover cards you didn't really mean to delete.

Ordering and Sorting New Stacks

Cards are arranged in the order you add them. Every new card you add is placed following the card you're looking at when you add the card. So if you want to keep the cards in alphabetical order, you could just create them that way—from A to Z. And when you want to add new cards, you can go to the card that

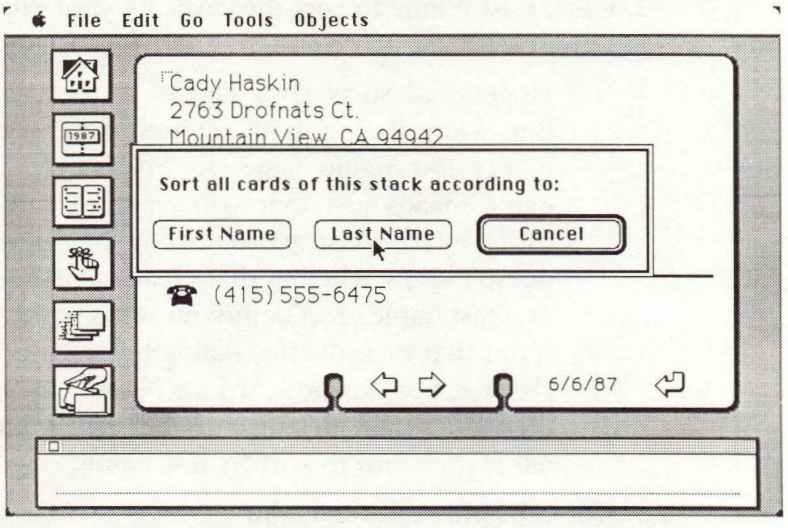


FIGURE 4–5 *The Sort button's script presents two possibilities for sorting the stack—by the first or last name in the first line of the Name and Address text field.*

would precede the new card alphabetically, create a new card there, and add the new name and address.

You don't really have to worry about whether you add cards in alphabetical order or not. You can change the order the cards are sorted in whenever you want, by following these steps:

1. Click the Sort button to the left of the Address card.

The Sort button is at the bottom left of the card. It has a hand rearranging cards. Clicking it presents a dialog box that gives you options for sorting the cards in the current stack—by first or last name—or to Cancel. Figure 4.5 shows this dialog box.

2. Click Last Name to sort the cards by your clients' last names.

HyperCard sorts your clients by last name. But what if you had entered your clients' names last name first? To HyperCard, last name means just that you want to sort the cards by the last word of the first line. It doesn't know whether that's really a first name or a last name. You'll find out ways to edit the script that presents this dialog box in the next chapter, but, for now, if Last Name were first on the cards, you'd just click First Name to tell HyperCard to sort by last name.

3. Click the Sort button again.

4. This time sort by First Name to alphabetize and reorder the cards by the first word in the first line of the field.

Although you can sort cards by whatever scheme you choose—last word of first line, for example, or first word of first line, it's generally a good idea to think about keeping consistency in the order of items on a card. It's true that HyperCard can find a name whether you've entered it first name first or last name first, but following a consistent order when you add names to cards will allow you to use the arrow keys (either the arrow keys on the keyboard or the buttons in the stack that point to the left or right) to move sequentially through the list in the order you want. Keeping the list consistent also lets you print the stack in a consistent order. As long as you keep them on the Macintosh screen, you probably won't notice any problem finding things in HyperCard. But once you have to

rely on your eyes scanning and juggling a printed client list, you might wish you'd kept it in alphabetical order.

Designing the Way the Bill Looks

Although you probably want your client list to keep track of your customers' names and addresses (just as the Address stack does), you could also use this list as customers' bills to send out at the end of the month. Traditionally, you'd probably keep track of your customers' names and addresses with a database and use a spreadsheet to keep track of what they owe you. But with HyperCard you can combine both functions into a single stack.

You can add fields to keep track of the amounts currently due and you can add buttons that calculate what your clients owe you. You can even change the entire look of the stack by adding a picture to its background.

Changing the Background Picture

You can borrow an entire background to use in your own stacks—as you just borrowed a background from the existing Address stack to use as a template for your client list. But you don't have to borrow an entire background. You can also borrow just a picture to use in a background or on a single card. Any picture in any existing stack is fair game.

Some stacks are collections, rather than "working" stacks. The Card Ideas stack is a collection of pictures to use generally as single cards within your stacks. The Stack Ideas stack generally has complete backgrounds to use as models for new stacks. But like any pictures anywhere in HyperCard, the pictures in the Stack Ideas stack can be copied and pasted onto other cards in other stacks—either in the background or on individual cards. And you don't have to take an entire picture. You can also select just part of one of these pictures and add only that part to a card or background anywhere.

In the following exercise, you'll learn how to change a background's look by replacing its picture with one you copy from somewhere else.

1. Go to the Stack Ideas stack, either by using the Open Stack command and selecting Stack Ideas or by using the Browse tool to type: Go stack "Stack Ideas."

Stack Ideas is in the Idea Stacks folder.

2. Press the Return key.

You need to include quotes around the words "Stack Ideas" because otherwise HyperCard would look for a stack named "Ideas" and be confused. Whenever HyperCard is confused about which stack to open, it presents the standard file dialog box with the question, "Where is Ideas" (the stack you asked for). Stack names that start with the word "Stack" do need to be in quotes, because HyperCard interprets the word "stack" as a general request for a stack rather than as part of the stack's name.

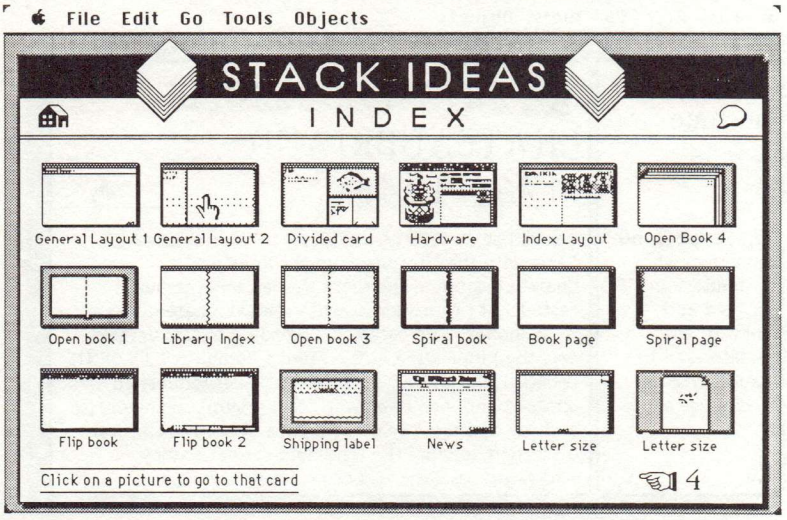


FIGURE 4–6 *Stack Ideas has several cards of seedlings to use in starting your own stacks. This one includes a General Layout design that's basic enough to be used for many different stack designs.*

Hint

Whenever a stack name is more than one word, it doesn't hurt to surround the stack name with quotes whenever you tell HyperCard to go to that stack. This tells HyperCard that you're specifically designating the name of the stack.

3. Click the hand that points to the right at the bottom right of the card three times to go to the fourth card of the Stack Ideas index, as shown in Figure 4.6.

The pointing hand is one of the conventions HyperCard authors use for "Go to the next card." Use the hand at the bottom right of the page to move through the cards in this stack.

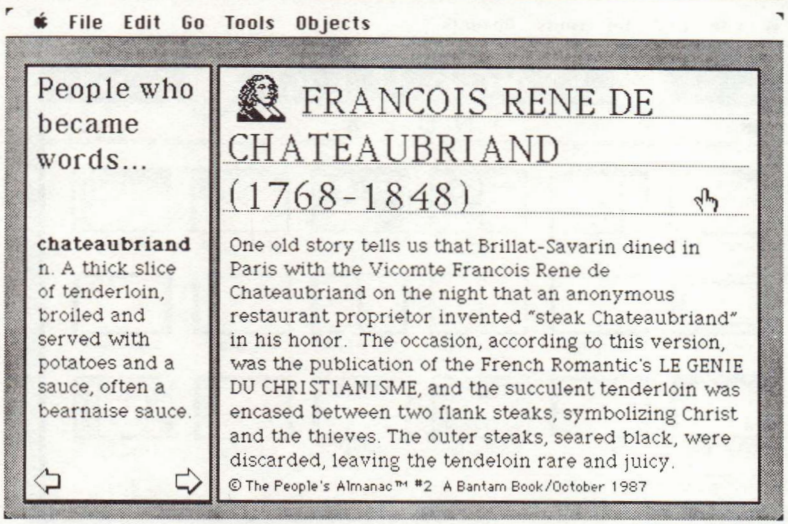


FIGURE 4–7 *Although stack seedlings contain a complete background you can use in new stacks, you don't have to copy the entire background. You may just want the picture—without copying the fields and buttons that are there.*

4. Click the "General Layout 2" miniature to go to the card shown in Figure 4.7.

Hint

You can make these miniature pictures yourself by going to any card in HyperCard, choosing Copy Card from the Edit menu, and then holding down the Shift key while you paste.

5. Drag down and off the Tools window to place it anywhere on the screen.

If you have a large-screen Macintosh, you've probably left the Tools window out to the side of the HyperCard window so you can get at tools easily without choosing from the menu.

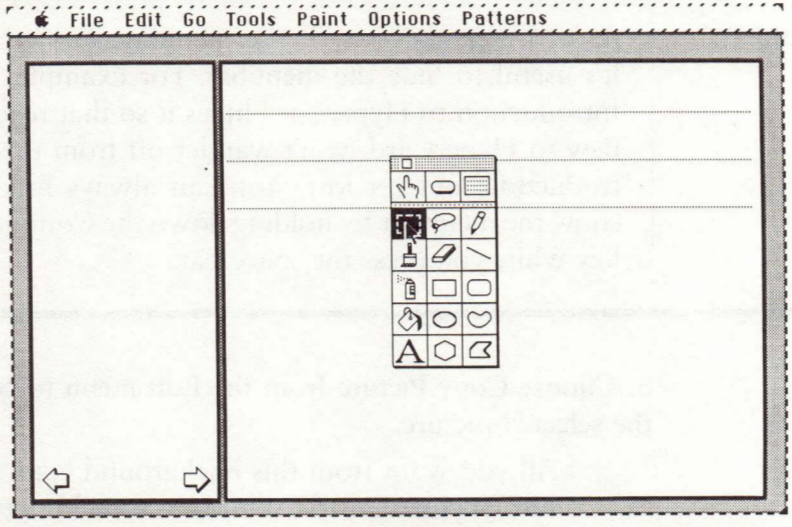


FIGURE 4–8 *Copying just the picture of the General Layout 2 seedling and pasting it into your Client stack's background changes the look but not the functionality of your stack.*

6. Choose Background from the Edit menu to edit the background.

The text and card picture momentarily disappears because it's in the card, not the background. But you still see the background picture—the card layout against gray—as shown in Figure 4.8. And if you clicked the Button or Field tool you'd see the background buttons or fields outlined in gray.

7. Double-click the selection rectangle to select the entire picture.

Marching ants appear around the entire card picture to show it's selected. You can't see the ants at the top of the picture because the menubar is hiding them, but they are selected anyway.

Hint

A card extends beneath the menubar. Sometimes it's useful to hide the menubar. For example, the Introduction to HyperCard hides it so that readers new to HyperCard won't wander off from the Introduction and get lost. You can always hide or show the menubar by holding down the Command key while you press the space bar.

8. Choose Copy Picture from the Edit menu to copy the selected picture.

All you want from this background is its picture. If you wanted the buttons or fields, you'd copy the entire card using the Copy Card command in the Edit menu. Copy Card copies the card and its background and adds the background to the stack you copy the card to.

Hint

Pressing the key at the top left corner of the keyboard right now will undo your latest painting action—in this case, copying the picture—rather than moving you back through cards. HyperCard does this to be consistent with MacPaint and other Painting applications. While you're using the Paint tools—as you are when you click the selection rectangle—you can go back by pressing Command tilde or Escape. You could also click the Browse tool to leave the Paint tools and then click just the top left key to go back as usual.

9. Choose Message box from the Edit menu (or type Command-M) and then type: Go stack "My Clients" and press Return. This takes you to your new stack.

10. Choose Background from the Edit menu or press Command-B to work with the stack's background.

If you don't choose Background, you'll be working just with the current card, and any picture you paste in will appear only on that card rather than on all cards you add to the stack.

11. Choose Paste Picture from the Edit menu or press Command-V.

HyperCard switches you back to the selection rectangle (if you left it). It pastes the picture you copied from Stack Ideas into your new stack's background. Because the new picture is the full card size, it completely replaces the picture that was there—the picture of the Address card.

Pasting a new picture into the background doesn't replace the other parts of the background that were there. It replaces only the background picture. So the fields that hold the name and address and the phone numbers are still there. And so are the buttons that take you Home, dial the phone, flip through the cards, and sort the cards. You've combined features from two stacks to give you exactly the look you want.

Hint

A picture is just a picture. You can copy a picture from a background and place it on a card, or vice-versa. Or you can grab a picture from a collection such as the Art Ideas, Card Ideas, or Stack Ideas stacks or from the Scrapbook or a MacPaint document, and paste it onto either a card or a background.

12. Choose the checked Background command from the Edit menu or press Command-B to return to the card.

Now you can see the card and its background on your screen.

Designing the Buttons

Your client stack inherited seven buttons from the Address stack you used as a model for the new stack. The Dial button might come in handy when you want to call one of your clients. You'll use the Sort button to keep your clients' names sorted, and you'll need the button that flips through the stack. But you don't really need the buttons that go to the calendars and To Do lists.

Hint

Even though these are background buttons, you don't need to be in the background to edit their structure.

Getting Rid of Buttons You Don't Need

Here's how to delete the unwanted buttons:

1. Click the Button tool in the Tools palette and then click the button that goes to the yearly calendar.

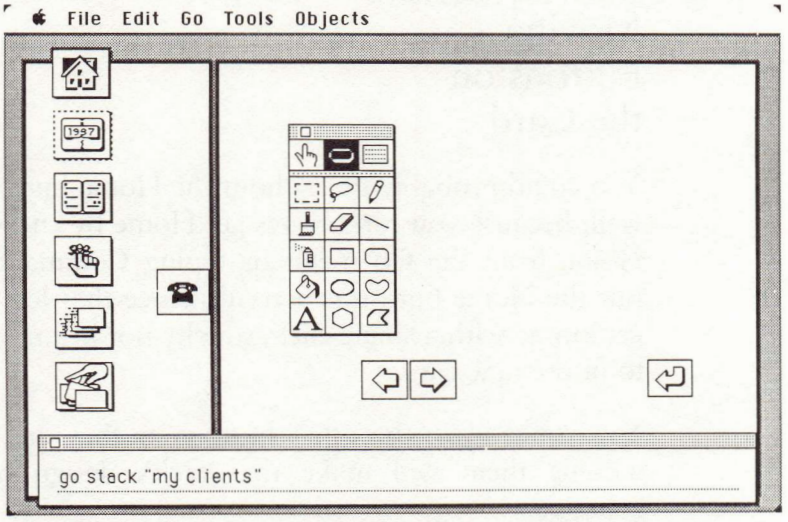


FIGURE 4–9 *The original buttons in your Client stack now need to be moved to fit the new picture. And you can delete any button you no longer need by selecting it and pressing the Backspace key.*

This selects the Yearly calendar button so you can delete it. Figure 4.9 shows the button selected.

2. Press the Backspace key.

The button is deleted.

Hint

You could also select the button by clicking it and then choose Cut Button or Clear Button from the Edit menu to delete it.

3. Do the same with the Weekly calendar button and the To Do button. Select each in turn and press the Backspace key.

Moving Buttons on the Card

You could probably do without the Home button as well, because you can always get Home by choosing Home from the Go menu or typing Command-H. But the Home button is a convenience that lets you get Home with a single click, so why not just move it to fit the new card.

You could leave the other buttons as they are, but moving them will make the stack's design more cohesive.

Here's how to move buttons:

1. Drag the Home button from its center downward to move it to the top left corner of the card.

Here's more HyperCard serendipity at work. In addition to taking you to the Home card, the Home button adds a nice graphic touch to your Housekeeping business stack.

2. Drag the Sort button from its center to move it to the bottom right side of the left panel.

Hint

Dragging from an edge outward changes the button's size; dragging inward makes it smaller; dragging from the center moves it.

You can also close the Message box if you want by clicking its close box.

3. Drag the button that flips through the stack to the left of the Sort button.

This arrangement visually balances the buttons.

Hint

You can line up buttons by dragging one over the other until they align perfectly, and then pressing the Shift key and dragging the top button away. Holding the Shift key down constrains the button's movement to the same horizontal or vertical plane.

4. Drag the arrow buttons that go to Previous, Next, or Return cards so they're near the bottom of the new card.

Leave the phone button where it is. It still lines up with the Phone Number field. Figure 4.10 shows the buttons in their suggested places. The Return button is selected.

Using Paint Text

You've been using the Browse tool to add text to fields. But HyperCard also has another kind of text—text you create using the Paint text tool.

Unlike regular, structured text Paint text can't be searched, and you can't edit it in the way you select and edit structured text. But Paint text has some def-

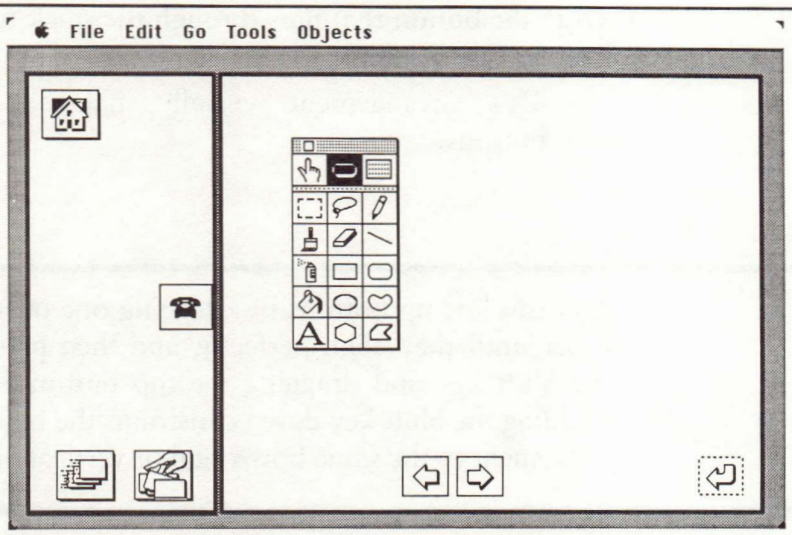


FIGURE 4–10 *You use the Button tool to rearrange the original buttons to fit the new picture.*

inite advantages as well. Because it doesn't need to be in a field, you can create it on the fly and position it anywhere you want it. And you can select it and rotate it, flip it, fill it with patterns, or do anything else you can do with a picture.

With Paint text you can also use text fonts that may not later be available in the System file. You do need to have the fonts you want to use for Paint text available in the System file when you create the text; you just don't need them later, when you're using the stacks you create. Using Paint text has the advantage that once added to a card or stack, it's stored not as text but as a picture, so it doesn't require that any special fonts be present in the System file. This gives you a lot more freedom in using unusual fonts.

Hint

If you're making stacks to be used by other people, remember that they will have available for real text only the fonts that are installed in their current System file. Try to use the fonts everyone has on their current System file.

Identifying the Fields

Now you'll add fields to this stack that show the amount the customer owes you. Using Paint text in the stack's background to identify these fields will make the text appear on each card in the stack.

1. Click the Paint text tool in the Tools window.

It appears as the letter A in the bottom left square.

2. Choose Text Style from the Edit menu or double-click the Paint text tool.

Double-clicking the Paint text tool is another way to get the Text dialog box. When you're using the Paint text tool, any choices you make apply to the current Paint text—the text you're about to type or the text you just finished typing.

3. Select the New York 14 font, or any font that appears on your list. Click Align Right.

This will align the text on the right side so that it lines up with the fields you're going to move here.

4. Click OK to confirm your changes.

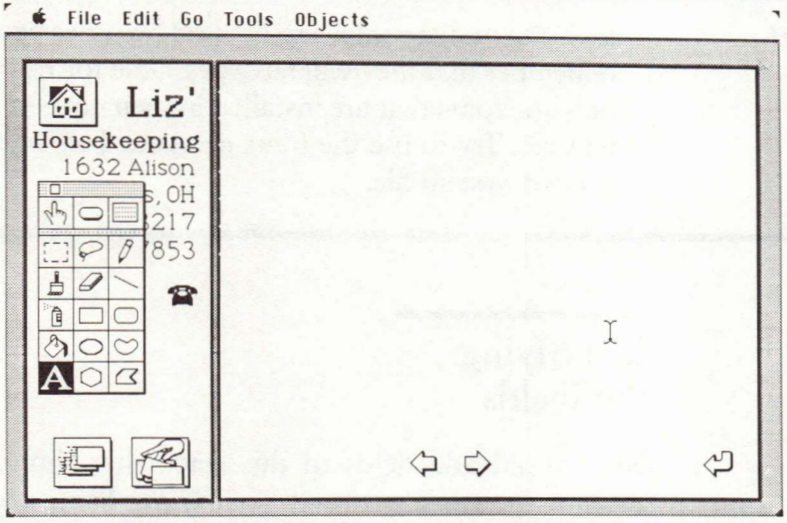


FIGURE 4–11 *Selecting a spot to add Paint text by clicking the Paint text tool and then clicking where you want the text to go.*

Hint

Clicking anywhere, going to another card, or choosing a different tool when you've just typed Paint text confirms the text you just typed. That means it is no longer text but a picture, and you can't change it in the way you usually change text—by Backspacing or dragging across it to select it, for example.

5. Choose Background from the Edit menu.

You want to add the Paint text in the background of the card so that it will appear on every card in the stack.

6. Click in the bottom right part of the card, at about the spot shown in Figure 4. 11.

You can move this text later, so don't worry if it isn't in exactly the right place.

Hint

Remember that the card includes everything you see on the screen—not just a part that looks like a "normal" card. The card may actually include the surrounding gray area that looks like background but isn't.

7. **Type: Past Due**

   As you type, the text scoots to the left because you've told HyperCard to align the text on the right side at the place you clicked.

8. **Press the Return key.**

   This moves the insertion point to the next line, just below the right side of the first line.

9. **Type: Charges**

   The right margin aligns with the field above.

10. **Press the Return key.**

    You move again to the next line, where you can type a label for the total amount now due.

11. **Type: Total Due**

Hint

You can Backspace to delete characters or double-click the Paint text tool to make new choices for the current text as long as you haven't yet confirmed the text by clicking, choosing another tool, or going to another card.

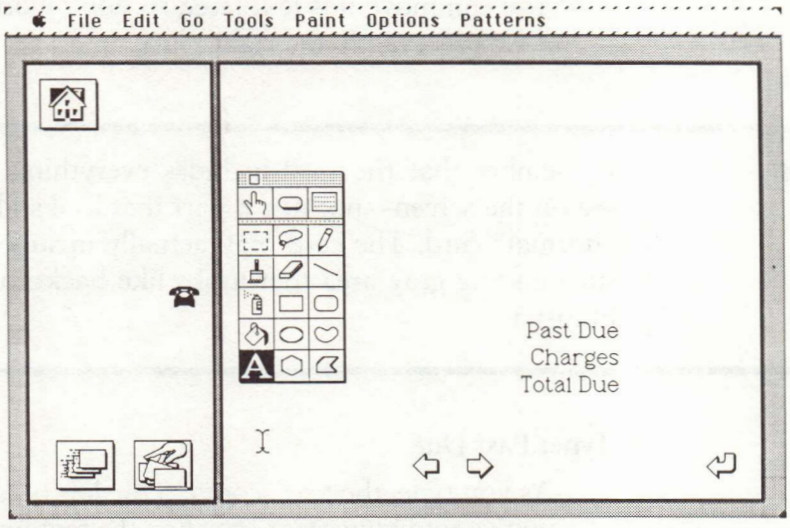


FIGURE 4–12 *Using the Paint text tool to add "Past Due," "Charges," and "Total Due" labels*

Figure 4.12 shows the "Past Due," "Charges," and "Total Due" Paint text. Note that the right margins are aligned.

12. Click anywhere to confirm the text.

Adding Your Business Logo

Paint text is also a good choice for your business logo. You don't need to search on this text, and you can include Paint text on every card by putting it in the background.

Hint	Whenever you want the same image (a business logo, for example) to appear on every card that shares a common background, use Paint text in the background.

Here's the way to add your logo:

1. Make sure Background is still checked in the Edit menu.

Remember, you can confirm this by looking for the short slashed lines in the menu bar.

2. Click the Paint Text tool if it's not already selected.

3. Choose Text Style from the Edit menu or double-click the Paint Text tool.

Double-clicking the Paint Text tool is a quick way to get to the text dialog box.

4. Choose New York 14 Bold. Align the text on the right side by clicking Align right and make it bold by clicking Bold.

5. Click OK to confirm your choice.

6. Click near the right side of the left panel and type your business name.

Use the Backspace key to correct mistakes and the Return key to start new lines.

Notice that if you type too close to the button, you obscure it with the Paint text you type. It's easy to fix this by using the selection rectangle to select the Paint text and then dragging it away from the button.

Now you'll add your business address.

7. Click directly under the last letter of the last line you typed. Then double-click the Paint text tool to get the Text dialog box back again.

Choose a smaller point-size number for your address and phone number, so that your logo stands out and uncheck the Bold style.

Hint

It's important to click again before you change the text style. Otherwise, the choice you make will affect all of the text you just typed as well as what you type next.

---

8. Type your address and phone number.

Use the selection rectangle or the lasso to move any text you want to adjust.

Hint

Choosing Select or pressing Command-S just after you've typed text selects the text you typed.

Stretching or shrinking Paint text

Because your logo is done with Paint text, you can stretch or shrink any of it. Here's how:

1. Use the selection rectangle to surround the text you want to alter—maybe the first part of your business name.

2. Hold down the Command key while you drag a corner of the selection to stretch or shrink it.

Dragging outward stretches the image vertically or horizontally or both, depending on the angle at which you drag. Dragging inward shrinks the image; dragging outward stretches it. Experiment until you have the shape you want. Use the Undo key to undo experiments or use the Eraser tool to erase.

3. Choose the checked Background command to return to the card.

Hint

Choose Keep to save the current picture as the one to revert to when you chose Revert. It's a good idea to choose Keep right before you're about to embark on any major change to a picture but want to preserve some changes you've made since you came to the current card.

Adjusting the Text Fields

The Address stack you used as a template for your new stack included three text fields that contain your clients' name and address, phone numbers, and the date of the last change to the card. (Whenever you make any text change to any card in this stack, the date in this field is automatically updated so that you know how current your records are.)

Although these fields are useful in your new stack, they're not positioned in the best place to work with your new picture. You can use the Field tool to adjust the fields.

Hint

Make sure your Macintosh clock is set right by choosing Alarm Clock from the Apple menu. HyperCard uses the same time and date settings your Alarm clock shows. It also uses them to record a modification date for stacks (just as any application records and shows in the Finder the date of modifications to its documents).

Hint

Even though these are background fields, you don't need to be in the background to edit their structures.

1. Click the Field tool.

The Field tool is the tool for adding, modifying, or deleting text fields—the containers that hold text. You can also use the Field tool to set a field's text properties such as font, size, and style. (To add or edit the text itself, you use the Browse tool to click whenever the pointer is an I-beam.)

Hint

You can change text style when you have a field selected with the Field tool, when you have an insertion point or any selection in a field using the Browse tool, or anytime you're using the Paint text tool.

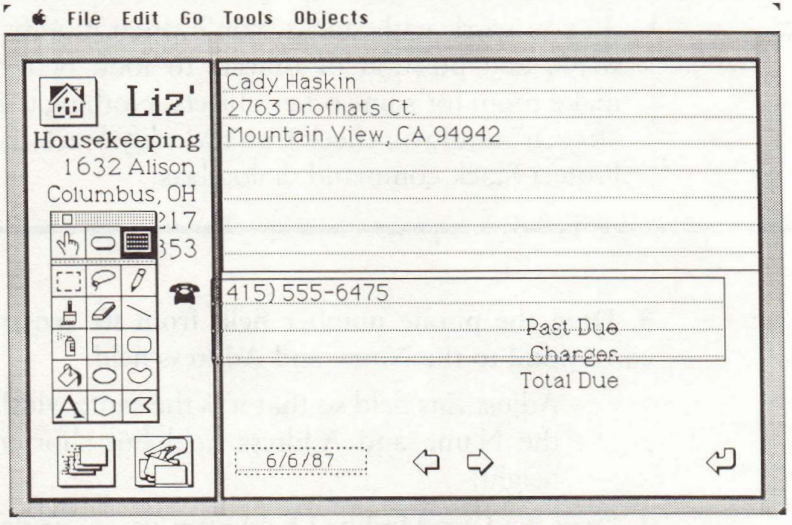


FIGURE 4–13 *You can use the Field tool to make any small adjustment you need to a field—moving it by dragging from the middle or changing its size slightly by dragging from a corner.*

When you're using the Field tool, you can see each field that appears on the card, including card fields and background fields. Remember that even though a field is in the background, the text within that field belongs to individual cards within the background. The field is just the container—what's in it varies from card to card.

2. Drag the Name and Address field from its center down and to the right. Drag inward from a corner to make it slightly smaller so that it fits better on the card.

Tweak its position by pulling on an edge to make it larger or grabbing it from the center to move it. Try to make it look like Figure 4.13.

Hint

As you work with stacks, you'll often modify the shape and position of objects to look better or make room for some other objects. Nothing is ever "frozen" until you check Can't Modify Stack in the Protect Stack command dialog box.

3. Drag the phone number field from its center to correspond to the Name and Address field.

Adjust this field so that it is the same width as the Name and Address field but shorter in height.

4. Drag the Last Modified field from its center down and to the left.

Make it line up at the left with the Name and Address field.

5. Adjust each field so they all sit where you want them on the white part of the card.

Don't worry if the fields aren't the perfect size, shape, or position. You can adjust them again later.

Creating a New Field

You'll use the existing fields for clients' names, addresses, and phone numbers, and for keeping track of when you last updated the bill. (That date will always be displayed in the date field automatically whenever you change any information on the card.) Now you'll add a new field to keep track of what each client owes you. Here's how:

1. Choose Background from the Edit menu to edit the background.

The fields you create should appear on each card in the stack. Although you can edit existing background fields without being in the background, new fields you create are created either on a specific card or in the background—depending on where you are when you create them.

Hint

If you forget to choose Background before you create a new field, you'll get a card field—a field that appears on the current card only. You can easily recover by selecting the field, choosing Cut Field from the Edit menu, choosing Background from the Edit menu, and choosing Paste Field from the Edit menu.

2. Use the Field tool to Command-drag to create a new field below the Phone Numbers field.

You can also create new fields by choosing New Field from the Objects menu. But Command-dragging a new field lets you immediately draw the field any size you want and place it anywhere you want.

You can now make some choices about the field—what it should look like, what its name should be, and what text style it should have, for example. You could choose a font in the same way you did in the last chapter, by selecting the field using the Field tool and choosing Text Style from the Edit menu. But there's another way to choose fonts and also make the other choices you have about fields.

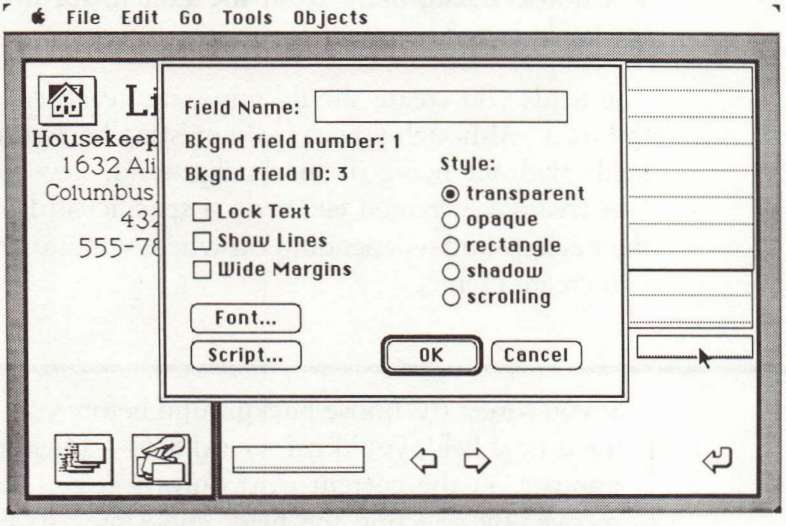


FIGURE 4–14 *The Field dialog box shows and lets you edit the field's name. It also tells you whether this is a card or background field, its unique Identification number within the stack, whether or not it's locked, whether or not it should show lines for the text, whether it should have wide margins, and what style it should be.*

3. Double-click the field to see its dialog box as shown in Figure 4.14.

You can also see this same dialog box by selecting the field and choosing Field Info from the Objects menu.

4. Name the field by typing "Past Due" in the blank.

You don't have to name fields, but if you want to refer to them later in scripts that let you control what HyperCard does, it makes the scripts easier to read and understand if you can just refer to fields by name rather than by number or ID.

5. Click Show Lines.

The field displays lines, just like ruled paper.

6. Click the Font button to set a text style for this field.

This is the same dialog box you see when you choose Text Style from the Edit menu. Once again you have choices for how the text in this field should look.

7. Click New York 14.

Then click OK to approve the choice, and return to the card.

8. Use the Field tool to drag the field inward from any edge to make it smaller and drag it from its center to move it so that it's aligned at the right with the other fields.

Make the new field just big enough to hold one line of text—you'll later use this field to keep track of any past balance due to you from the customer. The horizontal line within the field is the baseline for text you will type into the field. Adjust the Name and Address field, if necessary.

Hint

If you don't get a field exactly in the right place or the right size, you can change it later. Or you can change its text tyle by selecting it with the Field tool and choosing Text Style from the Edit menu (or double-clicking the field to see its dialog box and clicking Font to see the same dialog box).

9. Copy the new field by holding down both the Option and the Shift keys while you drag downward.

This creates a copy of the Past Due field. The new field has the same properties as the orig-

inal field—the same dimensions, the same text style, and the same name. You'll use the copy for the new field.

10. Double-click the new field and type: Charges to rename this field in its dialog box. Click OK.

You'll use this field to keep track of the current charges to the bill.

Hint

Holding down the Shift key while you Option-Copy the field constrains the copy to the same horizontal or vertical planes as the original. This same rule applies to pictures or buttons you Option-drag to copy. It's an easy way to make sure the copies are the same size and in the same place as the original.

---

11. Copy the Charges field by Option-Shift-dragging again.

The third new field will keep track of the amount this client owes you for this month's service.

12. Double-click the new field and type: Total Due in the name blank. Click OK.

Clicking OK confirms the field name and text style. Then it returns you to the card. Check the appearance of the card.

13. Make any adjustments necessary to line up the Paint text with the new fields.

Use the selection rectangle or the lasso to select the Paint text and then drag it to move it as shown in Figure 4.15.

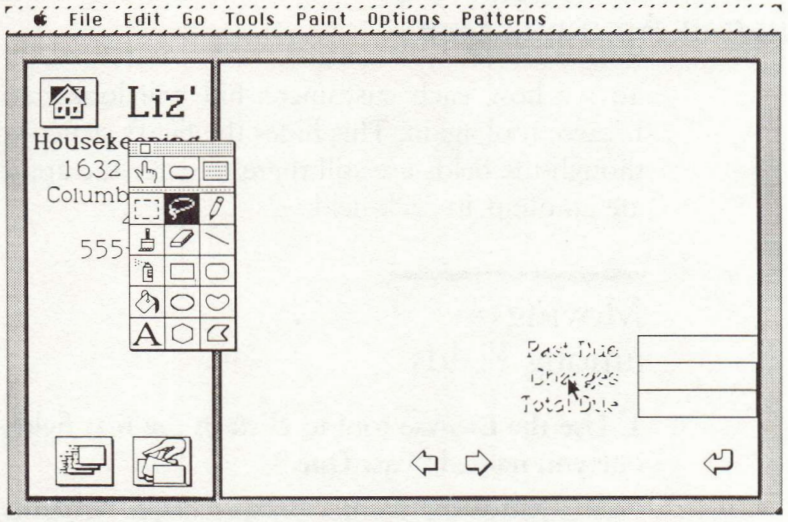


FIGURE 4–15 *Lassoing and dragging Paint text to line up with the fields*

Hint

The Paint text you typed identifies the field, but is completely separate from the field's actual name.

14. Choose Background from the Edit menu to return to the card.

Hint

It's generally easier to use the Field tool to select a field and move it or adjust its size than it is to move a picture to line up with an existing field. This is because you can't see the structure of fields when you're using the Paint tools. You can get around this problem by adding some text to the field so that you can see where it is, or by temporarily changing the field style to one that shows where the field is—rectangle or shadowed, for example.

Filling in Amounts Due

To see how each customer's bill will look, use the Browse tool again. This hides the field's structure, although the fields are still there to enter the appropriate amounts in each field.

Moving among Fields

1. Use the Browse tool to click in the first field—the one you named "Past Due."

Move the pointer on the screen. When it's an I-beam, you're over a field where you can click to get an insertion point. You can also press the Tab key to select an insertion point in the first field in the background. In this stack the first field is the Name and Address field. Pressing the Tab key puts an insertion point there. Pressing the Tab key again takes you to the next field—the phone numbers. The third press takes you to the Last Modified field, and the fourth puts the insertion point in the Past Due field, where you can type in past due charges.

Hint

The Tab key moves you first through background fields, and then through card fields.

2. Type: 40.00

Be sure to use zeros from the top row of keys on the keyboard instead of the character "o." To HyperCard

they're very different. Don't use dollar signs either. HyperCard needs to have plain old numbers to calculate correctly. Later, when you learn to use scripts, you could have HyperCard take off the dollar sign before calculating the numbers, but for now leave them off.

Hint

If the insertion point isn't exactly where you want it, use the Field tool to adjust the size or position of the field.

3. Press the Tab key again or click in the Charges field—the one below Past Due—and type: 40.00

Hint

What you call a field in identifying text you create with the Paint text tool and what you call it in its name blank are totally different things. Any scripts you write later should use the exact field name when referring to the contents of a field.

Hint

If pressing the Tab key doesn't take you to the field you expect, check the field numbers by choosing Field Info from the Objects menu. The tabs take you to the fields in numbered order. The fields are numbered in the order in which they're created. You can change that order using the Bring Closer and Send Farther commands in the Objects menu. Closer objects have higher numbers; farther objects have lower numbers.

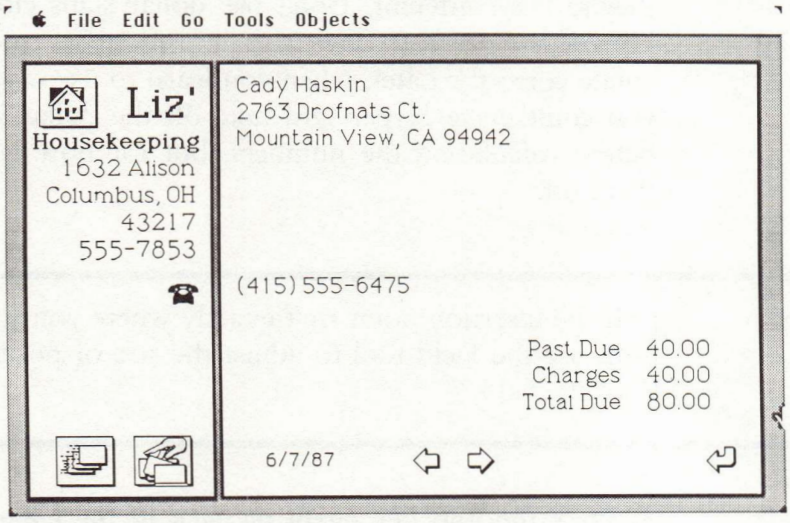


FIGURE 4–16 *The completed bill*

4. Press Tab or click in the third field—the one you named "Total Due" and type: 80.00

Figure 4.16 shows a card with the Past Due, Charges, and Total Due.

Adding Information for Other Clients

Add charges for each of your clients by moving to other cards in the stack, clicking in the text fields, and typing the amounts past due, the current charges, and the total amount due. Use the arrow keys to move to the previous or next card. Delete cards by choosing

Delete Card from the Edit menu. Add cards by choosing New Card from the Edit menu, clicking in the name and address field, and typing the new client's name.

Adding the New Stack to the Home Card

Now you have a stack to keep track of your clients' names and what they owe you. Although you can get to any stack by typing "Go" plus the stack name using the Open Stack command in the File menu, it's much more convenient to have the stack represented on the Home card, with a button that takes you to the stack.

The Home card isn't an actual directory of existing stacks in the way that folders and disks in the Finder are directories. And creating a new stack doesn't automatically add it to the Home card. The Home card is just a card with pictures and buttons. You can add to the Home card the same way you add to other cards.

Hint

The Home card is the first card in the Home stack, which, like any stack, can have as many cards as you want. In addition to the first card, which is a visual directory of stacks, the Home stack includes cards for setting your preferences and for telling HyperCard where to find stacks, applications, and documents you create using other applications.

Adding a Picture to Represent Your New Stack

The pictures on the Home card generally hint at what the various stacks do. The first step in adding your stack to the Home card is to add a picture that represents your new stack.

You've seen the miniature pictures HyperCard uses to represent cards. For example, when you choose Recent from the Go menu, you see miniature representations of the last 42 cards you've seen. And when you scout the Card Ideas or Stack Ideas stacks you also see miniatures of the cards within the stacks.

You can create your own miniatures of any card. Do the following now to make a miniature of your client list stack.

1. Click the Browse tool and go to the first card in the client stack, if you're not already there.

Hint

Pressing the Command and left arrow keys simultaneously or choosing First from the Go menu takes you to the first card in the current stack.

2. Choose Copy Card from the Edit menu.

Now this entire card is on the Clipboard, ready to be pasted anywhere. This is the same

command you use to cut and paste entire cards among stacks.

3. Choose Home from the Go menu or press Command-H.

4. Hold down the Shift key and then choose Paste Picture from the Edit menu.

You could also hold down first the Shift and also the Command keys while you type "V." The miniature picture of your new stack is pasted on the Home card.

Hint

HyperCard knows what's on the Clipboard—text, pictures, a button, a field, or an entire card. When you next choose Paste, HyperCard reminds you what you cut by including the object in the Paste command.

The picture appears in the middle of the card. While it's still selected, you can move it anywhere on the screen, change it from Opaque to Transparent (to let an underlying picture show through), or modify it using any of the commands in the Paint menu. Figure 4.17 shows the picture moved to an empty spot on the Home card.

Images are opaque when they're pasted in from the Clipboard. Since there's nothing under this picture that you want to show through, leave it opaque.

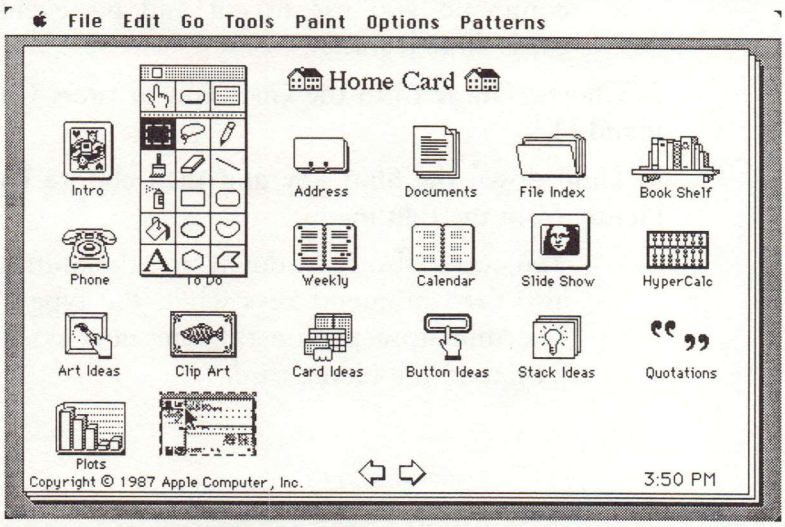


FIGURE 4–17 *Pasting a miniature onto the Home card. Holding down the Shift key while you paste a copied card turns the card picture into a miniature. This is handy for representing one card on another card—the Home card, for example.*

Hint

If you accidentally click anywhere outside the selection, the picture is no longer selected and no longer "floats" around on the screen without affecting an underlying image. If this happens, you can recover by choosing Undo from the Edit menu (to undo the Paste command), holding down the Shift key, and re-pasting the picture.

5. Drag the small picture to an available space on the Home card.

Hint

Use the selection rectangle to check the alignment of your picture with other pictures on the Home card: Drag the selection rectangle horizontally from an existing picture's baseline across to the new picture's baseline to make sure they line up. It's an aesthetic detail, but it can make a big difference in the overall effect of your HyperCard stacks.

Adding a Button that Goes to the New Stack

Now your Home card has a picture that represents your new stack, but you still need a button that takes you there.

1. Click the Button tool in the Tools palette.

When you click the Button tool, HyperCard shows you the existing buttons on the Home card by outlining them.

2. Holding down the Command key, drag diagonally across the picture to create a new button as shown in Figure 4.18.

This creates a button in the same spot you pasted the picture of the new stack. Command-dragging with the Button tool is a shortcut for creating a new button, just as Command-dragging using the Field tool is a shortcut for creating a field. You could also choose New Button from the Objects menu,

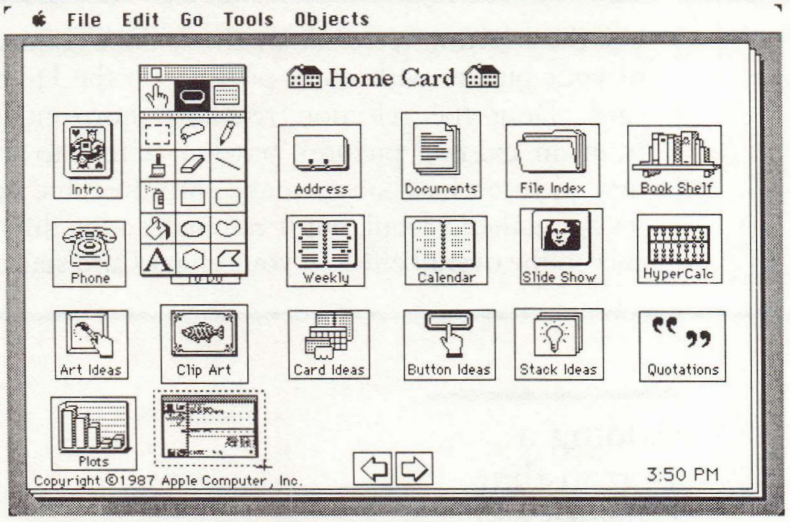


FIGURE 4–18 *Creating a new button by Command-dragging across the miniature picture.*

which tells HyperCard to create a familiar-looking button named New button. When you create a new button by Command-dragging, HyperCard creates a transparent, unnamed button in the size of the rectangle you drag out. You can customize the button by double-clicking it.

3. Double-click the new button.

The button dialog box appears. All you need to set is where to link this button to. Command-dragging the button shape made it transparent, and you don't need to name this button or set it to autohilite when you press it.

4. Click LinkTo.

HyperCard switches you to the Browse tool so that you can easily move to the card you

want to link the button to. And it presents the button destination palette with three choices—This card, This stack, and Cancel. Don't click any of them yet.

5. Go to the first card in your new stack by pressing the tilde key (or whatever key is at the top left corner of your keyboard) or choosing Back from the Go menu.

The destination window remains on the screen, waiting for you to show it the button's destination.

6. Click This Card in the destination window.

HyperCard switches you back to the Button tool and returns you to the Home card. The button you just linked is still selected.

Now you can get to your new stack by using the Browse tool to click its picture on the Home card.

7. Use the Browse tool to click the picture of your client stack.

You can also get to your stack from anywhere in HyperCard by choosing Open Stack from the Stack menu and opening the stack, or by choosing the Message box from the Go menu and typing, "Go to stack My Clients" (the "to" is optional) and pressing the Return or Enter key.

Printing the Bill

HyperCard gives you lots of printing choices. You can print single cards by choosing Print Card from the File menu. You can print entire stacks using the Print Stack command. You can print cards full-size,

half-size, or quarter-size. You can print to fit "minibinder" notebooks that are half the size of standard 8½ by 11-inch paper. You can print reports that include just the text on all the cards in the stack. Or, if you have cards with more than one background in the same stack, you can print just the cards in the current background. You can print only selected fields in the current background or you can print labels on special paper that has labels on it.

Here's how to print your client stack to send a monthly statement to each of your clients:

1. Use the Browse tool to go to your client list if you're not already there.

2. Choose Print Stack from the File menu.

The Print Stack dialog box appears. You can see this dialog box in Figure 4.19.

Hint

Import Paint, Export Paint, and Quit HyperCard are available when you're using the Paint tools, but all other commands in the File menu are dimmed. When you switch to any tool other than a Paint tool, the Import and Export Paint commands disappear from the menu.

If you want to send out 8½ by 11-inch sheets, check Print one card per page of paper. If you want to keep the records in a "minibinder" that uses pages half that size, click Print full size cards and Split-page format. (Split-page format adds space in the gutter so that you can separate the halves of the page to fit into a binder.)

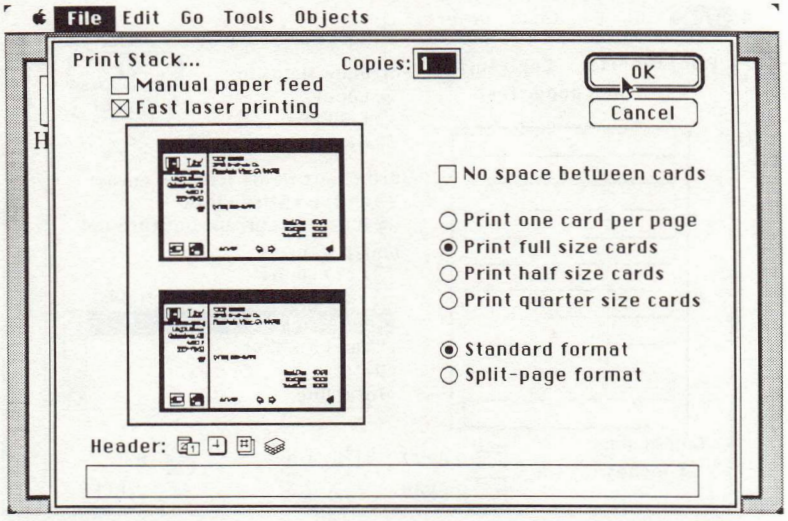


FIGURE 4–19 *The Print Stack dialog box. You can print cards full, half, or quarter size and you can use a format that fits a small notebook.*

Before you choose a printing command, make sure your printer is attached and switched on. Use the Chooser desk accessory to select a printer if you're on a network.

3. Print labels by choosing Print Report from the File menu.

You see the dialog box shown in Figure 4.20. Check the Labels check box and click "Name and Address" to print just that field. Now you see why you put phone numbers in a separate field—so you can print your clients' names and addresses without the phone numbers.

These settings are saved with the stack. The next time you choose Print Report while you're at your client list, this dialog box will be preset to print labels with the Name and Address field. Of course, you can

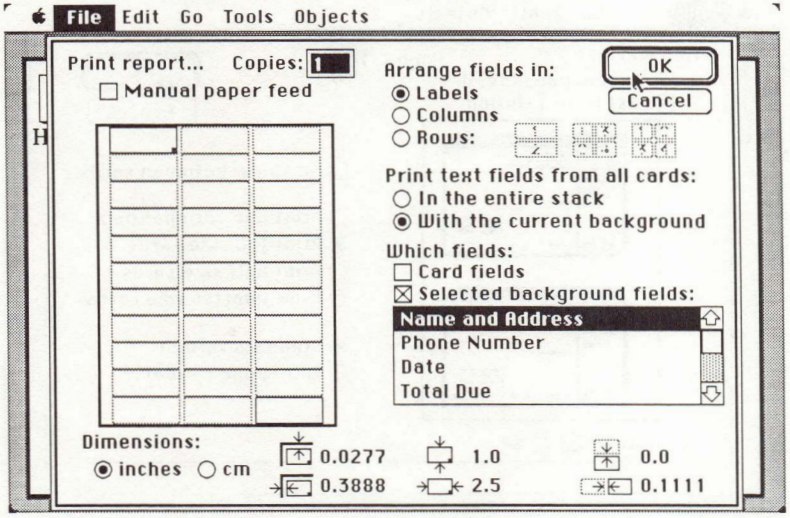


FIGURE 4–20 *The Print Report dialog box. You can print only the text of the current background or entire stack.*

change the setup whenever you want, but when you choose Print Report it will be preset to the way you set it last.

Summary

In this chapter you've seen how to design a stack from the bottom up. You've copied a stack to serve as a blueprint for your client stack, then copied a background picture to further modify the new stack. Getting information about stacks, cards, fields, and buttons is easy, as you have learned to use the Objects menu. With the click of a button, you can sort all the cards in a stack, or view them quickly. You've learned to delete buttons you don't really need, and moved buttons around on the card. You've seen how easy it is to add and link a new stack to the Home card.

You've seen the differences between Paint text and structured text. Finally, you saw how to print your work.

You've seen how much you can do in HyperCard without ever writing a single script. But editing and writing new scripts gives you access to the most power available in HyperCard—the power to control exactly what happens when you click a button or go to a card or take any other action in HyperCard. The next chapter introduces HyperTalk—the language you use to write scripts.

Part Three:
Scripting

Chapter 5

Gaining Power with Scripts

"The first installment of Macintosh applications made power available; the second will make information accessible. HyperCard provides a tool and a standard for this new class of information. Now there are information consumers and information creators, with a gentle pathway leading from one to the other."
Bill Atkinson

```
on mouseUp
  ask "Amount of payment?" with 0
  subtract it from field "Past Due"
  if field "Past Due" < 0 then
    add field "Past Due" to field "Charges"
    put 0 into field "Past Due
  end if
  send "mouseUp" to background button "Total Due"
end mouseUp
```

Like the best of all Macintosh applications, HyperCard gradually unfolds its power as you need it. In the beginning you see just four menus—the Apple, File, and Edit menus and the Go menu, which you use to navigate among cards and stacks.

You start to customize stacks by adding your own text to them. Next, you might check Painting on the User Preferences card in the Home stack and start drawing and copying pictures. After you've been painting for a while, you'll go to the Author level, copying buttons, linking them to new destinations, creating new fields, and copying and modifying existing fields. And when you're ready to modify scripts, you finally check Scripting as a user level. Scripts control objects in HyperCard—buttons, cards, fields, backgrounds, and entire stacks.

HyperTalk Scripts

Whenever you create a button, a field, a card, a background, or a stack, you're creating an object that can have a script. You can think of the script as the object's brain that tells it what to do. The language you use to create and modify these scripts is called HyperTalk. HyperTalk is built into HyperCard. Every object used with HyperTalk—buttons, fields, cards, backgrounds, and stacks—is an integral part of HyperCard itself. This makes the progression from Browsing to Typing to Painting to Authoring and finally to Scripting a very natural one that's as easy as clicking a new check box on the User Preferences card.

By starting with simple commands such as "Go" (to take you to a destination, just as LinkTo does), adding visual and sound effects that can take place along the way, and gradually adding other actions such as playing sounds, sorting stacks, calculating formulas, and much more, you can gradually learn to create stacks that are really small applications themselves.

But you don't have to sit down and study intensively or take a course in programming to get this power.

Using the Message Box

In Chapter 1, you learned how to use the Message box to search for text. You choose Find from the Edit menu. HyperCard displays the Message box and the command Find "|" with an insertion point between the quotes. You type the text you want to find, and then you press the Return key. "Find" is a HyperTalk command that's executed when you press the Return key.

You've also used the Message box for more than finding text. The Message box is a place where you can communicate any message to HyperCard. For example, "Go Home" is a message you type into the Message box to have HyperCard take you to the first card in the Home stack. Figure 5.1 shows an example of using the message box to Go Home.

Asking What Time or Date It Is

Besides using the Message box to tell HyperCard to find some text or to take you somewhere, you can use it to ask HyperCard for information. For example, you can type "the time" and press the Return key to ask HyperCard what time it is.

Or you can ask for the date by typing "the date" and pressing Return. HyperCard answers with the current date (as long as your Alarm clock is set correctly to

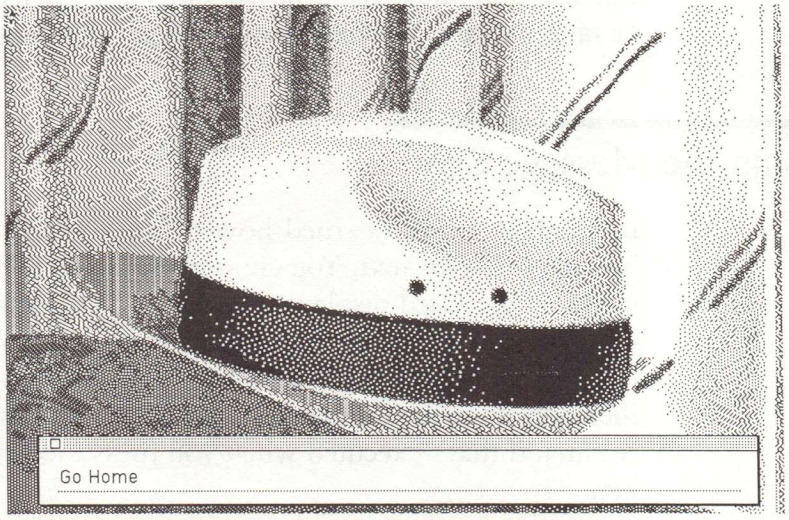


FIGURE 5–1 *Message box with "Go Home" in it. This card has no Home button and it hides the menus. But you can always use the Message box by typing Command-M and "Go Home" or the name of any other stack you want to go to.*

the current date)—month, day of the month, and year. You can get the date in its long form—Friday, June 19th, 1987, for example—by adding the modifier "long" to get "the long date." HyperCard knows about several ways to keep track of and show the date, the long date is one example. Figure 5.2 illustrates the long date in the Message box.

Hint

If there is currently any selection in the Message box, anything you type will replace just the selection. Otherwise, typing replaces any text in the Message box.

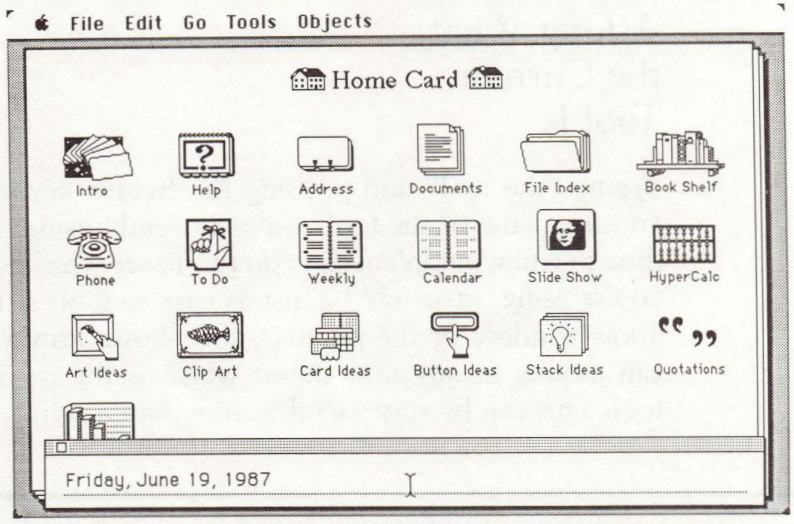


FIGURE 5–2 *You can type "the date" into the Message box and press the Return key to have HyperCard tell you the current date. You can also get "the time."*

Here are some other pieces of information you can ask HyperCard for by typing in the Message box.

Asking Where the Mouse Is

Typing "the mouseLoc" and pressing the Return key tells you exactly where the mouse is on the screen at the moment you press the Return or Enter key. Try moving the mouse to another place on the screen and pressing the Return key again. The numbers that HyperTalk gives you back are very useful in scripts you write that move objects on the screen. You use these numbers to tell HyperCard the position you want the object moved to or clicked at.

Asking What the Current Tool Is

Typing "the tool" and pressing the Return key tells you the name of the tool you're currently using. Although you wouldn't need to do this to see which tool you're using, (it would be just as easy to look at the Tools window or the pointer), this shows how you can inquire at any time about which is the current tool. This can be very useful from within a script.

Hint

When does typing go into the Message box? If there's an insertion point in a field when you open the Message box, HyperCard leaves the insertion point where it is so that what you type goes into the field. If there's no insertion point in a field, HyperCard assumes you want the typing to go into the Message box, replacing anything that was there. This is different from choosing Find, which always places the insertion point in the Message box.

Asking the Name of Objects

These messages are all functions: the date, the time, the tool. Functions return values you can use within scripts. You will learn more about functions later in this chapter. You can also ask HyperTalk about prop-

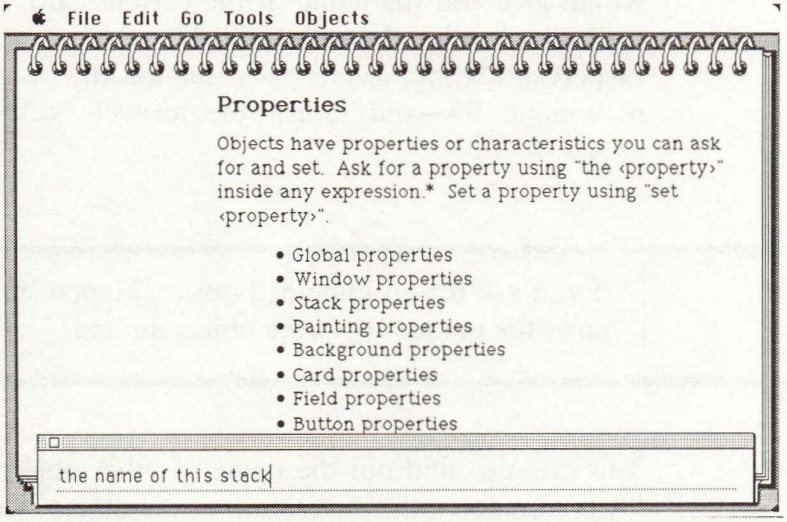


FIGURE 5–3 *Using the Message box to ask the name of an object. Pressing the Return key now will display the name of this stack—Help.*

erties of various objects. Properties are characteristics such as a text's font and size and the name of the object.

Try typing "the name" into the Message box.

Choose Message from the Go menu if the Message box is not visible.

HyperCard returns the answer "HyperCard" because you asked the question in the most general way possible. If you typed "the name of this stack" (as shown in Figure 5.3) you would receive as an answer the name of the current stack. (The answer would be "stack 'Help'" in this example because this is one of the cards in the Help system.) Likewise, if you typed "the name of this background" you would get the name of the current background ("bkgnd 'HyperTalk'" for this card) and "the name of this card"

would give you the name of the current card. (The card shown in the example has no name so HyperCard would use its pet name for any object—its unique ID—and return the answer "card ID 14303".)

Hint

If you ask for an unnamed object, HyperCard returns the unique ID of the object instead.

You can also find out the name of other objects by including more information about the object in your request. Here are some examples that show how you can tell HyperCard which object you're referring to:

Type: the name of field 1
Press Return.
Type: the name of background button 1
Press Return.

This would give you first the name of the first field and second the first background button on the card you're currently looking at. Unless you tell HyperCard otherwise, it assumes you're talking about the current card.

When you want to talk about an object that isn't on the current card, you tell HyperCard where the object is by adding modifiers to the object's description. For example, you might type any of the following lines, pressing the Return key each time to get the answer:

- the name of next card
- the name of previous card
- the name of first background button of previous card

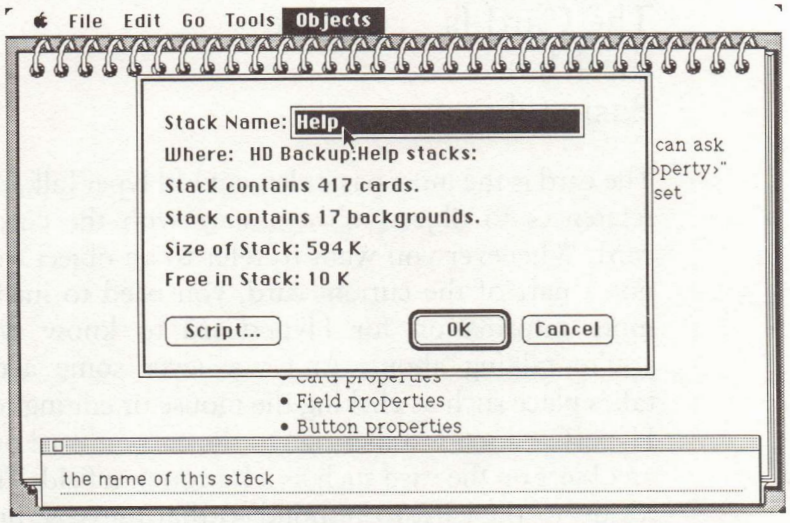


FIGURE 5–4 *There's another way to see a stack's name, plus more information about it. Choosing Stack Info from the Objects menu displays information about the current stack—the Help stack, in this case.*

Hint

You could also get most of this same information by using the Objects menu and choosing the object you want to know about. Each command in the top portion of the Objects menu presents a dialog box that gives you information about the selected button or field or the current card, background, or stack. (You can't use the Objects menu to find out about other objects—the previous card or the next background, for example—without first going to that card.)

Figure 5.4 shows a dialog box that describes the Help stack. This is what you'll see if you choose Stack Info while you're looking at any card within the Help stack.

The Card Is the Most Basic Object

The card is the most basic element of HyperTalk. Any references to objects always start with the current card. Whenever you want to refer to an object that's not a part of the current card, you need to include more information for HyperCard to know what you're talking about. And whenever some action takes place such as clicking the mouse or editing text, HyperCard sends a message to the current card or to an object on the card such as a button or a field. Then it's up to the card to respond to that message or to pass the message on somewhere else. This brings up the subject of inheritance, which is an important HyperCard topic. Before we get into inheritance as a way of passing messages within the HyperCard world, let's look at some other work you can do in the Message box.

Calculating

Although HyperCard isn't a full-fledged spreadsheet, you can use it for calculating formulas. For example, the HyperCalc stack on the Home card is a great example of a simple application that's very powerful at calculating formulas. You'll see the HyperCalc script in Chapter 8, "Hooked on Scripts." Right now, take a look at how you can use the Message box for simple calculation.

Type: 3 + 6 and press Return.

The result of the calculation appears in the Message box.

Hint	Pressing Return or Enter tells HyperCard to perform whatever action is in the Message box—either the Find command that is automatically inserted when you choose Find from the Edit menu or any other command or function (such as the date or the time) you type in or the result of a calculation you ask it to perform.

You can type in any calculation you want HyperCard to perform. For example:

- 7*2
- 4/3
- .0006 * 57
- 25483625-182635

HyperCard gives you an answer each time you press Return.

Below are some other examples of messages you can send to HyperCard by typing them into the Message box:

- DoMenu "New Card"
- Put "Hello" into Message box
- The number of cards (this tells you the total number in this stack)
- The number of this card (this tells you the sequential number of the current card)
- The number of fields (this tells you the total number of fields on this card)
- Choose Button tool

Hint

HyperTalk commands are separate from the commands you see in menus. In fact, all of the menu commands can be carried out by just one HyperTalk command—DoMenu—followed by the command name, such as "New Card" above. Within HyperTalk, the word "command" refers to the HyperTalk commands rather than to the commands you see in menus. The Find command straddles both worlds. It's a HyperTalk command that also appears in a menu.

Training Buttons to Issue Commands for You

You could use the Message box to type a HyperTalk command whenever you wanted HyperCard to do something or you wanted some information, but HyperCard also provides a way to string together more than one command and store them in a "script" of commands.

Any object—a button, a field, a card, a background, or a stack—can have a script. Let's look at one of these scripts.

You've seen how you could have HyperCard take you Home by typing "Go Home" in the Message box and pressing the Return key. But you can teach a button to do the same thing by putting the command in the button's script. Putting that command into a button's script tells HyperCard to carry out that com-

mand whenever you click that button—without your having to type the command into the Message box each time you want it to happen.

1. Go to your Client list by typing in the Message box: Go to my clients.

When HyperCard sees the word "Go," it checks for the keyword "stack" or "card" following it. If it finds neither of those keywords, it assumes that the next word or words are the stack name—"My clients" in this case.

Hint

The word "to" is optional in "Go to my clients." Also, you don't usually need to put quotation marks around the name of a stack.

Sometimes there can be ambiguity about how HyperCard should interpret information. For example, if you said "go card ideas" HyperCard would look within the current stack for a card named "ideas." It has no way of knowing that "Card Ideas" is the name of a stack unless you tell it so by enclosing the name in quotes. But in general you don't need to include quotes around stack names. You don't have to match uppercase and lowercase characters, either. For the most part, case doesn't matter to HyperCard.

2. Press the Return key.

HyperCard takes you to the first card in your client stack.

Hint

If HyperCard can't find the stack when looking at the disk level or in the HyperCard Stacks folder or any other folder named, it presents the standard file dialog box. When you open a stack from this dialog box, HyperCard adds its pathname to the list of places to look for stacks. The next time you ask for the stack, HyperCard will know where to look and won't ask you again unless you move the stack somewhere else.

3. Choose the Button tool.

Tear off the Tools window and click on the Button tool. Notice that all the buttons on the card are outlined. (Close the Message box if necessary.)

4. Select the Home button by clicking it.

If you were using the Browse tool, clicking the Home button would take you Home. But when you're using the Button tool, clicking the button selects it so that you can cut or copy it, move it, or find out more information about it.

5. Choose Button Info from the Objects menu.

Figure 5.5 shows you the dialog box that appears, with lots of information about this button.

Hint

Double-clicking a button using the Button tool is a shortcut for seeing its Button Info dialog box.

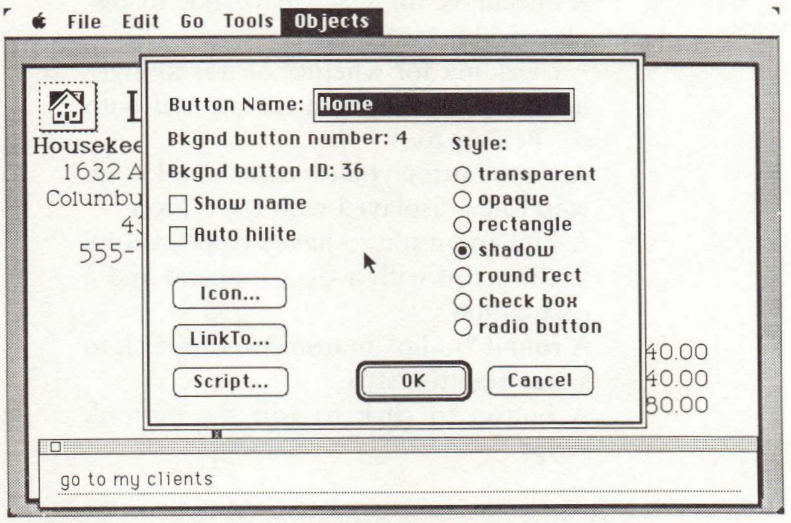


FIGURE 5–5 *You can get information about any button by selecting the button with the Button tool and choosing Button Info from the Objects menu. This is Button Info for the Home button in the My Clients stack.*

The Button Dialog Box

You've seen this dialog box before, in Chapters 2 and 3. There are places for:

- The button's name
- Its number within the current card or background (the dialog box tells you whether this is a card or a background button)
- Its unique ID number within the stack (arbitrarily assigned at creation to make it impossible to confuse this button with any other)

- A checkbox for whether or not to display the button's name
- A checkbox for whether or not to highlight the button when it's pressed with the Browse tool
- An Icon button you can click to select an icon to be displayed with the button
- A LinkTo button to have HyperCard fill in the script with a Go command and a destination
- A round "radio" button you can click to select a button style
- A button to click to edit the button's script

Hint	The Script button is dimmed unless you've checked "Scripting" on the User Preferences card in the Home stack. Get to the User Preferences card quickly by going Home and pressing the left arrow key. (It's the last card in the Home stack.) Click "Scripting" to be able to edit scripts. Click "Blind typing" to be able to type commands without first displaying the Message box.

Clicking the icon button presents a scrolling list of available icons. Any icon currently being used for the selected button is highlighted; clicking any other icon in the list selects and highlights it. Icons are available if their resource is in the current stack or in HyperCard itself. You can use a resource editor such as ResEdit to add or edit resources for icons. See Ap-

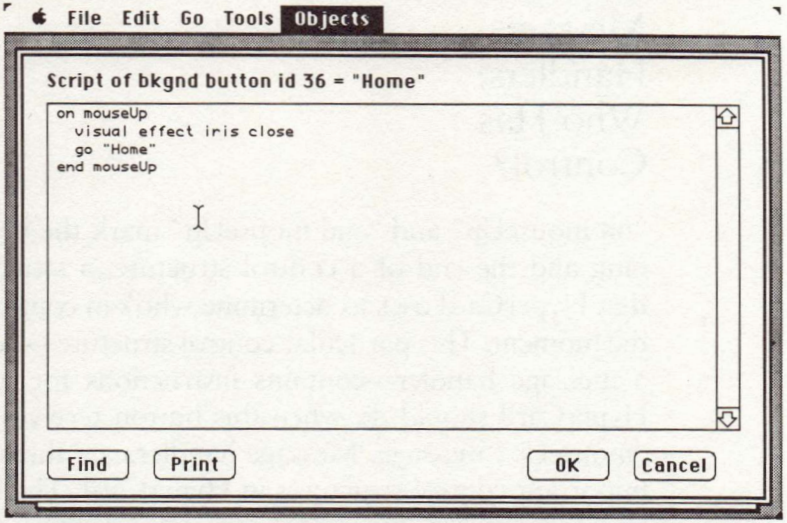


FIGURE 5–6 *The script for the Home button. You can see and edit scripts whenever your user level is set to Scripting on the Preferences card in the Home stack.*

pendix 3 for how to use Andy Hertzfeld's mini-servant program to work with HyperCard resources.

6. Click Script to see the Home button's script.

The Home button's script (shown in Figure 5.6) says:

```
on mouseUp
  visual effect iris close
  go "Home"
end mouseUp
```

There's an insertion point at the beginning of the script, where you could add more actions to the script.

Message Handlers: Who Has Control?

"on mouseUp" and "end mouseUp" mark the beginning and the end of a control structure, a structure that HyperCard uses to determine who's in control at the moment. This particular control structure—called a message handler—contains instructions for what HyperCard should do when this button receives the "mouseUp" message. Message handlers are the most important control structures in HyperCard. The button receives the mouseUp message whenever the mouse button is released while the pointer is over this button. In other words, whenever you click this button, here's what HyperCard should do.

Message handlers divide the script up into manageable chunks—one chunk for each how-to-handle message. Within the message handler are the commands you want HyperCard to carry out when the current object receives the message.

Between the "on mouseUp" and "end mouseUp" lines are the lines:

```
visual effect iris close
Go Home
```

The first line is a message—actually a command called "visual effect"—to HyperCard telling it to use one of the available visual effects as it takes you to the Home card. The visual effect command is really the one-word command "visual." "Effect" is a helper word that makes the meaning of the "visual" command clearer.

Hint

A message handler is a particular kind of control structure that tells HyperCard what to do when a certain message is received by the current object.

Hint

The unusual capitalization is just a convention in HyperTalk, not a rule. The uppercase "U" separates the word "mouseUp" without using any blank spaces. (Certain elements in HyperCard such as commands need to be one word for HyperTalk to be able to interpret them correctly.)

Hint

There are many visual effects you can use with the Go command. (See Appendix 2 for a list of the visual effects you can choose.) "iris close" zooms in to the center of the screen.

The second line of the script contains another message. It's the same message you might have typed into the Message box—Go Home.

Instead of being performed when you press the Return key (as it would be if you typed it into the Message box), this entire script is performed when you click the mouse button with the pointer positioned over the button. The button receives the message that the mouse button has been clicked, and this button is "trained" to use the visual effect of irising closed and to take you Home whenever it receives the mouseUp message.

7. Close the button's script by clicking the Cancel button.

Hint	Get into the habit of clicking Cancel rather than OK when you don't want to record changes to a script. Clicking OK can sometimes OK changes you may not be aware of—a character you may have typed, for example.

Naming Objects

HyperCard uses names so that you can specify objects for commands to operate on. You can name stacks, cards, or containers to place things in. A container is anything that can hold a value. One example of a container is the Message box. For example, "Put hello into Message box" specifies the container (the Message box) you want to put something ("hello") into.

Fields are another type of container. For example, you could type:

```
Put "hello" into field 1
```

In this case "hello" would be placed in field 1 whether or not the fields were visible. Until you take that value out of the field by deleting the word or replacing it with something else or by deleting the field itself, the field contains "hello."

Naming Hierarchies

You've seen how you can get the name of an object from HyperCard. Every object has a naming hierarchy that starts from its smallest subdivision and can be expanded to include increasing amounts of information. It's like your mailing address, with detailed parts and broad parts. If you live in the USA and are sending a letter to someone who also lives in the USA, you don't need to include "USA" in the address. In the address of someone who lives in another state, you do need to include that state's name to show that it's not the same state you are in. The address can get increasingly detailed—down to the apartment number if necessary—to differentiate this address from any other. You just include as much detail as you need to make the differentiation.

In HyperTalk, apartment numbers, street addresses, cities, zip codes, and countries look like:

```
character 1 of word 2 of line 5 of field 3 of card BestFriend of stack
My Stacks: Address
```

But if you were already in the Address stack in the folder named "MyStacks" you could just refer to:

```
character 1 of word 2 of line 5 of field 3 of
card 4
```

or

```
background button 6 of card 2
```

And if you were on card 12 of stack Help you could refer to:

```
background button 6
```

Whenever you refer to any named object—a button,

a field or part of a field, a card, a background, or a stack—include as much of the hierarchy of names as you need but no more. If you say to go to a stack, HyperCard will take you to the first card of that stack. If you want to go to another card in that stack, you can specify the second (or third, or last, etc.) card of the stack. For example, you could say, "Go to third card of stack "MyClients."

You don't have to use a name to designate an object. You can use its place within the card or the background or the stack to designate it. For example, you can designate card 1, first card, next card, or background button 3. These references aren't to a specific card but to any card or button that fits that description at the time the script is performed. You can also refer to cards by the unique ID they're assigned when they're created, or by any name you give them.

Hint

In general, weigh the extra work of naming objects against the amount of clarity it provides in reading scripts that use the name. Sometimes it's not worth the extra time it takes to name an object; its ID or numbered position are all you need. But sometimes using a name makes a script that refers to the object much clearer.

Absolute vs. Logical References

Look at the LinkTo Destination window shown in Figure 5.7 to see examples of using an absolute ref-

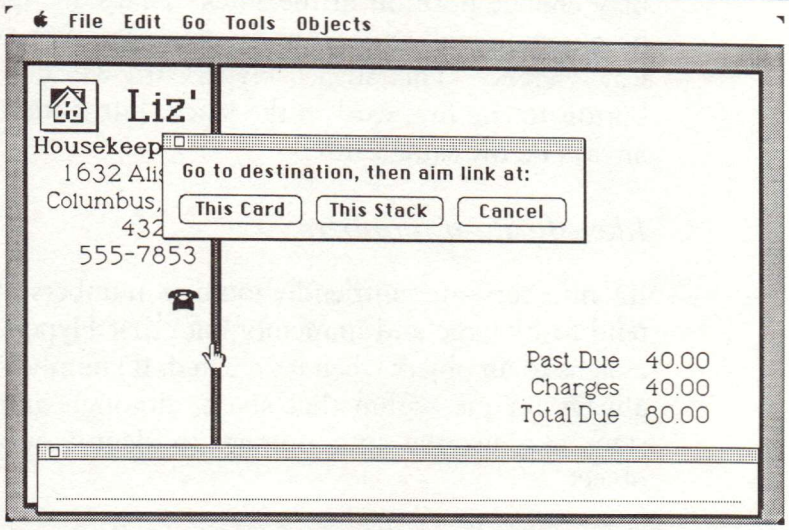


FIGURE 5–7 *The Destination window appears whenever you click Link To on a Button Info dialog box. While the window is visible you use the Browse tool to go to the destination you want for the current button. You can link the button to either a specific card—the card you're at when you click This Card—or the first card in the stack you're at when you click "This Stack."*

erence to a specific card and a logical reference. The choices in this window link you to either a specific card (absolute reference) or to a card determined by the current state of a stack (logical reference). You can see this Destination window by selecting any button using the Button tool, choosing Button Info from the Objects menu, and clicking LinkTo.

When you link a button to a specific card (This Card), you want HyperCard to take you directly to that card, regardless of where it is located in the stack. Therefore, your link destination is an absolute (unchanging) reference: the card ID number or unique name. Alternatively, if you want to link the button to the first card in a stack, you know that any given card

may change position in the stack. That's all right—because you really just want the stack, you use a logical reference (This Stack). HyperCard will link the button to the first card in the stack, but it may not always be the same card.

Identification numbers

ID numbers are unfriendly-looking numbers (they tend to be large and unmemorable) that HyperCard assigns to an object when it's created. ID numbers are always unique within that stack, although another stack may use the same number to identify another object.

IDs are absolute references. Absolute references have some of the same advantages post-office boxes have. Cards, like people, tend to move around. So "last" card may identify a certain card today, but by tomorrow there may be another "last" card. Using a card ID (or any other unique name) is like having a P.O. box instead of a street address. Even if you move, your mail will get to you. The ID number will always locate a particular card or button or field, regardless of its changing location within the stack.

When you copy a card or other object using the Copy command, it receives a new ID. But when you copy a stack in the Finder or with the Save a Copy command in the File menu, the objects in the stack that were copied are identical to the originals, including their ID numbers.

Sure IDs are ugly, hard-to-remember numbers, but you don't have to remember them or type them (LinkTo does it for you), and they ensure absolute identification of an object, no matter whether you

move it or change its name or position within the card or background or stack. If you find yourself typing in ID numbers, you're probably doing something wrong. Name the card or other object and refer to it by name instead of referring to it by its ID.

Hint

Carefully use the words "char," "word," "line," "field," "card," "background," and "stack" to avoid confusion. For example, never name a card "button 3." "Go to card button 3" would confuse both you and HyperCard.

Looking at a Script HyperCard Wrote for You

You've already seen how to link a button to a card: select a button, choose Button Info (or double-click the button), click LinkTo, go to the card you want to link the button to, and then click "This card."

When you click LinkTo and then designate a destination card, HyperCard automatically adds a line to the button's script that specifies the card by ID number. You don't have to know the ID number or write the script; HyperCard does it for you.

This is what happened when you used linkTo to link a button to your client list. Look at that button again.

1. Go Home by pressing Command-H.

Or just click the Home button.

2. Use the Button tool to double-click the button that goes to your client list.

Double-clicking takes you to the dialog box for that button.

3. When you see the button dialog box, click Script.

Figure 5.8 shows the script for that button. There's the "on mouseUp" and "end mouseUp" control structure. Between these two lines, HyperCard has inserted what to do when you click this button—Go to card ID 3342 of stack "My Clients."

4. Click cancel to close the script.

Hint

Whenever you're referring to a specific card, you have choices for ways to refer to it. If the card has a name, you can use that, or you can refer to it by location in the stack—last card, card 1, etc. When HyperCard fills in the script for you, it will still refer to the card by its unique ID number.

Hint

In this case you could also click the OK button. The OK button accepts the script as it is, including any changes you've made to it. The Cancel button ignores any changes.

When you use LinkTo to aim a button, HyperCard automatically writes that button's script to direct it to the place you choose. But you yourself can edit any button's script directly by using the Button tool to

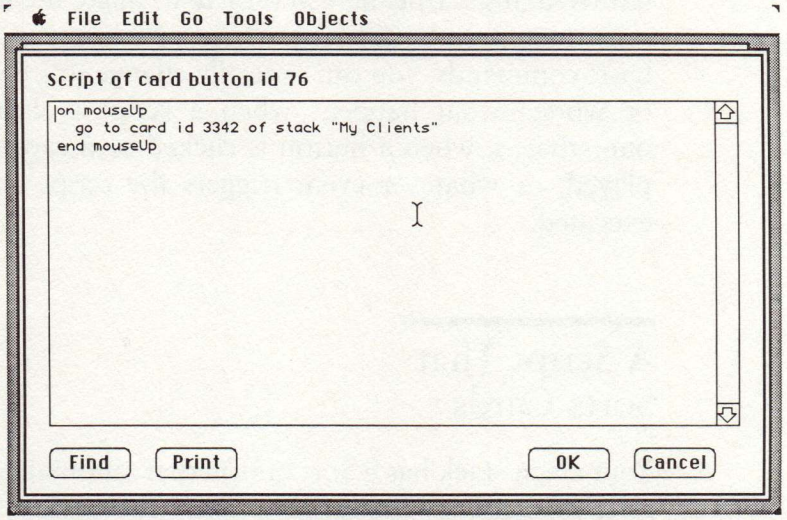


FIGURE 5–8 *The script HyperCard writes when you've linked a button to a specific card using LinkTo. The ID is a unique number each card has within its stack.*

open it and edit its text yourself. With LinkTo, HyperCard is saving you time in typing commands into the script, and it's making it possible for you to be a HyperCard author without being a HyperCard programmer. For example, you can put together a working stack without writing a single script—just by copying cards, pictures, fields, and buttons and by using LinkTo. But sooner or later, you're going to want to write a script of your own.

Modifying Existing Scripts

Things rarely are static in HyperCard. It's so easy to change the way something works that you find your-

self tweaking scripts here and there to make them do something a little different. Once you know a few basic commands, you can radically change (for better or worse) what happens when a script is carried out—that is, when a button is clicked or a card displayed, or whatever event triggers the script being executed.

A Script That Sorts Cards

Your client stack has a Sort button that automatically sorts the cards by first or last name. You copied this Sort button from the original Address stack in an exercise in Chapter 4. Let's look at the Sort script.

1. Go to the Address stack by choosing Message from the Edit menu and typing "Go Address" and pressing the Return key.

Or get there by clicking the Address card picture on the Home card or using the Open Stack command.

The Sort button is the one that shows a hand repositioning cards in the stack. Clicking the Sort button on this card presents a dialog box that lets you choose whether to sort the cards by first or last name. You can see this dialog box in Figure 5.9.

2. Use the Button tool to double-click the Sort button.

Clicking the button with the Browse tool resorts the cards, but clicking it with the Button tool lets you get information about the button, change its name or style, or edit its script.

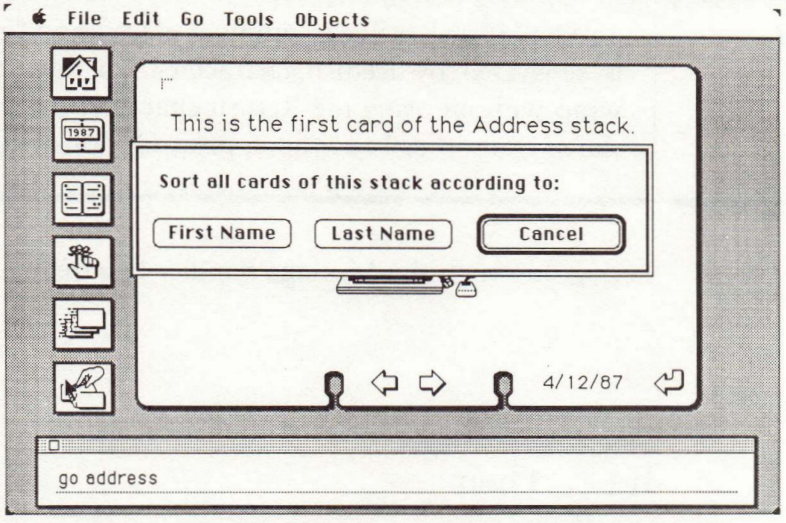


FIGURE 5–9 *The Sort button in the Address stack presents this dialog box. You can edit the script to have the dialog box present other choices.*

Hint

Looking at existing scripts is a good way to learn the structure and logic of HyperCard scripts.

3. Click Script to see the script for this button.

The script says:

```
on MouseUp
  answer "Sort all cards of this stack according to:" ¬
  with "First Name" or "Last Name" or "Cancel"
  if it is "First Name"
  then sort by first word of first line of field "Name and Address"
  else if it is "Last Name"
  then sort by last word of first line of field "Name and Address"
end mouseUp
```

Hint

To HyperCard, a line is simply a piece of text that is separated by Return characters. To have text wrap without using the Return character (without starting an official new line), press Option-Return.

Let's go through the Message handler line by line.

If . . . Then . . . Else Control Structure

This script has a new command—answer—and a new control structure—the if-then-else control structure—besides the "on mouseUp/end mouseUp" message handler you've seen before.

The Answer command will present the user with a dialog box with three possible answers: First Name, Last Name, or Date. You specify what should appear in the dialog box by enclosing the text you want in quotation marks. You use the word "with" to precede what you want to appear in the buttons below the specified text. In this example, the three buttons contain "First Name," "Last Name," and "Cancel." The word "or" separates the possibilities.

If the user clicks the first choice (if it is "First Name") HyperCard passes the control to the line:

```
then sort by first word of first line of
field "Name and Address"
```

Notice that the script specifies "first word of first line of field 'Name and Address.'" It goes into no more detail than word because that's the smallest unit it cares about, and it goes no larger than the named field because it assumes that the field is on the current card. It could have included "character 3 of first word" if you wanted to sort by that character. "Name and Address" is the name of the field. You could see this by using the Field tool to select the Name and Address field and choosing Button Info from the Objects menu.

The "if-then-else" control structure performs one of several actions depending on the answer to the question in the dialog box. If the user clicks the "First Name" choice in the dialog box, HyperCard sorts by the first word of the first line of the Name and Address field.

Similarly, if the user clicks the "Last Name" choice in the dialog box, HyperCard sorts by the last word of the first line of field "Name and Address."

Note that this script assumes you've ordered names on the card first name first and last name last. If you use a different order—last name first, for example—exchange the two "then" statements.

If the user presses the Cancel button, the dialog box goes away and the action is cancelled, because neither of the "if" statements apply. We could have caught this possibility and done something different with it by including an if statement for the Cancel button such as:

```
if it is Cancel then show card field 3
```

or any other action we want to have happen when Cancel is clicked.

Editing the Script to Sort by Date

Maybe sorting your clients by first or last name isn't enough for you. Let's say you always enter your clients first name first but you always want them sorted by last name to make a printed list alphabetized by last name. You don't really need the first name choice when you click the Sort button but you would like the ability to sort the list by the last date they were modified. You're already keeping track of the date that card was last modified in the Date field, which is updated whenever you modify any information on the card. (You can see this script by clicking Cancel to close the Sort button's script, choosing Background Info from the Objects menu, and clicking Script.) The script is in the background so that every card that shares this background (every one of your client cards) will have the date field updated whenever any field in the card is changed.

You can also use this date field to sort your cards by date. Here's how:

1. Use the button tool to double-click the Sort button in your client list if you're not already there.

First go to your client list if necessary.

2. Click Script to open the script if you're not already looking at it.

Figure 5.10 shows this button's script with the words "Last Name" and surrounding quotation marks selected. Figure 5.11 shows the script the way it will look when you're finished editing it.

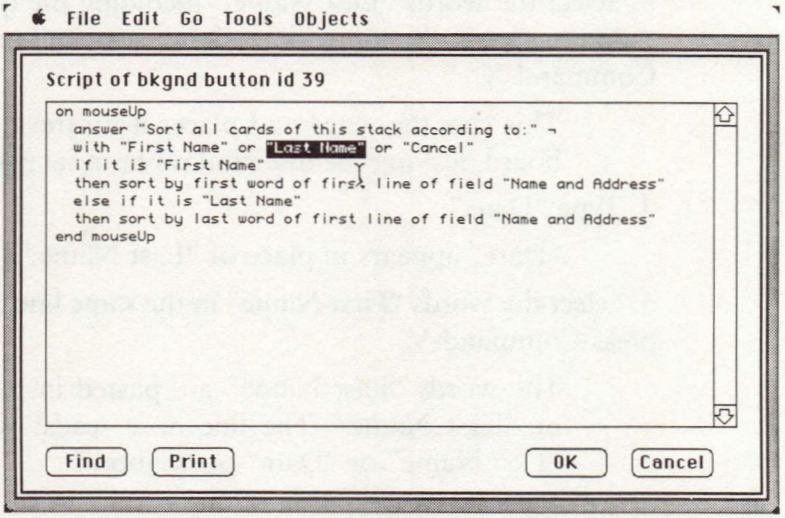


FIGURE 5–10 *The script for the Sort button. You can edit scripts by using the standard editing techniques, including cutting and pasting.*

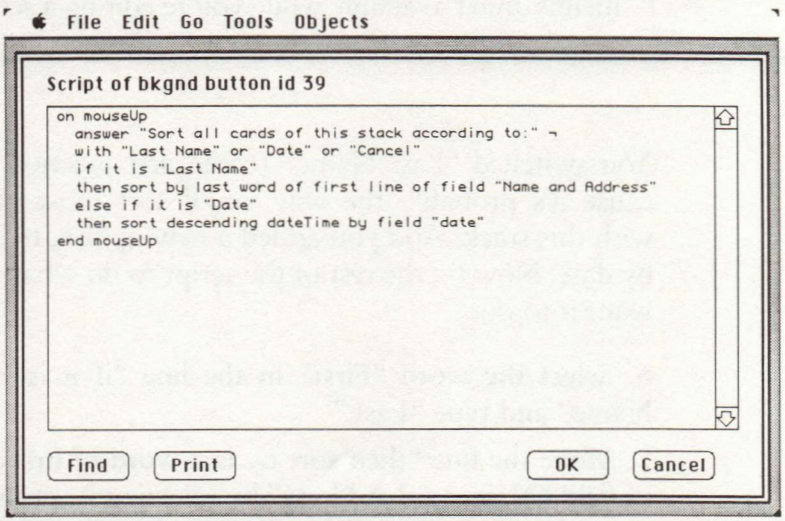


FIGURE 5–11 *The edited script for the Sort button*

3. Select the words "Last Name," including the quotation marks, in the second line of the script and press Command-X.

This cuts the word and places it on the Clipboard, leaving the insertion point in its place.

4. Type "Date."

"Date" appears in place of "Last Name."

5. Select the words "First Name" in the same line and press Command-V.

The words "Last Name" are pasted in place of "First Name." The line now reads: with "Last Name" or "Date" or "Cancel."

Hint

You cut and paste in and among scripts using the Command-key equivalents rather than choosing commands from the Edit menu. The HyperCard menus aren't available while you're editing a script.

You switched "Last Name" to the first position because it's probably the way you'll sort most often with this stack. And you added a new option, to sort by date. Now fix the rest of the script to do what you want it to do.

6. Select the word "First" in the line "if it is First Name" and type "Last."

7. Move the line "then sort by last word of first line of field "Name and Address" by selecting it, pressing Command-X to cut, clicking before the "then" in the line "then sort by last word of first line of field 'Name and Address,'" and pressing Command-V to paste.

Hint

If you accidently formed two lines together, separate them by pressing the Return key to move the insertion point to the next line.

8. Remove the next line—"then sort by first word of first line of field 'Name and Address'"—by selecting it and pressing the Backspace key.

You don't need this line anymore because you've removed the choice of sorting by first name.

9. Backspace again until there aren't any empty lines.

Empty lines don't hurt anything, but your scripts look neater without them.

10. Select "Last Name" in the line "else if it is 'Last Name'" and type "Date."

Now you need to tell HyperCard what to do if "it" equals "Date." "It" will have the value "Date" if the user clicks the Date button in the dialog box. HyperCard places either "Last Name" or "Date" into "it," depending on which button you click.

11. Click in the next line and type: then sort descending DateTime by field "Date"

HyperCard will sort the cards in descending order by their modification date so that the most recently modified cards will come first. To make the most recently modified cards come last in the stack you can use the word "ascending" to reverse the order.

DateTime is another modifier for the Sort command. It specifies the format used by the date field. Using the regular numerical order would cause the number

1/12/87 to come before the number 1/7/87 because it comes first "alphabetically." There are other formats for sorting as well. See the Sort command description in Appendix 2.

Hint

Pressing the Tab key and then looking at the last line of a message handler can tell you whether you've made a mistake, such as leaving out an "end" line. Each message handler should be balanced by an "on . . ." statement and an "end" statement. HyperCard tries to balance the script for you when you press the Tab key. If it can't, you'll notice that the last line of the handler doesn't line up at the left margin the way the first line does. This is your clue to go back and check for missing lines or typos.

---

12. Click OK or press the Enter key.

You want to save this modified script, so click OK instead of Cancel.

Hint

Pressing the Enter key is a shortcut for clicking OK when you're editing a script. It saves any changes you've made to the script.

---

13. Use the Browse tool to click the Sort button in your Client stack.

The dialog box that appears has the new choices—Last Name, Date, and Cancel, as shown in Figure 5.12.

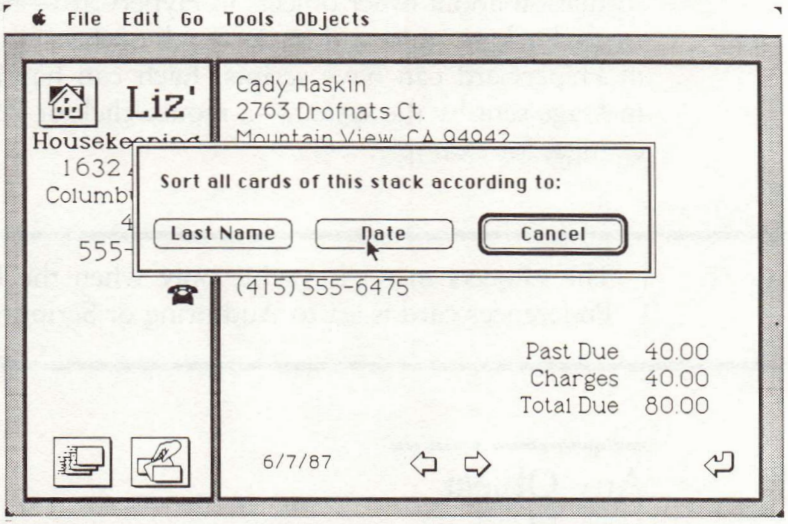


FIGURE 5–12 *The new dialog box for the Sort button. Choices for sorting now included by Date.*

Hint

Be sure to switch back to the Browse tool instead of the Button tool before you click the Sort button. Otherwise, you'll see the Button Info dialog box if you double-click, or a selected button if you single-click the button.

14. Click Date.

Your cards are sorted by modification date—most recently changed cards first.

The Objects Menu

You've already used the Objects menu to get information about buttons. The Objects menu also has in-

formation about other objects in HyperCard—fields, cards, backgrounds, and stacks. Each of these objects in HyperCard can have scripts. Each can handle a message sent by the system—a mouse click or a field change, for example.

Hint

The Objects menu is visible only when the User Preferences card is set to Authoring or Scripting.

Any Object Can Have a Script

Each object in the Objects menu can have a script that responds to various conditions, just as the button's script responded to the mouse click. Scripts can be triggered by any one of many messages the system sends in response to events. The script that updates the Date field in the Client stack is triggered by any modification to any field in the current background.

This script lives in the Background shared by all the cards in your Client list. You copied the Background from another Address stack, so the script in that Background was copied as well.

Where to Put a Script

In the case of the Sort button, you don't want anything to happen unless you click the Sort button using the Browse tool. If you click anywhere else on the

card—to change some text or to go Home or dial a phone number—you don't want the script that sorts to be performed.

Placing the message handler in the script of the Sort button confines it to events associated only with that button—clicking the mouse button or entering or deleting the button, for example. Similarly, placing the message handler in the script for a field confines it to events associated only with that field. For example, the closeField message is sent to a field when text within the field has changed.

To have more objects respond to the same message, you could duplicate the script for every object—button or field—that you want to respond to the message. Or you could put in the handler at a broader level—the card, the background, or the stack. Obviously, it's simpler to place the message handler in once, rather than many times. You need to think about whether this handler will work for every object that will inherit it.

If you put the Sort script in at the card level, the cards would sort whenever you clicked anywhere on the card. If it were in the background's script, it would sort with a click on any card that shares this background. If the message handler were at the stack level, any click anywhere in the stack would re-sort the cards.

More generality is sometimes what you're looking for. If a friend's phone number changes and you update it in your Address stack, you'd like HyperCard to update the Date field. You could put a message handler into the phone number field that would handle the mouse click by updating the date field. But you'd also like this to happen if you change the name

Hint	Put the Message Handler at the level in the object hierarchy—button, field, card, background, stack—that will be caught by all of the objects—but no more—that you want to respond to the message. If you want to respond when a button is clicked, put the Message Handler in the button's script. If you want to respond when something happens anywhere on the card, put the Message Handler in the card's script. If an action anywhere on any card that shares the background should elicit the response, put the script in the background, and if anywhere in the stack put the script in the stack.

and address field. You could duplicate the message handler in the second field, but by putting it in the background you can update the card whenever there's any change within *any* field that shares the current background. You're getting double your money for one script.

Before a closeField message goes to the background, the card has a chance at handling it. But putting the message handler in the card would limit updating to the single card that has the message handler. Instead, the script belongs to the background:

1. Choose Bkgnd Info from the Objects menu.

Go to your Clients stack if you're not already there.

2. Click Script to see this background's script.

You may have looked at this script earlier in the chapter. The background has the script:

```
on CloseField
  put the date into field "date"
end closeField
```

There's also another handler in this script that puts the insertion point in the first field whenever you make a new card. Ignore this handler for now.

When any field is closed (changes are made to it), put (put is the command—it moves information from one place to another) the date (just as if you had typed this into the Message box) into field "date." The result: the current date is displayed in the Date field.

3. Click Cancel to close the script.

You didn't change anything, so there's nothing to OK.

4. Change any information on a card by selecting text and typing to replace it.

5. Click outside the field and watch the date on the card change to the current date.

Because you changed information on a field within this background, the closeField message was sent first to the field, then to the card, and finally to the background, which had a message handler to handle it. The current date was placed in the Date field.

Messages the System Sends

You've seen the "mouseUp" message that's sent when you click with the mouse. HyperCard also sends other messages when events happen. For example, the "closeField" message is sent whenever a field's contents change—when you update a friend's ad-

dress, for example. (Strictly speaking, the message is sent when you click outside the field and the information in the field is different from what was there when you clicked inside the field.)

Just as the button received the message that a click had taken place and had the chance to respond with a message handler, so does the field you edited have the chance to respond to the message that the field has changed.

If the fields have no message handlers for this message (none do in this background), the message is handed off to the next larger object—the card—which in this case also gives up its chance to respond. But the Background does have a response:

```
Put the date into field "Date"
```

"The date" is a function that HyperCard can always get by reading the System clock and finding out what time it is, just as the Alarm Clock desk accessory does. It puts that current date into the field named "Date" and your friend's address card has a new modification date.

Hint

"Date" is in quotes because names of things that you (rather than HyperCard) assign are often enclosed in quotes as a safeguard against their ever being confused with something HyperCard might call by the same name. A word like "date" is especially suspect because "the date" is a function. Also remember, you don't have to use uppercase letters except as a convention for indicating separation in compound words.

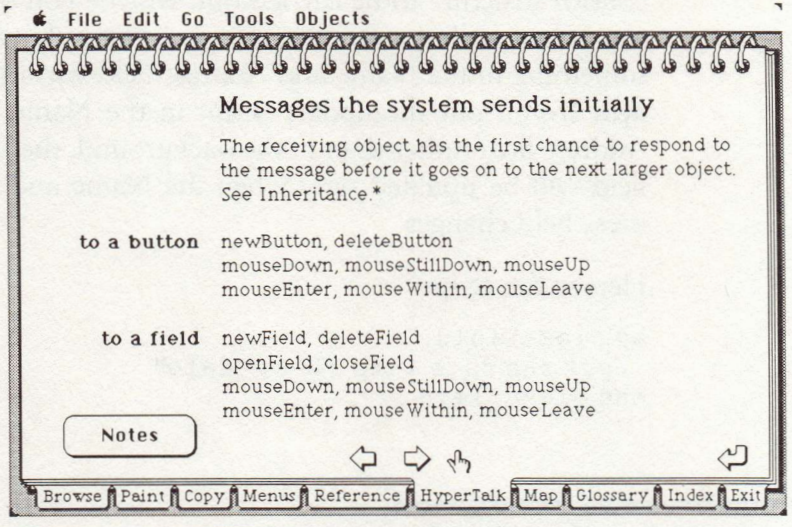


FIGURE 5–13 *Some of the messages the system sends initially. The receiving object has the first chance to respond to the message with a message handler such as "on mouseUp."*

Besides mouseUp and closeField, HyperCard sends many other messages in response to events that happen. See Chapter 8 for a complete list of the messages HyperCard sends, the objects that receive them, and the events that trigger the message. Figure 5.13 shows the messages that are sent to buttons and fields.

Putting a Message Handler Lower in the Hierarchy

Sometimes you want to limit rather than expand a response so that a smaller set of actions will cause the

control structure to be carried out. Maybe you don't care when a phone number changed, but only when something in the Name and Address field was modified. If you put the update script in the Name and Address field rather than in the background, the Date field will be updated only when the Name and Address field changes.

Here is the script:

```
on closeField
  put the date into field "date"
end closeField
```

Going to Another Stack Within a Script

HyperCard never leaves the current stack unless you tell it to. To get to cards in other stacks you use the Go command. (You can also use the Pop card command to go to another card. See Chapter 8 for how to use the Pop card command to return to cards you've put in a pile to remember using the Push card command.) So if you wanted a button to sort a field that wasn't in the current stack you'd say:

```
Go stack Address   --or any stack name, or go to card 3 of
                   --stack Address Clients
sort by first word of first line of field "Name and Address"
```

Summary

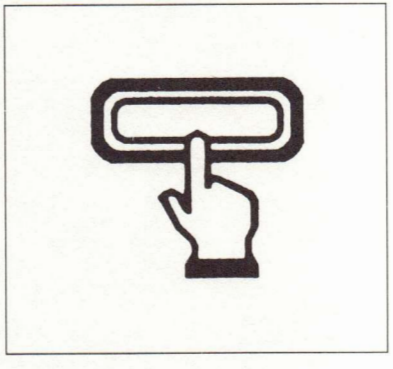
You can do a lot with the commands Go and Sort and with the message handler on mouseUp and end

mouseUp, especially combined with the If . . . then . . . else control structure. The next chapter tells you how to use some other commands to add some more power and flexibility to your client stack.

Chapter 6

"I think of the library as the research laboratory of the humanities."
Charles Kerns,
Stanford University

Expanding a Stack's Capabilities



As you begin to use the stacks that come with HyperCard and stacks you build yourself, you'll probably want to expand some of the stacks' capabilities. For example, if you create a stack to keep track of your clients and how much they owe you (as shown in Chapter 4), you'll

now have a customized Address stack you can also use as a bill to send to your clients. But as your business expands, and as you start becoming more aware of the power available in scripts you now know how to copy, edit, and write from scratch, you might want to use more of this power to give your client list stack more functionality. For example, you might decide to add a button that adds the current charges to the amount past due, and then places the total in the Total Due field.

With HyperCard it doesn't matter that you didn't think of everything you might want to do with this stack on the day you designed it. It's never too late to add new features to a stack or to modify features that are already there. So even though you created your client stack just to keep track of customers' names and addresses and what they owe you currently, it's not too late to modify it so it does a lot more.

Making the Client Stack More Powerful

In Chapter 5 you saw how HyperCard can calculate. Why not build some of that calculation power into your client stack by adding a button that does some of the arithmetic for you. This section shows you how.

Creating a Button that Totals a Bill

In the previous chapters, you copied buttons and changed their scripts. Although HyperCard encour-

ages copying and modifying wherever possible, you can also create new objects. For example, if you choose New Stack and uncheck Copy current background, you get a blank slate. In this new, blank background you can create everything you want in the background of the new stack—a picture, buttons, and fields. You can also use the Paint tools to draw your own pictures rather than relying on prefabricated pictures from existing background. With the Field and Button tools you're free to create brand-new fields or buttons as well.

In this section you'll create a new button and give it a script that will use HyperCard's calculating ability to total a bill.

1. Go to your Client stack by clicking the picture that represents it on the Home card.

First go Home if you're not already there.

2. Choose Background from the Edit menu to work on the stack's background.

This will place the button you're about to create on every card that shares this background.

3. Create a new button by Choosing the Button tool and Command-dragging.

The new button will present the total amount due. Command-drag across the existing words "Total Due" to place the button next to the Total Due field. Don't worry if you don't get the button in exactly the right place. You can move it or change its size later. Figure 6.1 shows where to drag.

4. Double-click the button to see the button dialog box.

The button is selected when you Command-drag to create it. You could also choose But-

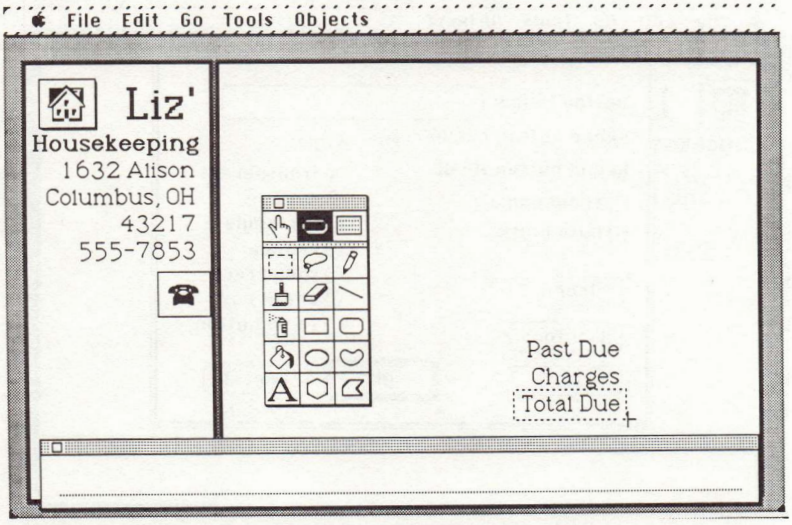


FIGURE 6–1 *Creating a new button by Command-dragging. Buttons can be transparent, showing through to anything under them such as this Paint text.*

ton Info from the Objects menu to see this selected button's dialog box. This dialog box is shown in Figure 6.2.

Hint

Use the button tool to adjust the size or shape of any button or field. You don't have to be in the background to adjust a background button or field.

Notice that the dialog box shows you that this is a background button by referring to its Background Button number (abbreviated "Bkgnd" to save space in the dialog box). It has an ID number that's unique within the current background. Its style is preset to Transparent, and the insertion point is in the Name blank, ready for you to name the button.

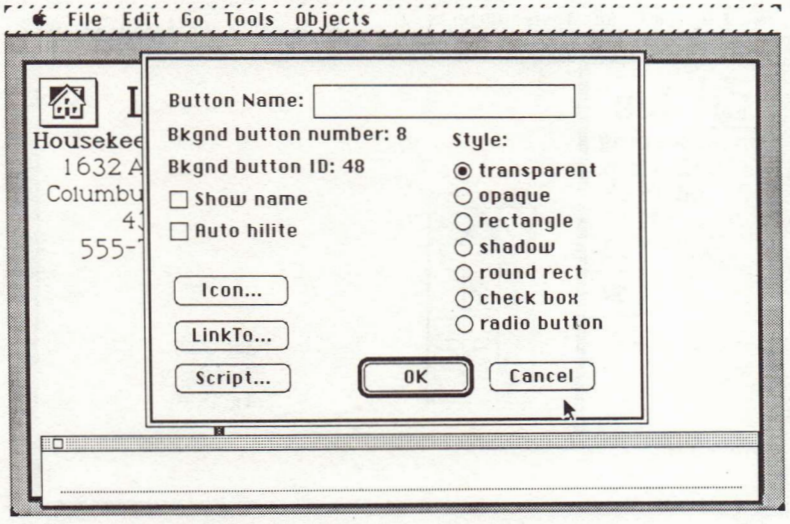


FIGURE 6–2 *The button dialog box has information about the selected button. It also has a LinkTo button for aiming the selected button at a destination you choose and a Script button for seeing and editing the button's script.*

Hint

Buttons you create by Command-dragging are preset to the transparent style; buttons you create with the New Button command are preset to rounded rectangles, with Show Name checked, and the preset name "New Button."

5. Name the button by typing "Total Due" into the Button Name blank.

Hint

You don't need to name buttons unless you want to display the name on the button itself or you want to refer to that button from other scripts. You won't show this button's name on the button itself, but you may want to refer to it later in a script. Naming an object never hurts.

6. Set the button to highlight when you press it by clicking Auto Hilite.

Auto Hilite specifies that the entire button will be highlighted whenever you click it using the Browse tool. If the button is transparent, as this one is, anything under the button—text pictures, or whatever—will be highlighted when you click the button.

7. Click OK or press the Return key.

Hint

Usually pressing the Return key is the "safest" thing in any dialog box. The safest thing in this case would cancel any setting you've chosen. But you use these dialog boxes most often to set some characteristic. So HyperCard interprets pressing the Return key as a go-ahead rather than a cancel and accepts any changes you've made in the settings. Click the Cancel button if you really do want to cancel.

8. Hold down the Shift key while you drag the button from one of its edges to adjust its size.

Notice that holding down the Shift key while you drag constrains movement to either the

horizontal or vertical plane. (If the button style is round rect, holding down the Shift key adjusts the button's height to the standard Macintosh button height.) Try to make the button match the Total Due text that's under it. Adjust the size so that the entire Total Due fits roomily within the button.

Hint

You can constrain the height of a button to be the height of a standard Macintosh button by holding down the Shift key while you use the Button tool to adjust the size and shape of the button.

9. Drag the button from its center to move it exactly where it should be.

Remember that you can always adjust it again later.

10. Use the Browse tool to see the effect of clicking the button.

When you click the button using the Browse tool, the Paint text beneath it is highlighted because you checked the Auto Hilite option.

You could probably spend a few more hours experimenting with the various button styles. For example, you could choose the round rect style (that looks like an ordinary button), name the button Total Due, and have Show Name checked on. This would obscure the Paint text in the background that says the same thing. (You could either delete the Paint text or leave it there; it's not hurting anything when it's not visible.)

Or you could try a rectangle or shadowed style of button to give a hint where you should click to make the button total the bill. Experiment with a few of the possibilities. (The check box and radio button styles probably aren't what you want here; they're for settings you click on and off.)

Editing a Button's Script

You've copied buttons to use in your own stacks and you've changed their destination using LinkTo and clicking a destination in the Destination palette. And you've seen scripts that tell HyperCard to go Home or to go to your new stack or sort cards in a stack. You've also seen how you can customize them to do what you want them to do.

When you create a new button, as you just did, the script is empty except for the message handler "on mouseUp" and "end mouseUp," as shown in Figure 6.3 and an insertion point where you can fill in the script. Now it's time to fill in this script.

1. Choose the Button tool.

2. Hold down the Shift key while you double-click the Total Due button.

> You can double-click any existing button whenever you're using the Button tool—to change the button's properties or to edit its script. Holding down the Shift key takes you directly to the button's script, without stopping at the dialog box.

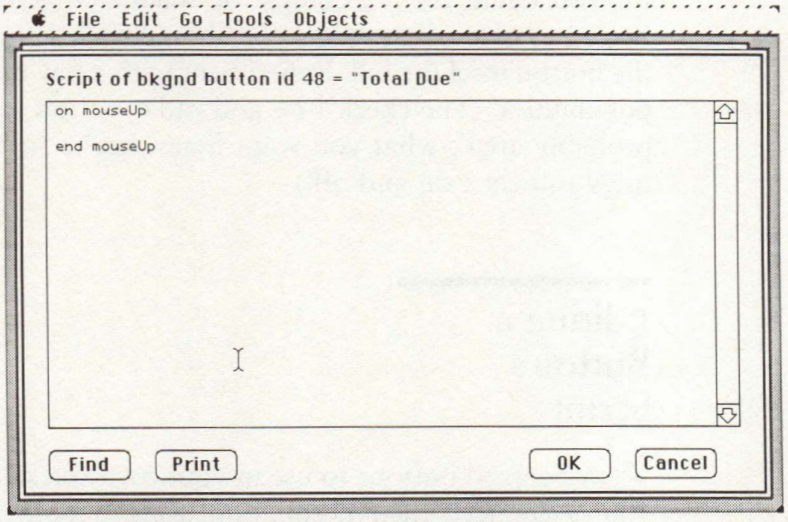


FIGURE 6–3 *The button's script is originally empty except for the on mouseUp and end mouseUp control structure. The insertion point is on the line between, so you can add a script that will be executed when the mouse button is pressed.*

Hint

Holding down the Shift key while you double-click a button takes you directly to the button's script, without bothering you with the Button dialog box.

3. Type:

```
Put field "Past Due" into field "Total Due"
Add field "Charges" to field "Total Due"
```

Type this exactly as shown, quotes and all. Check Figure 6.4 for how the script should look when you've finished typing the new lines into it. With two-word field names it's often necessary to include the quotes to prevent confusion, although they're not strictly

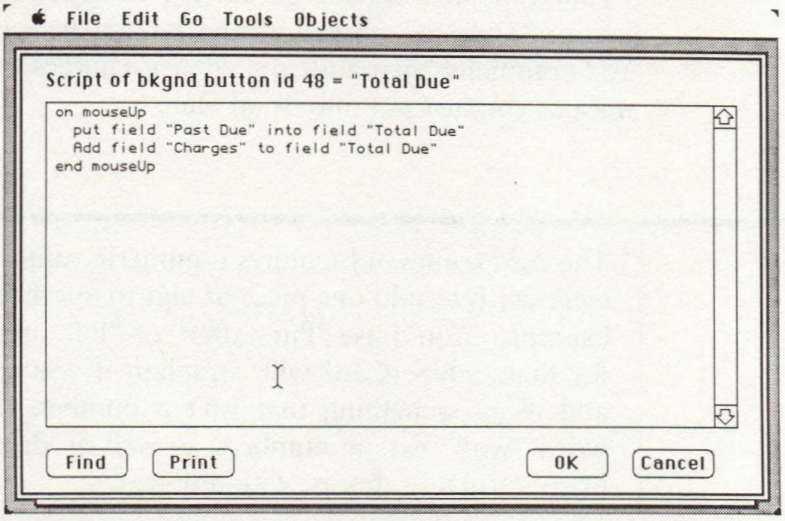


FIGURE 6–4 *The completed message handler for the Total Due button.*

required. Past Due, Total Due, and Charges are the names you gave these fields in Chapter 4.

4. Click OK.

You've told HyperCard that when the user clicks this button—causing the "mouseUp" message to be sent to the button—HyperCard should take the contents of the field Past Due and put them into the field Total Due. The next line contains a new command—Add. HyperCard should add the value in "Charges" to the total in Total Due.

You've used two HyperTalk commands in this script. The Put command places the amount currently shown in the Past Due field into the Total Due field. (The word "into" means replace anything that's currently there. You could also use "before" to append

the amount in front of the current contents of the field or "after" to append it after what's there.) The Add command then adds the current charges to the amount you just put into Total Due.

Hint

The Add command requires a numeric value. You can't use it to add one piece of text to another, for example. (You'd use "Put...after" or "Put...before" for that.) HyperCard will complain if you try to add using something that isn't a number. "Put" works with text or numbers as well as dates in many formats—that is, with any text.

5. Use the Browse tool to go to the card for any client whose bill you want to calculate.

Go to a card in My Clients that has a number in the Past Due field as well as in the Current Charges field.

6. Select any amount currently in the Past Due field and type: 0

You could also press the Tab key to select the first background field on the card—the Name and Address field—and then Tab repeatedly until you're at the Past Due field.

Hint

The Tab key takes you systematically to the next background field (from farthest to closest) and then through each card field on the current card.

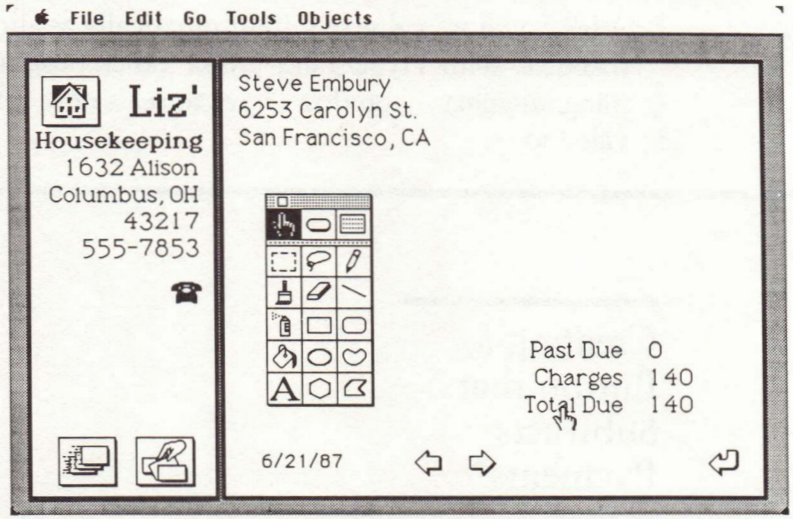


FIGURE 6–5 *A bill HyperCard calculated*

Hint

Don't use dollar signs when you want to do arithmetic on information in fields. And for now don't include cents in the amount.

7. Use the Browse tool to select the amount in the field "Charges" and type: 140

For the purpose of this exercise, let's say this is the amount you charge monthly.

8. Use the Browse tool to click the new button you created over Total Due.

HyperCard performs the button's script, placing the Past Due amount (0 in this case) into the Total Due and adding your Charges ($140) to the Total Due. Your client owes you $140. You can see how this looks in Figure 6.5.

Hint

A field with no value in it is automatically assumed to be 0. With HyperTalk, unlike other programming languages, you don't have to set a new field's value to 0.

Creating a Button that Subtracts Payments

If your customers send you payments, you can manually subtract the amount from the Past Due field and type in the current Past Due amount. But it would be much easier to create another button that subtracts payments automatically after asking you the amount of the payment. This section shows you how.

You've already used the Add command and the Put command to have HyperCard figure out the total amount due and display it in the right place. Now you will use the Ask command to present a dialog box for entering information. (The Ask command is similar to the Answer command you used when you modified the Sort button in Chapter 5.) The Ask command allows you not only to question, but adds a blank for an answer. The blank is preset to have some selected text in it—0 in this case. When you later use the button, you can change the content of the blank by typing or accept it as it is by clicking the OK button. If you OK it, HyperCard will subtract the amount you type in from the field that contains the Past Due amount. If the field then contains a negative number, because your client paid more than was

past due, that negative number is added to the Charges field (credited toward charges) and the Past Due field is set to 0. In other words, payments are first applied to past due amounts and then toward current charges.

Hint

Most of the commands you construct with HyperTalk can sound quite English-like. For example, Go to card 3 of stack "MyClients."

1. Choose the Button tool.

You're going to be working with a button again.

2. Choose Background from the Edit menu to work in the background.

Or press command-B to do the same thing.

3. Create a new button by holding down the Command key while you drag to the left of the words "Past Due."

You could also create the new button by choosing New button from the objects menu, but then it would be preset to a round rect with the name "New Button" showing.

4. Double-click the button to see its dialog box.

Type "Payment Received" into the button name blank.

5. Click Show Name to have HyperCard display the button's name and click Auto Hilite to have the button highlight when you click it using the Browse tool.

You could also make the button more traditional looking by clicking roundrect if you want.

6. Click the Script button to edit your new button's script.

Once again HyperCard has added "on mouseUp" and "end mouseUp" for you and put an insertion point between the lines.

7. Type in the new script:

```
ask "Amount of payment?" with 0
subtract it from field "Past Due"
if field "Past Due" < 0 then
    add field "Past Due" to field "Charges"
    put 0 into field "Past Due"
end if
send "mouseUp" to background button "Total Due"
```

Use the Backspace key to erase mistakes, the Return key for new lines. Press the Tab key when you've finished typing the lines to have HyperCard format them with the right indentation. After you have pressed the Tab key, your script should look exactly like the one in Figure 6.6.

Hint

HyperCard automatically indents certain lines of your script for you so that pairs of statements are indented the same amount. For example, "if field..." and "end if" are a pair of statements that are part of the same control structure and are indented the same amount so they visually balance each other. If the lines don't automatically shift to indent, you have a clue that you made a typing mistake. Check for typos.

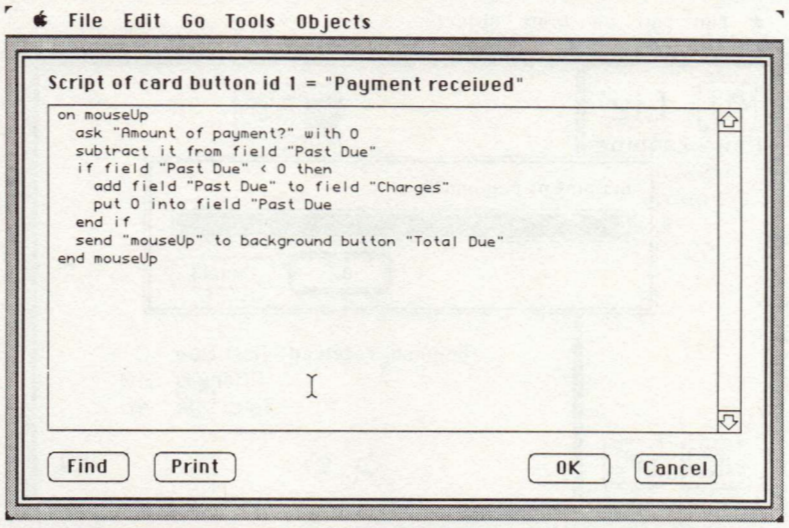


FIGURE 6–6 *The script for the Payment received button*

Don't forget the quotes around the field names and around the text you want HyperCard to preset as an answer.

8. Press the Enter key.

HyperCard indents the lines properly (if you haven't already pressed the Tab key to do this). You can see them properly indented the next time you open this script.

9. Adjust the button to fit its name.

Drag from an edge to make it larger.

10. Use the Browse tool to try out the button's script by clicking Payment Received.

Figure 6.7 shows you the dialog box that appears.

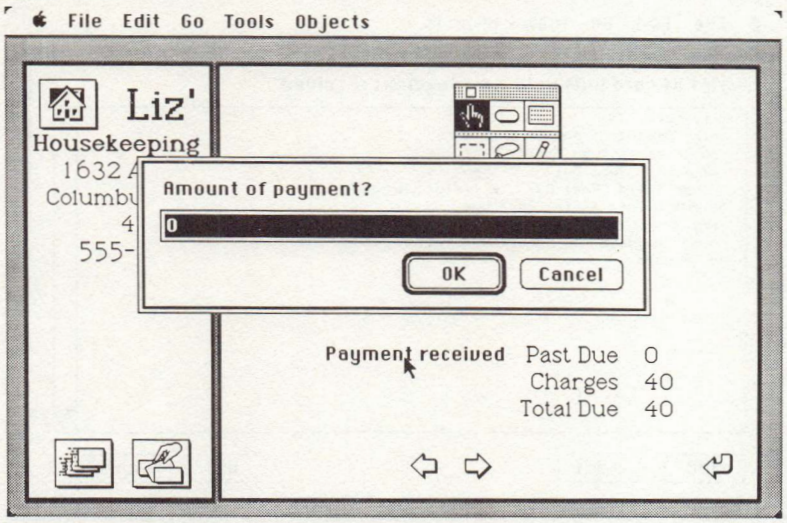


FIGURE 6–7 *Using the Payment received button. You enter the amount received and click OK. The Payment received button calculates the amount now due.*

11. When the dialog box appears, type: 50

This replaces the preset "0." Then click OK or press the Return or Enter key, to enter the Payment Received amount.

HyperCard shows that your client now has a credit of $50.

Examining the script more closely

Here's what each line in this script does:

```
Ask "Amount of payment?" with 0
```

presents a dialog box that asks the question "Amount of payment?" and provides the preset amount of 0, which is highlighted so that you can easily change it just by typing a new amount.

```
Subtract it from field "Past Due"
```

When you type in an amount and click the OK button, that amount is placed in a local variable called "it." HyperCard now takes that value (the value in "it") and subtracts it from the value currently in field Past Due.

```
If field "Past Due" < 0 then
```

Now we've placed a condition on the next action HyperCard should take. In other words, HyperCard should perform the next lines of this script only if the Past Due field contains a negative number—only if the customer caught up with past months and paid some of the current charges.

```
Add field "Past Due" to field "Charges"
```

You're telling HyperCard to put this negative amount into the Charges field.

```
Put 0 into field "Past Due"
```

The Past Due field is now 0 because the balance has been paid off.

```
end if
```

Hint

You can try out any one-line command in the Message box, without putting it in an object's script. Type the entire Command line into the Message box and press the Return key to perform the command, just as if you had created a button and put the command into the button's script. The Help system has examples of commands to try. Save yourself some typing by holding down the Command key while you drag across any one-line example there and press the Return key to try it out.

Sending a Message to Another Object

You have now completed reviewing the conditional part of the script. The script could end here, but the following line lets you pass the mouseUp message to another object:

```
send "mouseUp" to background button "Total Due"
```

Without this line you could click the Total Due button to have HyperCard perform the button's script. But by sending the mouseUp message to the Total Due button from within this script, you can have the Total Due script performed without clicking another button. The Past Due field (now 0) is placed in the Total Due field, and the Charges field (now a negative number) is added to the field Total Due.

Hint

The "Send" command lets any object send a message to another object. It's as if the second object had received the message from the system or through normal inheritance of the message.

Creating a Button That Moves Current Charges into Past Due

What you have now would probably work fine for the first month. But after a month, you'd need to add the current charges and move any unpaid balance from the Total Due into the Past Due field and recalculate the whole thing.

Here's how to make a button that keeps closer track of what your clients owe by moving the Charges into the Past Due field so that you can enter a new current charge:

1. Choose Background from the Edit menu, or type Command-B.

2. Hold down the Option and Shift keys while you use the Button tool to drag the Payment Received button.

Holding down the Option key copies the button; holding down the Shift key constrains the copy to the same horizontal or vertical plane as the original.

Hint

You can make a row of buttons that are all lined up by holding down the Shift key while you Option-drag to copy the original.

The new button has the same name as the original, and it has the same script. You'll change these, but keep the transparent style and leave Show Name unchecked.

3. Double-click the button to see its dialog box.

Be sure to use the Button tool to double-click.

4. Change this button's name to "New Month" by typing "New Month."

The typing automatically replaces the selected name. You'll see New Month when you put the dialog box away.

5. Leave Show Name and Auto hilite checked.

The name "New Month" will be displayed on the button and the button will be highlighted when it is clicked with the Browse tool.

6. Click OK.

Take a look at the new button to make sure you've set it to be like the original.

7. Use the Browse tool to click the New Month button.

The same dialog box appears that appears when you click the Payment received button because the New Month button has the same script as the button it was copied from. But the New Month button should do something different. Changing its script is easy enough.

8. Click Cancel.

9. Hold down the Shift key while you use the Button tool to double-click the new button.

10. Select all of the text between the "on mouseUp" and "end mouseUp" lines.

The selected script is shown in Figure 6.8. You'll replace these lines with a new script.

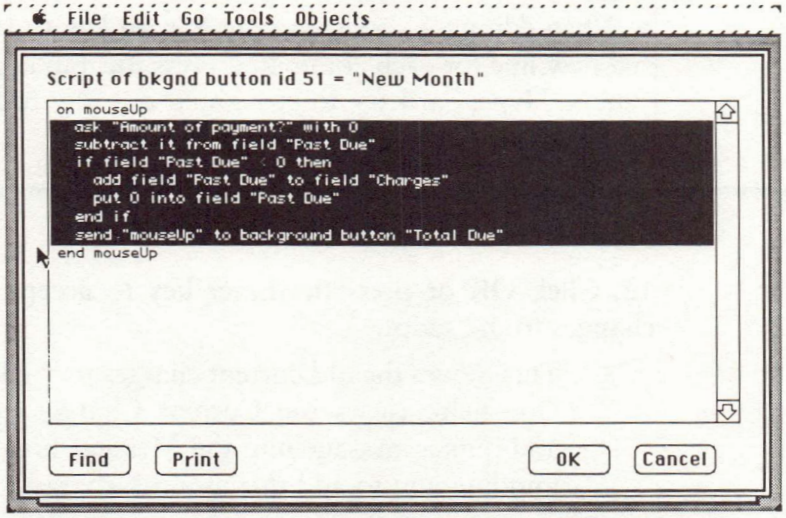


FIGURE 6–8 *You're going to replace the selected script with a new script. As with any Macintosh text editing, selecting text and then typing replaces the selection with what you subsequently type.*

Hint

Holding down the Shift key while you double-click a button is a shortcut for going right to its script, without stopping at the button dialog box.

11. Type:

```
add field "Charges" to field "Past Due"
put 0 into field "Charges"
put "Enter this month's charges" into Message Box
```

Type a Return to start each new line of the script. Be sure to include the quotes around any field name of more than one word to make sure HyperCard knows that a field name follows.

Hint

When editing a script, use the Return key to start a new line for each command. Press the Tab key to have HyperCard try to straighten out any indent errors you made while typing.

12. Click OK or press the Enter key to accept the changes to the script.

This moves the old current charges to the Past Due field, clears the Current Charges field, and puts a message into the Message box, reminding you to add this month's charges.

13. Use the Browse tool to click the New Month button.

The Charges field changes to 0 and the Past Due field changes to 90.

You're using the same commands you used before—"add" to add two numbers, and "put" to place a value in a field. You're also using the "put" command to place a message in the Message box.

Hint

The Message box is available to you within scripts as well as from the Edit menu. Any script can automatically display the Message box as part of its script. And you can also put something into the Message box—text or the result of calculation or the result of a function (the date, for example). Whenever you put something in the Message box from a script, the box is displayed automatically.

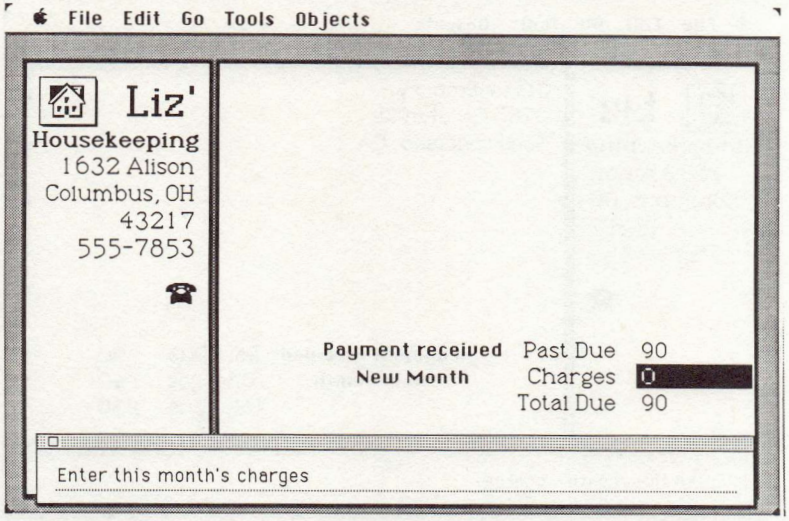


FIGURE 6–9 *Getting ready to add this month's charges.*

14. Add a new month's charges by selecting the zero in the Charges field and typing: 140

Figure 6.9 shows the Charges field selected, ready for you to type in the new charge.

15. Press the Total Due button

Pressing the Total Due button puts the amount in Past Due into Total Due and adds the current charges to it. See Figure 6.10.

Keeping Permanent Records

You can make this simple record system much more elaborate if you want to. It can keep a permanent record of the amounts charged and received by using another stack to keep track of the records.

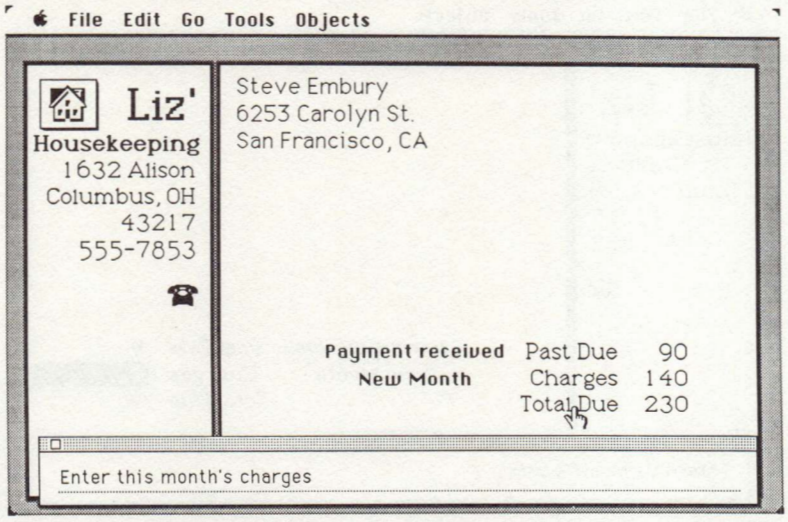


FIGURE 6–10 *Clicking the Total Due button to add this month's charges.*

Although your customers don't need a monthly history of the account, maybe you'd like to have this information available for yourself. Here's how to create another stack and modify the existing New Month button to create a permanent record of your clients' accounts.

Creating a New Stack Using the Current Background

In this section, you're going to create a new stack (Client History) to keep a history of your clients' past bills. To do this, you'll have HyperCard copy the client's current statement to another stack so that

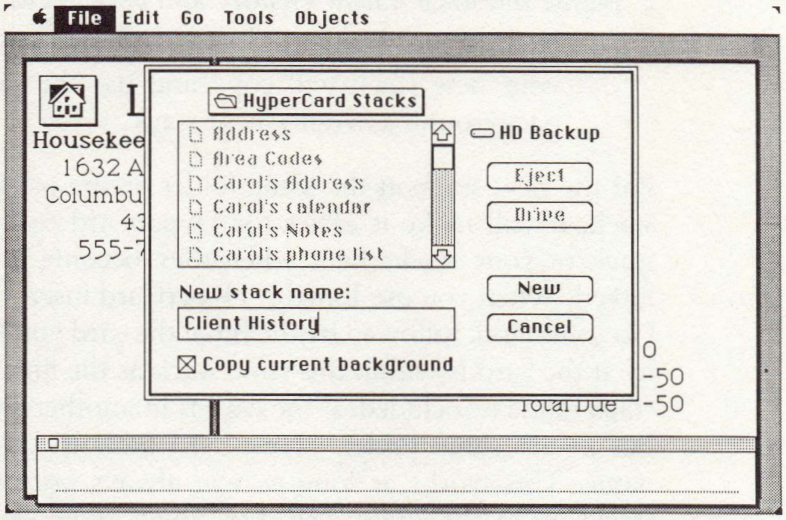


FIGURE 6–11 *Naming the new stack for Client History. The new stack uses the same background as the current stack unless you uncheck Copy Current background. Pressing HyperCard stacks takes you up the hierarchy to other folders and disks. Clicking Drive looks at folders in another disk drive. Clicking New creates a new stack with the name you give it in the name blank.*

your Client stack will always contain just the current month's activity. Then you'll be able to print out an account history of your Client History stack by choosing Print Report from the File menu.

1. Choose New Stack from the File menu.

It doesn't matter which card you're looking at when you choose the command, as long as you're somewhere within your Client stack. Each card in that stack uses the same background. Figure 6.11 shows the dialog box that appears when you choose this command.

2. Name the stack Client History and click New.

Leave Copy Current background checked so your new stack will copy and use the same background as your Client stack.

Put the new stack in the same folder as the original stack. It will make it easier for HyperCard to keep track of your stacks later when they become interlinked. When you use LinkTo, HyperCard inserts the Go command, followed by the ID of the card you link to. If the card is within the same stack as the link, no stack name is included. If the card is in another stack but in the same folder, HyperCard adds the stack name. This works as long as you always use these stacks from the same disk and don't move stacks from one folder to another. But if you move the destination stack into a different folder, or if you copy it to another disk, the link no longer works.

Hint

Not moving linked cards from their original folder ensures that the links will always work.

Hint

If you're building related stacks, keep all of the related stacks within the same folder. The "address" of the destination card will then keep its shortest form—name of stack only—and you won't have any broken links later on. You can fix broken links by including the correct path names in the "Look for Stacks in" card in the Home stack.

HyperCard takes you to the first card in your new stack. It has all the buttons and fields of your original

Client stack, but none of the text. It has a single card (because every stack has at least one card) but no real information on it. You'll add cards with information later, by creating a button that copies cards from the original stack to the new one.

3. Press the top left key to go back to the original Client stack.

Because these stacks look so much alike, check the stack name by choosing Stack Info from the Objects menu to see its name. Or put some Paint text in the new stack's background (such as "Client History") to distinguish it from your current Client stack (My Clients).

Hint

If you have a large-screen Macintosh, the stack name will always be visible at the top of the window just as document names are visible in other applications. Otherwise, choose Stack Info from the Objects menu to see the stack's pathname—the name of the current stack, any folders it's in, and the disk name—each separated by colons.

Moving Old Records into the Client History Stack

By now you know that you can always go back and change a button's script. You can edit the New Month button so that it automatically places a client's current records in the Client History stack at the beginning of each month. When you click the New

Month button (once each month), the client's current card will be copied to the Client History stack before you make any changes to the card to reflect the current month's transactions.

1. Choose the Button tool.

2. Hold down the Shift key while you double-click the New Month button.

This takes you directly to the button's script.

3. Click just before the second line—"Add field 'Charges' to field 'Past Due.'"

Now you can insert lines of script at this insertion point.

4. Type: DoMenu Copy Card

This will later copy the entire card, just as if you had chosen the command from the Edit menu.

5. Press the Return key to start a new line. Type: Push card.

6. Press the Return key again and type: Go to stack "Client History"

This script will go to the new stack where you want to keep the history of the transactions.

Hint

You could hide this action from the screen by including the line "lock screen" before going to the Client History stack and then including "unlock screen" when you come back to the original card. But going to the Client History stack will be a reminder to you about what is happening to the old card, to watch the script take you to the new stack and back again. Therefore, it's probably better to leave the screen unlocked.

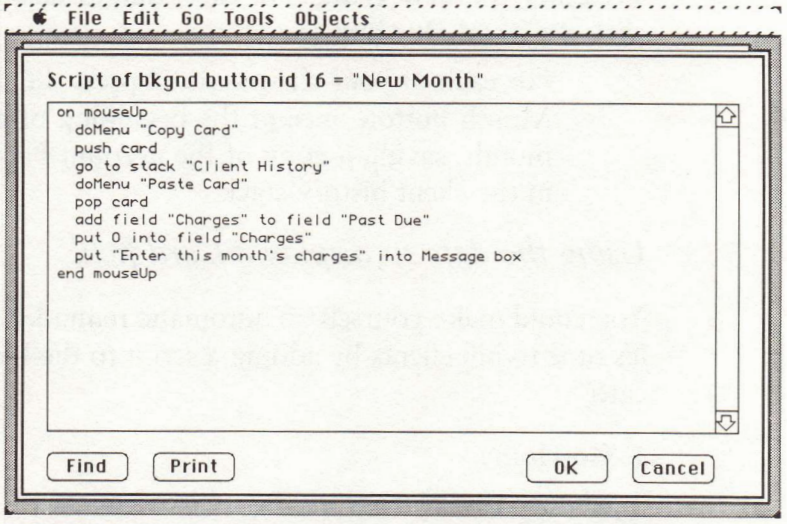


FIGURE 6–12 *The script that copies the current card to the Client History stack.*

7. Press the Return key and type: DoMenu Paste Card

This will paste a copy of your client's card into the Client History stack.

8. Press the Return key and type: Pop Card

This will later take you back to the original stack. (The card that was Pushed earlier in the script.)

9. Press Return

Figure 6.12 shows the completed script for the New Month button.

10. Click OK or press the Enter key and use the Browse tool to try the New Month button.

Say it's a new month and your client hasn't paid you anything.

11. Enter the new charges in the Charges field and click the Total Due button.

For each of your clients, you'll press the New Month button once at the beginning of each month, saving a copy of the account balance in the client history stack.

Using the date to display a Card field

You could make yourself an automatic reminder that it's time to bill clients by adding a script to the Home card.

1. Go Home.

2. Choose Card Info from the Objects menu.

3. Click Script and then type:

```
on openCard
  if word 2 of item 2 of the long date = 1 then
    show card field reminder
    show button "OK"
  end if
end openCard
```

4. Click OK.

5. Create a Card field by Command-dragging with the Field tool. Double-click the field and type into the field name: Reminder.

6. Click Shadowed and then click OK.

Choose a new font if you want by clicking Font before you click OK.

7. Use the Browse tool to click in the field and type: Remember to bill clients today!

8. Type into the Message box: Hide Card field "Reminder"

The card field will stay hidden until the first of the month.

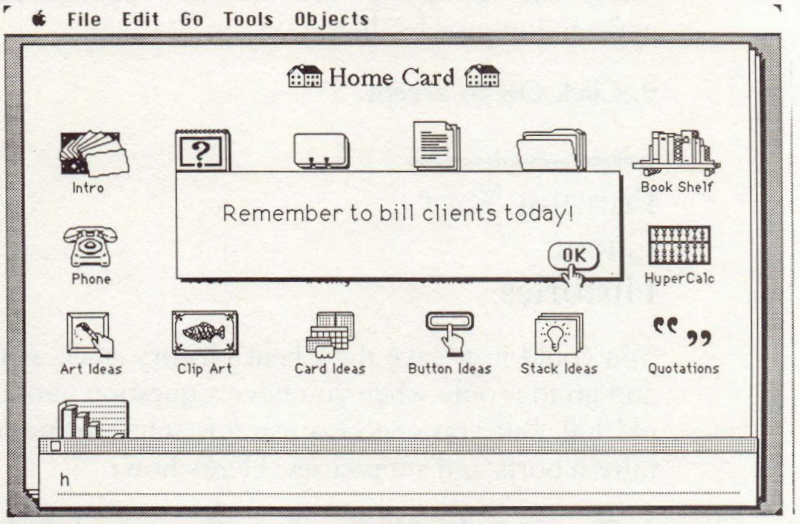


FIGURE 6–13 *You can create a field that reminds you to bill customers at the beginning of the month. HyperCard checks the Macintosh clock to know when to show the field.*

Now place a button in the field by following these steps.

1. Show the Card field by typing: Show Card field "Reminder"
2. Command-drag using the Button tool to create a small button as shown in Figure 6.13.
3. Double-click the button and type OK into the name blank.
4. Click RoundRect.
5. Click Show name and then click Script.
6. Type: Hide button "OK"
7. Press Return.
8. Type: Hide Card field "Reminder"

These lines should go between the "on mouseUp" and "end mouseUp" lines.

9. Click OK to accept.

Printing Your Client Histories

You could just leave the Client History stack as it is and go to it only when you have a question about an old bill. But you can also use this stack to produce nifty reports and summaries. Here's how:

1. Type into the Message box: Go to stack "Client History" Press the Return key.

This takes you to the first card in the new stack.

2. Flip through the cards and notice that they're arranged chronologically.

Most recent cards are at the beginning of the stack because that's where the script pastes them.

3. Choose Print Report to design a custom financial report.

This dialog box, shown in Figure 6.14, is full of riches! Print Report prints the text fields from either the entire stack or just within the current background. When the current background is checked, as it is here, you can print any or all of the fields in the background.

You click to select a field and you drag or Shift-click to make an additional selection. For this report, you probably want to include all of the fields, so leave them all selected.

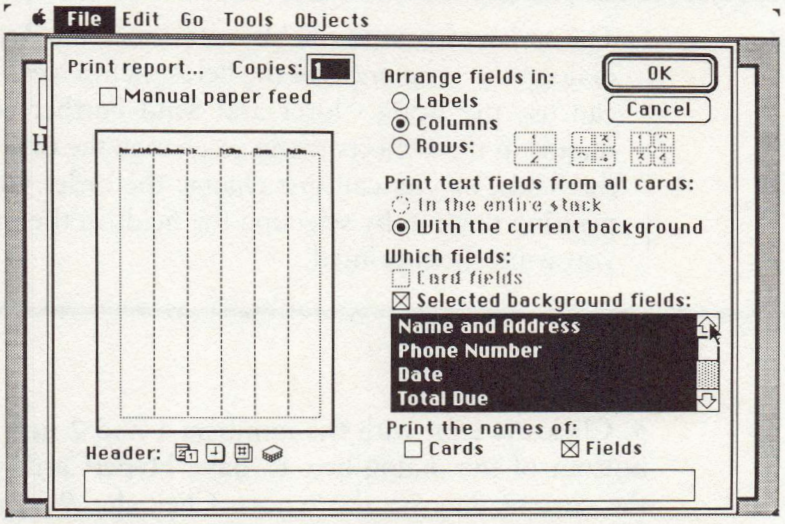


FIGURE 6–14 *The Print Report dialog box allows almost unlimited formats for printing only the text from stacks.*

4. Drag the vertical bars on the sample page at the left of the dialog box to adjust the spacing between the columns.

Leave the first column fairly wide so that it can hold the longest street address you have. Make the phone number and date fields slightly narrower, and make the columns that contain numbers quite small (unless you plan to charge a lot for your services!)

HyperCard shows the names of the fields in very tiny print across the top of the page—just legible enough for you to guess what they might be. You can change the order of fields by re-selecting them in the order you want them printed. For this report, the original order is probably fine to start with.

Leave Fields checked to include the names of each field at the top of each column.

Hint

The order of printing fields with the Print Report dialog box is the order of the fields themselves. You can use the Bring Closer and Send Farther commands in the Objects menu to change the order of the fields, or you can just change the order you're printing them in by selecting the fields in the order you want them printed.

5. Click the icon with the numbers 1 and 2 near the bottom of the dialog box to have HyperCard print the current date on the report. Click the # icon to have the pages numbered.

6. Click between the icons in the header blank.

7. Press the space bar to insert spaces between the date and the page number icons.

8. Click OK to have HyperCard print the report.

The next time you choose Print Report, HyperCard will remember the setting you chose last time. You can either print the report in the same format or change it.

Hint

In HyperCard, all print dialog settings are saved with individual stacks.

Summary

In this chapter, you've learned how to create buttons that perform the arithmatic functions on your billing

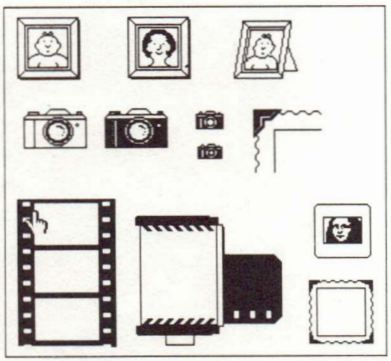
forms. You've created a button that totals the bill, one that subtracts payments, and one that moves current charges into the past due slot. This stack can provide you with a permanent history for each client. The HyperCard Help system completely documents the HyperTalk language, including all commands and functions. You can use the Help system not only to read about HyperTalk but to try out many sample scripts. Get to Help from anywhere in HyperCard by pressing Command -?.

The next chapter is a bagful of tricks to use in designing stacks.

Chapter 7

Practical HyperCard Tricks

"With HyperCard we have designers writing programs and programmers building rock videos."
Chris Espinosa



In an early draft of the MacPaint book, I wrote, "Watch the magic happen." The sentence was deleted when reviewers pointed out that it wasn't really magic to get the Macintosh to flip a graphic selection upside down. It could all be explained. But can't all magic be explained

as well? What follows is a collection of practical tricks for mastering HyperCard magic. I might call this chapter: "What I wish I had known when I started creating stacks."

Text and Graphics: A Good Marriage

Text and graphics have been tentatively coexisting in an uneasy truce since the early days of Macintosh, but it's always taken a bit of effort to combine them in one place. Often you had to do all the text in one application and all the pictures in another, and then you had to assemble them into one piece at the last possible moment because you couldn't edit them both in the same place. Or you would add pictures just before publication because scrolling was so slow when you added pictures to a word-processing application. In HyperCard, text and graphics seem really comfortable together at last. Figure 7.1 shows how a few words and a great picture can work together to get the message across better than either the text or the picture could alone.

Before I started writing the HyperCard Help system, Bill Atkinson looked me straight in the eye and gently pleaded, "Pictures. Lots of pictures." I wasn't too surprised, remembering his preference for graphics over words from my experience writing the MacPaint book. But even if I'd never heard it from Bill, I think I would have concluded before very long that HyperCard is at its best and most powerful when it exploits the graphic element, when it says it with a picture, even in applications that are not necessarily "graphic"—an expense report, for example.

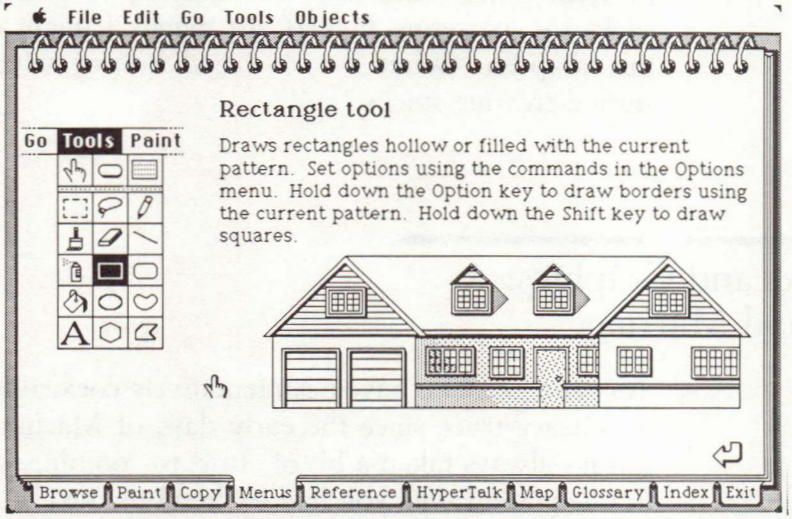


FIGURE 7–1 *Form and the shape of content.*

As beautiful as the Macintosh screen can be, it still feels better to curl up in a chair with a book when you have long passages to read. (I keep trying to imagine the square little Macintosh perched on someone's lap.) My rule of thumb became to use a picture instead of words wherever I could. Coworker Kristee Kreitman and I christened ourselves "Text and Graphics" (sung to the tune of "Love and Marriage").

What if you can't draw? It doesn't matter. You can use the great collection of card pictures in the Card Ideas stack and snippets of art in the Art Ideas stack included with HyperCard, or use the Import Paint command to use images from the many available collections of MacPaint or MacPaint-compatible documents. If you use the selection rectangle and the lasso to select and change the pictures, and use the other Paint tools to add your own flourishes or a personal logo you type in, it's almost as good as having your

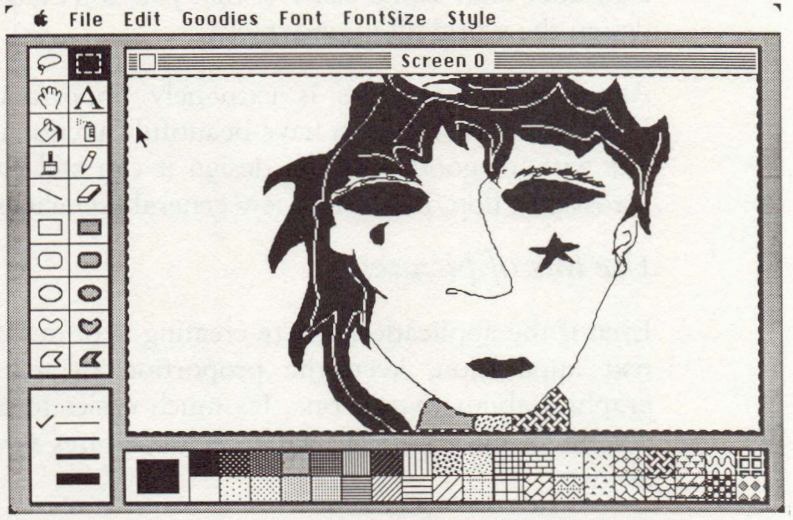


FIGURE 7–2 *You can import a MacPaint to HyperCard.*

own graphics department. Figure 7.2 shows a MacPaint picture being prepared in MacPaint before importing to HyperCard.

Restraint and Other Design Guidelines

But having art readily available is different from using graphics well. In fact, having abundant graphics available can sometimes create new problems. Just as the early days of Macintosh produced some untempered graphic flair in newsletters that used 32 typestyles in one page, all of these new artistic tools tempt us to abandon caution in graphic design. But after allowing yourself a brief fling, you'll probably take a good hard look at what you've done and hold back

a bit after that. You'll discover that you don't have to design the world all in one stack!

Attention to aesthetics is extremely important in HyperCard. A stack can have beautiful functionality, but without good aesthetic design it can still come across as a flop. Here are a few general guidelines:

Use lots of pictures

Even if the application you're creating is primarily a text application, keep the proportion of text to graphics about two to one. It's much easier to read and understand a screen that includes pictures as well as text.

It was interesting to watch test subjects as they tried out the HyperCard Help system during its development. Whenever new users came upon screens with pictures, their eyes went immediately there, and only after they had taken the picture in did they go back to reading the words. Soon they began to expect graphics on each card and seemed restless when they found cards with nothing but words on them.

Use traditional rules of book design

Many of the same principles apply when choosing text styles for text body and heads. Go light on the number of levels of heads you use. It's harder to keep track of what level you're at when you can see just one screen at a time than when you're thumbing through pages in a book. And you can't expect the readers of your text to remember that you're at the third or fourth level of an outline they can't see. Figure 7.3 shows the two levels of heads used in the HyperTalk section of the Help system. Every card has a title, which is actually the first-level head. If a card isn't the first card using that level head, it has

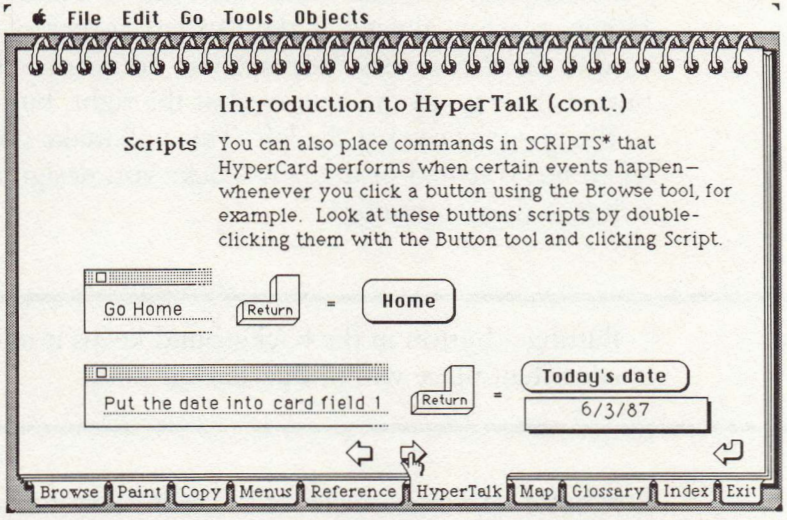


FIGURE 7–3 *The Introduction to HyperTalk uses both text and graphics to teach the programming language.*

"(cont.)" following it to keep the reader aware of the level they're currently at. In a book, you wouldn't have to do this, because the reader could see at a glance what level they were at (by looking at a running head or the actual level head itself). The second-level head—scripts—is set to the side of the main body of text.

Don't mix too many fonts within an application. Look at a book you love and see how many design elements it uses—one for the main text, one for chapter titles, two levels of heads, one for picture captions? Books with too many design elements are confusing and distracting. So are over-complicated stacks.

Consistency is also an important design principle. Put elements such as headings, graphics, or navigation buttons in a consistent place throughout the stack.

For example, if your stack includes a button to Home, place it always at the top left corner of the card or another consistent place you decide on. Place buttons that go to the next card at the right, buttons to the previous card at the left. This will make it easier to feel oriented in the new stacks you design and will give them continuity.

Hint	Putting a button in the background keeps it in one consistent place within that background.

But build in some surprises as well—just to keep things interesting.

Gray and black for definition and depth

Cards crave edges and borders. All-white cards tend to float away off the sides of the screen. One of the easiest tricks is to create a "drop shadow" effect around a rectangle by adding a one- or two-pixel border on two sides of a card. Be consistent in where you have the "light source" coming from—for example, if it comes from the top left, as it does in Figure 7.4, the shadows will always fall on the bottom right of the image. And try for some consistency in how deep the shadows are—always one or always two pixels deep, for example. Your graphics will look more pulled together if they have similar proportions.

Use rectangle or shadowed fields (choose these styles from the Field dialog box) to define the areas where text should go. Figure 7.5 shows some examples of the styles of fields you can use. Notice that the field at the top right of this picture doesn't have an outline, but it does use underlining (you get this by clicking

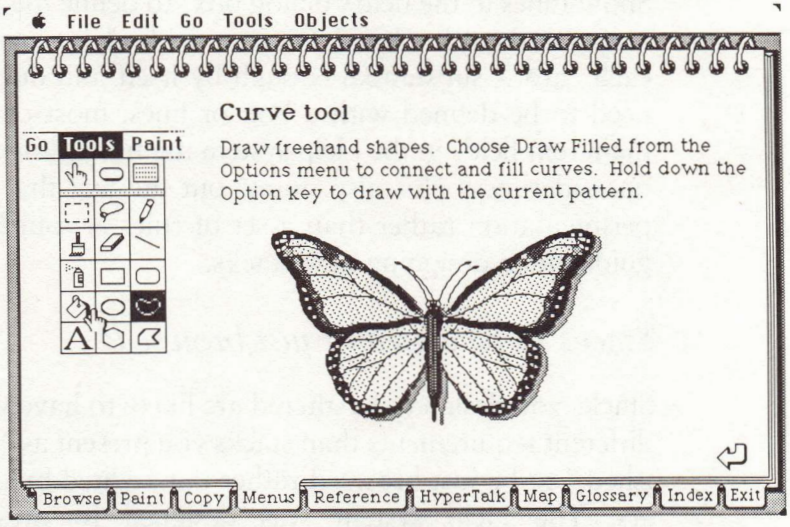


FIGURE 7–4 *The butterfly is shadowed on the right. Cards that explain what commands do display the menubar so that the menu won't be confused with the real Tools window.*

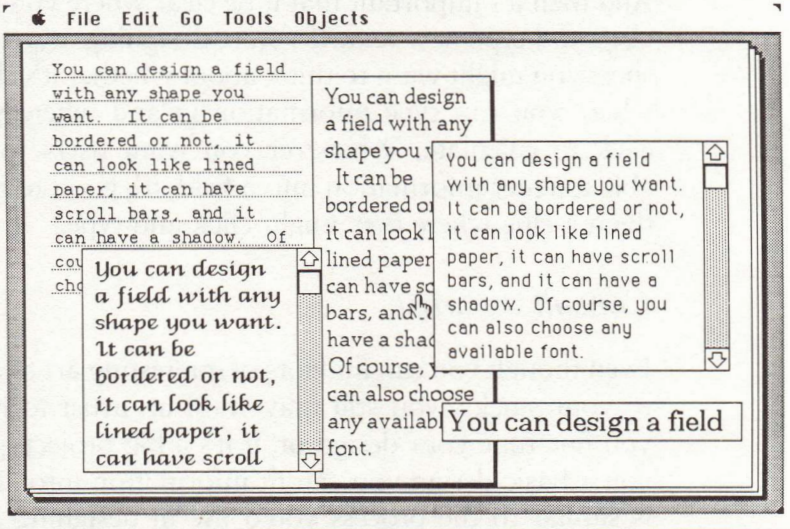


FIGURE 7–5 *A variety of field styles*

Show Lines in the field's dialog box) to define the text area. On the other hand, because the body of text on each card is substantial enough by itself and doesn't need to be defined with a box or lines, most of the main text fields in the Help system use no outline and no underlines. This also points out the fact that experimentation rather than a set of rules is your best guideline in designing new stacks.

Stacks to be edited or just browsed?

Stacks you design to be altered are likely to have very different requirements than stacks you present as "finished," to be just browsed rather than edited by others. The Help system was designed to present information, not as a place for users to enter their own information (although you could easily set up a Help system meant to be annotated, by including a field labeled "Notes"). But a calendar is set up to be filled in later. Therefore, it's important to determine whether your stacks will be edited or just browsed. And then it's important that it be clear where you can click and type new text. If you're designing a spreadsheet you might want to think about whether it's clear where you can type information in and where you click to calculate. Whenever you want users to be able to enter information into a field, it's good to give them a clue where they might click and type.

Consult an artist

Even though you can find lots of interesting art to add to your stacks, you still may want an artist to help you fine tune your design or, if it's a big project, give you a basic design you can fit information into. This is similar to the process you'd use in designing and writing a book. You may have a very clear idea of

what you want to say, but without a structure to put it into you're likely to feel lost. Working closely with an artist from the beginning can help you get the design to take the shape of the content.

Identify the elements you're going to need—what kind of divisions and how many, what kind of illustrations and how many. It's a lot easier to work on stacks if you have a rough design to work with from the start, even if the design isn't fleshed out or if it changes radically later on. And until you're an experienced HyperCard author, you may want to stick with one of the stack designs included in the Stack Ideas stack.

Starting Simple

Lay out the basic content structure or functionality of your stack before you build it in HyperCard. Do this on paper. It's very easy to get carried away in HyperCard, following a branch of a plan that may not work out to be the best one in the end. It's fine to follow flights of fancy, but you may be in for more than you bargained for unless you set forth with and at least try to stick to a plan. A simple outline on paper is usually all you need. If you're led to abandon the plan later, fine. You'll have started off with sure footing.

My first attempt at building a Help stack found me inching my way along a limb whose plan was far grander than anything I could complete in a reasonable amount of time. And it wasn't a plan that would have worked out in the end anyway. Quite naturally taken with the whole idea of linking together information in a "hypermedia" application, I began fol-

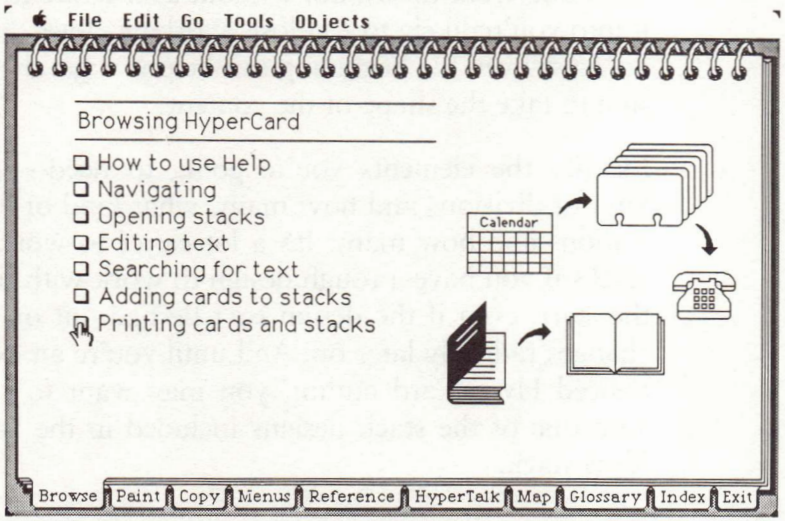


FIGURE 7–6 *This basic topic list was the backbone of the Help system design. From it grew an extended list covering several more cards.*

lowing a tiny branch to its ultimate extension. Then I looked back at the whole thing and realized I could never build the tree like that. It branched out too quickly and into too much detail for me to handle. It was hard not to get lost, even within just twenty or thirty cards. The design needed a consistent framework and a simple structure.

I started over with a small list of topics—each accessible from a central place, each branching no more than one additional level, and each with the same look and a consistent way to get back to the main topic card (see Figure 7.6.) Starting with this single list of topics, I was eventually able to build in all the complexity I needed, without getting bogged down in detail at the beginning.

Don't be afraid to throw out what doesn't work! Starting with a plan doesn't mean you need to stay

with it forever. Sometimes the form needs to change as the content becomes clearer. Realize that you probably learned something in the process. In the long run it's a lot better to throw out a design that doesn't work than to hang onto it because you feel you spent so much time developing it.

The first couple of weeks I worked on HyperCard, I tried using the bulletin board analogy, where all of the cards were stuck on this magical bulletin board and the links were pieces of yarn draped from card to card. But I was missing the main points I needed to convey with all of my "yarn draping." Bill Atkinson interrupted my yarn drawing with six or seven main topics that needed to be explained. We put each topic on a small piece of paper and lined them up on the floor, rearranging topics and subtopics to get them in the right relationship. That same basic structure is still in the Help system, embellished and sometimes twisted into something quite different. But I was always glad I had it as a foundation.

Layers and Backgrounds

There's a simplicity in the basic elements of HyperCard—text, pictures, and buttons. But this simplicity can be somewhat deceiving because although there are just these three elements, they can be combined in ways that produce a rich and complex fabric. This complexity can be confusing until you're used to the idea of layers.

We're not used to "layers" on the Macintosh screen. In MacPaint there's a single layer of bits—each pixel is either black or white—and that's it. The only exception is when you paste a picture in and it "floats" in its selection rectangle above the rest of the picture

until you "set it down" by clicking somewhere outside the selection.

HyperCard has this same "floating" layer, but it has other layers as well. If you understand how layers work and how to use them together with transparency and opacity to present exactly the information you want, you'll love all the flexibility they give you as an author.

Background and Card Pictures

Every card can have both a card picture as well as a background picture it shares with other cards that have the same background. These card and background pictures are each fixed in place front to back. Figure 7.7 shows the relationship of objects within a card and its background.

The background picture is the deepest layer because it is behind everything else. The card picture is the intermediate layer, because it is between the background picture and you. The background picture is always opaque, because there's nothing behind it to hide or show through, but the card picture can be either opaque or transparent, depending on whether you want to see through to or cover up what's behind it.

Pictures you paint with the Paint tools or paste in from the Clipboard are initially opaque when you paint them or paste them, but you can make any picture transparent by selecting it and choosing the Transparent command. Figure 7.8 shows a picture

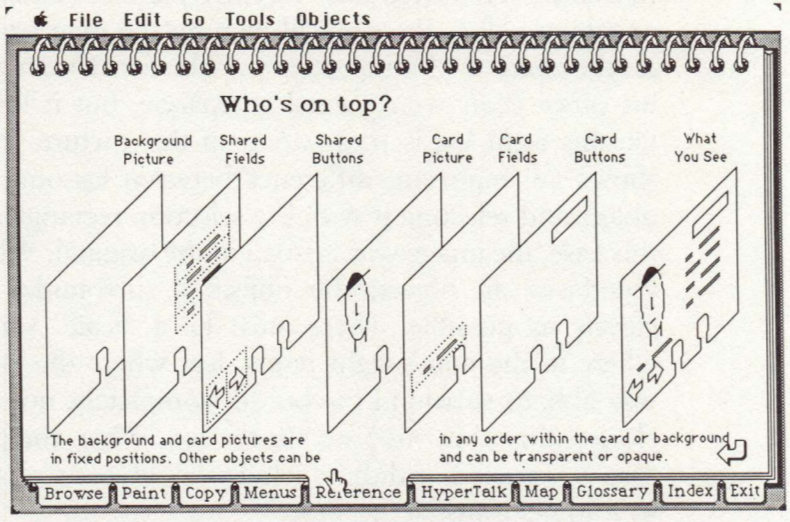


FIGURE 7–7 *Layers in HyperCard. The background and card pictures are in fixed, back-to-front position. Other objects can be in any order within the card or background and can be transparent or opaque.*

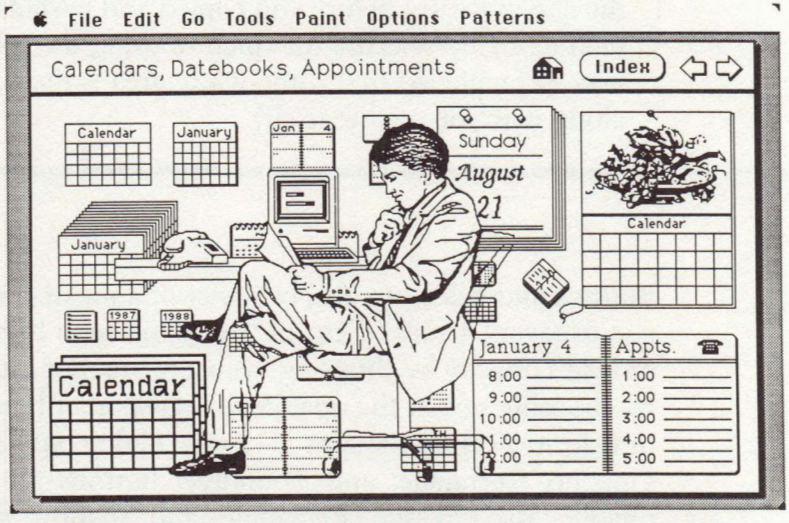


FIGURE 7–8 *Opacity. An area must be enclosed to be opaque. The pants' leg remains transparent because the outline has a hole in it.*

that has been pasted over another picture. (The calendars are all in the original picture; the man sitting on the chair has been pasted in.) The entire man and his office chair were pasted in opaque, but it looks like his right leg is transparent in this picture. This shows an important difference between lassoing an object and selecting it with the selection rectangle. In this case, the image was lassoed in the original. When you lasso an object, the object is surrounded as closely as possible. There must be a "leak" somewhere in the man's right pants' leg, where the lasso was able to surround the border completely, not enclosing the white area within the leg. When the picture was pasted, it didn't include the white area, and so it looks transparent there.

Hint

The "transparent leg" could be avoided by finding the gap in the leg before you copied and pasted the picture, or by selecting the picture using the selection rectangle (so that some white area is included along with the picture itself).

Buttons and fields can also be either opaque or transparent layers. Unlike pictures, you don't set a button or field's opacity by choosing from a menu, but rather by choosing one of the opaque or transparent button or field styles from the button or field dialog box. The Opaque, Rectangle, and Roundrect buttons are all opaque; the Opaque, Rectangle, and Shadow field styles are all opaque.

Background and Card Buttons

Like pictures, buttons can be either in the background layer or on the card layer. The background buttons are shared by every card that uses the background and the card buttons belong to individual cards. Unlike pictures, buttons aren't fixed at certain immovable front-to-back positions within the layering. They can be pushed back or pulled forward (using the Send Farther and Bring Closer commands). The only limitation is that background buttons can't move any closer than the card picture (which is fixed in place). Nor can card buttons move behind the card picture. Like pictures, buttons can also be either opaque or transparent to cover up or show through to objects behind them.

Bring Closer and Send Farther

Two commands—Bring Closer and Send Farther in the Objects menu—are just for rearranging the front-to-back-order of objects on a card. When you create a new button or field, it's automatically placed closer to you than the last button or field you created. But you can change that order by using the Bring Closer and Send Farther commands. Selecting an object and choosing Send Farther pushes the object one level back. If the object at the next level back is of the same

type—another button if it's a button, another field if it's a field—the numbers are reassigned to show who's currently on top. (The higher the number, the closer to the top the object is.) An object's number is its number within objects of the same type—field 3 among 4 fields, for example. If the object at the next level is of a different type, the numbers don't change, but their relative positions do.

This same ordering scheme is used by HyperCard to determine the order fields should be tabbed through—the order fields will be highlighted in if you press the Tab key anytime you have an insertion point or other selection in text. This is something to think about if you're designing a stack where Tab order is important—any application where you'll later be entering text such as a calendar, for example. Look at one of the monthly calendars in Stack Ideas to see how this works. If you press the Tab key while you're editing text in any of the days' text field, you select the following day's text. If the fields for the days in the month had been created in the wrong order, pressing the Tab key might take you to a random place in the calendar. (All would not be lost if you did do this; you could use the Send Farther and Bring Closer commands to reorder the fields.)

Tab order is also used when you want to print only the text in a stack. The text fields are preset to be printed in their back-to-front order.

Knowing about layers and opacity and using the Bring Closer and Send Farther commands will let you display exactly the information you want on any card.

The Order of Objects on a Card

Fields can also belong to the background or the card, and they, also, can move freely in front to back place, as long as they stay within either the background or the card domain. You use the Send Farther and Bring Closer commands to move a selected field in either direction—farther away from you or closer to what you see. Each object is assigned a number by HyperCard. The objects furthest away have the lowest numbers (they're numbered in the order of their creation).

Because objects are numbered only with other objects of the same type, when they're mixed together with objects of another type (buttons with fields, for example), bringing them closer or sending them farther may not change their number. For example, background field 3 may move in front of background button 3 but still be behind background field 2. It's number won't change unless it moves in front of background field 2.

There are several reasons to care about the order of objects on a card:

- The order sometimes determines what you can see.
  If an object is opaque you won't see an object behind it. When you don't want to make an object transparent (if it's a button or field, for example, making it transparent means it can't have a border), you change the object's order to be able to see something behind it.

- The order determines which object gets clicks.
  If there's more than one object front to back in the location you click on the screen, the closest object will receive (and be able to handle with a message handler) the mouseUp message that's sent when you click. Any buttons underneath that first button will never get the mouseUp message unless the first button's script explicity sends the mouseUp message using the "send" control structure ("send mouseUp to button 2", for example).

- The order determines Tab order.
  When you press the Tab key, fields are tabbed through in lowest to highest order. All background fields on the current card are tabbed through first, and then the card fields on the current card. If you want a field to be tabbed to before another field, use the Bring Closer or Send Further commands to reorder the fields.

- The order is used for presetting the Print Report dialog box.
  When you choose Print Report from the File menu, the fields are presented in their lowest to highest order. You can actually print the fields in a different order by Shift-clicking them to select them in the order you want them printed, so you may not want to change the fields' order just for printing. But it's something else to think about in deciding which order fields should be in.

- The order is used in searching for text. When you search for text, HyperCard searches lower-numbered fields before higher numbered fields. (It uses the same order as it uses for Tab order.) If you want a field to be searched before another field, change their order.

Using card buttons to hide stack buttons

Often there are buttons you'd like to appear on all but a few cards that share a background. For example, in the Help system, I included three arrows in most of the backgrounds. An arrow that points right takes you to the next card on the same subject; an arrow that points to the left takes you to a previous card on the same subject; and a return arrow takes you back to the topic card or a cross-reference that brought you to the current subject.

Placing these arrows in the background made all three of them appear on every card with the same background. But when a card was the first one on a topic I didn't want a left arrow. And unless the card was the last on the topic I didn't want the return arrow to show up. To cover up a stack button so that it can't be seen (and clicking it will have no effect) on a card, I placed a "do-nothing" button over the stack button on the cards I didn't want to use that background button. You make a "do-nothing" button by using the Button tool to drag a new button over the area. Click Opaque in the Button Style dialog box. Don't link the button anywhere, and put nothing in its script. Figure 7.9 shows using a card button on the first card of HyperCard Help to cover up a background button that on all other cards takes you back to where you've been before.

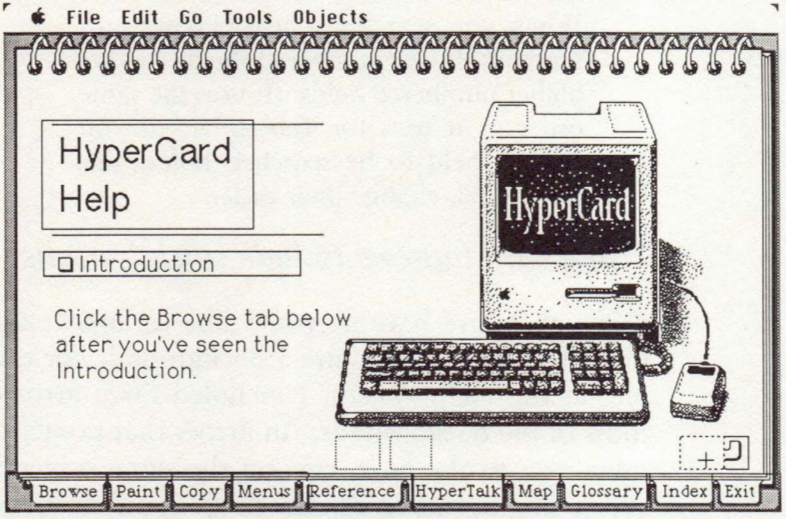


FIGURE 7–9 *An opaque card button covers up the background button in the bottom right corner.*

You might also use a card button to override an action that normally takes place with a different action. For example, a stack might include a background button that takes you to the next card. The background button's script says:

```
on mouseUp
  go next card
end mouseUp
```

This might be what you want the button to do for all of the cards in the background except one or two. Maybe you want the last card of the stack to go to the Home card, for example. To have clicking in the same spot take you back to the Home card instead of to the next card, you'd add a card button on top of the background button with the script:

```
on mouseUp
  Go Home
end mouseUp
```

Now whenever you click in this spot on this card only, you'll go Home instead of to the next card.

Using More than One Background

A stack starts out with one background. When you choose New Stack from the File menu, you create the stack either from scratch or by copying another background. But because stacks often grow to need more than one background—either because you need a different background picture or different background fields or buttons—you don't want to divide your application up into more than one stack. This happened with the Help system. It started with one background, but eventually grew to 17! Why?

My first divergence from the simple outline of six topics came when I was working out how to present explanations of the commands in menus. I wanted to be able to show a picture of the menu and to have clicking on any command in the menu take the reader to a description of the command.

Using a separate background for each menu let me put the picture of the menu in the background, and for each command create one corresponding background button that takes you to the description of that command (see Figure 7.10.)

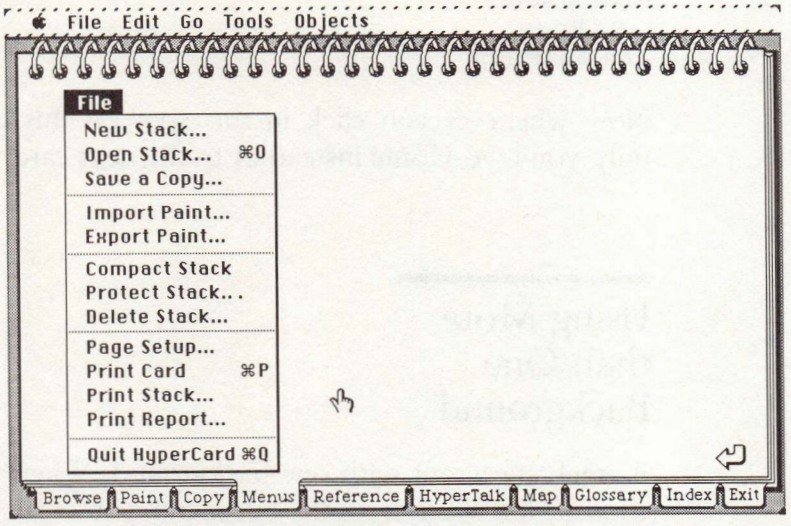


FIGURE 7–10 *One of the backgrounds used in the Help stack. Putting this picture and buttons to each menu command in the background saves space and effort.*

Without using the separate backgrounds, each card that described a command in a menu would have had to have a picture of the menu and a card button to each of the commands in the menu—for a total of 144 buttons for a menu with 12 commands! Having that same menu in a separate background meant I could use 12 background buttons instead of 144 card buttons, and I could use a single picture of the menu instead of 12 card pictures. I did use card pictures to show the highlighted command being described on that card; the technique I used to do this was to go to the background picture, select a rectangular area around the command being described, copy it, go out of the background, paste the picture, and choose Invert (to highlight the command).

Adding a New Background to a Stack

You can use the New Background command to add a background to an existing stack. This gives you a blank new background with one blank card in it. But this means you have to create the contents of that new background yourself, separately copying whatever picture, fields, and buttons you want in that background. This is what I did to create the various sections of Help. I'd create the new background, copy the original background picture, modify the picture to suit the new background (usually changing the tabs so that a different one was in the front), copy the fields that were appropriate, and then copy the background buttons I needed, often modifying their scripts to do something a little different from what the original did.

There's another way to do the same thing that might be a little faster, depending on how much of the original background you want to keep:

1. Copy a card that uses the original background.

2. Paste it into another stack temporarily and make a small change to it to differentiate it from the original background.

3. Paste it back into the original stack.

HyperCard recognizes it as a separate background even though it's almost identical to the original. Now you can make it as similar to or different from the

original as you want. The only cards affected by subsequent changes to this background are those you create while viewing the card that has the new background or while viewing any of its descendents.

Adding Backgrounds by Copying a Card

The easiest way to add a new background in a stack is to copy an existing background from another stack.

The way you do this is a little sneaky. You can't just go to a background you like, copy it, and paste it into another background. And you can't paste an existing background alone into another stack. *Backgrounds don't travel by themselves.* They always are attached to at least one card.

The way to copy a background is to get a card that uses that background and later discard the card itself if you don't want what's on it. Here's how you do it:

1. **Go to the card whose background you want.**
2. **Choose Copy Card.**
3. **Paste the card into the stack you want to add the background to.**

   Position it in the place in the stack where you want the new background to begin by going to the card that should be before it and then choose Paste Card from the Edit menu.

4. Get rid of the contents of the card if you don't need them in the new stack.

Deleting the card at this point will also delete the background, because you're deleting the only card in the stack that uses that background. But once you've added another card to the background (by choosing New Card while you're viewing the original) deleting the original won't delete its background. Backgrounds stick around as long as there are cards that use them.

You can also propagate backgrounds into completely new stacks by choosing New Stack while you're at any card. If you leave Copy Current Background checked in the dialog box that appears, your new stack will have a copy of the same background as the card you were at when you chose the command.

How does HyperCard know which cards should use which background? A card is born with a background when you create it. It has the same background as the card you're looking at when you choose the New Card command. When you add a new background (either by creating a new one or adding a card that has a different background), you also add the ability to add other cards that have that background. Where you are when you choose New Card makes a big difference. You can't replace a card's background with a different one—you can only modify the background (and thereby modify every card that shares it). To change the background of a specific card, you need to create a new card with the background you want, and then copy each separate element of the original card to it—the pictures, text, and card fields or buttons one at a time.

More than One Stack

Sometimes another background isn't enough. The Help Index is a good example. I could have added both of its fields to the main background of Help. (A stack can have as many fields as you have room to create.) But having many text fields on the same card (as you would if you added fields to the ones that already exist in Help) makes it difficult and confusing to add text to the right field, with insertion points popping up where you least expect them, and having the Tab key bring you to fields you don't really want to use.

I could have added a new background to the main Help stack, still keeping it within the same stack. But the Help Index also has special search requirements. While someone was in the Index I wanted searching to be limited to just the index entries. I could have put the Index in a separate background and limited the searching to certain fields, as I did in the Glossary. But it always felt as if the Index should remain a separate piece. So the index is in a separate stack altogether.

I also put the Help Samples in a separate stack. Help Samples contains an assortment of cards used in the Introduction to HyperCard. These also could have been incorporated into the main Help stack, but I kept them out because they were such an odd assortment of different looks that it seemed as if they'd intrude in the flip book design of the rest of Help (see Figure 7.11.) Also, Help Samples contains a small tutorial used to show how searching works. I wanted to limit that searching to just a few cards. Putting

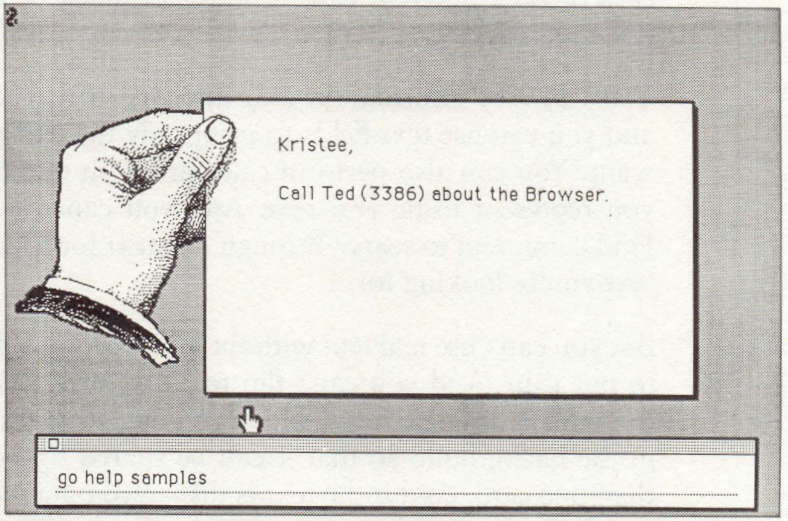


FIGURE 7–11 *One of the cards in the Help Samples stack. Help Samples is a separate stack because it looks different from the Notebook design used in Help.*

them in the separate Help Samples stack accomplished that.

Real Text and Paint Text

Why include two ways to produce text in the same application? Although both kinds of text in HyperCard use the same fonts and produce text that looks the same, the two kinds of text are very different. Text you create by creating a field with the Field tool, clicking inside the field using the Browse tool, and typing is "structured" or "real" text. This text is much easier to boss around than Paint text. You can select it just as you would select any Macintosh text, and cut, copy, or paste it into any field on the same

card or on a different card in the same stack or in a different stack.

You can sort information that appears in real text, and you can use text fields to print only the text you want. You can also perform calculation on numbers you represent using real text. And you can use the Find command to search through real text for specific text you're looking for.

But you can't use real text without first creating a field to put it in. And you can't flip real text over, invert it, or fill it with patterns. Nor can you put real text in the background so that it can be shared by more than one card. And if the text is displayed on a Macintosh that has different fonts in the System file, the text won't look like what you put in there because the Macintosh will substitute an available font for any that are missing.

Fancy Fonts and Freedom from the System File

Paint text gives you a lot of freedom. With Paint text it doesn't matter whether the font you originally used is available later or not. Although you do need the fonts when you type in the Paint text, once you click anywhere else or move to another card, the characters are just bits, stored not as text but as a picture. In the Help system I used mainly the New York and Geneva fonts because they're available to everyone. When I wanted to use a fancier or less available font I used Paint text instead of real text. Figure 7.12 shows some of the possibilities for Paint text.

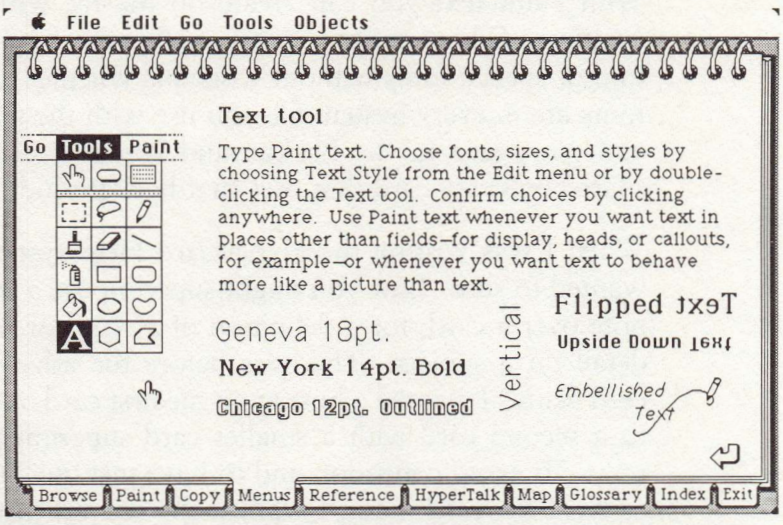


FIGURE 7–12 *Paint text gives you a lot more flexibility in fonts and styles you can use.*

Shared Text

You can also use Paint text when you want text to be shared by many cards. For example, the text used in the small business example in Chapters 4 and 6 of this book uses Paint text for the name and address of the company. Putting that Paint text in the background makes it appear on each card, which is used as a bill to send to clients. If you used real text for the name and address of the business, you'd have to cut it and paste it to each card you wanted it to appear on. Therefore, sometimes you don't want real HyperCard text, but rather the more freeform Paint text you created with the Paint text tool.

With real text you can edit, search, sort on, and change font, size, and style.

With Paint text you can create on the fly, without creating a field to contain it; invert, flip, etc.; use custom or special fonts without worrying whether those fonts are in every System file you use with the stack; and place text in the Background where it can be shared by every card that uses that background.

When I was writing the HyperCard Help system, I wanted to show how you might superimpose a small note over a card, for a definition of a word or more detail on a subject. (This was before the advent of card fields.) I wanted a button on the first card to lead to a second card with a smaller card superimposed on it. To avoid confusion and to have just one "live" copy of the card's text, I wanted the second card to use MacPaint text rather than the real text.

I made a screen shot of the original card and pasted it onto a second card, where I also added the small card overlay. Pretty clever, I thought. Just to check what I'd done, I even went to the original card and used the Text tool to select the text there, confirming that it was real text. Then I went to the second card and tried selecting the text there using the Text tool. The beep confirmed that this was Paint text.

Several days later, I started to write out a "bug report" describing an intriguing bug. Whenever I moved the text field on my original card, a "shadow" of the original text stayed behind. But of course it wasn't a HyperCard bug. I had pasted the snapshot, with its Paint text, right under the original text. The card had both the real text and a picture of the text. The real text was duplicated exactly by the Paint text and, until I moved the text field, it wasn't visible because the two kinds of text were in the same place. (When I had checked my work by selecting the text

with the Text tool, it had selected the real text; if I had tried selecting it using the selection rectangle, it would have selected the Paint version.)

By the way, don't automatically discard the possibility of using this "archaic" method of showing a "pop-up" field by creating and going to a second card that mimics the original with the addition of a box of additional text. Although real card fields as pop-up fields are great, you can't use any visual effects when you display them, nor can you include pictures in a pop-up field. Sometimes it's worth creating an additional card to get those effects.

Lining Things Up

It might not seem important whether a picture on one card lines up exactly with a picture on another—not until you move between the cards and watch the picture jump around on the page. If the pictures are even one pixel askew, the effect can be jarring when you move between the cards.

For example, you might want some Paint text to appear in a box on a number of cards. You haven't put the text in the Background picture, because it appears only on a few cards. Instead, you're going to add it to the card picture on just the cards that need it.

If you copy the text from one card and then paste it onto subsequent cards, it will automatically be placed in the same spot you copied it from, so it will line up without your having to do anything. There are also a number of other ways to keep Paint text in line, as seen below.

Lining up Paint Text

One of the problems with using Paint text rather than real text is that it's difficult to get Paint text lined up with other text on the card you've already typed—either real text or Paint text. Here's a trick to help.

1. Type the added Paint text.

2. Select it by pressing Command-S.

3. Make it transparent by choosing Transparent (or just typing "T" if you have Power Keys checked in the Options menu).

4. Move the selection over the existing text until it lines up with what you want to line it up with.

5. Hold down the Shift key while you drag the selected text to a new location. Holding the Shift key down while you move the text constrains it to the same horizontal or vertical plane so that it will line up neatly with the existing text.

Fixing mistakes

There's another Paint text trick that also comes in handy: If you want to add something to a line of Paint text or correct an error at the end or the beginning of a line, follow this procedure:

1. Use the eraser to get rid of the error.

2. Retype the addition, including one or two characters that you're not replacing.

Figure 7.13 shows the error erased and the new text ready to drag over.

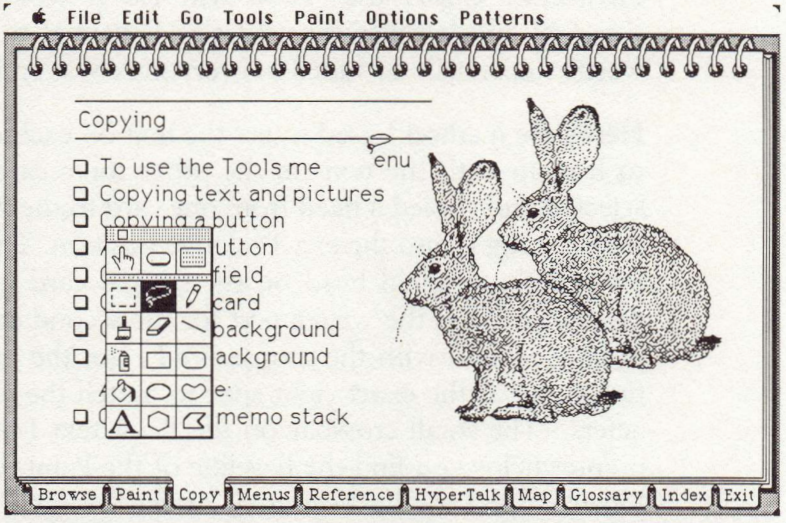


FIGURE 7–13 *Fixing a Paint text mistake*

3. Make the text transparent and drag it over the original text until the duplicated characters match exactly. The new text now lines up exactly with the old.

Lining up heads

Sometimes you want to match the position of heads that are different in content but appear in the same place on more than one card. This happens automatically with real text, for the fields are in exactly the same position on each card and so the text within the fields lines up. But when you're using Paint text, the fields appear wherever you click, making it very difficult to get them in the same position on more than one card. For example, each of the topic cards in the HyperCard Help system (the cards you see when you press the tabs along the bottom of the screen such as

Browse or Copy) uses Paint text (so it won't be searched and because the text size I wanted to use wasn't necessarily included in everyone's System file).

Here's the method I used to get the text on each card to line up with the text on the other topic cards: I selected and copied a head from one card to the next, thus pasting it into the exact relative position. Then I retyped the original head on the second card (even though this was the wrong text for the second card), poking around with the mouse until I got the insertion point at the exact right spot to match the characters. (The small crossbar on the Paint text I-beam pointer helps you find the baseline of the Paint text.) I backspaced to the beginning of the heading and typed the correct text for that card. Although this sounds a little picky and difficult, it's really pretty easy after you've done it a few times. And it's amazing how good you can get at predicting where to click to get the insertion point in the right place.

If you really want to make it easy on yourself, use real text in a field. And let HyperCard worry about lining it up for you.

Combining Paint text and real text

You can't combine text styles within a single field of structured text. But you can use Paint text within a field. For example, if you wanted to show a single word in boldface, you could leave the amount of space required for the word by pressing the space bar a number of times, and then use Paint text with the style choice you want in the space left. You probably won't want to do this with text that's meant to be edited, because every time you change anything in the field, the Paint text won't word wrap and you'll find yourself constantly having to come back to it, select

it with the selection rectangle, and move it to where it should be.

Aligning Pictures You Cut and Paste

Pictures you cut and paste in HyperCard stay in the same relative position that they were cut or copied from. This means that lining things up from card to card is a snap. (You can also move the pasted selection to another part of the screen if you don't want it to be in the original position.)

Visual Effects

Because it's so easy to browse cards, you'll want to learn to move gracefully between them. How the screen looks while you're going from one card to another can greatly affect the "feel" of the stacks you create. Below are a few effects that work well:

• zoom out	Going from a topic in a list to another card
• iris close	Returning to the topic list
• zoom in	Returning to where the mouse currently is
• dissolve	Picture to text
• venetian blinds	A spiffy effect you notice most when there's a big contrast between the cards
• wipe up or down	Movement from the bottom up or top down
• checkerboard	Noticed most when there's a lot of contrast

- barn door open — Unfolds from the center, like curtains parting
- barn door close — Folds back up again

The visual effect always precedes a Go command in a script, as shown in these lines:

```
visual effect barn door open
Go Home
```

In general, use the same effect for parallel actions. Topics in a Table of Contents or other launching place should be uniform. For example, each topic on the HyperCard Help topic cards zooms out. This consistency helps the user feel more sure about where he or she is; it provides a familiar framework. Similarily, you might have a return button always zoom in on returning to a topic list.

Dissolves are good for leaving graphic images, especially for going from a fairly dark, dense picture to a lighter one or to a card of text. Use Wipe up or down for flipping pages. Use Barn door open and shut for opening pages from the center of the screen—especially if you're using a "two-page" layout that looks like an open book. Use special effects such as checkerboard and venetian blinds sparingly.

The visual effect command can also include special effects you add using the word "to" in the script:

```
visual effect zoom open to black
visual effect zoom open to white
visual effect zoom open to gray
visual effect zoom open to card
```

(to stay on the same card rather than using the next Go command to go to a different card.)

Follow any of these special effects with the usual Go command.

Animation

Because HyperCard can move among cards so fast, you can produce animation by displaying a set of cards in sequence, much like a movie cartoon. The down side of this technique is that you have to store a complete picture for each tiny change you make. But if you have plenty of disk space to spare, it works.

Here's the basic technique:

1. Draw a picture.
2. Copy the entire picture to another card by double-clicking the selection rectangle, choosing Copy Picture from the Edit menu, choosing New Card from the Edit menu, and choosing Paste Picture from the Edit menu.
3. Make a small change to the second picture. (The difference between the two images becomes the animation.)
4. Copy the altered picture and paste it onto another new card.
5. Modify this new image and continue in this same way until you have the entire sequence of images, each on a separate card.
6. If the cards aren't in the order you want the images displayed, use the Cut Card command and cut and paste the cards into the order you want them.
7. If you want to have the sequence back up the original image, copy and paste images from existing cards to new ones in reverse order.
8. Create a background button that goes to the next card to move among the images, adding any visual effects you want to happen along the way.

Creating animation by moving among cards eats storage space because each separate image you use has to store the entire card picture. Here are a couple of tricks that can save you some space:

Use a background picture to show the parts of the animation that are constant. For example, if you want to animate a car driving down the road, you might use the entire road scene as the background picture and store the pictures of the car in different locations on card pictures. Use the Go command to show the cards in order—giving the illusion that the car is moving down the road.

You can also create animation within a single card. Here's a script that uses screen coordinates (the horizontal and vertical position on the screen) to animate an object on the screen. When you click Frog Jump, HyperCard runs a script that makes a frog jump around the screen. See Figure 7.14.

```
on mouseUp
put the userLevel into saveLevel
set userLevel to 3
if the userLevel < 3 then exit mouseUp
choose lasso tool
click at 442,294 with commandKey-select the
¬frog
set dragSpeed to 400
play boing c e f g d c e
drag from 442,294 to 385,201
drag from 385,201 to 330,296
drag from 330,296 to 321,136
drag from 321,136 to 198,208
drag from 198,208 to 114,58
drag from 114,58 to 36,296
drag from 36,296 to 199,143
drag from 199,143 to 305,264
drag from 305,264 to 439,183
drag from 439,183 to 442,294
click at 0,0--remove the selection
doMenu "Revert"
```

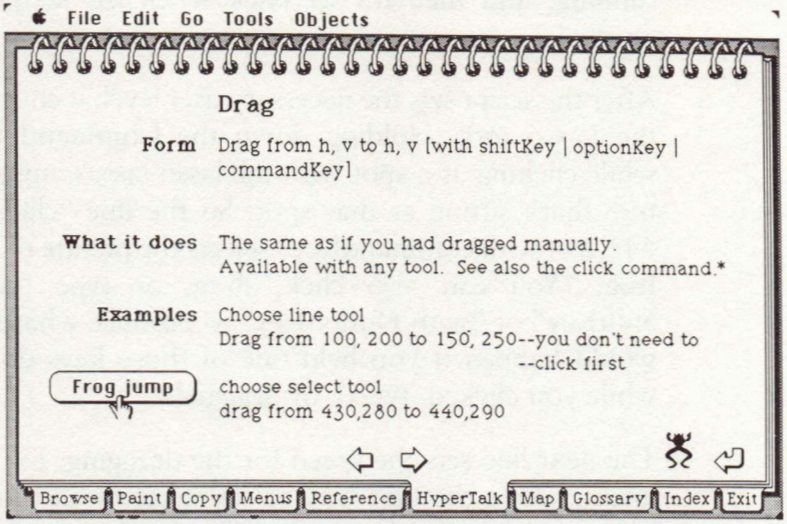


FIGURE 7–14 *The Frog jump button shows you how to animate. You can use the Button tool to open the button's script.*

```
set userLevel to saveLevel
choose browse tool
end mouseUp
```

In this script the user level must be at 3 or above to be able to use the Paint tools. This script temporarily sets the user level to 3—the first level at which the Painting tools are available—saving the "real" userLevel in the variable "saveLevel." If HyperCard isn't able to set the userLevel to 3 (if there's not enough memory, for example), the script is exited and the frog doesn't jump. "SaveLevel" is put back into the actual userLevel at the end of the script. In general, it's a good idea to put things back the way you found them in HyperCard. Temporarily changing the user level allows the script to work no matter what user level is set on the Preferences card of the Home stack. The user level is set to 3 just while the script is

running, and then it's set back when the script is finished.

After the script sets the necessary user level, it chooses the Lasso tool. Holding down the Command key while clicking at a spot with the lasso lassos any picture that's sitting at that spot. So the line "click at 442,294 with commandKey" selects the picture of the frog. (You can also click, drag, or type "with ShiftKey" or "with OptionKey" to produce whatever would happen if you held one of those keys down while you clicked, typed, or dragged.)

The next line sets the speed for the dragging. Experimentation is the best way to see what speed to set. The speed is in pixels per second for each drag statement. Between 200 and 800 is the usual range, but there's no actual limit to the speed you can set. Clicking at 0,0 removes the selection so that the following Revert command will revert the entire card. Now the user level is set back to the level saved at the beginning of the script and the script chooses the Browse tool.

Hint | Always put things back the way you found them.

Keeping Track of Buttons

While you're working on a stack, you can keep track of buttons under construction or use large background buttons to provide a default action.

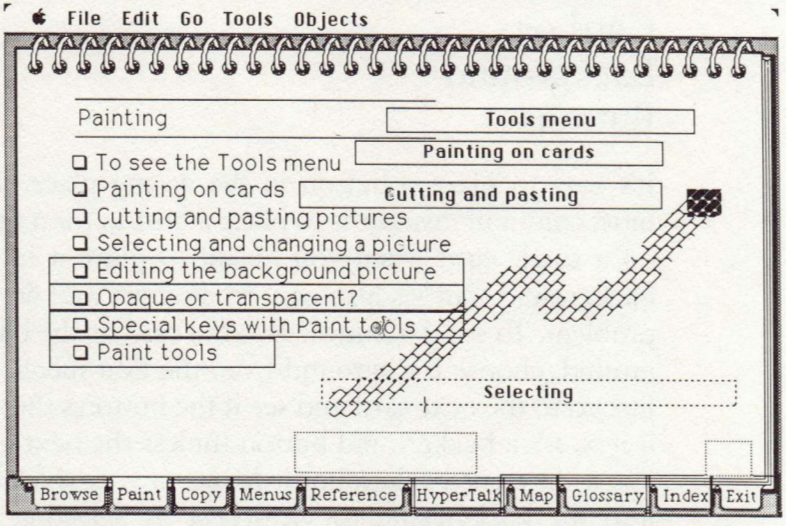


FIGURE 7–15 *Keeping track of buttons. The buttons are temporarily named to show where they're linked.*

Buttons Under Construction

Keep track of buttons under construction (or reconstruction) by naming them with their function and clicking Show Name. This keeps them visible and you can see what they do without opening their scripts. Later you can click Show Name again to make them not display their name. Figure 7.15 shows some buttons labeled with their function temporarily while the card is under construction. That way you can see what each button does without opening its script.

Card or Background Buttons

It's easy to place a button in the wrong place. The most common mistake is to place a button (or a field) on a single card when you meant to place it in the background. But it's also easy to discover and fix the problem. To see if a button is in the card or the background, choose Background from the Edit menu. Or just go to the next card and see if the button's there—if it is, it's a background button (unless the next card has a different background). Move any card button into the background or vice-versa by selecting the button with the Button tool, choosing Cut, choosing Background, and choosing Paste Button.

Creating Big Background Buttons

You might decide to use a convention that clicking near the outside edges of a card brings you Home. Make a large, invisible button that covers the entire card. (You can make it invisible by dragging the top left corner to "tuck it under" the menubar and then dragging the bottom right corner down and to the right.) Make this large button's destination the Home card. Select the button and choose Send Farther until the button is at the very back of all your buttons. Now whenever you click outside of all other buttons, the large button will take you Home.

Controlling Headers

If you're creating a stack with headings or chapter title, having the text for that element in its own field means you can change the characteristics of all the text in the field—every chapter name or every first-level head on every card that shares the background, for example—by selecting the field and changing it in just one place. Every chapter title or every first-level head that shares the background changes. This is quite different from using most word processors, where you have to separately select and change the characteristics of each separate head.

Keeping Your Stacks Private

Here's a way to disguise what you're doing, so if you keep HyperCard in a public place you can still have your privacy. HyperCard lets you protect stacks with passwords by choosing Protect Stack from the File menu and checking Private Access so that you can officially prevent access to any of your private stacks. But there's another way to keep your work from casual public view. You can "hide" stacks by making a screen shot (Command-Shift-3) of a spreadsheet or any other application. Make the spreadsheet be the Home card picture by copying the MacPaint document produced by Command-Shift-3 and pasting it onto the card. Whenever you go Home in HyperCard, you'll see the spreadsheet. You can then "hide" your buttons to various stacks in some of the spreadsheet's cells. Figure 7. 16 shows a picture of the Finder

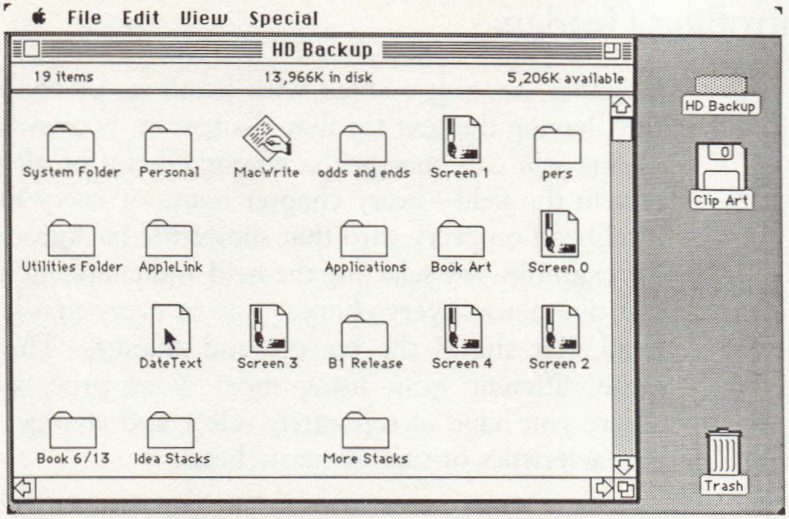


FIGURE 7–16 *Using a picture of the Finder to hide HyperCard. Each icon can be a button that takes you somewhere.*

being used as a Home card. Buttons in this "Finder" could lurk in the Finder's icons.

Using Pop-up Fields

Any background or card field can be alternately hidden and shown on a particular card to create the effect of a pop-up field. You can hide it or show it using the "hide" and "show" commands. Pop-up fields are great for adding a little bit of extra information (especially when you're out of space on the card, as I often was in the Help system, or when the extra information would detract from getting your main point across). You can also use pop-up fields to give alternative responses to actions the user takes, or to provide clues in an adventure game, for example. You

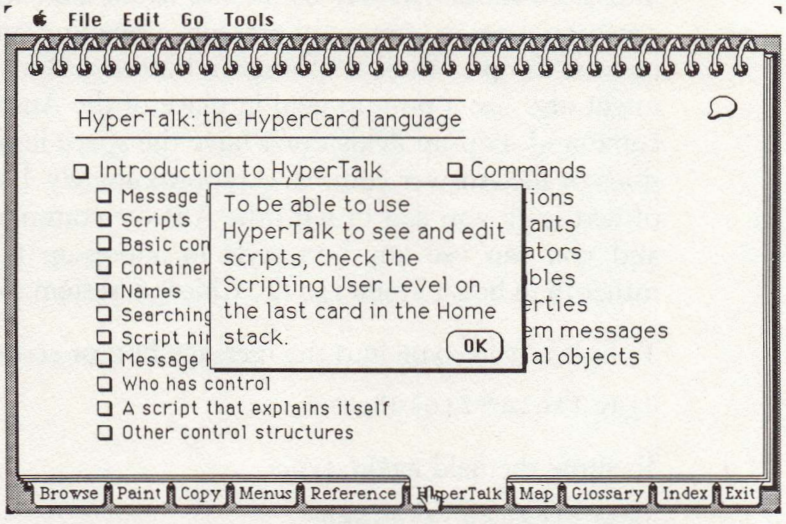


FIGURE 7–17 *A pop-up field you can hide or show*

can use pop-up fields as "About" boxes, telling the world who created this stack. Or put a footnote in a pop-up field.

You can show or hide a field depending on the userLevel set on the Preferences card or with the Protect Stack command. (HyperCard always keeps track of both the user level you set on the Preferences card and any that's set for the specific stack. It uses the lower of the two levels.) For example, I wanted to keep new users from accidentally wandering into the HyperTalk section of Help. I used the following script in the card that checks the current userLevel setting before displaying the card field (shown in Figure 7.17):

```
on openCard
   get the userlevel
   if it is < 5 then show card field 1
end openCard
```

You can use the Answer command as an alert message (the Answer command presents a one-line statement with up to three responses in buttons). But you might also use a pop-up field in place of the Answer command. Pop-up fields don't have the space limitations of the Answer command (approximately 1 line of text is all you can fit with the Answer command) and you can use any text style in a pop-up field, rather than being limited to the Chicago System font.

To hide a field, type into the message box or script:

```
Hide field "fieldName"
```

To show the field again, type:

```
Show field "fieldName"
```

Include the word "card" if it's a card field:

```
Show card field "fieldName"
```

Fields are shown until you hide them, and they're hidden until you show them. Remember that a background field has separate text on each card, so you can use a single background field for many cards' pop-up field. But sometimes you want the field to have a different look on different cards—a different font, border, size, etc. In this case it's better to use a card field. Card fields and background fields take about the same amount of space and are equally fast, but if you're using many identical card fields within the same stack, it would save you some space to use a background field instead.

In the Help system topic cards I used transparent buttons over Paint text for each topic. Most of the buttons take you to other cards in the Help system, but some, such as "To see the Tools menu" just show a pop-up field that explains about the subject.

This button's script shows the card field whenever you click the button:

```
on mouseUp
  show card field 1
  show button OK --"OK" is the button's name
end mouseUp
```

The OK button sits on top of the card field. Its button script says:

```
on mouseUp
  hide button OK
  hide card field 1
end mouseUp
```

To show a field when the pointer is positioned over a button (without clicking it) the button's script would say:

```
on mouseEnter
  show button OK
  show card field 1
end mouseEnter

on mouseLeave
  hide button OK
  hide card field 1
end mouseLeave
```

This script shows a field only while the mouse is still down (when you release the mouse button, the script goes away):

```
on mouseStillDown
  show button OK
  show card field 1
end mouseStillDown

on mouseUp
  hide button OK
  hide card field 1
end mouseUp
```

Sorting Tricks

In the example in Chapters 4 and 6 you used a separate field for the phone number. You could then sort your client stack by phone number by sorting by that field. Just add this line to the script:

```
sort ascending by field "phone number"
```

You don't have to have something in a separate field in order to sort by it. HyperCard lets you sort by any word in any line of any field. For example, you might sort an address list by the last word in the last line of the address field to sort by zip code. You could do the same thing with phone numbers or any other element of a list. You do need to keep these elements in a consistent order on cards—phone number always on line 5, for example—to have this work correctly.

Creating a Hidden Field to Sort By

You can always use Cut Card and Paste Card to move an entire card from one place to another. But if you're doing massive reorganizations of many cards in the stack, it's sometimes useful to create a temporary field to use just for sorting. Create the field, type numbers or letters into it in the order you want the cards to be in the stack, create a button with a script that sorts the cards by that field. You can copy the Sort button from the button collection and modify it to do exactly the right thing. When you've got the cards sorted, delete the field, or hide it by typing Hide field "fieldName" in the Message box. It will disappear until you tell HyperCard to show it using the Show command.

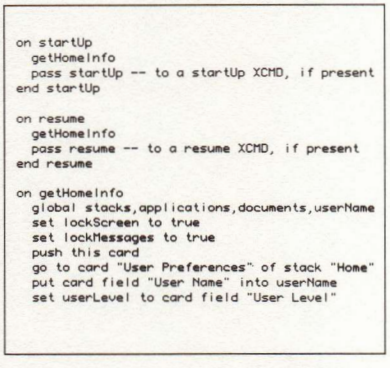
Summary

This chapter has shown you a bagful of tricks you can use when designing your own stacks. The next chapter will introduce and explain some useful scripts you can add as they are or modify to do something a little different.

Chapter 8

Hooked on Scripts

"Most programming languages come data-empty. You define or type in all of the data you want to manipulate. In Hypercard you start with thousands of words, cards, and pictures. And you can see them. Programming is much easier when you can see all of the data."
Ted Kaehler



```
on startUp
  getHomeInfo
  pass startUp -- to a startUp XCMD, if present
end startUp

on resume
  getHomeInfo
  pass resume -- to a resume XCMD, if present
end resume

on getHomeInfo
  global stacks,applications,documents,userName
  set lockScreen to true
  set lockMessages to true
  push this card
  go to card "User Preferences" of stack "Home"
  put card field "User Name" into userName
  set userLevel to card field "User Level"
```

This chapter is a compendium of useful scripts, with explanations of what they do and how you might modify them to do something slightly different. Reading an existing script is one of the best ways to learn how to write your own scripts. You can start by using the script

exactly as it is. Then you might modify it by changing details such as visual effects or text to search for.

Seeing how a script is put together can also be helpful in a general way by showing how commands and control structures work together to cause HyperCard actions.

Note: The on mouseUp/end mouseUp control structure is used in many of these examples, but these scripts could be in response to any message. Each message has a handler that begins with "on <message name>" and ends with "end<message name>."

Where to Put a Script

In general, place a message handler script where it will reach all of the objects that you want to be able to respond to the message it handles but no more. For example, if you want just one button to respond to the message mouseUp, put the script in that button. If you want some but not all of the buttons on a card to respond, duplicate the script for the buttons you want to respond. If you want clicking anywhere on the card to activate the handler, put the handler in the card's script, etc.

Inheritance

Messages are sent and received according to a set of inheritance rules. Every object that receives a message, either from the system or from another object, has a chance to respond to that message with a message handler. If the object doesn't have a handler for that message, the message is passed on to the next

larger object, which then has its chance to respond. The message is automatically passed through the entire path of objects in this same way. Each receiving object has its chance to respond.

A receiving object can also pass a message on to the next object even though it has already handled the message. To do this, the handler's script ends with the line:

```
pass <message>
```

The message (such as mouseUp) is then passed on to the next object as though it had never been caught.

You can also "break the rules" of inheritance using the send command to send a message to a particular object that might not be in the normal inheritance path:

```
send <message> to <object>
```

For example, you might say "send mouseUp to button 3" in the script for button 2. In this case, when the user clicked button 2, button 2 would catch it with an "on mouseUp" message handler, execute any script in the handler, and then pass the mouseUp message on to button 3, which might contain its own mouseUp handler. The basic difference between pass and send is that pass sends the message along the normal route and send breaks the inheritance rules by sending it to any object within the current stack.

You can also go to distant objects (objects on any card, background, or stack other than the current one) by using the Go command to go to the card whose buttons, fields, card, background, or stack you want to get the message. Once you're at the card you want, inheritance works just as it always does, using

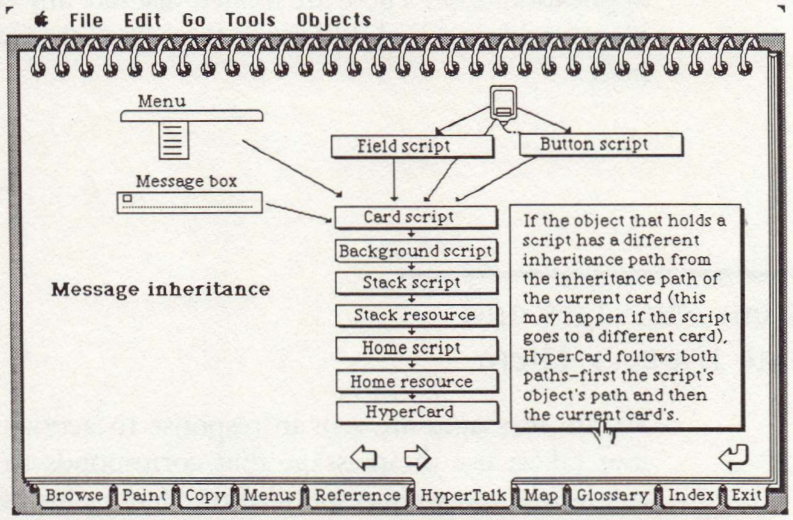


FIGURE 8–1 *The Inheritance path. All system messages are sent to the current card or an object on the card such as a button or field. If the receiving object doesn't have a handler for the message it is sent to the next larger object in the path, which then has a chance to respond to the message with a handler. Messages can also be sent among objects using the send or pass control structures, which send the message to a particular object or pass it on to the next object in the normal hierarchy.*

the path of the current card and then in addition checking the path of the object the original script is in.

Figure 8.1 shows the inheritance path of objects in HyperCard. The same inheritance is used for any message—including messages the system sends, such as mouseUp or openCard, functions, and messages you define yourself using other HyperTalk commands and functions. In addition, developers can define external commands by using another programming language, such as C or Pascal, to build a module that can be attached to any stack or to the HyperCard

application itself. These are treated just like any command in HyperCard, following the same inheritance path.

System Messages and What Triggers Them

System messages are sent in response to actions the user takes; use the message that corresponds to the conditions you want responded to. Here's a list of the system messages, which objects they're sent to initially, and when they're sent:

Message	Sent to	When
mouseDown	the frontmost object (field, button, or card) at the location clicked	when mouse button pressed
mouseStillDown	the frontmost object (field, button, or card) at the location clicked	continuously while the mouse button is being pressed
mouseUp	the frontmost object (field, button, or card) at the location clicked	when mouse button is pressed and released. The message is sent to a button or field only if the mouse button is both pressed and released over the object.

Message	Sent to	When
newButton	the newly created button	when created
deleteButton	the button being deleted	just before button is deleted
mouseEnter	the frontmost field or button under pointer	when the mouse pointer enters a button or field
mouseWithin	the frontmost field or button under pointer	continuously while the mouse pointer is over a button or field
mouseLeave	the frontmost field or button under pointer	when the mouse pointer enters a button or field
newField	the newly created field	when created
deleteField	the field being deleted	just before field is deleted
openField	the field being clicked in	just before it gets its insertion point
closeField	the field recently edited (sent only when there's a change)	when you click somewhere else or leave the card
newCard	the newly created card	when created
deleteCard	the card being deleted	just before card is deleted
openCard	the current card	when card becomes visible
closeCard	the current card	just before going to another card
startUp	the first card in the Home stack if application itself is started or the first card in any stack started directly from the Finder	when HyperCard is opened

Message	Sent to	When
idle	the current card	continuously while nothing else is happening
returnKey	the current card	when the Return key is pressed
enterKey	the current card	when the Enter key is pressed
tabKey	the current card	when the Tab key is pressed
arrowKeyLeft	the current card	when the left arrow key is pressed
arrowKeyRight	the current card	when the right arrow key is pressed
arrowKeyUp	the current card	when the up arrow key is pressed
arrowKeyDown	the current card	when the down arrow key is pressed
newBackground	the current card	when the background is created
deleteBackground	the current card	when only one card in a background, sent just before that card is deleted
openBackground	the current card	when a card of this background becomes visible after being in another background
closeBackground	the current card	just before going to a card with a different background
newStack	the first card in the new stack	when the stack is created
deleteStack	the current card	just before it's deleted
openStack	the first card in the stack	when the stack is opened

Message	Sent to	When
closeStack	the current card	just before going to a card in another stack or quitting HyperCard
help	the current card	when you choose Help from the Go menu or press Command-?
suspend	the current card	when you open another application from HyperCard using the Open command
resume	the card you return to (the one you left) after opening another application	when you return to HyperCard
quit	the current card	when you quit HyperCard by choosing Quit from the File menu or pressing Command-Q

A Moment in the Life of HyperTalk

Here is an imaginary stream of consciousness of what the HyperTalk interpreter does while you're looking at a card on the screen:

Send idle to the current card, send idle to the current card, send idle to the current card, . . . Oh, the mouse moved. Did it move over a button or field? It did, so

send mouseEnter to that button. Send mouseWithin, send idle, send mouseWithin, send idle, . . . Did the mouse button go down? It did, so send mouseDown to the button. Send mouseStillDown, send mouseStillDown, . . . Oh, the button went up! Send mouseUp to the button. Ah, the button has a handler for mouseUp in its script. Let's see, what do these commands say to do. . . ? Go to stack "Address." That means we're leaving this stack. Send closeCard, send close Background, send closeStack. Look for that other stack. What folder is it in, anyway? OK, bring in the first card, send openStack, send openBackground, send openCard. What's the next statement in that script we were executing? Ah, the mouseUp script is over. Is the pointer over any buttons or fields in the new card? It's not, so send idle, send idle, send idle, . . .

For each of the messages it sent, HyperTalk searched the entire inheritance chain to see if any of the scripts had a handler for that message.

Scripts You Can Use

The rest of this chapter contains sample scripts to try in your own stacks. Each script is presented in full with some comments inserted within the script (all comments follow "--") and an explanation following the script. When the explanation is more extensive, specific lines of the script are repeated with an explanation of that line following.

Choosing a Command

You can do just about anything you can do manually in HyperCard from a script as well. For example, you can choose any command from a HyperCard menu using the HyperTalk command DoMenu.

```
on mouseUp
   doMenu New Card   --to add a card following the current card
end mouseUp
```

When HyperCard encounters two hyphens in a row, it ignores the rest of the line. This is how you insert comments into your scripts, to help you or others who read the script remember what the script is doing. Including comments in your scripts makes it much easier for other people to interpret and modify them for their own use.

Choosing a Tool

You can also choose a tool from within a script. The drag command in this script moves the button from the center of button 1 (loc is the center horizontal and vertical screen coordinates) to a new location.

```
on mouseUp
   choose button tool
   drag from the loc of button 1 to 200,250
end mouseUp
```

"the loc" is the center of the button. Button 1 could be adjusted to any button name or number. "299,250" is its new horizontal and vertical screen coordinate position.

Hint

An item is a group of characters, including spaces, separated by commas. It is larger than a character and can be larger than a word, but smaller than a line.

```
on mouseUp
   choose field tool   --work on the field itself, not the text
   click at loc of field 1   --select the field
   doMenu Text Style...
   choose Browse tool
end mouseUp
```

DoMenu Text Style displays the Text dialog. Make sure there really is a background field so this script can work on it. Making choices in the dialog box is one thing you can't do from a script. You can set the properties of a field, however, using the form "Set <property> of <field>." Add some text to field 1 to see this script change its font. For example:

```
on mouseUp
   choose Browse tool   --edit text
   click at the loc of field 1   --click in center of field
   set textFont of field 1 to Chicago
   set textSize of field 1 to 12
end mouseUp
```

In the following script, the contents of field 1 is automatically typed into field 2 (First, create field 2 so there's someplace for the text to go.):

```
on mouseUp
   choose Browse tool
   click at loc of field 2   --get an insertion point in the middle line of
                             --field 2
   type field 1   --as if you typed what is in field 1 into field 2
end mouseUp
```

And this script draws rectangles:

```
on mouseUp
   choose rect tool   --for making rectangles
   set filled to true  --change a property-they will be filled with
                       --pattern
   set pattern to 12  --patterns are numbered from 1 to 12 starting at
                      --the top left corner of the Pattern window
   drag from 100,150 to 200,250   --draw the rect from top left to bottom
                                  --right
end mouseUp
```

Every tool has an official name. "Rect" is the rectangle tool. And every pattern has a number, starting from the top left of the pattern window, going across the first row and similarly left to right across each row.

Tools include:

browse tool
button tool
field tool
select tool
lasso tool
pencil tool
brush tool
eraser tool
line tool
spray tool
rectangle tool
round rect tool
bucket tool
oval tool
curve tool
text tool
regular polygon tool
polygon tool

Searching a Script for Text

Here's a script that searches for text within scripts—something you might do to change all occurrences of

an object's name to a different name you've given it, for example.

The Get command puts whatever follows into the local variable "it." "Pattern" is a variable that represents the text you're looking for. It's the first argument in this message. Whenever there's more than one argument, you separate them by commas (no spaces required) so HyperTalk can tell where the first argument ends and the second argument begins. By the way, the things in a list that are separated by commas are items, just as the horizontal and vertical coordinates in the examples above.

This script is included in the HyperCard Home stack, so it's available anywhere in HyperCard as long as you're using the original Home stack. (Messages are sent to the Home stack after they've made the route through the current card, background, and stack.)

```
on searchScript pattern, stackName
  --search all scripts of a stack
    if stackName is not empty then go to stackName

    get script of stack
    if it contains pattern then edit script of stack   --start the script
         --editor if the pattern is contained in this script

    repeat with i = 1 to the number of bkgnds   --cycle through the
         backgrounds
     go to card 1 of bkgnd i   --show a card of this background
     get script of bkgnd
     if it contains pattern then edit script of bkgnd

     repeat with j = 1 to the number of bkgnd buttons   --all background
                                                        --buttons
        get script of bkgnd button j
        if it contains pattern
        then edit script of bkgnd button j
     end repeat
```

```
    repeat with j = 1 to the number of bkgnd fields  --all background
                                                     --fields
       get script of bkgnd field j
       if it contains pattern then edit script of bkgnd field j
    end repeat

 end repeat  --finished with the backgrounds

   repeat with i = 1 to the number of cards  --go through all cards
                                             --individually
    go card i
    get script of this card
    if it contains pattern then edit script of this card

    repeat with j = 1 to the number of card buttons  --the buttons on
                                                     this
                                                     --card
       get script of card button j
       if it contains pattern then edit script of card button j
    end repeat

    repeat with j = 1 to the number of card fields  --the card specific
                                                    --fields
       get script of card field j
       if it contains pattern then edit script of card field j
    end repeat

   end repeat  --finished with the cards
end searchScript
```

Whatever piece of text you supply as the first argument is stored in the local variable "pattern" and the program searches all scripts for that text. If you leave off the stack name, the script searches scripts in the current stack. "Edit script" is a command that opens an object's script so that you can edit it. "Repeat with i = 1 to the number of bkgnds" is a control structure that steps through each of the backgrounds. This process is known as a "loop." It temporarily assigns the number of each background in turn to "i," so you can refer to the background within that part of the script.

To execute the SearchScript script from the Message box, you might type:

```
searchScript "button", Home
```

When you press the Return or Enter key, HyperCard goes through each script within the Home stack, stopping to open any script that contains the word "button." Once in a script, you can type Command-F to see a dialog box where you can specify the text you are looking for within the script. In this case you'd type "button" and then press the Return key to search the current script. When you click the OK button, HyperCard continues looking through subsequent scripts, opening in turn each script that contains the word "button." Type Command-period to stop searching.

Finding Backgrounds

If you're doing a project that has many backgrounds, you often want to go to each background to make the same change everywhere. For example, in the Help system, whenever I changed my mind about what should be on the tabs along the bottom of the cards, I wanted to make that change in each of the 20 or so backgrounds' background picture. I put this simple message handler into the stack script of my Home stack so that typing "bk" into the Message box would take me to the next background in the stack.

```
on bk
   go to first card of next background
end bk
```

You could also add another line to save yourself some more work:

```
on bk
   go to first card of next background
   doMenu Background
end bk
```

This would not only take you to the next background, but choose Background from the Edit menu so that you could work on the background.

Merging Stacks

There are times when you want to consolidate two stacks into one. For example, you might want to add your office Address stack to one you keep at home. You can do this "manually" by cutting and pasting text or cards one by one. Or you could merge the stacks using this script. Once the stacks are merged together, you can delete any duplicate cards and re-sort them using one of the Sort buttons. This script is based on one by Mike Farr. No links from other cards to the cards that are moved are carried with them when they move, so don't use this script to move individually linked cards.

```
on mouseUp
   global ToStack   --global variable for name of destination stack
   global FromStack   --global variable for name source stack
   put "Merge appends a copy of one stack to the end of another."
   ask "Copy cards from stack:" with "<stack name here>"
   put it into FromStack
   ask "Append cards to end of stack:" with "<stack name here>"
   put it into ToStack

   answer "Append stack" & quote & FromStack & quote & "to" & quote ¬
   & ToStack & quote & "?" with "Yes, Merge" or "Cancel"
   if it = "Cancel" then exit mouseUp   --give the user a chance to bail
                                        --out
```

```
   go to stack FromStack
   repeat for the number of cards   --stop at last card of FromStack
      doMenu "copy card"
      go next card   --go to next card to copy
      put the long ID of this card into other   --remember it
      go to last card of stack ToStack   --paste card at end of stack
      doMenu "paste card"
      go other   --go to next card to copy
   end repeat

   go to stack ToStack   --go to beginning of ToStack
   put "Merge Complete! You're at the beginning of stack" & ToStack

end mouseUp
```

The first portion of the script asks the user the name of the two stacks to move cards between. The middle portion shuttles back and forth between the stacks, copying a card in one and pasting it into the other. The last portion goes to the completed stack and tells the user the program is finished.

The global command makes a variable accessible from anywhere in HyperCard. In this case it's being used to make the access to the names of the stacks faster.

Let's examine the middle of the script. The current card is in the "From" stack.

```
repeat for the number of cards   --stop at last card of FromStack
   doMenu "copy card"
   go next card   --go to next card to copy
   put the long ID of this card into other   --remember it
   go to last card of stack ToStack   --paste card at end of stack
   doMenu "paste card"
   go other   --go to next card to copy
end repeat
```

The statements are repeated for as many times as there are cards to move. Each time through, copy a

card from the "From" stack, go to the next card. Get the complete name of this card (by ID number) and put it into the local variable "other." Go over to the end of the stack we are adding onto. Paste in the card and then go to the card whose name we remembered (back in the first stack). This is the next card to copy, and when the repeat loop goes back to the top again, the first statement copies it.

Displaying Windows on a Large Screen

If you have a larger than normal screen, you may want to automatically place the message window and tool window outside the normal boundaries of the HyperCard window. The message "startUp" is sent to the current card when HyperCard starts up. You can place the tool window where you want on the screen. The first number (112) is the number of screen dots from the left side (the horizontal coordinate), and the second number is how far down (the vertical coordinate). Don't use this script unless you have a large screen. You could lose the Message box off the bottom of your screen.

Add these lines to both the startup and resume message handlers in the stack script of the Home stack:

```
on startUp
   set loc of card window to 112,50  HyperCard uses absolute positions
                                     --for the card window
   set loc of msg to 0,358   --these positions are relative to the card
                             --window
   set loc of tool window to -88,39
   show tool window
```

```
   get HomeInfo
end startUp

   on resume
   set loc of card window to 112,50
   set loc of msg to 0,358
   set loc of tool window to -88,39
   show tool window
   get HomeInfo
end resume
```

To substitute horizontal and vertical screen coordinates that you like better, place the windows at the spot you want them by dragging them there and type "the loc of msg" into the message box. Press the Return key to see the current location in the message window. Copy that number and use it in the script for the message window (msg). Do the same for the other two windows.

A Spreadsheet

The HyperCalc stack is a mini-spreadsheet. It's included here because it's an elegant example of the kind of spare, clean code you can write in HyperTalk. This script is executed whenever any information in any of the fields on the card changes. Dan Winkler, the author of HyperTalk, wrote this script.

```
on closeField
   repeat with i = 1 to the number of fields
      get line 3 of field i--the formula
      if it is not empty then put the value of it into line 2 of field i
    end repeat
end closeField
```

The Target

In this example, a card contains many buttons. The buttons themselves don't have scripts. Instead the card script relies on the button names to go to the right stack. (The names are showing.) None of the buttons has a handler in it. If you click on the button, mouseUp is sent to the button. But, because the button doesn't handle the message, it is sent to the card that has this handler:

```
on mouseUp
   if the target contains "button"
   then go to stack the short name of the target
end mouseUp
```

"The target" is the name of the object that originally received the message. For example, button "Address." The first line asks if the target is a button (it could be a field or this card). If it is a button, get the name of the target ("short" means without the word "button"); the Get command puts it into it. Go to that stack: the Address stack, in this case.

If the target does not contain the word "button," don't do the "then" part; just leave the mouseUp handler.

Push and Pop Card

We just saw how to get a "computed go." There is another way. Often you want to make a note of where you came from so that you can go back there later. In the Address stack, for example, the return

arrow takes you back to where you were when you entered the stack. In the stack script:

```
on openStack
   push recent card  --save the card you were just on
end openStack
```

In the return button's script:

```
on mouseUp
   pop card  --go to the last card we saved
end mouseUp
```

Push and pop use what is called a "push-down stack." Not the stack of cards that we know, but an analogy to what you find in cafeterias. The dinner plates are stored in a device that lets you put a lot of plates in, and then take the top plates out again.

"Push recent card" puts the name of the card you were just on at the top of the "stack" of remembered cards. "Pop card" removes the top card name and goes to that card.

See the Help system for a wrinkle on this. The on openStack message handler in the Help stack script decides if the recent card was inside or outside of the Help system, and saves it in a global variable.

```
on openStack
   global helpExit, crossRefDepth
   push recent card
   pop card into it
   if "help" is not in it then
      put it into helpExit
      put 0 into crossRefDepth
   end if
end openStack
```

The script in the Exit Help button uses the global variable as a destination:

```
on mouseUp
   global helpExit
   visual effect dissolve
   go to helpExit
end mouseUp
```

Finding Where a Message Is Defined

Put the following script in your Home stack. It allows you to find the scripts that define any mysterious message you find in another handler.

```
on def theMessage
   put "on" & & theMessage into pattern   --search for "on xxx" which
         --defines xxx; run up the inheritance chain, looking for it
   if script of this card contains pattern then   --start with the card's
                                                  --script
      edit script of this card
      exit def
   end if
if script of this bkgnd contains pattern then
   edit script of this bkgnd
   exit def
end if

if the script of this stack contains pattern then
   edit script of stack
   exit def
end if

set lockScreen to true   --check the Home script
push card   --remember where we are
go to stack home
if script of this stack contains pattern then
   set lockScreen to false   ---we'll look at it
   edit script of stack
   pop card
   exit def
end if
pop card   --back to original card
set lockScreen to false
```

```
  if theMessage is in "add subtract multiply divide go play put find" & &
  "open close read write convert visual effect do doMenu click drag" & &
  "type get set dial"
  then put theMessage && "is a HyperTalk command. See the Help system"
  else put "It is not accessible from this card (unless it is a XCMD)"
end def
```

Imagine you're in a script and you see a single command you don't understand. It was defined somewhere by the author of the stack, but you don't know where. Close the script and type:

```
def startup
```

Press Return. The Home stack script will find the script that defines the message.

An XCMD is an extension of the HyperCard language made by writing Pascal or "C" code, compiling it into a code resource, and installing it in a stack. Commands defined in this way can't be discovered by the "def" script.

Adding a Card

The "New" button in the File Index stack does some interesting things. It adds a new card and numbers it one more than the last card in the stack. It also puts a text insertion point in the first line of the main text field so that you can type the entry without clicking there.

```
on mouseUp
   go to last card
   get field "Index Number"   --serial number of last card
   doMenu "New Card"
   put it + 1 into field "Index Number"   --add 1 and insert in new card
```

```
   choose browse tool
   click at 100,100--put a text insertion point on first line
end mouseUp
```

"It + 1" is number one larger that the serial number of the previous card. We put that into field "Index Number" of the new card. The coordinates were determined by placing the pointer at the appropriate spot on the screen, typing "the mouse loc" into the Message box, and pressing Return.

If you want to get coordinates within a field, don't click in the field, just move the mouse over it.

Plots

The buttons in the plots stack (you can get there from the Home card) draw a bar graph, a stack of coins, or a pie chart from a list of numbers in a field. The script that follows is in the Coin button. There are two handlers. The main one sends a message to ask the other handler to compute the largest number in the list. "What" is a parameter (or argument). It is a value passed to the handler so it can be used. In this case the variable called "what" contains the text "field Data."

```
on mouseUp
   put the userLevel into saveLevel
   if the userLevel < 3 then set userLevel to 3  --Painting level
   if the userLevel < 3 then exit mouseUp  --stack is protected or memory
                                           --is low
   clearScreen
   reset paint  --to be sure of starting conditions
   choose oval tool
   set lineSize to 2
   set filled to true
   set cursor to 4  --the watchCursor
```

```
put frameLeft() into horiz   --which pile to start on
put maxLine(field "Data") into maxValue   --the tallest pile
put the number of lines in field "Data" into coins
if coins > 10 then
   answer "Only the first 10 data items can be plotted." with "OK"
   put 10 into coins
end if

repeat with i = 1 to coins
   get line i of field "data"
   if it is not empty then
      put round(frameHeight()/8.2 * it/maxValue) into coinCount
      set pattern to 13
      drawCoins horiz, frameBottom()-2, coinCount
   end if
   add 40 to horiz
end repeat

reset paint
choose browse tool
set userLevel to saveLevel
end mouseUp

on drawCoins left, bottom, howMany
   put bottom into y
   repeat with i = 1 to howMany
      subtract 8 from y
      drag from left,y to left+30,y+10
      play "harpsichord" c7s   --this is an added line that makes a
                               --clicking sound as the coins pile up
   end repeat
end drawCoins
```

This is the kind of program one normally thinks of when one hears the term "computer program." HyperCard can handle these too, and even makes them easy to read. The main program, "mouseUp," has a section for setting up the Paint tools, a set up section for that data, and a repeat loop that calls a subroutine to actually draw the coins. The subroutine, "drawCoins," can be in any script at this level or above in the inheritance. It has three input parameters, "left, bottom, howMany" that tell where to draw the coins. The main program calls several "functions," such as frameLeft() and maxLine().

These are handlers that return a value when they are done. The value goes right into the expression that was being computed when the function was called. Functions can be defined by the user, and frameLeft() and maxLine() appear in the background and stack scripts of the Plots stack.

Playing a Tune

Sound adds an interesting new dimension to your stacks. The harpsichord and boing sounds are built into HyperCard, available to any script. You can use other sounds as long as you include their resources in the stack that uses them. This script plays the Big Ben chime when you click.

```
on mouseUp
   play "Boing" tempo 200 e c d g3h. gq d4 e cw   --play the Big Ben chime
end mouseUp
```

on mouseUp and end mouseUp make up the message handler. "Play" is the command. "Boing" is the name of the sound to play. Tempo is an optional setting. "Normal" tempo is 200; higher numbers are faster. The comment at the end of the line tells you what this command will do.

The letters each represent a note on the musical scale. Octave 4 (middle C's octave) is assumed if none is specified. This tune starts at the E above middle C. It goes down to C, up to D, then down to the G below middle C, which it holds for a dotted half note. All notes are the same length as the previous note unless otherwise specified. If you don't specify any value in

the script, HyperCard uses quarter notes. Here are the other values:

h = half note
w = whole note
q = quarter note
e = eighth note
s = sixteenth note
t = thirty-second note
x = sixty-fourth note

A period means make the preceding note dotted (1.5 times as long). The following notes in the command continue to use a duration setting until you change it by including a new duration after the note you want to be of a different value. Changing octaves works the same way.

There are several examples of the Play command in the Help system. The music example in the Introduction to HyperCard contains a script you can use to build a keyboard that places a corresponding note on a staff when you use the Browse tool to click any key.

The music example is a fairly simple one, limited to the key of C. (With a little work it could be expanded to include other keys). The example shows a keyboard and a staff. "Playing" a note on the keyboard produces a corresponding note on the staff. One of the time-consuming tasks in learning to play any musical instrument is learning this correspondence. Another is learning to recognize the pitch of a sound that's played or a note on the staff. One of the first things I wanted to build in HyperCard was a musical example that would give you instant feedback so that you'd know if you'd identified the right note. Ted Kaehler wrote this script that shows some of the possibilities of what you can do with just a few commands.

```
on mouseUp
   get the clickloc
   put item 1 of it into noteNum
   put 1 + (noteNum - 52) DIV 13 into noteNum
   put "36 38 40 41 43 45 47 48 50 52 53 55 57 59 60 62" ¬
   && "64 65 67 69 71 72 74 76 77 79 81 83" into keymap
   put word noteNum of keymap into realNote
   play "harpsichord" realnote & "q"
   if noteNum < 15 then
      put 220-((noteNum-17)*3)-10 into noteLocY
   else
      put 204-((noteNume-17)*3)-10 into noteLocY
   end if
   show button Note at 317, noteLocY
end mouseUp
```

"Get the clickloc" returns the horizontal and vertical screen coordinates last clicked. It gives a position on the keyboard to be used in determining the notes "played." The values are in the form of items, separated by commas.

Because clickloc's answer or "result" is automatically placed in "it," a local variable used to store temporary results, you can now put "it" into another local variable—NoteNum.

Hint

A variable is a container, like a field. You can store a value in it. But unlike a field, a variable doesn't exist outside the script as an object. It's a place to put something while the script is running, to be forgotten when the script is done.

The next few lines of the script calculate the note that matches the place clicked on the picture, using the pixel width of each key to determine which note was clicked. The next line describes the intervals in the

key of C and puts the table into the variable "keymap." "Put word NoteNum of keyMap into realNote" pulls the correct note pitch out of the table based on the location you clicked on the keyboard. (HyperCard doesn't know the difference between text and numbers, so word "noteNum" may represent word 3, which in this case would actually be the number "40."

Now comes the "Play" command. Instead of a note value such as a, b, c, etc., the variable "realnote" tells HyperCard which note to play (the numbers in the table above are another way of specifying note pitches). The "&" is used to *concatenate* or string together the notes and the "q" that specifies quarter notes.

To put the correct note on the staff, this script uses a button with an icon that looks like a quarter note. (The resource for this button is in the Help Samples stack. You can use a resource editor such as ResEdit to edit or add new resources to any stack.) The horizontal location is fixed for all the notes at 317 (see the last line of the script), but the vertical location changes, depending on how high or low in the staff the note falls.

NoteNum is the number of the white key clicked. Lines "If NoteNum < 15. . ." through the "end if" statement calculate the position based on whether the note clicked is above or below Middle C using arithmetic I'd rather not talk about. Show button Note at 317,noteLocY moves the button on the screen.

Limiting the Scope of the Find Command

In the Help system I wanted to limit finding text in the Glossary to terms defined within the Glossary. The normal Find command searches the entire stack and looks in all fields. The glossary is just one background of the stack and the terms being defined are listed in field 1 of each card. There is a "find" button on each glossary card that prompts the user for a term and it does a restricted search. We want all searches after the user clicks the "find" button to be restricted. Any search that starts outside the glossary and happens to land in it should continue searching the whole stack when the user presses return. For example, if you were searching anywhere in Help for the word "button," HyperCard would find any instances of the word in the Glossary terms or definitions as well as any other place in Help. But if you clicked the "find" button in the glossary in order to search for the word "button," HyperCard would search only the first field in the Glossary background—the field that contains the terms. To make this happen, let's define a global variable "findInGlossary" that's true after the user clicked the find button and false if the user just happened into the glossary.

Add these lines to the background script:

```
on openBackground
   global findInGlossary
   put false into findInGlossary
   push recent card
end openBackground
```

These lines are in the Find button's script:

```
on mouseUp
   global findInGlossary
   put true into findInGlossary
   doMenu "Find..."
end mouseUp
```

When you click the "find" button in the glossary, set the variable to true and prompt the user for the term. Subsequent searches will look only in field 1 of the glossary, because of this script. To start the search, the user presses the Return key. That sends the message "returnKey" to the current card. This background script catches the message and modifies the find command:

```
on returnKey
   global findInGlossary
   if first word of msg is not "find" or findInGlossary <> true
   then pass returnKey   --do any other msg box command normally
   else send msg && " in field 1" to HyperCard  --just in title field
end returnKey
```

If the command in the message box is not a Find command, pass returnKey to let HyperCard do what it would normally do. If findInGlossary is false, also pass the command through unchanged. Else add "in field 1" after what is in the message box (Find "button") and send the whole command to HyperCard.

And finally, set "findInGlossary" back to false when you leave the glossary with these lines:

```
on closeBackground
   global findInGlossary
   put false into findInGlossary
end closeBackground
```

Generating Bar Codes

James Redfern wrote this script that generates bar codes using the Paint tools:

```
--Puts 3 of 9 bar code of a card's id on the card.
--Bar code will be in upper left corner of card, under the menubar.
--jfr

on code
   put "0,0" into origin   --location of upper left corner of bar code
   put 20 into height   --height of bar code
   put 1 into narrow   --width of smallest line
   put narrow * 2 into wide   --use for compressed ratio
   put narrow * 3 into wide   --use for preferred ratio
   put the short id of this card into info   --or anything to 31 chars

   put "QGNTDJ?4OLSBIPF296-+X•/$V ZKRAHOE185MC3UYW" into alpha
   put "000000111,000001101,000010011,000010110," into trans39
   put "000011001,000011100,000100101,000110001," after trans39
   put "000110100,001000011,001000110,001001001," after trans39
   put "001001100,001010010,001011000,001100001," after trans39
   put "001100100,001110000,010000101,010001010," after trans39
   put "010010001,010010100,010100010,010101000," after trans39
   put "011000001,011000100,011010000,100000011," after trans39
   put "100000110,100001001,100001100,100010010," after trans39
   put "100011000,100100001,100100100,100110000," after trans39
   put "101000010,101001000,101100000,110000001," after trans39
   put "110010000,111000000" after trans39
   put item 1 of origin into orgH
   put item 2 of origin into orgV
   put narrow * 7 + wide * 3 into codeWidth
   repeat with cardInx = number of this card to number of cards
      put the short id of this card into info   --or anything to 31 chars
      put "*" & info & "*" into info
      put number of chars in info into infoSiz
      put orgH + codeWidth into horz
      put (infoSiz + 2) * codeWidth into width
      choose select tool
      drag from origin to orgH + width*2 -1, orgV + height -1
      doMenu "Clear Picture"
      drag from origin to orgH + width -1, orgV + height -1
      doMenu "Opaque"
      choose line tool
      put true into black
```

```
      repeat with inx = 1 to infoSiz
         put item offset(char inx of info, alpha) of trans39¬
         & "0" into codeBuff
      repeat with bar = 1 to 10
         if char bar of codeBuff = "0" then
            set lineSize to narrow
            put narrow div 2 into toolTLCenter
            put (narrow +1) div 2 into toolBRCenter
         else
            set lineSize to wide
            put wide div 2 into toolTLCenter
            put (wide +1) div 2 into toolBRCenter
         end if
         if black
            then drag from horz + toolTLCenter, orgV + toolTLCenter ¬
            to horz + toolTLCenter, orgV + height - toolBRCenter
            add the lineSize to horz
            put not black into black
         end repeat
      end repeat
      go to next card
   end repeat
   choose browse tool
end code
```

This script adds a bar code to the card picture. Change the 0,0 at the beginning in order to change the location of the code on the card.

Change the repeat count to "number of cards" and un-comment the "put" and "go to" to apply the script to an entire stack of cards. Change what's put into the variable "info" to represent something other than the card's ID number.

This is the 3 of 9 bar code. Each character is represented by nine bars, of which three are wide, and the rest narrow. A bar can be white or black, but the first and ninth bars are always black. A narrow white bar separates contiguous characters. A wide bar is three times the width of a narrow bar, as a preferred ratio. If space is tight, a compressed ratio of one to two can be used. Up to 31 characters can be represented with

a single bar code. A special character begins and ends this sequence. Look under the menubar (Command-space) to see this.

This handler paints a bar code on a card's picture, clearing and masking the area of the picture where the bar code will go. This area will be a rectangle, with an upper left corner and height selected by the user. The width is determined by the number of characters to be coded, and the selected width of a narrow bar. If need be, the width can be made smaller by using the compressed ratio, but this may increase read errors.

Automatic Indexing

Ken Doyle has written a very nice "stack indexer." In a field on the first card of a stack, it creates a line for each card in the stack and puts the title of the card there. At one glance you see all the titles of the cards in the stack. Clicking on one takes you instantly to that card. Whenever you leave the stack after changing a title, adding a card, or deleting a card, the index is automatically updated. Figure 8.2 shows what the stack looks like—two fields and a button.

In the stack script, this handler creates the index. Make sure there's a background field named "title" with a title in each card and a card field named "index" on the first card.

```
on index
   --create an index of each card in this stack
   global changed
   put "" into card field "index" of card 1
   put the number of cards into cards
```

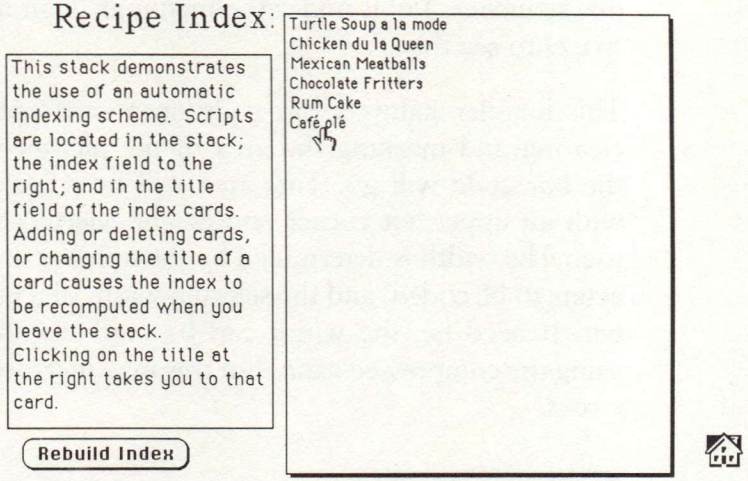


FIGURE 8–2 *The first card in a stack that automatically indexes*

```
   repeat with cardNum = 2 to cards
      put line 1 of field "title" of card cardNum into ¬
      line cardNum - 1 of card field "index" of card 1
   end repeat
   put false into changed
end index
```

The repeat loop spins through the cards, putting the title field of each into the corresponding line of the index field on card 1.

The field that contains the index is locked, so that when the user clicks on it, the field's mouseUp handler gets a chance to run (if it weren't locked, you'd get an insertion point for typing):

```
on mouseUp
   put the textheight of card field "index" into
height
   put the rect of card field "index" into rect
```

```
    put item 2 of rect into lineLoc
    put item 2 of the clickLoc into clickY
    put 1 into cardNum
    repeat while clickY > lineLoc
       add height to lineLoc
       add 1 to cardNum
    end repeat

    if cardNum > 0 then
       visual effect iris open
       go card cardNum
    end if
end mouseUp
```

This script computes which line you clicked on and goes to that card. Now all we have to do is figure out when to update the index (by running the "index" script). The global variable "changed" will be true when the user has done something to modify one or more cards in the stack. It starts off false:

```
on openStack
    global changed
    put false into changed
    push recent card
    hide message box
end openStack
```

And gets set to true when you change something:

```
on closeField
    global changed
    put true into changed
end closeField

on deleteCard
    global changed
    put true into changed
end deleteCard

on newCard
    global changed
    put true into changed
end newCard
```

When you leave the stack, decide if it was changed this time. If so, recompute the index. The field "alert" has text that tells the user to wait while the index is rebuilt. It is normally hidden.

```
on closeStack
   global changed
   if changed = true then
      go first card
      index  --actually do the work
   end if
end closeStack
```

And there is a background button named "Rebuild Index" that lets you manually order the index to be rebuilt.

```
on mouseUp
   index  --rebuild it
end mouseUp
```

A Pictorial Index

The following script, written by James Redfern, creates a picture index using the miniature form of a card generated by pasting an entire card while holding down the Shift key:

```
  --This makes a miniature picture of every card in a stack, and places
  --them on a newly created card (and bkgnd), which will then be the
  --second card. Each miniature on this new card will be a button that
  --will go to the corresponding card.
  --Only the first 49 cards will be indexed, but this can be extended
  --by changing the script to create multiple cards.
  --Note: the screen is 512 by 342, and the miniture is 64 by 42 pixels.
  --jfr

on picIndex
  put "256,171" into center
  put 61 into col   --64=2 + 29, half miniture width plus border
```

```
  put 43 into row   --42=2 + 22, half miniture height plus border
  go to card 1
  set lockScreen to true
  doMenu "New Background"
  set name of this background to "Not This Bkgnd"
  set name of this card to "Put Them Here"
  go to card 1
  repeat with cardInx = 1 to number of cards
    push next card
    if short name of this bkgnd is not "Not This Bkgnd" then
    choose browse tool   --avoid btn outlines in miniature
    set lockScreen to false   --must be to get miniature
    doMenu "Copy Card"
    set lockScreen to true
    put "on mouseUp" & return & "go to" into goToScript
    put id of this card & return & "end mouseUp" after
    goToScript
    go to card "Put Them Here"
    type "v" with commandKey, shiftKey   --this pastes a miniature card
                                         --pic
    drag from center to col,row
    choose btn tool   --Objects menu must be present
    doMenu "New Button"
    put id of card btn (number of card buttons) into newBtnID
    set rect of card btn id newBtnID to ¬
    col-32,row-21,col+32,row+21
    set style of card btn id newBtnID to transparent
    set showName of card btn id newBtnID to false
    set name of card btn id newBtnID to empty
    set autoHilite of card btn id newBtnID to true
    set script of card btn id newBtnID to goToScript
    add 43 to row   --42 + 1, miniture height plus gap
      if row > = 321 then   --342 - 21
      put 43 into row   --42=2 + 22, half miniture height plus border
        add 65 to col   --64 + 1, miniture width plus gap
        if col > = 480 then exit picIndex   --512 - 32
      end if
    end if
    pop card   --go to the next card
  end repeat
  choose browse tool
  go to card "Put Them Here"
  set lockScreen to false
end picindex
```

To carry out this script, type "picIndex" and press Return or place the script in a button you create.

Summary

This chapter has included scripts contained in HyperCard and scripts contributed by early HyperCard hackers. You can use these scripts exactly as they are or modify them to do something slightly different. Modifying scripts is the best way to learn your way around HyperTalk.

Chapter 9

"We've witnessed the transition into another era. This is the HyperCard era."
Lee Felsenstein

Talking to the Rest of the World



There's a world of connected information within HyperCard, but there's also an outside world you might sometimes want to communicate with—using desk accessories or sending or receiving information from word processors, graphics programs, or other HyperCard files on a network.

The Scrapbook and Notepad

The Scrapbook desk accessory is a handy place for pictures or text you use often. It can't store buttons or fields, but the Scrapbook is still a great help in juggling the odd pieces of information you need to store temporarily, and for storing the pictures or text you use frequently and don't want to chase down in various stacks.

The Scrapbook can accept any picture from HyperCard. Even though HyperCard uses a different format than the Scrapbook uses, its pictures can be stored in the Pict format the Scrapbook uses. HyperCard pictures can subsequently be pasted into any application that can accept the Pict format. Occasionally a HyperCard picture may be too large to be stored in the Scrapbook, which is limited to the size of a MacPaint window.

Unlike MacPaint, which plops a pasted picture into the center of the screen, HyperCard pictures are pasted into the exact spot they were copied from. (This makes aligning things a dream.) This position is remembered by the Scrapbook as well, so you can store things temporarily in the Scrapbook without losing their original position on the screen. Of course, you can move any pasted image while it's still selected, and you can undo and re-paste any picture.

Opening Other Applications

You don't have to leave HyperCard to open another application. The Open command lets you open any available Macintosh application. Use it in the Message box or any script: Open MacPaint or Open "My document" with MacWrite.

When you quit the other application, you'll return to whatever card you were at when you left HyperCard.

Importing and Exporting Text

If you currently have text in a word-processor or database file that can be saved as a text-only file, you can bring the text-only (ascii) file into a HyperCard stack. And you can create a text-only file from the text in any existing stack. To later open the document from the Finder, select both the text-only document and the application you want to use with it. Then choose Open from the File menu.

Most word processors and data bases let you save documents as text-only, without any formatting information. You can use the following script to bring these documents into HyperCard:

```
on mouseUp
  --This button lets you bring in lots of data from another
  --application. First figure out how to get that application
  --to write out its data as a plain ASCII text file. Most
  --databases have an easy way to do this. Only write out a very
```

```
  --small part of your data to make a small file. After you get the
  --procedure down correctly, then write a file with the bulk of your
  --data. After writing a small sample file from the application, get
  --into a text editor and look at the data. What characters appear
  --between records (where you will want a new card)? What
  --characters appear between fields? Remember that Return is a
    --character (a
  --second Return shows as a blank line). Tab is also a character and
  --is sometimes hard to see. The separator characters must not
  --also appear in the data itself.

  --To use separators of more than one character, you must rewrite
  --this script to look for the second character.

  put ";" into fieldMark   --Put your field separator between
    --the quote marks. It must be only one character.
  put tab into recordMark.   --Put your record (card)
    --separator on this line. Use the words return, tab, and quote
    --to mean those characters.

    --The text file should be in the same folder as HyperCard.
  ask "Import text from what file?"
  if it is empty then exit mouseUp
  put it into fileName
  open file fileName
  repeat
    doMenu "New Card"
      repeat with i = 1 to the number of fields
        read from file fileName until fieldMark
      if it is empty then   --end of file
        if i = 1 then doMenu "Delete Card"
          close file fileName
          exit mouseUp
        end if
        put empty into last char of it   --remove last separator
        put it into field i
      end repeat
      read from file fileName until recordMark
  end repeat
end mouseUp
```

125

126

And to export text from HyperCard into an ascii file:

```
on mouseUp
  --Write out the text in the fields of all cards in this stack.
  --Put it into an ASCII file.
  --These markers must not appear in the text. (If you are going to
```

```
  --use the accompanying Import text button to read the file, use
  --single character markers.)
  put ";" into fieldMarker
  put tab into recordMarker

  put the short name of this stack & "text" into fileName
  ask "Export text to what file?" with fileName
  if it is empty then exit mouseUp
  put it into fileName
  open file fileName
  go to first card
  repeat for the number of cards
    repeat with i = 1 to the number of fields
      write field i to fiel fileName
      write fieldMarker to file fileName
    end repeat
    go to next card
      write recordMarker to file fileName
  end repeat
  close file fileName
end mouseUp
```

Some databases don't use a separate marker to signify that a field is also the end of a record. The following script imports text from such databases:

```
on mouseUp
  --For data with no end-of-field marker before the end-of-record
  --marker, use the following script. (Some data bases write this type
  --of file.)

  --This button lets you bring in lots of data from another
  --application. First figure out how to get that application
  --to write out its data as a plain ASCII text file. Most
  --databases have an easy way to do this. Only write out a very
  --small part of your data to make a small file. After you get the
  --procedure down correctly, then write a file with the bulk of your
  --data. After writing a small sample file from the application, get
  --into a text editor and look at the data. What characters appear
  --between records (where you will want a new card)? What
  --characters appear between fields? Remember that Return is a
  --character (a
  --second Return shows as a blank line). Tab is also a character and
  --is sometimes hard to see. The separator characters must not
  --also appear in the data itself.
```

```
  --To use separators of more than one character, you must rewrite
  --this script to look for the second character.

  put ";" into fieldMark   --Put your field separator between
  --the quote marks. It must be only one character.
  put tab into recordMark   --Put your record (card)
  --separator on this line. Use the words return, tab, and quote
  --to mean those characters.

  --The text file should be in the same folder as HyperCard.
  ask "Import text from what file?"
  if it is empty then exit mouseUp
  put it into fileName
  open file fileName
  put the number of fields into lastField
  repeat
    doMenu "New Card"
      repeat with i = 1 to lastField-1
        read from file fileName until fieldMark
        put empty into last char of it   --remove field separator
        put it into field i
      end repeat
      --the last field
      read from file fileName until recordMark
      if it is empty then   --end of file
        if field 1 is empty then doMenu "Delete Card"
        close file fileName
        exit mouseUp
      end if
      put empty into last char of it   --remove last separator
      put it into field lastField
    end repeat
  end mouseUp
```

And this script is for exporting to those same databases:

```
on mouseUp
  --Write out the text in the fields of all cards in this stack.
  --Put it into an ASCII file.
  --These markers must not appear as part of your data in the text.
  --(If you are going to use the accompanying Import text button to
  --read the file, use single character markers.)
  put ";" into fieldMarker
  put tab into recordMarker
```

```
  put the short name of this stack & "text" into fileName
  ask "Export text to what file?" with fileName
  if it is empty then exit mouseUp
  put it into fileName
  open file fileName
  go to first card
  repeat for the number of cards
    repeat with i = 1 to the number of fields
      write field i to file fileName
      write fieldMarker to file fileName
    end repeat
    go to next card
    write recordMarker to file fileName
  end repeat
  close file fileName
end mouseUp
```

You do need to tell HyperCard what characters in the text-only document should separate fields and cards in the HyperCard stack. You can use any character on the keyboard to separate fields and cards. Tab and Return are the most commonly used.

Hint

If you're having trouble getting text into HyperCard or if you have a large file to import and want to troubleshoot before you try importing the whole file, try this. Make a copy of the text-only file you want to import; delete all but a small portion of the file; then try importing it. If you're still having trouble getting the file into HyperCard, make two or three new cards in the HyperCard stack you want to import to. Type information into each of the cards; export the text by choosing Export Text from the File menu and look at the resulting file to see what HyperCard did to it. Make the file you want to import look like that.

Importing and Exporting Pictures

The format of HyperCard graphics is different from the format of MacPaint documents. But HyperCard can read MacPaint format and transform it into the format HyperCard understands. The Import and Export Paint commands let you translate entire documents. (You can also use the Clipboard and the Scrapbook to translate the picture into the generic picture format used to trade pictures among Macintosh applications.) To bring an existing MacPaint document into HyperCard, choose Import MacPaint from the File menu. This command is available whenever you're using any Paint tool. Select a document from the list of MacPaint documents that appear. HyperCard pastes the picture onto the card picture of the current card. You now have one brief moment to change your mind by choosing Undo or pressing the top left key on the keyboard. Once you do anything else, you can't receive any picture that may have been there before you imported the MacPaint picture. Use the Transparent command (or just type "T") to see any underlying background picture.

Networks: AppleTalk, Phone Lines

You can send HyperCard stacks over AppleTalk and you can share stacks with other HyperCard users with AppleShare. You can also send stacks using phone lines, just as you send any other Macintosh file.

Sound

With HyperCard, sound moves in along with visual effects as a new element of the interface. HyperCard comes with two sounds built in: harpsichord and Boing. Both of these sounds are used with the Play command to produce a certain sound color (called a "timbre"). You can control both the duration and the pitch of any note you play using a musical shorthand. For example, you can use this line to play "Mary had a Little Lamb," on the harpsichord:

```
Play "harpsichord" "e d c d e e eh dq d dh eq g gh"
```

The first letter in each group is the pitch (c through g, then a and b). Each note is a quarter note unless you tell HyperCard otherwise by giving it a different duration, such as "h" for a half note, "w" for a whole note, or "e" for eighth note. HyperCard makes all following notes the same length until you change it again.

The full variety of notes is:

```
g#3h.      --G sharp in the third octave for a dotted half note
```

Sharp is # and flat is b. The number is the octave (C to C) with 4 being the middle octave. Any note can be stretched to 1 1/2 times its length (made dotted) by adding a period after it. A "t" after a note makes it a triplet (1/3 length). A command to play a tune (the one above plays "Mary Had a Little Lamb") is most useful in the script of a button. However, play-

ing a sound in the "openCard" handler of a card often gives a nice effect.

You can substitute a number for the pitch by using this line:

```
Play "harpsichord" "56q 58"
```

This allows computed note values. Middle C is 60 and the number is in half-steps.

You can control the speed of a tune by using the "tempo" option by using this line:

```
Play "harpsichord" tempo 200 "e d c d e e eh dq d dh eq g gh"
```

In a script, the function "the sound" returns the name of the sound that is playing at this moment. The answer is "done" if the sound has finished. "The sound" is useful because the script goes on to the next command before the notes in a play command are made. The notes come out at their own speed. Any disk that is not a SCSI disk (any floppy disk or non-SCSI HD-20) uses some of the same hardware as the sound. If you get data off the disk (go to the next card) while sound is playing, the sound gets scratchy. Here is a script that waits for the sound to finish before going to the next card:

```
Play "harpsichord" tempo 200 "e d c d e e eh dq d dh eq g gh"
repeat until the sound is "done"
  --wait here for the sound to finish
end repeat
go to next card
```

The sound itself (also called an instrument, sound color, or timbre) is actually a resource file. It is at-

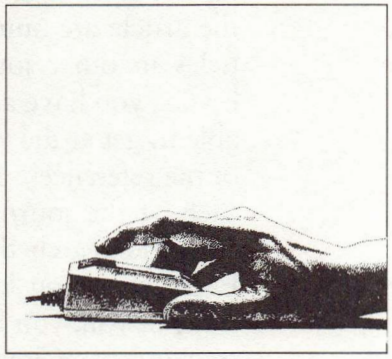
tached to a Macintosh file and HyperCard finds it by looking for it by name. When you specify the name of a sound, such as "harpsichord," HyperCard looks at the resources of the current stack, the Home stack, HyperCard itself, and the system file. If it finds a resource of type 'snd ' of that name, that is the sound that will play.

There are two sources of new sounds for HyperCard: You can get a sound from another stack, or you can move a new sound into HyperCard from an application that manipulates sound. Some programs that let you record, synthesize, and edit sounds on the Macintosh can also write the sounds out in the format that you want. If the application can write 'snd ' format II resources, then you can make it to put the sound directly into your stack. If you are getting an existing sound from another stack, you can move it with the ResEdit program or Andy Hertzfeld's new resource mover (see Appendix 4).

Summary

In this chapter, you have learned how to get information into and out of HyperCard using the Scrapbook and commands within HyperTalk designed to deal with information in other formats.

Afterword



This book is full of practical examples of using HyperCard. It has detailed exactly how to get HyperCard's power into your own work.

But the most exciting power of HyperCard is its way of opening our minds to new ways of doing things with a computer—ways that can't

be prescribed in a book, ways that haven't even been thought of yet. If you have followed the examples throughout this book, you'll see some new ways of using the Macintosh to organize your work. But more importantly, you'll also start to imagine new approaches to old problems. You'll find that HyperCard encourages you to approach your work from a new slant.

HyperCard and Hypermedia

HyperCard is "hypertext," or, more accurately, "hypermedia," which means that it gives you the ability to move around in and edit large amounts of information that's represented in many different forms—text, pictures, sounds, animation, and video disk frames, for example.

Imagine an article in a scholarly journal. At the bottom of each page and in a bibliography at the end of the article are numerous cross-references to other articles in other journals. In the ancient medium of books, you have a tedious process ahead of you to be able to get at the exact information you want in each of the references. You need to go to the library, locate each of the journals, see if they're already checked out, and search the stacks for the issues you want. Even when you have all the materials assembled in front of you, you have the task of tracking and backtracking from journal to journal, discarding the information you don't want, and trying to make sense of the mess of paper in front of you. And that's just the access to the words. What if you want to contribute something to the discussion at hand? There's no good way to annotate without taking copious notes,

including full references, and being the world's most organized person. And you still have no way to communicate your observations with others, aside from corresponding with them individually.

Now imagine the same scene with HyperCard: You open a stack that contains an index of articles on a subject. As you browse through one of the articles, searching for text that interests you and clicking buttons that take you to other places in the article, you see cross-references to other articles. Clicking on the cross-references takes you instantly to the cross-reference. And, wonder of wonders, you're able to annotate any of the text there, keeping notes for yourself or adding to a discussion you share with other people interested in the same subject. And you're not limited to text, of course, because you have the Paint tools at hand. This may dramatically change the way we exchange information with other people.

Education

HyperCard promotes student-initiated learning. It's a tool not just for the teacher, but for the student. With HyperCard, learning doesn't have to be a process of digesting a predefined lesson plan, but of each student defining their own unique learning experience.

Learning by exploring. Education is communication in time, space, and the intellect. HyperCard plays on the curiosity that makes this kind of education work. A major problem in traditional education is "You have the answers, and I don't." Or "Get me to the answers section." HyperCard invites a different interaction between students and teachers, between students and the information to be learned.

HyperCard can amplify the power of a teacher and allow the teacher to become a facilitator, rather than a dictator, of the learning process.

Form and the Shape of Content

There are probably whole schools of philosophy based on the study of the relationship of form to its content. I do remember and recommend one remarkable book by artist Ben Shahn—*The Shape of Content*—that explores beautifully the inextricable interweavings of the two.

As a musician I was used to studying this elusive relationship—watching melody shape harmony, words dictate risings and fallings of pitch. Writing books about software brought up many of the same issues. How does content relate to form? It's not likely that you could design a book and then fit the writing into the design. However, writing a book without a design in mind is even less likely.

At first glance, HyperCard neatly solves the problem. The process of creating a form, of designing a stack, can be completely intermingled with the content of the stack. One doesn't have to come before the other; each can be done concurrently, mutating in the process, or rebounding from drastic changes and resisting either bad design or bad content. And each can be done by the same person—one head, one vision.

But as you start the work of designing a stack, the problems of unifying form and content can get worse. Why is this? With HyperCard you can totally lose yourself in one or the other part of the process, neglecting the other or getting bogged down in detail. I

would often say to myself, "I've got to get back to the *words* one of these days," so fascinated was I with my new power as a stack designer.

I was entranced with the form. I wanted to spend all of my time with it—endlessly seeing what I could make of it next. But every once in a while, usually faced with a very real release deadline, I had to abandon the form to deal with the content.

Gathering

While writing Macintosh manuals, I would sometimes think of or hear a phrase I liked and wanted to use somewhere in a book I was writing. But I couldn't think of a way to work a phrase like "the pain screamed through her body" into a book about computer hardware. It was even harder to imagine recounting the feeling of my mother licking her handkerchief and wiping my sticky, five-year-old face with it.

I started writing down a few things I drew out of the past and that I wanted to remember for a book some day. But the papers got lost in notes I was keeping, or they survived only as shreds that fell out of my purse months later. At best they remained tucked in a drawer in my desk,—not forgotten, but not accessible whenever I needed them, either.

My husband Ted began typing these realizations and memories into HyperCard whenever I mentioned one. HyperCard seems to elicit every tendency you might have to gather things together in one place.

Last night one of the other guests at a dinner with our friends Margie and Merrill was a man named

Scott. Scott told us about someone he knew in college. One time Scott walked into the computer room and discovered this man sitting quietly in the corner of the room, reading the newspaper. The paper was upside down. Slightly embarassed, the man quickly turned the paper rightside up again. When he was a small boy, he had loved to hear his parents read to him, but had also been chomping at the bit to read to himself. His parents had heard that it wasn't good for a child to read too early, so they had positioned their son at their feet, where he had taught himself to read *upside down*. Of course, he later learned to read in the usual way, but whenever he was tired or, say, alone in a computer room he read the way it was easier for him.

That's the kind of thing I'd type into HyperCard . . . to go in a book someday.

Becoming a Minimalist

When I wrote the MacPaint manual I had to learn a new way of writing; omitting unnecessary words. This was good training for HyperCard writing because writing for the Macintosh screen requires saying things simply. Save the wordy explanations for someplace else. It is tempting to sum up my experiences with HyperCard and offer here the guidelines for creating great new stacks. But to do that would be the opposite of HyperCard's purpose. Instead, this book has shown what I have learned, without prescribing any pat formulas for success. HyperCard will bring its own serendipitous discoveries as you master the basics and incorporate the techniques others have developed. Whatever else, it's going to be fun.

Annotation

One of HyperCard's best features is that everything in it is editable. Therefore, any user can be an author, and vice-versa. But sometimes you want the original to stay intact and not be editable by anyone. For example, a card catalog in a library could quickly be mangled by readers jotting down their private notes or by would-be librarians with good intentions "fixing" a card catalog number. In public places such as libraries or schools, you could create stacks that can be browsed but not edited by locking the text and hiding the menubar. And someday you may be able to "edit" the stack, by saving a file with only the differences between the library's stack and "your" stack.

Appendix 1 HyperCard Windows and Commands

Like other Macintosh applications, HyperCard presents commands in easy-to-see-and-use, pull-down menus. The shortcuts for the most part are exactly that—faster ways to do things you can also do by choosing from menus. A few of the Painting actions aren't in the menus, but if you've used MacPaint, you already know how to do nearly all of these.

Which menus and commands you see depends on your User Level and the tools you're using. HyperCard is preset to the Typing level, which is the second User Level. (The lowest level—Browsing—limits you to just viewing the stack without making any changes to it.) With your User Level set to Typing, you see just the Apple, File, Edit, and Go menus, and the File and Edit menus are shortened to nonauthoring commands.

To see the Tools menu, you need to be at the Painting level

or above. At the Painting level, you also get the additional items in the File and Edit menus. Although you can see the Field and Button tools in the Tool menu, they're disabled at this level.

At the Authoring level, you have full use of all tools and you see the Objects menu, which you can use to see the dialog boxes that let you set properties for buttons, fields, cards, backgrounds, and stacks. You can't see or edit scripts, however, until you change your User Level to Scripting.

You set your User Level on the last card of the Home stack. The easiest way to get to this card is to go Home and then press the left arrow key to go to the previous card (the last card in the Homestack).

HyperCard Help and the HyperCard manual contain complete descriptions of all HyperCard commands. The windows and commands are summarized here.

The Tools Window

The Tools window is the torn-off Tools menu. See the Tools menu for a description of each of the tools. If you have a large screen such as a Radius, or if you have a Macintosh II, you can keep the Tools window handy and out of the way off to the side of the HyperCard window.

Add these lines to your Home script to display the windows separately on a large screen:

```
on startup
  show card window at 90,55
  show tool window at -80,50
  show message at 20,370
end startup
```

The numbers are horizontal and vertical screen coordinates. Adjust them for a different arrangement. (Lower numbers are higher on the screen and further to the left.) Whenever you start up HyperCard (or type "startup" into the Message box to send the startup message), the Message box or the Tools window will be displayed at these coordinates unless another script changes their position.

The Pattern Window

The Pattern window is the torn-off Pattern menu. See the Pattern menu for a description.

The Destination Window

This window appears when you start a link and remains until you complete the link by pointing to a destination. The two choices on the window are to link the selected button to a specific card or to the first card in a specific stack. The advantage of the latter is that when you want a link to go to a stack you don't usually want it to go to a specific card but to the first card of that stack. This destination choice links the button to whatever card is currently the first card of the destination stack.

The Message Window

This is the way you communicate with HyperCard—by issuing the Find command or any other one-line HyperTalk command. And HyperCard uses this window to respond back to you—with the results of a command or function or any text a script puts there. For example, a script might say "Put hello" or "Put the time" to automatically display the Message box with "Hello" or the current time.

The Apple Menu

About HyperCard gives the version number.

The Apple menu also contains the desk accessories installed in the System file on the current startup disk.

You can read about the System file and installing fonts and accessories in it in your owner's guide.

The File Menu

New Stack

Creates a new stack. With Copy Current Background checked, the new stack inherits the fields and background picture and buttons (including their scripts) of the card you were viewing when you chose the command. Otherwise, the new stack is a blank slate, ready for you to create everything from scratch.

Open Stack . . .

Presents the usual standard file dialog box you see with any application. Move among folders and disks by double-clicking folders, pressing the Drive button, or dragging from the name at the top of the list.

Save a Copy . . .

Copies the current stack exactly as it is and places it in the folder on the disk you specify.

Compact Stack . . .

Rearranges the current stack on the disk as compactly as possible. You can find out the new file's size by

choosing Stack Info from the Options menu. HyperCard will work faster with the compacted stack, and it will take up less space on your disk. The minimum space you'll gain is included as "Free in Stack" when you choose Stack Info from the Objects menu.

Protect Stack . . .

Provides options for limiting access to or modifying the contents of the current stack. You can also use this menu to limit the user level while within this stack. HyperCard uses the lower of the two limits set—either here or on the User preferences card in the Home stack. Passwords you set are encrypted so that you can have private access to a stack. (There's no way to get at the stack if you forget the password, however.)

Delete Stack . . .

Deletes all cards in the current stack and takes you back to the Home card.

Import Paint

Appears only when you're using a Paint tool. Presents a list of MacPaint documents on the current disk. Clicking a document name and then clicking Open or double-clicking the document name pastes a copy

of the MacPaint document. You get a card's worth of the top left corner of the MacPaint document.

Import Paint is available whenever you're using a Painting tool.

Export Paint

Creates a MacPaint document from the current card and background picture. The picture is placed at the top left corner of the MacPaint document.

Export Paint is available whenever you're using a Painting tool.

Page Setup . . .

Gives you options for the printer you're using and the physical aspects of what you're printing. Be sure you have LaserWriter 4.0 or later on your disk. Earlier versions can't handle HyperCard.

Print Card

Prints the current card.

Print Stack

Prints the current stack back to front with whatever options you set. These options are remembered with stacks.

Print Report . . .

Presents formats for printing text-only reports. You can print only certain fields or all fields in the current background or all fields in the current stack. Formats include labels and the fields can be arranged across the page in rows or down the page in columns of adjustable width.

Quit HyperCard

Does just what you'd expect, except that you aren't asked if you want to save because HyperCard saves your information while you work.

The Edit Menu

In general, the Edit commands affect the current selection. The Cut, Copy, Paste, and Clear commands affect whatever's selected.

Undo

Undoes your last editing or painting action, depending on which tool you're using. With the Paint tools, pressing the accent/tilde key is the same as choosing Undo.

Cut

Removes the currently selected text, picture, button, or field. Whatever is cut is placed onto the Clipboard.

Copy

Removes the currently selected text, picture, button, or field. Whatever is copied is placed onto the Clipboard.

Paste

Places the text, picture, button, field, or card you last cut or copied at the insertion point in a field or whatever position it was copied from. Pictures are pasted in opaque. You can make them transparent by choosing Transparent from the Paint menu or typing "T" with Power Keys checked.

Clear

Removes the currently selected text, picture, button, or field without placing it on the Clipboard.

New Card

Adds a new card (after the current card) to the current stack, with the background picture and background buttons of the current card.

Delete Card

Deletes the current card and brings the next card in the stack into view. HyperCard won't let you delete the last remaining card in a stack. For that use the Delete Stack command.

Cut Card . . .

Removes the entire current card—buttons, fields, pictures and all, including the card's background (which, of course, also remains behind if any other cards are using it)—and places it on the Clipboard.

Copy Card . . .

Copies the entire current card—buttons, fields, pictures and all, including the card's background—and places it on the Clipboard.

Text Style . . .

Presents a dialog box with font, size, alignment, and style choices for the currently selected field.

The best sizes for each font appear in the list. You can override these choices by typing a new size. You can also change the vertical space between lines by typing in a new number in the Height box. Text Style is available whenever there's an insertion point in a field or a field is selected using the Field tool.

Background

Lets you edit the background rather than the card objects. Choose Background again to return to editing the card picture or button.

The Go Menu

Back

Takes you to the most recent card.

Home

Takes you to the first card in the Home stack.

Help

Takes you to the first card in the Help stack.

Recent

Displays a dialog box with the 42 most recent cards. Clicking any card takes you to that card.

First

Takes you to the first card in the current stack.

Prev

Takes you to the previous card in the current stack.

Next

Takes you to the next card in the current stack.

Last

Takes you to the last card in the current stack.

The Tools Menu and Window

Browse Tool

Use the Browse tool for exploring stacks by clicking buttons and searching for text, and for editing text when the pointer is an I-beam.

Button Tool

Use the Button tool to make buttons that link to another card or that specify other HyperCard actions.

Field Tool

Use the Field tool to control the size, shape, and position of boxes that text goes into. Background fields are inherited by every card in the stack; card fields belong to individual cards.

Selection Rectangle

Use the selection rectangle to select a rectangular area.

Lasso

Use the lasso to select nonrectangular images. If there's enough space around the image you can also select with the selection rectangle and then choose the Lasso to have the selection rectangle "hug" the shape.

Lines

Draw straight lines. Press the Shift key to constrain to 45- or 90-degree angles. Hold down the Option key to draw lines using the current pattern.

Eraser

Erases where you drag. Double click to erase the entire picture.

Pencil

Draws a thin line—white on black or black on white. This is the tool of choice for FatBits.

Rectangle

Draws rectangles, hollow or filled with the current pattern. Hold down the Option key to draw borders using the current pattern. Click the rectangle at the top left of the Pattern window to draw filled rectangles.

Freeform Shape

Paints freeform shapes. Hold down the Option key to draw shapes using the current pattern.

Hold down the Option key to draw border with the current pattern. Click the rectangle at the top left of the Pattern window to draw freeform shapes filled with the current pattern.

Paintbrush

Paints with the current pattern. Choose Brush Shape from the Paint menu or double-click the paintbrush.

Spraypaint

Spray paints with the current pattern.

Rounded Rectangle

Draws rounded rectangles. Hold down the Option key to draw borders using the current pattern. Click the rectangle at the top left of the Pattern window to draw filled rounded rectangles.

Irregular Polygon

Draws lines to form a polygon. Click to turn a corner. Choose Draw Filled from the Options menu to fill in a connected polygon.

Hold down the Option key to draw border with the current pattern. Click the rectangle at the top left of the Pattern window to draw filled polygons.

Paint Text

Type Paint text. Choose fonts, sizes, and styles by choosing Text from the Edit menu or by double-clicking the Paint text tool. Confirm choices by clicking. Use Paint text whenever you want text in places other than fields—for display, heads, or callouts, for example—or whenever you want text to behave more like a picture than text.

Paint Bucket

Position the tip of the pouring paint bucket in an outlined area and click to fill the area with the current pattern. The paint will leak through any gaps in the outlined area. To correct, choose Undo and then fix the gaps.

Oval and Circle

Draws ovals, hollow or filled. Hold down the Option key to draw borders using the current pattern. Click the rectangle at the top left of the Pattern window to draw filled ovals. To draw circles, hold down the Shift key.

Regular Polygon

Drag to draw a polygon. Specify number of sides by choosing Polygon Sides from the Options menu. Draw filled polygons by choosing Draw Filled from the Options menu.

The Paint Menu

Select

Selects a recently drawn shape.

Select All

Selects entire card or background picture.

Fill

Fills the selection (or last shape drawn) with the current pattern.

Invert

Inverts the selected area of the picture. You can also type "i" with Power Keys checked.

PickUp

Picks up a copy of an underlying image in the shape of what you last drew or a selection you dragged

there. The copy is selected and you can then drag it off the underlying image.

Darken

Makes the selected area of the card or background picture darker.

Lighten

Makes the selected area of the card or background picture lighter.

Trace Edges

Outlines the black in the selected area. If nothing is selected, all black in the picture is outlined. You can also type "e" with Power Keys checked.

Rotate Left

Rotates the selected area of the picture to the left. If nothing is selected, the entire card or background picture is rotated. You can also type "[" with Power Keys checked.

Rotate Right

Rotates the selected area of the picture to the right. If nothing is selected, the entire card or background picture is rotated. You can also type "]" with Power Keys checked.

Flip Vertical

Flips the selection or the last drawn object vertically.

Flip Horizontal

Flips the selection or the last drawn object horizontally.

Opaque

Makes the selected area of the card picture opaque. If nothing is selected, the entire picture becomes opaque. You can also type "o" with Power Keys checked.

Transparent

Makes the selected area of the card picture transparent. If nothing is selected, the entire picture becomes transparent. You can also type "t" with Power Keys checked.

Keep

Save the current picture as the one to revert to.

Revert

Go back to the last kept version.

The Options Menu

Grid

Constrains drawing, typing Paint text, selecting, or moving a selection along the lines of an invisible grid.

FatBits

A zoom lens for detailed drawing. Command-click with the pencil to zoom in at an exact spot; otherwise zoom into what you most recently drew. Use the Option key to get the Grabber.

Power Keys

Performs painting commands with a single keystroke. The Power Keys are documented in online Help and in the user's manual.

Line Size . . .

Choose a width for lines, rectangles, ovals, and polygons.

Brush Shape . . .

Choose a shape for the paintbrush.

Edit Pattern . . .

Edit the current pattern. Patterns are saved with each stack.

Polygon Sides . . .

Choose the number of sides for regular polygons.

Draw Filled

Allows you to fill rectangles, rounded rectangles, ovals, polygons, and closed curves with the current pattern.

Draw Centered

Allows you to draw shapes from the center rather than from a corner.

Draw Multiple

Allows you to draw repeated images using the current line thickness.

The Patterns Menu

Lets you choose a pattern to paint or fill shapes with. The patterns, which you can edit by double-clicking them and clicking dots on or off, are saved with stacks.

The Objects Menu

Button Info . . .

Displays a dialog box describing the selected button—its name, number within the card or background, unique ID, and Script, Icon, Cancel, and OK buttons.

Field Info . . .

Displays a dialog box describing the selected field—its name, number within the card or background, unique ID, and Font, Script, Cancel, and OK buttons.

Card Info . . .

Displays a dialog box describing the current card—its name, number within the stack, unique ID, number of buttons or fields it contains, and Script, Cancel, and OK buttons.

Bkgnd Info . . .

Displays a dialog box describing the current background—its name, unique ID, number of cards that share it, number of background fields and buttons it contains, and Script, Cancel, and OK buttons.

Stack Info . . .

Displays a dialog box describing the current stack—its name, where it is located (with colons indicating folders), number of cards and backgrounds it contains, its size in Kbytes, the amount of space that would be freed up by compacting the stack, and Script, Cancel, and OK buttons.

Bring Closer

Brings the selected button or field one level closer and raises its number by one when appropriate (when it bumps an object of the same type out of its relative position).

Send Farther

Brings the selected button or field one level closer and lowers its number by one when appropriate (when it changes the relative position of an object of the same type).

New Button

Creates a new button named New Button. You can double-click it or choose Button Info from the Objects menu to see its button dialog box.

New Field

Creates a new field. You can double-click it or choose Field Info from the Objects menu to see its field dialog box.

New Background

Creates a new, blank background, ready for you to create or paste in a background picture, background fields, and background buttons.

Appendix 2 | HyperTalk Commands

The notation < > means whatever is within the angle brackets represents something else you type. It might be a value or the name of the field or any container such as a local variable or the Message box. The text within the brackets tries to give you an idea of what that text might represent. <Source> means where the value comes from. <Destination> means where a value goes. <h,v> means a location on the screen (h is the horizontal coordinate, v is the vertical coordinate).

Many commands put the result of the command into "it," a local variable used commonly because it reads like English. To see the contents of "it" in the Message box, type "it" and press the Return key. For example, you'd type "get name of field 1," press Return, type "it," and press Return again to see the name of field 1.

Hint | In other languages "it" is called an "accumulator" but in HyperCard it's just called "it."

Add <source> to <destination>

Adds the number in the source to the number in the destination and puts the answer in the destination. The destination must be a container, and the source can be a container, a calculated expression, or just a number. Use Add instead of just an expression to specify where to put the result.

Examples:

```
Add 5 to total   --same as Put total+5 into total
Add number of cards to field "total cards"
Add (amount + principle) * 0.015 to field "payment"
```

Answer <question> with <reply> or <reply> or <reply>

Puts up a dialog box on the screen asking the user to choose between the replies. The question appears in the dialog box. There can be one, two, or three possible replies. The answer that the user chooses goes into "it," a local variable, and can be tested with the "if" command. Maximum length of each reply is 13 characters (depending on character width). Contrast with "Ask."

Examples:

```
Answer "How many copies?" with 1 or 2
Answer "Are you happy?" with "yes" or "no" or "maybe"
Answer "What's the answer?" with field 1 or field 2
```

Ask <question> with <default>

Puts up a dialog box and allows the user to type a reply. The reply goes into "it," a local variable. The question appears in the box and the default reply is filled in and selected. The question and default can be any source. Contrast with Answer.

Examples:

```
Ask "Please elaborate:" with "I can't think of anything clever."
Ask "Enter number of copies:" with 1
Ask password "Please type password"   --encrypts it
  --as a number for developing custom protection
```

beep <count>

Makes a short system beep on the loudspeaker. The count is the number of beeps. It can be a constant or anything that returns a number.

Examples:

```
beep 3

on doMenu
   beep
end doMenu

if card field 1 > 1 then beep card field 1
```

choose <name of a tool>

Chooses a tool from the Tools Menu.

Examples:

```
choose rectangle tool
choose browse tool
```

Click at h,v [with shiftKey | optionKey | commandKey]

The same as if you had clicked manually

Examples:

```
Choose Button tool
Click at the loc of card button 2
doMenu Button Info...

Choose text tool  --for Paint text
Click at 50,100
type "Hi"
```

Close file <fileName>

Close a file that you have previously opened and read from or written into. Used for manipulating ASCII text files from HyperTalk. See Open.

Example:

```
close file "export 1"
```

Convert <container> to <format>

Converts the date or time in the container to the format specified. The formats are:

```
seconds          seconds since 1904
dateItems        integers representing year,month,day,
                 hour,minute,sec,day of week
long date        Monday, May 11, 1987
short date       5/11/87
abbreviated      Mon, May 11, 1987
date
long time        2:05:15 PM
short time       2:05 PM
```

Delete <text>

Removes specified text from the current card.

Examples:

```
delete word 4 of line 5 of field Backorders
delete field pastdue
delete char 1 of line 1 of first field
```

Dial <source> [with modem [<modemParameters>]]

Makes the touch tone sounds for the phone number. Enclose phone number in quotes to keep the "-" from causing subtraction. You can buy a device that connects the Macintosh sound output to the phone. To use the Dial command, you need to first pick up the phone and get a dial tone, and then issue the command.

Examples:

```
dial "408-555-1212"
dial "415-555-1212"
```

Dialing with a Modem

Set up voice telephone calls by using a modem to dial. Pick up the phone to get a dial tone and then click a button whose script uses the Dial command.

When you're using the serial port for other devices, click the "dial with": setting on the Phone card to change it from "modem" to "speaker." Using modem dialing when another device is attached to the serial port may cause the system to lock up.

Examples:

```
dial "1-408-996-1010" with modem "ATS0=0S7=1DT"
dial "1-408-996-1010" with modem
```

If you don't have touch-tone service, use pulse dialing. In this case, don't pick up the phone until the number has been dialed:

```
dial "1-408-996-1010" with modem "ATS0=0S7=1DP"
```

You can dial voice calls with the Apple Modem 300/1200, the Apple Personal Modem, or any Hayes-compatible modem.

Divide <destination> by <source>

Divides the number in the destination by the number in the source. Puts the answer in the destination. The destination must be a container, and the source can be a container, a calcuated expression, or just a number. Divide A by B is the same as Put A/B into A.

Examples:

```
Divide total by 5
Put principle into field monthly payment
Divide field monthly payment by years*12
```

Do <source>

Treats the source as if it were one or more lines in a script and executes it.

Examples:

```
do card field 1
do card field 2
```

DoMenu <name of menu item>

Chooses the menu item named, just as if the user had selected it from a menu. You do not have to specify the name of the menu, only the name of the item.

You can write your own message handler to intercept the doMenu (or keyboard equivalent) message.

Examples:

```
doMenu Quit HyperCard
doMenu Paste Button
doMenu Print Stack
doMenu Calculator
```

Drag from h, v to h, v [with shiftKey | optionKey | commandKey]

The same as if you had dragged manually. You don't need to click first. Available with any tool. See also the click command.

Examples:

```
Choose line tool
Drag from 100, 200 to 150, 250

choose select tool
drag from 430,280 to 440,290
```

Edit script of <object>

Opens the script editor so you can edit the object's script.

Example:

```
Edit script of button 3

Repeat with bb = 1 to the number of buttons
   edit script of button bb
end repeat
```

Find <source> [in field n]
find chars <source> [in field n]
find word <source> [in field n]

Searches the current stack for the source text, looking in the field specified (or in all fields if none specified.) You can use a Go command to move to a different stack before starting the search. After a search, the card of the first match becomes the current card. If no match, the current card does not change.

Examples:

```
Find "Hyper" in field 1   --finds at word beginnings
Find chars "Talk"   --finds anywhere within words
Find word "HyperTalk"   --finds exact word matches
```

Get <expression>

Puts the value of the expression into the local variable "it."

Example:

```
Get the textFont of field 2
```

Global <name of a variable>

Variables used within a message handler are normally local to that handler, and go away when the message

has been handled. This command defines a variable whose value is carried until you open another application or quit HyperCard. Each script that uses the global must redeclare it within its script before using it.

Examples:

```
on startUp
  global userName  --get the current value
  put "Willy" into userName  --change it
end startUp
```

Go [to] <destination>

Goes to the card with any visual effect that has been specified. If you don't specify a stack, you stay in the current stack; if you don't specify a card, you go to the first card in the specified stack.

Examples:

```
Go back  --or any commands in the Go menu
go Accounts  --first card in another stack
go card id 45A of Server:HyperCard stacks:rolo
go mid  --the middle card in this stack
go any  --a random card in this stack
go to next card of this background
go to card 1 of next background
```

Hide menuBar
Hide card window
Hide pattern window
Hide Tool window
Hide message window
Hide button ID 3
Hide field <name>

Removes the specified window from the screen, just as if the user had clicked on the window's close box or typed Command-space bar. Hidden fields aren't tabbed through. You can find text in hidden fields (but it won't be highlighted).

Multiply <destination> by <source>

Multiplies the number in the source by the number in the destination and puts the result in the destination. (Different from * because it tells where to put the answer).

Example:

```
Multiply total by 5   --Put 5*total into
total
```

Open <document> with <application>

Leaves HyperCard and starts another application. If you don't specify a document name, the application opens with a new document. Use a full pathname or let HyperCard look through the paths defined on the Homestack. You can also open an application or document named in a field.

Examples:

```
Open MacPaint
Open "Memo to Ted" with MacWrite
Open field 3
Open file "raw text"   --in preparation for
--using Read
```

Open printing [with dialog]
close printing

Start and stop a printing job, using stacks' current Print Stack dialog box setting or optionally allowing the user to make new choices.

Examples:

```
open printing
go to card Tutorial
print 30 cards
go to stack Reference
print 30 cards
close printing
```

Play

Make musical sounds using the specified voice. Play the notes in the list. "Boing" and "harpsichord" voices are included in HyperTalk. To use other voices, include their resources in your stacks.
note with # for sharp or b for flat
octave (4 = the middle octave)
time value—w, h, q, e, s, t, x (whole thru 64th)
c#5q. optional "." for dotted, 3 for triplet

Examples:

```
play "Boing" tempo 120 "e c d g3h. gq d4 e cw."
```

Everything except accidentals stays in effect until it is changed. Put a space between notes. If no notes are specified, the sound plays for its natural length.

While sound is playing, HyperTalk scripts go on running and all tools and buttons continue to work. If

your stack is on a 3½-inch disk or an HD20, sound will be scratchy whenever HyperCard reads the disk.

You can also specify a note value with a number rather than a letter: 60q is Middle C. 61 is C#; 59 is B, etc. To play sounds other than "harpsichord" or "Boing," you must make the data for the sound available. The actual data for sounds is stored in resources. HyperCard looks for the resources in the current stack, the Home stack, the HyperCard application, and the System file. If the sound exists in any of those places, it will play.

Pop card

Goes to the card most recently pushed by the push command.

Examples:

```
Pop card
pop card into field 3  --stores the long
  --name and pops the card without
  --taking you there
```

Print <document> with <application>

Leaves HyperCard, opens the specified document, and prints it using current printing specifications. Use a full pathname, or let HyperCard follow the search order you've set in the "Look for Stacks In" card in the Home stack. You can also run an application or document named in a field.

Examples:

```
Print "Memo to Ted" with MacWrite
Print field "Weekly report" with field 3
  --(the name of the application it uses is
  --in field 3)
```

Push Card

Marks the current card. The next "pop card" command will return to this card.

Examples:

```
push card
  {later}
pop card

Push recent card   --pushes the card you were
     --at before you came to the current
     --card
```

Put <source> before | into | after <destination>

Places the contents of the source into the destination. Use "before" to insert before, "into" to replace, or "after" to append after. The destination must be a container. If the destination is not recognized, HyperCard creates a temporary variable of that name. If the destination is omitted, it is the Message box. Use with the Go command to put in a remote field.

Examples:

```
put   --put the contents of it into Message
      --box
put "completed" after line 3
```

```
put "Bye now" into field 2
put "Hello"
```

Read from file <fileName> until <character>
read from file <fileName> for <number of bytes>

Reads from an external ASCII text file into "it." You must use Open to open the file and Close after you are done. See the "Import Text" button in stack Button Ideas.

Examples:

```
read from file "Import 2" until tab
if it is empty  --end of file
   then close file "Import 2"
```

Reset Paint

Puts the options in the Options menu and the characteristics of Paint text back to their preset vaiues.

Set the <property> of <object> to <value>

Sets the specified property for the object.

Examples:

```
Set textFont of button 1 to New York
Set style of button 1 to transparent
Set userLevel to 2  --Typing
```

Show menuBar
show card window at h,v
show pattern window at h,v
show Tool window at h,v
show message window at h,v
show tool window at h,v
show button ID 3 at h,v
show field <name> at h, v

Puts the specified window on the screen at the same location as the last time it appeared (just as if you'd torn off the menu or chosen Command-M) or at any horizontal or vertical location you specify.

Examples:

```
Show Message at 14, 65  --h and v of button
center
Show 6 cards
Show all cards
```

Sort [ascending | descending] [text | numeric | international | dateTime] by <field or expression>

Sorts the current stack, moving cards as necessary. Defaults are ascending order and text.

Examples:

```
Sort by field 1
Sort by last word of field Name
Sort by field LastName
Sort descending numeric by field 3
Sort by field (Termdate - field HireDate)
Sort by date
Sort by dateTime
```

Subtract <source> from <destination>

Subtracts the number in the source from the number in the destination and puts the answer in the destination. The same as Put <destination>- <source> into <destination>. Note that the order of the numbers is backwards from using a minus sign (A-B is written Subtract B from A). Destination must be a container.

Examples:

```
Subtract 5 from total
subtract field 1 from field 2
```

Type <source> [with shiftKey | optionKey | commandKey]

The same as if you had typed manually.
Available whenever you could manually type text.

Example:

```
Choose Browse tool
Click at the loc of card field 1
Type "I can type this all by myself."
```

Visual [effect] <name of effect>

Sets up a visual effect to occur at the next Go command. You can stack up several visual effects to occur one after another. The modifiers slowly, fast, very slowly, and very fast will change the speed.

Examples:

```
visual effect zoom open
visual wipe left  --the word "effect" is optional
visual dissolve slowly
visual effect scroll left to gray --"to gray" is an added effect
Visual effects you can type include:

zoom open              iris open          barn door open
zoom close (or         iris close         barn door close
  zoom out, zoom       wipe left          scroll left
  in)                  wipe right         scroll right
dissolve               wipe up            scroll up
checkerboard           wipe down          scroll down
venetian blinds
```

You can combine effects with the adverbs very fast, fast, slow, very slow, slowly, very slowly. (You see this difference between fast and very fast only on a Macintosh II.)

Wait [for] <number of ticks>
wait [for] <number> seconds
wait until <true or false expression>
wait while <true or false expression>

Pauses doing the commands in the script for a certain amount of time.

Examples:

```
wait 20  --ticks (60th of a second)
wait 3 seconds
wait until the mouse is up
wait while the mouse is down
wait for 4 seconds
```

Write <source> to file <fileName>

Write text to an ASCII file. Additional commands

write where the previous ones left off. See Open file, Read file, and Close file.

Examples:

```
write field 3 to file "export 1"
write "The topics are" & return & "as follows" to file "export 1"
```

New Commands

You can also define your own commands that use existing HyperTalk commands.

Define new commands by including the name of the command in a Message Handler:

```
on bk
  go to card 1 of next background
end bk
```

When you send the message "bk" in the Message box or a script (by saying "send bk to <object>," HyperCard knows what to do by what follows "on bk."

Commands you define follow the same rules of inheritance as any message.

Appendix 3 | Moving Resources

Sounds and icons are stored as Macintosh resources. With most applications you never deal with resources; they belong to the technical world of Macintosh programmers. But with HyperCard you may want to add sound or icon resources from other stacks or create your own new resources from scratch.

A resource can be attached to either a stack or to Hypercard itself. The advantage of adding a resource to HyperCard is that it's then available to any stack. The advantage of adding a resource to a stack is that it stays with that stack when you give it to a friend who may be using a different copy of HyperCard or a different Home stack. Of course, you can attach resources to more than one stack. If you attach a resource to more than one stack, however, resources will take up more space on the disk than if you attach them to the Home stack or to HyperCard itself.

One way to copy icon resources among stacks is to copy a button that uses the resource you want. The button's icon resource is copied along with the button. Whenever you subsequently click Icon in any of this stack's Button Info dialog boxes, you'll see the new icon in the icon directory. The icons displayed in the icon directory are those attached as resources to the current stack or to HyperCard itself.

A more generalized way to get at and move any resource, including icons and sound, is to use a resource editor. You can use ResEdit or MiniServant, which was written by Macintosh programmer Andy Hertzfeld. MiniServant is available from most user groups and software bulletin boards. It greatly simplifies moving and editing resources, especially sound (resource type snd), functions (XFCN), and custom commands (XCMD) that Macintosh programmers write to extend HyperCard's capabilities. Note that there's a space character following the snd for sound resources. It's important that you include this space character.

The "Flash" XCMD, written by Bill Atkinson, is included with HyperCard as an example of what you can do with an XCMD. Stacks you purchase may include these resources.

MiniServant

Here's how to use MiniServant to move existing resources:

Double-click the MiniServant application to start the MiniServant application. It behaves very much like the Finder with some differences—notably a grabber

hand to move around, and a zoom lens to move in for a closer look.

To copy a resource from an existing stack:

1. Select the stack whose resources you want to copy.

2. Choose Resource Open from the File menu.

This command gives you a new view into the information in a stack, letting you see and edit its resources.

3. Select the stack you want to copy the resource to.

4. Choose Resource Open from the File menu.

If the stack doesn't have a resource fork, MiniServant creates a new one.

5. Drag the resource you want to copy from the stack it's in to the stack you want it copied to. (Double-click any sound resource to hear what it sounds like.)

You can copy resources only among resource windows. The Macintosh will beep if you try to drag a resource anywhere else. You can copy multiple resources by Shift-clicking to select more than one resource. And you can open resources for HyperCard, the System, or any Macintosh file.

To edit an icon resource:

1. Select the stack that contains the icon you want to edit.

2. Choose Resource Open from the File menu.

3. Double-click the "Type icon" resource icon.

4. Double-click any icon resource.

5. Click dots black or white to change the icon.

ResEdit

You can also use another program called ResEdit to copy or create new resources. You can purchase ResEdit from the Apple Programmers and Developer's Association (APDA), 290 SW 43rd Street, Renton, WA 98055. Their phone number is (206) 251-6548.

Appendix 4 | System Requirements and Comforts

Memory

Hypercard requires 750K of memory to work with the Paint tools. You'll need about 600K to work without them. (You'll get a dialog box informing you that the Paint tools can't be used if there isn't enough memory.) Don't use the RAM cache, Switcher, or any multiapplication environment unless you have more than one megabyte of memory.

Disk Space

You can run HyperCard from a single disk if you're limiting your work to phone lists and calendars. (One disk in the

HyperCard set is a minimal collection of stacks, HyperCard, and a slimmed-down System Folder.) But realistically you need at least an external 800K disk drive in addition to the built-in disk drive. And to be really comfortable you'll probably want a hard disk.

The following are the minimal versions of system software required to use HyperCard. (Check version numbers in the Finder by selecting the icon and choosing Get Info from the File menu.)

Mac Plus	**Macintosh SE**	**Macintosh II**
System 3.2+	System 4.0+	System 4.1
Finder 5.3+	Finder 5.4+	Finder 5.5
LaserWriter 4.0	LaserWriter 4.0	LaserWriter 4.0

Index of Hints

Chapter 1
Opening an application or a stack from the Finder. 15
The Home card versus the Open File command. 18
The Undo Key, Undo Paint, and Recent dialog box. 19
Using the Button tool to copy and create new buttons. 24
Enter text in a consistent way for easier printing and sorting. 26
The shape of the pointer tells you when you can edit text. 27
Pick up text into the Message Box by Command-clicking or dragging using the Browse tool. 28
Found text is highlighted with an enclosing rectangle rather than inverse video. 29
Including three beginning letters of one or more words you're searching for allows HyperCard to do its fastest search. 30
A stack, background, or card's script can specify automatic opening of the Message box. 30
Type "Go" followed by the name of a stack (the entire stack name enclosed in quotation marks), and press the Return or Enter key as a shortcut to opening a stack. 31
Make a copy of any stack you don't want to alter. All changes in HyperCard are automatically saved. 31
Set the level of changes you're allowed to make by clicking a checkbox on the User Preferences card — the last card in the Home stack. 32
An I-beam appears over a text field only when the field is unlocked. 33

Chapter 2
Differences between MacPaint and HyperCard pictures. 38
Cards in a stack are arranged in a circle. You can get to the last card by pressing the left arrow from the first card. You can get to the first card by pressing the right arrow from the last card. 39
Choose Alarm Clock from the Apple menu to check that it's set to the current time and date. Many stacks depend on the clock being set right. 40
Use the Open Stack command to open copies you've backed up by copying them in the finder or choosing Save A Copy. 42
"Tear-off" the Tools or Patterns menus by dragging down and beyond the menu. 42
Keep the tools window out while you're making design changes to your stack. Drag it by its top bar to move it. Click the close box to put it away. 42
When you're using the Paint tools, the Undo key undoes your last painting action. To back up to other cards you've seen, hold down the Command key while you press the Undo key. 44
When you're about to embark on a major change using the Paint tools, keep interme-

diate changes by choosing Keep from the Paint menu. Later choose Revert to revert to those changes, as long as you're still at the same card and are still using the Paint tools. 45
When any Paint tool is selected, the Cut and Copy commands in the Edit menu apply to pictures. 48
Checking Power Keys in the User Preferences card automatically checks it until you uncheck it (even if you've quit and restarted HyperCard). 49
You can revert to kept changes or to the way the card was when you came to it as long as you don't leave the current card or choose a tool other than a Paint tool. 50
Get in the habit of Copying rather than Cutting images, especially when you're copying from Art ideas or one of the other collection stacks. 50
The Datebook contains two calendars plus a To Do list. Three buttons on the Home card take you to various parts of the Datebook. 52
Pictures within HyperCard are always pasted opaque. 54
Checking Power Keys in the Options menu turns them on for this session only. Check them on the User Preferences card to keep them checked across sessions. 55
Whenever you want to use the same picture on more than half of the cards in a stack, put it in the background. 56
The Weekly Calendar, Yearly Calendar, and To Do list, are examples of using more than one background in the same stack. 56
When there is no insertion point in the Message Box or anywhere else, HyperCard puts anything you type into the Message Box. 57
HyperCard maintains consistency with MacPaint by using the top left key to undo changes. Press Command as well as the top left key to retrace steps. 58
When you're in the background, slanted lines appear in the menu bar. 59
When you're looking at a card's background, you see only the background picture, but when you are looking at the card, you see both the card picture and any background picture underlying it. 60
In HyperCard, the marching ants that surround a selection disappear when you drag the selection. 61
Each card is the full screen size of a Macintosh Plus. 65
Choosing Grid from the Options menu avoids gray seams. 66
Holding down the Option key copies the selected area. Holding down the Shift Key keeps the copy aligned. 67
You don't have to be in the background to change the shape, style, font, or size of a background field. You need to be in the background only when you create a background field. 70
Use the Address stack for any kind of miscellaneous information. 72
Create different text styles by using separate fields for the text you want to differentiate. 72
Use Paint text for special cases when you want to embed bold or italic words within a plain text style. 74
Ways to determine whether a button belongs to a background or a single card. 75
Upper or lowercase in button names. 79
The button icon dialog box. 81

Chapter 3
Cards that use the same background don't necessarily have to be next to each other in the stack. 91
Copy a stack in the Finder to make an identical copy; use the New Stack command to copy just the current background without the text or any card-specific information. 91

Pressing the Tab key is a shortcut for selecting the next field. 96
You can change a field's style when it's selected using the Field tool, or when its text is selected using the Browse tool. 99
Although all text in a field shares the same font and style choices, you can add a new style by creating a new field. 99
Double-clicking either a font or size choice also confirms all style changes and puts away the text style dialog box. 99
Choose background from the Edit menu to make any changes to the background picture, but modify background fields and buttons from either the card or background. 100
Create "shared" text by using Paint text in the background. 102
Normally, pictures are pasted into the same relative position they were cut or copied from, but when you paste in text as Paint text, HyperCard pastes it into the center of the screen. 103
Holding down the Shift key while you draw a line with the Line tool constrains the line to horizontal, vertical, or 45 degrees. 108
Command-click with the Pencil tool to enter Fatbits at an exact spot. 108
Where you are when you choose New Stack determines the background for your new stack. 113
If HyperCard can't find a stack it needs, it presents the Standard File dialog box, where you can click on the intended stack. 114
Remember to check your Alarm clock if a stack is not working properly. 115
Command-2 and Command-3 do the same things as the left and right arrow keys, in case your Macintosh keyboard doesn't have arrow keys. 115
Some buttons you can copy and paste without knowing how to write scripts. 117
The Message box isn't just for Find. You can also type "Go" followed by the name of the stack you want to go to. 117
Use icons with buttons when you can, to save yourself having to create a separate picture. 119
You don't have to be in the background to modify a background button or field. 121

Chapter 4
Try to anticipate what your stack will do and plan it from the start. 126
If you don't see the Objects menu, click Author or a higher level on the User Preferences card. 129
Compacting stacks for speed. 130
Choose Cancel in a dialog box when you don't want to change anything. 131
HyperCard builds a path name from choices you make in the Standard File dialog box. 132
Finding out how many backgrounds exist in the current stack. 132
Be consistent in the order you enter names, last name first or first name first. 133
Pressing the Tab key selects the next field. 133
Enter phone numbers in a consistent way. 134
Immediately choose Undo to recover a card you really didn't want to delete. 136
Surround stack names of more than one word with quotation marks. 141
Make a miniature picture by copying a card and holding down the Shift key while you paste. 142
Hide or show the menu bar by holding down the Command key while you press the space bar. 144
When you are using the Paint tools, the top left key does what it's always done in MacPaint. 144

You can copy any picture to paste onto a card or background. 145
You don't need to be in the background to edit background button structure. 146
You can get rid of a button by clicking it with the Button tool to select it and then choosing Cut Button or Clear Button, or pressing the Backspace key. 147
Dragging from an edge outward changes a button's size; dragging inward makes it smaller; dragging from the center moves it. 148
Lining up buttons. 149
For obscure fonts, you might want to use Paint text because a person you give your stack to might not have the specified font in their System file. 151
Hints about Paint text. 152
A card includes everything you can see on the screen. 153
Ways you can modify Paint text while you are typing it. 153
Use Paint text in the background to create "shared" text. 155
Still more about Paint text. 156
Choosing Select or pressing Command-S just after you've typed text selects the text. 156
Choosing Keep gives you a picture to revert to. 157
Another warning about setting your clock. 158
You can edit background fields without being in the background. 158
When you can change text style. 158
Nothing is frozen until you check Can't Modify Stack from the Protect Stack dialog box. 160
How to move a field from a card to the background. 161
You can always modify a field later. 163
Aligning fields. 164
Labels you type to identify a field are completely separate from the field's actual name. 165
Moving a field to line up with Paint text. 165
The Tab key moves you through background fields and then card fields. 166
Moving fields to line things up. 167
When referring to fields in scripts, use the exact field name. 167
Changing the order of fields. 167
The Home card is a visual directory, not an official directory. 169
How to get to the first card in a stack. 170
HyperCard knows what's on the clipboard-text, pictures, buttons, fields, or entire cards. 171
Losing a selection by clicking outside it. 172
Checking alignment among pictures. 173
Some Commands are dimmed when you are using the Paint tools. 176

Chapter 5

If there is a selection in the Message box, typing replaces only the selection; otherwise typing replaces everything. 186
Where does your typing go? 188
When an object is unnamed, HyperCard uses its ID. 190
About the Objects menu. 191
Pressing Return or Enter performs whatever action is in the Message box. 193
HyperTalk commands are separate from the commands you see in menus. 194
The word "to" and quotation marks are usually optional in going to a stack. 195
HyperCard builds pathnames. 196
Double-clicking a button is a shortcut for seeing its Button Info dialog box. 196

Set your level to scripting to be able to edit scripts. 198
A message handler is a control structure. 201
Capitalization is just a convention, not a rule. 201
You can choose from a list of special visual effects. 201
Click Cancel when you don't want to keep changes to a script. 202
Weigh the extra work of naming objects against the amount of clarity it provides in reading scripts that use the objects. 204
Avoiding confusion about object names. 207
The ways in which you can refer to an object. 208
The OK button accepts the script including any changes you have made. 208
Looking at existing scripts is a good way to learn the structure and logic of HyperCard scripts. 211
A line is a piece of text that is enclosed by Return characters. 212
Cut and Paste in and among scripts using the Command key equivalents. Menus aren't available while you're editing scripts. 216
Separate lines in a script using Return characters. 217
Use the Tab key to check scripts' alignment. 218
Pressing the Enter key is a shortcut for clicking OK when you are editing a script. 218
Clicking on a button with the Browse tool carries out its actions; clicking with the Button tool selects the button for further action such as changing its size or looking at its Button Info dialog box. 219
The Objects menu is visible only when the User Preferences card is set to Authoring or Scripting. 220
Put message handlers at the level in the object hierachy that includes all the objects (but no more) that you want to respond to a message. 222
When you name an object the same name as a HyperCard command or keyword, be sure to use that name in quotes. 224

Chapter 6
You don't have to be in the background to adjust a background field or button. 231
Buttons you create by Command-dragging are preset to the transparent style. Buttons you create with the New Button command are preset to rounded rectangles. 232
You don't have to name buttons. 233
In most Macintosh applications, pressing the Return key is the safest way to close a dialog but in HyperCard Return often accepts the changes you have made. 233
Constrain a button's height to the height of a standard Macintosh button by holding down the Shift key while you adjust its size or shape. 234
Holding down the Shift key while you double-click a button, takes you directly to the button's script. 236
The add command requires a numeric value. 238
The Tab key takes you to the next field. 238
Don't use dollar signs when you want to do arithmetic on information in fields. 239
A field with no value in it is automatically assumed to be 0. 240
Most of the commands you construct with HyperTalk sound quite English-like. 241
HyperCard tries to indent scripts automatically. 242
Try out commands using the Message box and pressing Return. 245
The "send" command lets any object send a message to another object. 246
Hold down the Shift while moving buttons to align them. 247
Pressing the Shift key while you open an Info dialog box is a shortcut to get to scripts. 249
Use the Return key to start each new line in the script. Press the Tab key to straighten out any indent errors. 250

The Message box is available from within scripts as well as from the Edit menu. 250
Not moving linked cards from their original folder ensures the links will always work. 254
Keeping linked stacks within the same folder keeps path names shorter. 254
If you have a large-screen Macintosh, the current stack name is always visible at the top of the window. 255
The "lockscreen" command hides the action from the screen. 256
The order of printing fields with the Print Report dialog is the order of the fields themselves from back(1) to front. 262
All print dialog settings are saved with individual stacks. 262

Chapter 7
Putting a button in the background keeps it in one place within that background. 270
If you want to lasso a picture and make it opaque, be sure it does not have any "leaks" in its outline. 278
As your Mother said, always put things back the way you found them. 304

Chapter 8
An item is a group of characters separated by commas. 324
A variable is a container, like a field. 341

Chapter 9
Hints about importing text. 359

Index

Abbreviated date format, 399
About command, 376
Absolute references, 204–207
Accumulator, it variable as, 397
Add command, 238, 397
Adding
  of backgrounds to stacks, 287–289, 395
  of buttons, 173–175
  of cards to stacks, 34, 100–105, 132–135, 336–337, 381
  of pictures to weekly calendar, 38–45
  of resources to stack, 416
  of text to fields, 31, 71–72
  of text fields, 97
Address stack
  enlarging appearance of cards in, 64–67
  enlarging fields in, 68–71
  searching in, 26–31
After, with Put command, 238
Alarm clock, 10, 40, 115, 158
Alert messages, Answer command for, 310
Align Right command, 151
Alignment
  of buttons, 149
  of margins, 153–154
  of pictures, 295–299
  and selection rectangle, 173
Alphabetical order, sorting stacks into, 26, 136–139
Ampersand for string concatenation, 342
Animation, 301–304
Answer command, 212, 398
  as alert message, 310
Apple menu, 376–377
AppleTalk, 360
Arithmetic. *See* Calculations
Arrow keys for navigation, 20
ArrowKey system messages, 320
Art Ideas stack, 89, 266
  borrowing pictures from, 45–55
Artists for graphics design, 272–273
Ascending option, 217
Ask command, 240, 244, 398
Atkinson, Bill
  design philosophy of, xxiv-xxv
  Flash XCMD by, 415
  on HyperCard, xiii-xix
  quoted, 1, 87, 183
Author User Level, 68, 374
Auto hilite check box, 121, 233

B, for musical flats, 361, 407
Back command, 18–20, 383
Background buttons, 118–119, 279
  compared to card buttons, 75, 306
  linking of, to cards, 105–111
  modification of, 100
  moving of, 121
  and new stacks, 90
Background command, 58–59, 97, 100, 103, 152, 382
Background fields, 32–33, 384
  creation of, 160–165
  deletion of, 104
  modification of, 70, 99–100
  and tab key, 96, 133, 166–167, 238
Background pictures, 16–17, 37, 276
  and animation, 302
  compared to card pictures, 55–60
  modification of, 99–100, 139–146
  shrinking of, 62–63
Backgrounds, 17
  adding of, to stacks, 287–289, 395
  and BackInfo command, 394
  editing of, 97
  finding of, with scripts, 328–329
  inheritance of, 32–33, 91
  and layers, 275–291
  multiple, 56, 91, 132, 285–286
  and New Stack command, 113, 125, 128, 252–255
  and Paint text, 151–155, 292–293
  and Stack Ideas stack, 92, 140

and Stack Info command, 132
system messages for, 320
Backspace key for deletion, 242
of buttons, 147
of cards, 136
of characters, 34, 153
of fields, 104
Bar codes, script to generate, 345–347
Barn door open and close effect, 300, 413
Beep command, 398
Before, with Put command, 237–238, 409
Big background buttons, 306
Billing stack example. *See* Designing of stacks
Bills, designing, 139–146
Bkgnd Info command, 394
Black for definition and depth, 270
Blind typing check box, 198
Boing sound, 339, 361, 407
Borders, 270
for background buttons, 75, 118
and rectangle tool, 385
Borrowing
of pictures, 45–55
for stack building, 88
Bring Closer command, 167, 262, 279–281, 394
Browse tool, 384
clicking buttons with, 21
Browsing, stacks for, 272
Brush shape command, 391
Building of stacks, 87
by borrowing and collecting, 88–89
by copying buttons, 116–121
new monthly calendar stack, 111–121
with Stack Ideas stack, 92–111
starting, 90–91
Built-in information and HyperCard, 88
Burke, James (quoted), 36
Business logos, adding of, to stacks, 154–157
Button Ideas stack, copying buttons from, 116–121
Button Info command, 196–199, 393
Button tool, 24, 68, 118, 173, 210, 384
for customizing stacks, 74–80
for deletion of buttons, 146–147
for size and shape, 231
Buttons, 21–25, 74
alignment of, 149
background vs. card, 75
for calculations, 229–235
clicking of, with Browse tool, 21
commands with, 194–202
copying of, 116–121
creation of, 77–80, 395
deletion of, 146–147
designing of, 146–149
editing scripts of, 208–209, 214–219, 235–240
highlighting of, 118, 121, 233
home, 21
and icon resources, 416
icons for, 80–84
IDs for, 79
keeping track of, 304–306
library for, xvii
linking of, to cards, 81–84, 105–111
moving of, 148–149
naming of, 232–233
navigation with, 75–77
for new stacks, 173–175
size and shape of, 148, 231, 233–234
for subtraction, 240–245
system messages for, 318
testing of new, 115–116
See Background buttons; Card buttons

C language
and HyperCard, 317
and XCMD, 336
Calculations, 192–194
button for, 229–235
and Paint text, 292
See also specific arithmetic commands
Calendar button, 111
Calendars
monthly, creation of new, 111–121
weekly, adding pictures to, 38–45
Cancel in dialog boxes, 131, 202
Can't Modify Stack check box, 160
Card buttons, 118–119, 279
compared to background buttons, 75, 306
hiding stack buttons with, 283–285
Card fields, 161, 384
Card Ideas stack, 89, 140
Card Info command, 393
Card pictures, 17, 37, 276
compared to background pictures, 55–60
for differentiating cards, 107–111
opaque, 61–63
Cards, 16–17
adding of, to stacks, 34, 100–104, 132–135, 336–337, 381
arrangement of, 39
copying of, 288–289, 382
deletion of, 34, 136, 378, 381
IDs for, 204
inherited backgrounds, 91
linking buttons to, 81–84, 105–111
and messages, 192
printing of, 175, 379
pushing and popping of, 333–335, 408–409
referring to, 208
separating fields from, in files, 359
system messages for, 319
See also Navigation
Case sensitivity, 28, 79, 195, 201

Centering of fields, 96
Characters, deletion of, 34, 153
Chars with Find command, 403
Checkerboard effect, 299, 412
Choose command, 323, 399
Chooser desk accessory, 177
Circles, tool for, 387
Circling of dates, 38–45
Clear Button command, 147
Clear command, 381
Click command, 399
Clicking, 21
  and object order, 282
Client billing stack. *See* Designing of stacks
Clip Art stack, 45
Clipboard
  and pictures, 54, 64, 360
  and text, 102–103, 382
Clocks, 10, 40, 115, 158, 224
Close file command, 399
Close printing command, 406–407
CloseField message, 222–224
Coin button, script in, 337–339
Collecting for stack building, 89
Command key
  dragging of, 28, 77
  for scripts, 216
  for shrinking and enlarging pictures, 63
Command-1 for First, 107
Command-2 for Previous, 115
Command-3 for Next, 115
Command-? for Help, 21
Command-backspace for card deletion, 136
Command-down arrow to push cards, 20
Command-eraser to erase lines, 108
Command-H for Home, 18, 20
Command-left arrow for First, 170
Command-N for New Card, 101
Command-option for background buttons, 75, 118
Command-S for Select, 48–49, 156
Command-shift-3 for screen shots, 307
Command-space bar for hiding menus, 144
Command-tilde to retrace steps, 19, 44
Command-up arrow to pop cards, 20
Command-V for pasting pictures, 64, 145, 171
Commands, 194, 396–414
  and buttons, 194–202
  external, 317, 336
  and Message box, 31
  in scripts, 323
  testing of, in Message box, 245
  See also specific commands
Commas with arguments, 326
Comments in scripts, 322–323
Compact Stack command, 129–130, 377–378
Computed go command, 333
Concatenation of strings, 342
Consistency
  of data, and sorting, 26, 133
  in design, 269–270
Containers, 202
  variables as, 341
Control structures, 200, 242
  if . . . then . . . else, 212–213
Convert command, 400
Coordinates
  screen, 323, 341
  of tool window, 331, 375
Copy Button command, 119
Copy Card command, 170–171, 382
Copy command, 381
  and ID numbers, 206
Copy Current Background check box, 113, 377
Copy Picture command, 144
Copying
  of buttons, 116–121
  of cards, 288–289, 382
  of fields, 163–164
  and IDs, 206
  and Paint text, 291–292
  of pictures, 48
  and Stack Ideas stack, 93–94
  of stacks, 10, 90–91, 377
  of text, 33
Creation
  of buttons, 77–80
  of fields, 160–165, 395
  of new monthly calendar, 111–121
  of stacks, 94–97, 125–128, 252–255
Cursor keys for navigation, 20
Customization
  and HyperCard, xv
  of stacks, 36
    with Button icons, 80–84
    with Button tool, 74–80
    with Field tool, 67–74
    with Paint tools, 37–67
  of text appearance, 72–74
  See also Designing of stacks
Cut Button command, 147
Cut Card command, 382
Cut command, 380
Cut Text command, 102–103
Cutting
  and Paint text, 291–292
  of pictures, 299
  in scripts, 216

Darken command, 49–50, 389
Data in card pictures, 109–110
Database files, importing and exporting of, 355–359

Date, 185–187
  and Convert command, 399
  sorting by, 214–219
Datebook stack, 41, 52
DateItems format, 399
DateTime modifier, 217–218
Definition
  of messages, location of, 335–336
  screen, gray and black for, 270
Delete Card command, 34, 381
Delete command, 104, 400
Delete Stack command, 378, 381
Deletion
  of buttons, 146–147
  of cards, 34, 136, 378, 381
  of characters, 34, 153
  of fields, 104
  of lines, 108
  of stacks, 378, 381
  of text, 34, 400
Depth, screen, gray and black for, 270
Descending option, 217
Designing of stacks, 123
  adding of features, 229–251
  appearance, 139–146
  buttons, 146–149
  functions, 124–139
  and Home card, 169–175
  with Paint text, 149–157
  text fields, 157–165
Destination window, 376
  with LinkTo, 82–83, 204–205
Dial button and command, 146, 400
Dialing with a modem command, 401–402
Directory, stack, 17, 169
Disk space requirements, 418
Dissolve effect, 299–300, 413
Divide command, 401
Do command, 401
Dollar signs and arithmetic operations, 239
DoMenu command, 194, 323, 402–403
Do-nothing buttons, 283
Double borders for background buttons, 75, 118
Doyle, Ken, stack indexer script by, 347–350
Drag command, 402
Dragging, setting speed of, 302, 304
Draw Centered command, 392
Draw Filled command, 392
Draw Multiple command, 392
Drop shadow effect, 270
Duplication of stacks, 10, 90–91, 377

Edit menu, 380–382
Edit Pattern command, 392
Edit script of command, 402–403
Editing
  of backgrounds, 97
  of button scripts, 208–209, 214–219, 235–240
  and I-beam cursor, 27, 33, 40–41, 133, 384
  and Paint text, 149
  of scripts, 403
  stacks for, 272
  of text, 32–34
Education and HyperCard, 367–368
End if statements, 242
End statements, 200, 218
Enlarging
  of buttons, 148
  of Paint text, 156–157
  of pictures, 63
  of text fields, 67–71
Enter key
  and Message box, 193
  for OK, 218
EnterKey message, 320
Eraser tool, 385
Escape key to retrace steps, 19–20
Espinosa, Chris (quoted), 264
Export Paint command, 360, 379
Exporting, 378–379
  of pictures, 360
  of text, 356–359
External commands, 317, 336

Farr, Mike, stack merging script by, 329–331
Fast, with visual effects, 412
FatBits command, 108–110, 385, 391
Felsenstein, Lee (quoted), 353
Field Info command, 167, 393
Field tool, 97–99, 158–159, 384
  for customizing stacks, 67–74
  for field creation, 161
  and insertion point, 167
Fields, 32
  adding of, 97
  adding text to, 71–72
  adjustment of, 157–165
  as containers, 202
  copying of, 163–164
  creation of, 160–165, 395
  deletion of, 104
  enlarging of, 67–71
  filling in of, 166–168
  fonts, size, and style in, 72–73, 97–99
  hiding and showing of, 309–310, 312
  identification of, 151–154
  inheritance of, 32–33
  locating of, with tab key, 96, 133, 238
  locked, 71, 348
  naming of, 162, 164, 167
  numbering of, 167
  ordering of, 281
  and Paint text, 150, 292, 294, 298
  pop-up, 295, 308–311

Fields (*continued*)
printing of, 176, 262
properties of, 97–100, 158, 324, 384
separating cards from, in files, 359
sorting by, 312
system messages for, 319
and text, 292, 294
values in, 240
See also Background fields; Card fields
File Index stack, 336–337
File Menu, 377–380
Files
closing of, 399
importing and exporting of, 355–360, 378–379
reading from, 409
writing to, 412–413
Fill command, 388
Filling in fields, 166–168
Find command, 25–31, 194, 403
limiting scope of, 343–344
and Paint text, 292
See also Searching
First command, 18, 107, 170, 383
Flash XCMD, 415
Flats, musical, 361, 406
Flip Horizontal command, 390
Flip Vertical command, 390
Folders, 114, 130
and linked cards, 254
Font button, 163
Fonts
in fields, 72–73, 97–99
in graphics design, 269
and Paint text, 150–151, 292–294, 387
and System file, 72, 151
and Text Style command, 382
Form and content, and HyperCard, 368–369
FrameLeft() function, 338–339
Free space and Free in Space, 129, 378
Freeform shape tool, 386
Frog Jump script, 302–303
Front-to-back-order of objects, 279–281
Functions, 188, 338–339. *See also* specific functions

Gathering of information, 369–370
Get command, 326, 403
Global command, 330, 403–404
Glossary stack, limiting scope in, 343–344
Go command, 20–21, 195, 404
computed, 333
and Find command, 403
and inheritance, 316
and Message box, 117
and visual effects, 300
Go menu, 18–19, 383–384
Graphics. *See* Pictures
Gray, for definition and depth, 270
Grid command, 66, 391

Harpsicord sound, 361, 406
Headers
control of, 307
lining up of, 297–298
Height box, 382
Help command, 18–19, 383
Help Index stack, 290
Help Samples stack, 290–291
Help stack, 191, 290
Help system message, 321
Hertzfeld, Andy, MiniServant by, 415–416
Hide command, 308, 404–405
Hiding
of fields, 308–311
of menu bar, 144
of stacks, 307
Hierarchies, naming, 203–204
Hierarchy and message handlers, 225–226
Highlighting
of buttons, 118, 121, 233
of text, 29
Hint bits for searching, 30
Home buttons, 21
Home card, 14–15, 18
adding new stacks to, 169–175
Home command, 18–19, 383
Home stack, 14–15, 18, 169
Horizontal screen coordinates, 323, 331, 341
HyperCalc stack, 23, 191, 332
HyperCard
as bulletin board, 1–3
as software erector set, xiii-xvii
Hypermedia, HyperCard as, 366–367
HyperTalk scripts, 184–185, 194

I-beam cursor and editing, 27, 33, 40, 133, 384
Icon resources and buttons, 416
Icons, 198
for buttons, 80–84
copying buttons with, 119
dialog box for, 81
Idle message, 320
IDs, 190, 206–207
for buttons, 79
for cards, 204
If . . . then . . . else control structure, 212–213
If statements, 242
Import MacPaint command, 360
Import Paint command, 266, 360, 378–379
Importing, 379
of pictures, 360
of text, 355–358

Indentation with scripts, 242–243
Indexer
  pictorial, script for, 350–351
  stack, script for, 347–350
Information base, personal, 92–97
Inheritance, 377
  of backgrounds, 91
  of fields, 32–33
  and scripts, 315–318
Insertion point
  and editing, 29, 33
  and Field tool, 167
  and Message box, 57, 188
Interaction and HyperCard, xiv-xv
Into, with Put command, 237
Invert command, 388
Iris open and close effect, 201, 299–300, 412
Irregular polygon tool, 386–387
It local variable, 245, 326, 341, 397
  and Ask command, 398
  and Get command, 403
Items, 324

Kaehler, Carol, and HyperCard project, xvii-xix
Kaehler, Ted
  music script by, 340–342
  quoted, 314
Keep command, 44–45, 157, 390
Kelly, Kevin (quoted), 123
Kerns, Charles (quoted), 228
Keyboard
  musical, 340–341
  for navigation, 20–21

Labels, printing of, 176–178
Largest number in list, script to find, 337–339
Lasso tool, 385
  compared to selection rectangle, 47–49, 278
Last command, 19, 384
Layers
  and backgrounds, 275–291
  temporary, 63–64
Learning and HyperCard, 367–368
Levels for objects, 222
Lighten command, 49–50, 389
Line Size command, 391
Lines, 212
  drawing of, with Shift key, 108
  space between, 382
Lines tool, 385
Linking and LinkTo command, 106–107, 174–175, 204–205
  of buttons to cards, 81–84, 105–111
  and folders, 254
  script for, 207–209
  and stacks, 131–132
LinkTo button, 81–83
Lists, finding largest number in, 337–339
Local variable it, 245, 326, 341, 397
Lock screen command, 256
Lock Text command, 71
Locked fields, 71, 348
Logical references, 204–207
Logos, adding of, to stacks, 154–157
Long date format, 399
Long time format, 399
Look for Stacks In card, 130–131, 254, 407
Loops, 327, 331

MacPaint
  compared to Paint tools, 38
  history of manual for, xxiii-xxiv
  importing and exporting files of, 360, 378–379
Margins, alignment of, 153–154
MaxLine() function, 338–339
Memory requirements, HyperCard, 418
Menu bar, 59, 144
Menus, 373–374, 376–395. *See also* specific menus
Merging of stacks, script for, 329–331
Message box
  and command key dragging, 28
  as container, 202
  entering commands in, 31
  and Go command, 117
  and insertion points, 57, 188
  and scripts, 185–194, 250
  testing commands in, 245
Message handlers
  and control, 200–202
  hierarchy of, 225–226
Message window, 376
Messages
  and cards, 192
  script to locate definition of, 335–336
  sending of, 246
  system, 223–225, 318–321
Miniature pictures, 142
  creation of, 170–173
Minimalism and HyperCard, 370
MiniServant, 415–416
Modems and Dial command, 400–401
Monthly calendar, creation of new, 111–121
Mouse
  location of, 187
  system messages for, 318–319
MouseDown command, 200
MouseUp command, 200–201
Moving
  of buttons, 121, 148–149
  of text, 33
Multiply command, 405
Music. *See* Sound

Names, 188–191, 202
  for buttons, 232–233
  and case sensitivity, 201
  for fields, 162, 164, 167
  quotation marks for, 140–141, 195, 224
  for stacks, 95–96, 113–114, 127–128, 195
Naming hierarchies, 203–204
Navigation
  with buttons, 75–77
  and Go menu, 18–19, 383–384
  with keyboard, 20–21
Networks, 360
New Background command, 287–288, 395
New Button command, 77, 232, 395
New Card command, 34, 101–102, 289, 381
New Field command, 97, 395
New Stack command, 90, 94–96, 126–127, 377
  and backgrounds, 113, 128
  compared to Save a Copy command, 91
  See also Building of stacks
Next command, 115, 383
Notes, musical, 339–342, 361–362, 406–407
Numeric values, 238

Objects, 128
  IDs for, 206
  and inheritance, 315–318
  names of, 188–191, 202
  numbering of, 167
  order of, 279–281
  scripts for, 219–226
Objects menu, 77, 128–129, 191, 219–226, 393–395
Octaves in music, 340, 406
On statements, 200, 218
Opacity
  and backgrounds, 276–278
  and object order, 280–281
  of pictures, 37, 53–55, 61–63, 110–111, 381
Opaque command, 62, 390
Open command, 355, 405
Open printing command, 406
Opening of stacks and Open Stack command, 15, 18, 169, 377
Option key
  for borders, 385–387
  for copying, 66–67, 106–107, 163–164
  for drawing shapes, 386
  for lines, 385
Option-command for buttons, 118
Option-D for data, 109–110
Option-O for opaque, 110
Option-Return for lines, 212
Option-shift for copying, 66–67, 163–164
Options menu, 391–392
Ordering of objects, 279–281. *See also* Sorting
Outlining of buttons, 118
Oval and circle tool, 387

Page Setup command, 379
Paint bucket tool, 387
Paint menu, 388–390
Paint text, 149–157
  alignment of, 295–299
  for common backgrounds, 151–155
  and "real" text, 102–103, 291–299
  for text styles, 74
  tool for, 387
  See also Text
Paint tools
  adding pictures to stacks with, 37–67
  user level for, 38–39
Paintbrush tool, 386
Pascal
  and HyperCard, 317
  and XCMD, 336
Pass command, 316
Passwords
  and Protect stack command, 378
  and stack privacy, 307
Paste Button command, 120
Paste command, 381
Paste Picture command, 64, 145, 171
Paste Text command, 103
Pasting
  and Paint text, 291–292
  of pictures, 50–53, 64, 299
  in scripts, 216
Pathnames and StackInfo command, 255
Pattern window, 375
Patterns menu, 375, 393
Pauses, 412
Pencil tool, 108, 385
Periods with musical notes, 340, 361, 406
Permanent records, 251–262
Personal finances stack, creation of, 92–97
Phone lines, 360
  and Dial command, 401
Phone numbers, consistency in entering, 134
PickUp command, 388–389
Pictorial indexer, script for, 350–351
Pictures, 16–17, 145
  adding of, to weekly calendar, 38–45
  alignment of, 295–299
  background vs. card, 55–60
  borrowing of, 45–55
  and Card Ideas stack, 140
  copying of, 48
  design considerations for, 267–275
  enlarging of, 63
  importing and exporting of, 360
  miniature, 142, 170–173
  opaque or transparent, 53–55, 171
  Paint text as, 150–152
  using Paint tools to add, 37–67
  pasting of, 50–53, 64, 299
  to represent new stacks, 170–173

shrinking of, 62–63
with text, 265–273
See also Background pictures; Card pictures
Pitch, musical, 361–362
Play command, 339, 342, 361, 406–407
Plots stack, 337–339
Polygon Sides command, 392
Popping of cards and Pop Card command, 20, 333–335, 407
Pop-up fields, 308–311
and Paint text, 295
Pound sign for musical sharps, 361, 406
Power Keys command, 40, 49–50, 55, 391
Previous command, 19, 115, 383
Print Card command, 175, 379
Print command, 407–408
Print full size cards check box, 176
Print one card per page of paper check box, 176
Print Report command, 177–178, 260–262, 380
and object order, 282
Print Stack command, 175, 379
Printing, 407–408
of cards, 175, 379
page setup command for, 379
of stacks, 138, 175–178, 260–262, 379, 406
and tab order, 280, 282
Privacy of stacks, 307–308
Private Access check box, 307
Properties
of fields, 97–100, 158, 324, 384
of objects, 189
and Set command, 409
of text, 158
Protect Stack command, 160, 307, 309, 378
Pulse dialing, 400
Pushing cards and Push Card command, 20, 333–335, 409
Put command, 237–238, 408–409

Quit HyperCard command, 380
Quit system message, 321
Quotation marks
in answer commands, 212
for names, 140–141, 195, 224, 236

Read from file command, 409
Recent command, 19, 383
Records, permanent, 251–262
Rectangle tool, 325, 385
Rectangles, script to draw, 325. *See also* Selection rectangle
Redfern, James
bar code script by, 345–347
pictorial index script by, 350–351
References, absolute vs. logical, 204–207
Regular polygon tool, 388
Repeat command, 327, 331
Replacement of text, 34, 133, 186
ResEdit, 198, 417
Reset Paint command, 410
Resources, 198–199, 362–363, 414–417
Resume system message, 321
Retracing steps, 19
Return arrows, 283
Return key
in dialog boxes, 233
and Message box, 193
for new lines, 212, 242, 250
ReturnKey message, 320
Revert command, 44–45, 157, 390
Rotate Left command, 389
Rotate Right command, 389
Round rect check box, 79
Rounded rectangle tool, 386
Rounded rectangles, buttons as, 232

Save a Copy command, 10, 31, 377
compared to New Stack command, 91
Scope of find command, 343–344
Scrapbook desk accessory, 354
Screen
coordinates of, 324, 341
displaying windows on large, 331–332
locking and unlocking of, 256
and Tools window, 375
Script command, 199
Scripting user level, 198
Scripts, 183–185, 207–209
and buttons, 194–202
commands in, 323
going to stacks with, 226
and hierarchies, 203–204, 225–226
and inheritance, 315–318
and Message box, 185–194, 250
modification of, 209–219, 235–240, 402
and objects, 202
and Objects menu, 219–226
placement of, 220–223, 315
and references, 204–207
Scroll visual effect, 412
Searching, 65
and object order, 283
and Paint text, 102, 149, 292
script for, 326–328
See also Find command
Seconds time format, 399
Select All command, 388
Select command, 156, 388
Selection, message box, 186
Selection rectangle, 384
for checking alignment, 173
compared to lasso, 47–49, 278
Send command, 246, 316
Send Farther command, 167, 262, 279–281, 394
Serial port and Dial command, 400
Set commands, 324, 409

Shadowed fields, 270
Shape of buttons, 231
Shared text and Paint text, 293–295
Sharing
  of buttons, xvii
  and HyperCard, 3–4
Sharps, musical, 361, 406–407
Shift key
  for aligning copies, 66–67, 107, 149, 163–164, 234
  with buttons, 234–236
  and circles, 387
  to draw straight lines, 108, 385
  for miniature pictures, 142, 171
Short date format, 399
Short time format, 399
Show command, 308, 410
Show Lines command, 272
Show name check box, 79, 162, 232, 305
Shrinking
  of buttons, 148
  of Paint text, 156–157
  of pictures, 62–63
Simplicity and graphics, 273–275
Size
  of buttons, 148, 231, 233–234
  of fields, 97–99
  and Paint text, 387
  of stack, 129
  of text, 382
Slanted lines in menu bar, 59
Slow, with visual effects, 411–412
Sort button, 137, 146, 210
Sort command, 410
Sorting
  and data consistency, 26, 133
  by date, 214–219
  by fields, 312
  and Paint text, 292
  script for, 210–212
  of stacks, 136–139
Sound, 361–363, 406–407
  music, 24, 339–342
  resources for, and MiniServant, 415–416
Space between lines, 382
Space free, stack, 129–130
Spaces and names, 201
Speaker, Beep command for, 398
Split-page format check box, 176
Spraypaint tool, 386
Spreadsheets, 23
  script for, 332
Stack buttons, hiding of, 283–285
Stack Ideas stack, 89
  and backgrounds, 140
  and calendar alternatives, 111
  starting stacks with, 92–94
Stack Info command, 128–130, 132, 191, 394
  and pathnames, 255
Stacks, 14
  adding backgrounds to, 287–289, 395
  adding buttons for, 173–175
  adding cards to, 34, 100–105, 132–135, 336–337, 381
  adding resources to, 414
  cards in, 16–17
  compaction of, 129–130, 377–378
  copying of, 10, 90–91, 377
  creation of, 94–97, 125–128, 252–255
  deletion of, 378, 381
  directory of, 18, 169
  expanding capabilities of, 228–263
  going to, with scripts, 226
  indexer for, 347–350
  merging of, script for, 329–331
  multiple, 290–291
  names for, 95–96, 113–114, 127–128, 195
  new, adding of, to Home card, 169–175
  opening of, 15, 18, 169, 377
  printing of, 138, 175–178, 260–262, 379, 406–408
  privacy of, 307–308
  protection of, 160, 378
  push-down, 334
  saving of, 10, 31, 91, 377
  size of, 129
  sorting of, 133, 136–139
  system messages for, 320–321
  transmission of, 360
  See also Building of stacks; Customization of stacks; Designing of stacks
Stackware, 4
StartUp message, 331
Strings, concatenation of, 342
Style
  button, 79
  of fields, 97–99
  and Paint text, 387
  of text, 74, 156, 158
Subtraction and Subtract command, 240–245, 412
Suspend system message, 321
System clock, and the date function, 224
System file, and fonts, 72, 151, 292
System messages, 223–225, 318–321
System requirements, 418–419

T, for musical triplets, 361
Tab key
  for finding fields, 96, 133, 166–167, 238
  and object ordering, 280, 282
  for script checking, 218, 242
TabKey message, 320

Target script, 333
Tear-off windows, 42–43
Tempo in sound, 339, 362
Temporary layers, 63–64
Text, 17
  adding of, to fields, 31, 71–72
  compared to Paint text, 102–103, 291–299
  copying of, 33
  customizing of, 72–74
  deletion of, 34, 400
  editing of, 32–34
  with graphics, 265–273
  importing and exporting of, 355–359
  moving of, 33
  and New Stack command, 128
  properties of, 97–100, 158
  replacement of, 34, 133, 186
  searching of, 25–31, 65
  styles of, 74, 98, 158, 324, 382
  See also Paint text
Text fields. *See* Fields
Text Style command, 73–74, 98, 324, 382
The clickloc function, 341
The date function, 185, 224
The loc function, 323
The loc of msg message, 332
The long date function, 186
The mouseLoc function, 187
The name function, 189–190
The short name function, 333
The sound function, 362
The time function, 185
The tool function, 188
This Card with LinkTo, 82–83
This Stack with LinkTo, 82
Tilde key to retrace steps, 19–20
Time, 185–186
  and Convert command, 399
To, 195
  with visual effects, 300
To Do list, 52
Tools
  choosing of, 323–325, 398
  current, 188
Tools menu, 40, 374, 384–388
Tools window, 375
Trace Edges command, 49–50, 389
Transparency of pictures, 37, 53–55, 171, 381
Transparent command, 54, 64, 276, 278, 390
Triplets, musical, 361
Tunes. *See* Sound
Type command, 411

Underlining, 270, 272
Undo command, 43–44, 380
  and card deletion, 136
  for painting actions, 58, 144
Unlock screen command, 256
Unlocked fields, 71
User level and Preferences card, 6–7, 32, 129, 169, 198
  and menus, 374
  and Objects menu, 220
  for Paint tool, 38–39
  and pop-up fields, 309
  Power Keys preference on, 49
  and Protect Stack command, 378
User-defined commands, 413

Variables
  as containers, 341
  global, 330, 403–404
  it, 245, 326, 341, 397
Venetian blinds effect, 299, 412
Vertical coordinates of screen, 324, 331, 341
Very fast, with visual effects, 411–412
Very slow, with visual effects, 411–412
View button, 118
Visual effects and Visual command, 200, 299–300, 411–412

Wait command, 412
Weekly button, 111
Weekly calendar, adding pictures to, 38–45
White space, adding to, 64–67
Windows, 375–376
  displaying of, on large screen, 331–332
  tear-off, 42–43
  See also Destination window
Winkler, Dan, spreadsheet script by, 332
Wipe up or down effect, 299–300, 412
With, for answer commands, 212
Word, with Find command, 403
Write command, 412–413

XCMD functions, 336
  and MiniServant, 415–416
XFCN functions and MiniServant, 415–416

Zoom in and out effect, 299–300, 412
Zoom lens, FatBits command as, 391

CAROL KAEHLER

Addison-Wesley Publishing Company

HyperCard™ Power

Techniques and Scripts

"Carol Kaehler knows the tips and tricks inside HyperCard, and she shares them with you in this book. In an afternoon with Carol, you could learn a lot."

—Bill Atkinson, creator of HyperCard

HyperCard™ is Apple® Computer, Inc.'s, revolutionary new Macintosh™ program that organizes and molds information. As a member of Apple's HyperCard development team, Carol Kaehler brings an insider's experience to ***HyperCard Power: Techniques and Scripts,*** revealing hints, tips, and tricks for customizing HyperCard to take advantage of its amazing power and flexibility. The book goes far beyond an introduction to HyperCard's features to teach you the skills needed to create your own custom HyperCard applications. In an informal style it provides simple, practical, hands-on lessons in every chapter.

The first part of ***HyperCard Power: Techniques and Scripts*** focuses on using HyperCard, including how to browse and customize "stacks," the ordered collections of cards that constitute a HyperCard database. The second part teaches you how to become a HyperCard author by building and designing new stacks. You will learn everything you need to know about writing "scripts" in the HyperTalk language: how to create them, how to use them, and how to plug in and customize preexisting scripts.

HyperCard Power: Techniques and Scripts is jam-packed with useful tips and tricks, and it includes complete, usable scripts for merging stacks, adding a new card, generating bar codes, creating automatic indexes, creating a pictorial index, and much more. It also includes step-by-step tutorials to show you how to:

- use pictures, buttons, and sounds to enhance your stacks
- copy and paste buttons and sounds
- write scripts to link buttons, include visual effects, sort cards, and more
- use FatBits to work on fine detail in Paint documents
- import and export text to and from other applications

Useful appendices include HyperCard windows and commands, and an index summarizes all of the hints contained in the book.

Carol Kaehler has over five years' experience as a technical writer at Apple Computer, Inc. She is the author of the popular MacPaint and MacProject manuals and of the award-winning Macintosh Plus owner's manual. Carol was a part of the HyperCard development team and, working with Bill Atkinson, wrote and designed the HyperCard on-line Help system.

9 780201 067019 51795