

EVERYTHING
you need to create multimedia

Multimedia Starter Kit

for Macintosh®

**All the Best
Software and Advice!**

Michael D. Murie

- Develop multimedia projects
- Try out working demos
- Integrate text, graphics, sound
- Explore environments
- Add 3D and animation
- Get product info

Multimedia Starter Kit for Macintosh



Hayden
Books

Multimedia Starter Kit for Macintosh

Michael D. Murie

Multimedia Starter Kit for Macintosh

©1994 Hayden Books, a division of Macmillan Computer Publishing

All rights reserved. Printed in the United States of America. No part of this book may be used or reproduced in any form or by any means, or stored in a database or retrieval system, without prior written permission of the publisher except in the case of brief quotations embodied in critical articles and reviews. Making copies of any part of this book for any purpose other than for your own personal use is a violation of United States copyright laws. For information, address Hayden Books, 201 West 103rd Street, Indianapolis, Indiana 46290.

Library of Congress Catalog Number: 94-77937

ISBN: 1-56830-113-8

This book is sold as is, without warranty of any kind, either expressed or implied. While every precaution has been taken in the preparation of this book, the publisher and author assume no responsibility for errors or omissions. Neither is any liability assumed for damages resulting from the use of the information or instructions contained herein. It is further stated that the publisher and author are not responsible for any damage or loss to your data or your equipment that results directly or indirectly from your use of this book.

96 95 94 4 3 2 1

Interpretation of the printing code: the rightmost double-digit number is the year of the book's printing; the rightmost single-digit number is the number of the book's printing. For example, a printing code of 94-1 shows that the first printing of the book occurred in 1994.

Credits

Publisher

David Rogelberg

Managing Editor

Karen Whitehouse

Developmental Editor

Marta J. Partington

Copy/Production Editor

Meshell Dinn

Technical Reviewer

Cathy Clarke

Publishing Coordinator

Stacy Kaplan

Interior Designer

Fred Bower

Cover Designer

Jay Corpus

Indexer

Chris Cleveland

Production Team

Dan Caparo, Brad Chinn, Kim
Cofer, Lisa Daugherty, David Dean,
Jennifer Eberhardt, Erika Millen,
Beth Rago, Bobbi Satterfield, Karen
Walsh, Robert Wolf

To Our Readers

Dear Friend,

Thank you on behalf of everyone at Hayden Books for choosing *Multimedia Starter Kit for Macintosh* to help you learn about the exciting world of multimedia on the Macintosh. We think you'll enjoy the examples in this book and on the disk, while getting a true understanding of the conceptual nature of multimedia.

What you think of this book is important to our ability to better serve you in the future. If you have any comments, no matter how great or small, we'd appreciate you taking the time to send us email or a note by snail mail. Of course, we'd love to hear your book ideas.

Sincerely yours,



David Rogelberg
Publisher, Hayden Books and
Adobe Press

You can reach Hayden Books at the following:

Hayden Books
201 West 103rd Street
Indianapolis, IN 46290
(800) 428-5331 voice
(800) 448-3804 fax

Email addresses:

America Online:	Hayden Bks
AppleLink:	hayden.books
CompuServe:	76350,3014
Internet:	hayden@hayden.com

For Susan...

Dedication

About the Author

Michael D. Murie

Michael Murie is a co-author of Hayden's *QuickTime Handbook*, a Macintosh media consultant, and is president of Multimedia Workshop.

When Michael's not trying to pay the bills by developing animations and interactive programs for clients, or by writing for *MacWEEK* and *New Media* magazines, he works on mad, crazy projects with no commercial appeal. Michael's favorite group is The Beatles: "I always liked the white album because of the diversity of the material, but I can appreciate Sgt. Pepper's place in history."

An acknowledged Macintosh freak, Michael is *still* working on the motion picture *The Catnappers*. He has no spare time.

Acknowledgments

As Claude Raines said in *Casablanca*, "Round up the usual suspects!" Thanks to all my friends and colleagues for their help and support both in this book and other ventures.

I'd like to thank those who worked on the first edition of this book when it was called *The Macintosh Multimedia Workshop*, particularly Laura Wirthlin, Dave Ciskowski, and Bob Hone. Their hard work made the first edition a success. And now we have the new edition.

To everyone at Hayden who worked on this edition and made it possible — thanks. I'd especially like to thank David Rogelberg, Karen Whitehouse, Meshell Dinn, Cathy Clarke, and Fred Bower.

As always, I'd like to thank my editor, Marta Partington, for her tireless efforts and understanding. She remained unfrazzled in the face of my continued tardiness.

Kent Borg and David Drucker provided demonstrations of Mosaic. Ric Ford constantly supplied me with new information, some of it actually useful. He's a useful drummer, too. Thanks to The Grimster. The support of my friends make this and other ventures possible.

Particular thanks to the representatives from the many companies that provided demos and samples.

Extra special love and affection to Skye.

Trademark Acknowledgments

All products mentioned in this book are either trademarks of the companies referenced in this book, registered trademarks of the companies referenced in this book, or neither. We strongly advise that you investigate a particular product's name thoroughly before you use the name as your own.

Apple, Mac, and Macintosh
are registered trademarks of
Apple Computer, Inc.

Table of Contents at a Glance

Part I Introduction

- 1 What Is This Book?**
- 2 What Is Multimedia?**

Part II Creating Multimedia Projects

- 3 Developing a Multimedia Project**
- 4 Dissecting a Multimedia Project**

Part III Multimedia Tools

- 5 Hardware**
- 6 Text**
- 7 Graphics**
- 8 Animation**
- 9 3D**
- 10 QuickTime**
- 11 Sound**
- 12 Clip Media and Copyright**

Part IV Multimedia Environments

- 13 Choosing the Right Development Environment**
- 14 Interactive Presentations**
- 15 Authoring Environments**

Part V Multimedia Workshops

- 16 Putting It to Work**
- 17 Putting Director to Work**

Appendix: What's on the Disc

Index

Contents

Part I	Introduction	2
1	What Is This Book?	3
2	What Is Multimedia?	7
	So What <i>Is</i> Multimedia?	8
	What Is Multimedia “Good For”?	9
	Can You Do Multimedia?	10
	How Do You Get Started?	11
Part II	Creating Multimedia Projects	14
3	Developing a Multimedia Project	15
	The Steps of Development	16
	Having the Idea	17
	Example: electronic newsletter	17
	Establishing the Constraints	17
	Obvious constraints	18
	Cost	18
	Time	18
	Materials	18
	The not-so-obvious constraints	19
	Who will use the product?	19
	What will they use it on?	20
	QuickTime	21
	Color	21
	What version of the System?	21
	How much memory?	22
	Variations in playback speed	22
	Screen size and bit depth	22
	How will the product be delivered?	23
	Unique constraints	24
	Planning the Production	26
	The authoring environment	27
	The data	28
	The user interface	28
	But how to get started?	29
	The Macintosh OnLine Reference	31
	The structure of the Macintosh OnLine Reference	31
	Evaluating the Macintosh OnLine Reference	33

QuickTime Intro News	35
The structure of the QuickTime Intro News	36
Evaluating the QuickTime Intro News	37
Titles Sampler	38
The structure of the Titles Sampler	40
Evaluating the Titles Sampler	41
Multiple Media Tour	41
Macintosh Human Interface Guidelines Companion	43
It's Time to Start Designing	45
Creating the Prototype	49
Testing the Prototype	50
Making the Final Delivery	51
All the Things We Forgot	52
Things to Watch	52
Moving on	53
4 Dissecting a Multimedia Project	55
Overview of the Project	56
How I became involved	57
Planning the first prototype	57
Reviewing the first prototype	59
Switching development tools	61
The final phase	62
Producing the final version	65
Considering the Future	67
Part III Multimedia Tools	70
5 Hardware	71
A Brief History of the Macintosh	72
The original Macintosh	73
The color, expandable Macintosh II	74
Quadra 630	75
The Portable, the PowerBooks, and the Duos	75
The Performas	77
The AV Macintosh	77
Processor and clock speed	78
Enter PowerPC	78
Emulation	79
Sound	79
Which chip?	80
The future	80

Choosing the Right Macintosh	80
The multimedia user's Macintosh	81
The multimedia developer's Macintosh	81
Other Hardware	82
Graphics display boards and graphics accelerators	82
Storage devices	83
Hard disk drives	83
Removable media	84
CD-ROM	85
Laser discs	86
Still other hardware	88
The Future	88
6 Text	89
A Brief History of Text on the Macintosh	90
What Text Communicates	94
Text	94
Hypertext	94
Getting around in hypertext	96
Getting back to where you were in hypertext	97
Using Text in Productions	98
Fonts	98
Hyperfonts?	101
Internet HyperText	101
Disk Utilities	104
7 Graphics	107
A Brief History of Graphics on the Macintosh	108
Tools of the Trade	114
Graphics editing programs	115
Basic object-oriented graphics programs	116
PostScript graphics programs	117
Graphs or graphics?	117
File formats and applications that work with them	117
MacPaint	118
PICT	118
TIFF	118
RIFF	118
EPS	119
Targa	119
GIF	119

	BMP	119
	JPEG.....	120
	Other Sources of Graphic Images	122
	Video images	123
	Still video images	124
	QuickTake 100	126
	Scanned images	127
	Kodak Photo CD images	129
	What is Photo CD?	129
	How does Photo CD work?	129
	How do single-session and multi-session drives differ?	130
	Clipped images	131
	Unusual Graphic Tools	131
	Pressure-sensitive drawing tablets	135
	The Future	135
8	Animation	137
	Working with Animation	139
	A few principles of animation	139
	A few problems of animation	141
	File formats	143
	Some Animation Tools	144
	Life Forms	144
	PROmotion and ADDmotion	146
	Animation Works	148
	Cinematic	149
	MovieWorks	150
	Other animation packages	151
	The Future	152
9	3D	153
	The Three Functions of 3D Modeling	154
	Basic modeling	155
	Extrusion	156
	Lathing	157
	Sweeping	158
	Freeform	158
	Advanced Freeform	159
	Exchanging Models—DXF format	160

Rendering	161
Shading	162
Wireframe	162
Flat shading	162
Gouraud shading	163
Phong shading	163
Ray Tracing	164
Surfaces	164
Procedural surfaces	165
Texture mapping	165
Lighting	167
RIB format and RenderMan	168
Network and accelerated rendering	169
Power Macintosh	170
Animation	170
Creating animations	170
PICT, PICS, and QuickTime	172
The Categories of 3D Applications	172
Architectural modeling 3D applications	172
Simple extrusion 3D programs	173
General purpose 3D applications	175
Virtual Reality	175
Considerations When Choosing a 3D Application	176
A Look at Some 3D Applications	177
Ray Dream Designer	177
Specular International Infini-D	180
Strata Inc's StrataVision 3D	182
Macromedia Three-D	183
Alias Sketch!	184
All the rest?	185
A couple of unusual tools	185
Hardware Requirements	185
The Future	186
10 QuickTime	187
Understanding the Importance of QuickTime	188
What's so neat about digital video?	191
What's the difference between animation and video?	191
How does QuickTime work?	193

Installing QuickTime and Playing Movies	194
Recording Video	195
The compressor's role in digitizing	197
QuickTime compressors	199
Some tips for compressing video clips	199
Selecting and Using Digitizing Hardware	200
Video digitizing boards	200
SuperMac's VideoSpigot	201
RasterOps boards	202
Radius VideoVision	205
Hardware acceleration	206
Other compression algorithms	207
Audio	207
QuickTime VR	208
Using QuickTime Movies	208
Play them	208
Edit them	209
Using Programs That Support QuickTime	211
Utilities	212
Editing	213
Post-processing tools	215
Cross-platform Applications	217
The Future	217
11 Sound	219
Storing Sound	220
Macintosh built-in sound hardware	220
Macintosh sound quality	222
Lower sample rates	222
Sound compression	223
Recording Sound	223
Editing Sound	228
Saving Sound and Using Sound in Other Applications	229
Using Advanced Sound Techniques	230
Get the best sound at 11 kHz	230
Use a tape recorder	231
Watch out for clipping	231
Avoid recording too low	231
Get a good original for better sound	232
Additional Hardware	232

	Using CD	232
	Using MIDI	234
	Making the Macintosh Speak	237
	Looking Ahead	238
12	Clip Media and Copyright	239
	Stock Houses	240
	Clip Media	241
	Licensing	244
	Media Cataloging Utilities	245
	Creating Your Own Clip Media	246
	Copyright Laws	246
	Copyright and your work	246
	Obtaining Copyright for your work	246
	Employee and Work for Hire	247
	Copyright and the work of others	248
	Releases	248
	The Future	249
Part IV	Multimedia Environments	252
13	Choosing the Right Development Environment	253
	The Different Environments	254
	Slide presentations	254
	Interactive presentations	256
	Authoring programs	256
	Custom programming	257
	QuickTime	257
	The Decision	257
14	Interactive Presentations	259
	Understanding Presentation Programs	260
	Using Interactive Presentations	261
	Using Action! and Special Delivery	261
	Action!	261
	Special Delivery	264
	Producer Pro	266
	Authorware	268
	Apple Media Tool	269
	The Future	269
15	Authoring Environments	271
	HyperCard—What Is It?	272

	How does it work?	272
	Fields	274
	Backgrounds	279
	Buttons and scripts	281
	HyperTalk	284
	AppleScript	286
	Color	287
	What is it good for?	289
	Where to go from here?	290
	The future	291
	SuperCard—What Is It?	291
	How does it work?	291
	What is it good for?	293
	Where to go from here?	294
	The future	295
	Director—What Is It?	295
	How does it work?	296
	What is it good for?	301
	Where to go from here?	303
	Other Programs to Consider	303
Part V	Multimedia Workshops	306
16	Putting It to Work	307
	What is the Project?	308
	Getting Started	308
	Step 1: Creating a New Stack	310
	Displaying color graphics	310
	Adding a Quit button	314
	Step 2: Adding Another Background	315
	Add a second background	316
	Add a background field and buttons	316
	Step 3: Adding Content	319
	Add a title	320
	Copying text	320
	Formatting text	323
	Step 4: Adding a Picture to a Card	324
	Step 5: Adding the Other Cards	324
	Manually wrapping text	325
	Cards 5 through 10	326

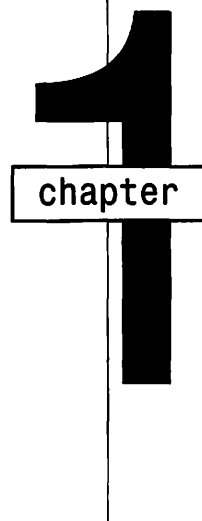
Step 6: Some Finishing Touches	329
Adding a content list	331
Things That Haven't Been Done	333
Moving On	334
17 Putting Director to Work	337
The Project	338
Getting Started	339
Importing graphics	340
Placing a graphic on the Stage	342
Arranging more objects	345
Creating the opening	347
Adding effects and setting the Tempo	349
Adding Scripts	351
Navigating within a Movie	353
Adding the underground	355
Problems!	357
Adding an information page	358
Advanced Scripting	360
Where to from here?	363
Appendix What's on the Disc	365
Tour the Disc	366
QuickTime	366
Cover Art	367
Software Developers	367
Vividus	367
Specular International	367
Playmation	367
No Hands Software	368
Virtus	368
Heizer Software	368
Fractal Design	368
Multiple Media Tour	368
Strata	368
Adobe	369
Morph	369
PROmotion	369
VideoFusion	369
VideoToolKit	369

CameraMan	369
Kodak	369
Michael's	370
QuickTime Handbook movie	370
Now & Zen	370
Farmers Market	370
Dogcow	370
Interactive Skye	371
Exploring Multimedia BBS	371
EM BBS Demo Movie	371
Exploring Multimedia BBS	371
Getting Started with TF	371
TeleFinder/User	371
TF Prefs 1 bit	371
Library	372
Images	372
QuickTime clips	372
Sounds	372
Book chapters	372
Graphics	372
Pictures	373
Photo CD stuff	373
PCD1642 Slide Show	373
Photos	373
Animation	373
3D	373
QuickTime	373
Image Test	373
Movie Test	374
Sound	374
Sound Quality	374
Workshop	374
Index	375



Introduction

What Is This Book?



Welcome to the *Multimedia Starter Kit for Macintosh*! The goal of this book is to help those of you who are new to multimedia on the Macintosh get started “doing it yourself.” With the help of this book, you’ll soon be creating your own multimedia projects.

If you already bought the first edition of this book, *Macintosh Multimedia Workshop*, then you may be wondering what’s changed in this book. First of all, in the year since the release of the first edition, just about all the software applications have been updated. The book has been updated to reflect this. Second, there have been a few new software developments (Apple Media Tool), although we’re still waiting for some things to happen (Kalieda’s ScriptX).

You’ll see new hardware developments covered (most notably, PowerPC) and operating system changes updated (QuickTime 2.0, QuickDraw GX). You’ll also find a new chapter in the “Multimedia Section.” This chapter will take you through the basics of creating a multimedia project using Macromedia Director.

It’s our hope that these changes will make this book even better! As always, your comments about how we might improve the book are welcome.

The first section of the book, “Introduction,” begins with an introduction to the book (you’re standing in it) and then attempts to explain what multimedia is. Maybe you’ve seen this word splashed about and wondered what the hype is all about. Chapter 2, “What Is Multimedia?,” will try to answer all your questions (including some you didn’t even want to ask).

The second section, “Creating Multimedia Projects,” describes the steps involved in developing multimedia projects. Chapter 3, “Designing a Multimedia Project,” discusses how to go about designing a project, illustrating the principles with “case studies” of existing projects. Chapter 4, “Dissecting a Multimedia Project,” provides a retrospective look at a real-world project developed using the steps in Chapter 3. If this section seems like a lot to master immediately, don’t worry—just skip ahead to read about the types of media you can use in multimedia projects and come back to this section later.

The third section, “Multimedia Tools,” introduces the different media (text, graphics, animation, 3D, QuickTime, and sound) you can use in multimedia projects. The chapters in this section discuss how you can use and manipulate these media, and introduce some of the applications you may need to start working with them. (True, 3D isn’t really a medium, but it’s such a vast and complicated subject that I didn’t think I could spread it across graphics, animation, and QuickTime without a lot of confusion!) The section begins with Chapter 5, “Hardware,” which describes the equipment you need to run multimedia applications and to develop multimedia projects, and ends with Chapter 12, “Clip Media (and Copyright),” which provides information about using material created by others.

The fourth section, “Multimedia Environments,” introduces the environments in which you put together multimedia projects. Chapter 13, “Choosing a Multimedia Environment,” discusses in broad terms how to do just that. Chapter 14, “Interactive Presentations,” describes new developments in interactive presentation environments, including Action! and Special Delivery. Chapter 15, “Authoring Environments,” describes the authoring environments of HyperCard, SuperCard, and Director in more detail.

The last section, “The Multimedia Workshop,” shows you how to create a multimedia project from materials included on the accompanying CD-ROM. The first chapter does this with HyperCard, while the second chapter covers Multimedia Director. You’ll need a copy of Director or HyperCard in order to complete this chapter. When you finish these projects, you can apply the same ideas to create multimedia productions of your own.

The CD-ROM included with this book also contains some multimedia illustrations for the book, a variety of utilities, a selection of demonstration applications and sample multimedia projects from companies that sell multimedia products, and some examples of multimedia projects that I have created (most of them for my own amusement!). The appendix describes the contents of the CD-ROM in more detail.

It can be a daunting task to start developing a multimedia project. It is my hope that this book will lead you in the right direction and give you the encouragement to begin developing your own projects. Remember: start small, work a chunk at a time, and keep adding things until you have a complete project. Good luck!

The following conventions used in this book were established to help you use the book more easily.

Important terms appear in **boldface** type where they are defined.

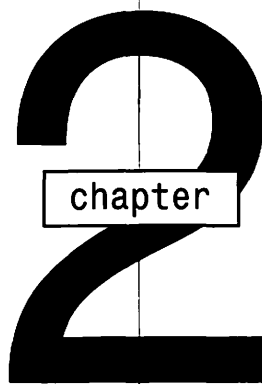
Information that appears onscreen and information that you type appears in a special typeface.

When you see two or more keys separated by a hyphen (-), you hold down the first key(s) as you press the last key.



Multimedia illustrations, utilities, demonstration versions of programs, and sample projects included on the accompanying CD-ROM are designated with this icon. Remember, the appendix describes the contents of the CD-ROM in more detail, and ReadMe files on the CD-ROM provide more information about specific items.

What Is Multimedia?



chapter

You could fill several pages of a book with answers to the question: “What is multimedia?” One of my favorite definitions is from a computer salesman, offered when he was setting up hard disks, VCRs, monitors, and other equipment for a demonstration at a local computer user group. When asked what multimedia was, he surveyed the pile of equipment he had just carried into the room and said: “Anything that requires more than two trips to the car!”

So What *Is* Multimedia?

Multimedia is often defined simply as the simultaneous use of more than one media type on a computer. That’s a very broad definition, which isn’t precise enough for many. The current use of the term multimedia is particularly curious considering that multimedia used to refer to a slide/sound presentation! Then there’s the issue of interactivity—some users think that you must have interactivity in your presentation to have multimedia. The truth is that multimedia has been co-opted by the computer industry, and it can be (and has been) used to describe just about anything!

The basic types of media available on a computer are: text, graphics, animation, video, and sound. Another part to many multimedia productions is interactivity—the capability for the user to interact with the multimedia production, causing things to happen or interrupting the flow of information.

Some of you probably are thinking that, by the definition above, your word processor qualifies as a multimedia tool because you can use it to combine two media types—text and graphics. But word processing and even desktop publishing is generally considered an exception to the rule. Why? Perhaps because this combination isn’t exotic enough to be deemed “multimedia.” Alternatively, a QuickTime movie, which may contain only animation and sound (or maybe *only* animation!), is considered multimedia.

Gee, only two paragraphs past my definition of multimedia, and it already begins to look like it doesn’t hold water.

In many ways, defining multimedia is like trying to define desktop publishing, but worse. Most people probably would define desktop publishing as the use of a product such as Aldus PageMaker or QuarkXPress to create

materials that will be mass produced. Writing a letter with Microsoft Word is not generally considered desktop publishing. But you *could* use Microsoft Word to create a flyer, brochure, or even a newsletter, which may well be considered desktop publishing.

So, writing a letter and placing a graphic in that letter probably does not qualify as creating multimedia, but writing a letter and attaching a QuickTime movie to that letter probably does!

Maybe a simpler definition of multimedia would begin “If it looks like a duck”

What Is Multimedia “Good For”?

Perhaps more important than any definition are examples of what you can use multimedia to do. Multimedia is most often used for presentations, training and educational programs, sales demonstrations and museum kiosks, games, and database front-ends. But multimedia also can be used for correspondence, artistic programs, and even wedding invitations! There’s really no end to what you can do with multimedia.

If a picture is worth a thousand words, what is a movie worth? Or an animation, or a piece of music? Rather than a text-only explanation of how to assemble a bicycle, why not have a step-by-step animation that shows how all the pieces fit together, and why not enable the user to click the animated parts to find out their part names and numbers? Similarly, an electronic encyclopedia may use many words to describe the sounds that whales make, without really communicating those sounds. Enabling the readers to click buttons to hear the sounds is a huge benefit. Video is even more powerful—when used correctly—whether as part of an encyclopedia or a sales presentation.

These possibilities are why multimedia has generated so much excitement, and why you need to know about multimedia and how to create it. Multimedia is not a buzz word for a nonexistent technology or a fad that will be around for a couple of years and then fade away. Multimedia exists and is being used for many applications already. In the future, multimedia will be integrated into more applications and become accepted as part of what a computer “does,” rather than being thought of as an unusual new feature.

What’s a front-end? “Front-end” is a term often used to describe an easy-to-use interface for a large collection of data that previously had been accessible only through a clumsy command-line interface.

Can You Do Multimedia?

Many multimedia productions available today, particularly commercial games and educational products, would be almost impossible for one person to produce alone. The combination of professional skills needed to produce a high quality production—graphics, animation, video production, sound engineering, and so on—are difficult to find in any one person.

By necessity, multimedia is often a collaborative art form. You may want to specialize in one or two areas (for example, sound and video, or maybe animation or 3D work). Or you may want to be the creative force and guiding light behind a multimedia project rather than get your hands dirty using the application software. This role is comparable to being the director or producer of a movie, and as the director or producer of the project you must bring together its different elements—some of which you have created yourself, others which have been created by people you've hired, or have been purchased from clip media collections.

Still, there are many situations where you may be able to produce the whole project yourself. Different needs have different requirements. A commercial product has a very different expectation attached to it than does an internal training product for a large corporation or an educational game for a school.

I don't want to scare you into thinking that a multimedia project is some large, difficult development project that is beyond your skills, time, and budget. In many cases, one person *can* produce the entire project—particularly if the production involves only one or two media. I want to encourage you to start developing your own projects while remaining aware of the possibilities for improving a project by working with others—whether those other people are colleagues in your department, friends, or professionals you hire to put the finishing touches on the production.

Designing and producing a multimedia project involves three major tasks, each of which can be done by one person or by several people. The first task is design—this is the process of coming up with the idea for the project and specifying how it will work. The second task is producing the different elements—for example, the art work, the text, and the movies that you will use in the project. The third task is implementation—taking the elements and putting them together based on the original design.

How Do You Get Started?

Whether you want to design, create, or implement a multimedia project, this book will help you. The next part of this book, “Creating Multimedia Projects,” describes the creative process—how to go about designing and analyzing a multimedia project. The third part, “Multimedia Tools,” describes the tools for creating and manipulating the individual media. The fourth part, “Multimedia Environments,” describes how to implement a project. The final part of the book, “The Multimedia Workshop,” provides an opportunity for you to create sample productions from elements provided on the accompanying CD-ROM. (You’ll need a copy of either HyperCard or Macromedia Director to complete this section of the book.)

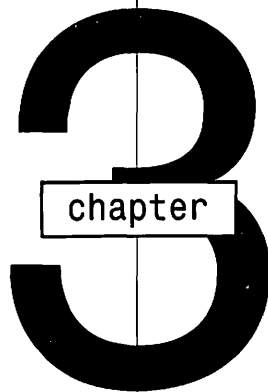
Can you do it yourself? Yes! And in the rest of this book, you’re going to learn how.

II

PART

Creating Multimedia Projects

Developing a Multimedia Project



There's more to developing a multimedia project than having an idea, knowing how to use the tools, and putting it all together. You must consider design—graphics, pictures, and user interface elements, such as buttons and menus—as well as content preparation and implementation of the project. At the same time you must consider how users will use your project and whether they will be able to get the information they need from your product without confusion. All these issues can make the task of creating a multimedia project seem daunting to the beginner.

But you can do it! This chapter provides an overview of the process; a step-by-step guide to developing multimedia projects from beginning to end. The chapter also introduces a specific example—an electronic newsletter—the development of which is described in greater depth in Chapter 16. Hopefully, by following this example, you will be able to take the principles and apply them to your own development efforts.

Of course, you also need to know which tools (applications) are available and how to use them. This topic is covered in the two separate sections on Multimedia Tools and Multimedia Environments. Although you can read the chapters in this book in any order, if you are a beginner you may want to read through this chapter and the following chapter quickly first, and then read the sections on the specific applications. Then, when you are ready to actually create your own project, you can reread this chapter again.

The Steps of Development

Having worked on a number of different projects, I have come up with the following steps in the development of a multimedia project:

1. The idea
2. Establishing the constraints
3. Planning the production
4. Creating the prototype
5. Adjusting the prototype
6. Final delivery of the project

This outline is not some hard and fast formula I have spent years refining. Rather, it's an assessment of the steps I usually go through when working on a project. Each project has its own quirks, of course, and this outline is only the most common steps of development. You may work differently, and that's okay, too. But one thing I encourage you to do is to start small and build up. That way you can have some success ("We finished the first phase!"), which will give you the incentive to keep going. It will also confirm that you are heading in the right direction—something that's important to know when you are doing work for others.

Having the Idea

Every good project starts with an idea. You have to conceive of some idea, goal, or purpose before you can start working on a project. The clearer the idea, the easier the rest of the development should be. Unfortunately, ideas are often kind of fuzzy—"I want to create a neat product"—sums up the goal of many development projects.

Example: electronic newsletter



For the sample project in this chapter, we are going to create an electronic newsletter. This idea is logical because a) I already have the material I can use as an example, and it is in electronic format; b) This is the kind of thing that might be distributed by modem or on disk; and c) It will be a good example for the book! The materials and the complete example can be found on the CD-ROM included with this book.

Pretty simple, really. Now that we have the idea, it's time to move on to the next stage.

Establishing the Constraints

You have probably already started thinking about how this product might work and look. That's great! But before you get carried away with the details, there are some more things you need to know. What are the

constraints for the project? Whether you are developing something for yourself, your company, or a client, every project has constraints that will affect the final results. Some constraints are more obvious than others.

Obvious constraints

The most obvious constraints are cost, time, and materials.

Cost

Cost is usually the biggest constraint. How much money is available for the project? Money, of course, frequently influences how much time and effort you can put into creating a project. Limited funds may mean that you can't afford a professional sound track or narration. Or maybe you can't afford the cost of licensing software or distributing several floppy disks to every user. These are all issues that will affect the final design.

Because I am paying for the development of the electronic newsletter, cost is a serious issue. Luckily, the major expense is my time, so as long as I don't overpay myself, I should come in within budget!

Time

Sometimes clients will tell you they have unlimited funds. Usually, that means they have a very tight schedule, and they know there is no way you can outspend their budget in the limited amount of time!

When is the project due? Whether developing a project for a client or for yourself, you will have some kind of deadline. Even if you have "unlimited" funds, you may not have the time to do everything you have planned.

This project is a prototype version of the newsletter. I will only implement one article and part of the structure as an experiment. If the experiment works, then I will implement the rest of the issue; otherwise, I may try some other method for distributing the newsletter electronically. Because it's a prototype, the finished piece will be produced in less than a week.

Materials

Multimedia developers need to have the right equipment and the right software available. If you don't have the right equipment, you may not be able to develop the project at all. For example, if you don't have a color-capable machine, you won't be able to make QuickTime movies. The right

software also can mean the difference between a successful and an unsuccessful project. You should refer to the chapters in the section on Multimedia Tools to establish the hardware and software that you may need for your project.

Note

Software and hardware can be really hairy for the beginner. Often you know what you want to do, but you don't know what you need to do it. If you want to create some illustrations, should you buy Illustrator, Freehand, Photoshop, Painter, Canvas, or some other program? You may not know the answer until you buy the program and invest some time in learning it.

To help determine which programs are appropriate for which tasks, read the chapters on Tools in this book, and read reviews in Macintosh magazines. If you can at least identify what you need to do, you will narrow the field down to only a few programs.

The cost, time, and materials constraints are, in a sense, the macro-constraints. Multimedia projects live or die by the *micro*-constraints—the things that affect the end-user, rather than the problems the developer had when putting the product together.

The not-so-obvious constraints

These constraints include such things as: Who will use the product? What will they use it on? What do they expect from the product? You might think that these things are obvious, and simply skip to the next section. However, before spending a great amount of effort creating a sophisticated product, you should be careful to make sure that all your users will derive the most benefit from it!

Who will use the product?

It's important to know who is going to use the product and what they will expect from it. In reality, very few products are “for everyone.” Even programs such as spreadsheets and word processors are for particular

segments of the computer-user market. An exhibit at a museum or an electronic sales brochure will be used by a very different audience than an electronic help system. Each group of users has different needs and expectations.

The user environment may have different requirements, too. This is especially true for programs running in public places. A museum exhibit or a shop demo may require a limitation on the maximum amount of time a user will spend using the system. For example, the average user must spend no more than five minutes using the product. To achieve this goal, you may have to cut the material so that there is only so much for a user to see.

Applications for public installations often require some kind of automatic reset and attract mode. After a period of nonuse, the product will go back to the beginning and run some animation in an attempt to attract a passerby's attention.

The example project: The electronic version of *Exploring Multimedia* is primarily for people who are working with multimedia tools, or who are seriously considering doing so. It's therefore unlikely that the casual Macintosh user will use this product. We will assume (it is dangerous to assume, but at some point you have to) that these serious users will want solid information. Although they may be impressed by (or may even expect) a sophisticated-looking product, these users don't want a lot of fluff.

What will they use it on?

What kind of computer will these users have access to? This question is important. Hardware restrictions often limit what you can do. If a large proportion of the users only have a Mac Plus or SE, then you must cater to the black-and-white graphics and small screens of these machines. You can, of course, create multiple versions of the product—for example, one for color and another for black-and-white systems. But this requires that an inventory of two versions be maintained, and/or that both versions be provided to users, dramatically increasing the amount of information being distributed. Also, fixing problems in two versions takes longer (and inevitably causes problems).

Alternatively, you can have the product itself determine what kind of machine it is running on, and have two sets of graphics stored in the production. This also increases the physical size of the production, though there will be some savings where graphics and code aren't being duplicated.

In either case, developing for this kind of situation will increase production costs because of the additional time required to create and cater to the multiple versions of graphics.

You may prefer to just produce a black-and-white production since it will be capable of running on a color machine. But will those same color users be disappointed if your production isn't in glorious color?

Some other examples of hardware limitations are listed below.

QuickTime

At this writing, QuickTime runs only on computers that support Color QuickDraw. None of the 68000-based Macs meet this requirement, so Mac Plus, SE (except for the SE/30), and PowerBook 100 users are unable to play QuickTime movies. If your product requires QuickTime, then those users will be unable to play it.

Color

Color animation or graphics will not always work effectively on black-and-white computers. At best, color graphics may appear murky; at worst, they may display as a solid black square. Is this a problem? Of course it is a problem, but the real question is: "Will it be a problem for the vast majority of users?" If most of them will have color machines, then you may just ignore the problem and require that the user have a color monitor.

What version of the System?

System 7 added color icons and a new desktop structure. If you want to make use of color icons, aliases, and other System 7 features, don't forget that users running System 6 will not see these things. More applications

are taking advantage of the new features that System 7 has added. Although today only a few applications will not run under System 6, this could become more of a problem in the future; you really need to consider what version of the System your users will be running.

System 7.5 adds to System 7, but at the moment it doesn't seem to greatly impact multimedia productions.

How much memory?

Memory became a concern with the arrival of System 7—which takes up even more memory than System 6 did. Most multimedia productions require a lot of memory, and it is not unusual to require a minimum of four or possibly eight megabytes of memory to run a presentation. At the same time, it is perfectly possible to produce something that will run in two megabytes (the minimum for System 7), and enable most Macintosh owners to run your product.

Variations in playback speed

A fantastic animation created on a Quadra 800 may run so slowly on a Mac II that the animation loses its effect. QuickTime alleviates some of these problems (or attempts to), but the reality is that you will always be able to create something that looks spectacular on the fastest, hottest machine—and looks abysmal on the slowest and lowliest. Always check your work on the slowest machine that you expect users will use for the production.

Screen size and bit depth

In the early days, every Macintosh had the same size screen—512 by 342 pixels. Then came the Mac II, and for a long time the standard color screen size was 640 by 480 pixels. When the LC was released, Apple unveiled a 12-inch color monitor with a pixel resolution of 512 by 384. Things have only gotten worse—the color PowerBook 165c has a completely different pixel resolution of 640 by 400.

If you create a presentation that completely fills a 640 by 480 screen, what happens when you play it on a smaller screen? In most cases, the rest of

the image will be cropped. Unfortunately, most multimedia authoring tools do not enable you to change the size of the presentation on the fly (while the production is being played). For this reason, it is probably best to use the smallest screen size that your audience is likely to have.

Pixel depth—the number of colors possible onscreen—is important. It's often safe to assume that color monitors have 8-bit pixel depths (and can display 256 colors at one time), but 24-bit screen depth cannot be assumed. Also, you probably should avoid 24-bit screen depth because animations and graphics become very large and animations play slower when they are in 24-bit color (after all, three times the amount of information is being moved around).

Tip

The safest solution is to only work in 1-bit (black and white), and always optimize for playback on a non-System 7 Mac Plus. If you do this, then you know that anyone who owns a Macintosh will be able to play and see your production. However, sometimes you have to make trade-offs. For each case, you have to ask “Is this a problem?” and “Can we live with the outcome?” Which is more important, a state-of-the-art presentation that blows the doors off the competition, or something that the widest audience can view? For different applications, the answer will be different.

How will the product be delivered?

This is an important question that is sometimes overlooked. How will you get the product to the end user? For one-off presentation situations like a presentation to a potential client, you can use a SyQuest cartridge, DAT tape, Magneto-optical drive, or even a portable hard disk, and then hand-deliver and install the product.

If you are sending copies to multiple locations, then you have to consider what equipment the users have access to, and how much it will cost to

deliver on a given medium. For example, although many users may have SyQuest drives, the cost of the cartridges makes these drives unattractive for mass distribution (see Chapter 5, “Hardware”).

Even using floppy disks is a problem. Can your users read high-density disks? Although high-density (HD) disks hold 1.4 MB of “stuff” and make distribution much easier, not everyone has those drives, so you may have to go with 800K, double-density disks instead.

It doesn’t take many disks to make distribution on CD-ROM cost-competitive. If you are distributing several hundred copies, then a production that takes up more than two or three floppy disks will cost less to distribute on CD-ROM. Of course, not everyone has CD-ROM drives. Do you want to limit yourself to those users?

Unique constraints

Some constraints may be unique to your application. To determine all the constraints, create a list for each of the categories like the one in Table 3.1.

Table 3.1 Unique Project Constraints

<i>Element</i>	<i>Constraints</i>
Development	
Authoring tools	What tools will you need to use? Do you have the skills to use them? Do you have to learn, or hire someone else? Any licensing requirements? Programs such as Apple Media Tool and HyperCard have licensing requirements, whereas others (SuperCard and Director) do not.
Hardware	What equipment do you have? What might you need? This equipment may include video or sound-digitizing equipment, or a hard disk for mastering a CD.

<i>Element</i>	<i>Constraints</i>
Data	In what format is the information used in the product? If the project uses a large amount of data from different sources, how is the data formatted and how will it be accessed by you?
Resources	What other resources—for graphics, video, sound, or data formatting—will you need? What will it cost?
Production	
Updates	Will the production or the data need to be updated? How will the production be updated? Who will do it, and what is involved?
Implementation	You might design and build the product, but some of the information may be added by someone working for the client. Who will do this work? Will they need special instruction? Special software?
Distribution	
Media	How much material do you need to distribute? What equipment do users have? What is cost effective?
Duplication	How and who will duplicate and distribute the production? What are the costs?
Users	
Hardware	What hardware do users require to run the production? What hardware do most users have?
Use	How will users use the production? Will they need any additional help (manuals, written instruction)? Will users require any other support—where will they get it?

At this point, you should be able to define a minimum configuration. The minimum configuration is the minimum amount of equipment a user will need to use or play your production.

For the electronic newsletter, it's probably safe to assume that the complete version should be smaller than 750K when compressed so that a) it fits on a floppy disk and b) it doesn't take too long to download using a modem. I'm also going to assume that users have a color-compatible Macintosh. This is a safe assumption because the audience for the newsletter is people using Macintosh multimedia tools, and most of these people will be using color-capable machines.

This is not really a minimum. I would like the electronic newsletter to have acceptable performance playing on a Color Classic.

This is the minimum configuration for the electronic newsletter:

- System 6.0.7+
- Color-compatible Macintosh
- 256 color monitor or better
- Screen resolution 512 by 342
- Color Classic

Planning the Production

Creating a multimedia project is difficult because everything is dependent upon everything else. The choice of tools and authoring environment, the users' requirements, and the material or subject matter of the production all limit the design, and all three interact—the choice of development tool affects how the material can be presented, but so does the expectations of the user.

Working on your first project is particularly difficult because you probably don't fully understand the capabilities of your tools or the material, and no one ever fully understands the requirements of the user. These three elements are interrelated, and while you can consider them separately, they also must be considered together. Often it is only at the very end of a project that you understand the project well enough to actually create a good plan for implementing it. And by then it's too late! All you can do is hope that you get to produce a similar project and apply some of what you learned to that one.

Let's look at the three elements that make up a production in depth. They are:

- The authoring environment
- The data
- The user interface

The authoring environment

The authoring environment is the program you use to create your multimedia production. These programs enable you to take graphics, sound, and other media types and link them together into a production to add interactivity—buttons that affect what happens and other methods for interaction between the user and the production. The authoring environment should not be confused with tools such as paint programs, QuickTime editing programs, and other programs you will use when creating the elements of your production. While such tools will be necessary to create elements of the production, they do not provide the possibility of interactivity.

Director, HyperCard, Passport Producer, and Special Delivery are all authoring environments. They are described in greater depth in the section on multimedia environments. If you want to learn about them, see the chapters in Part IV, "Multimedia Environments."

This section assumes that you are not going to program your production from scratch using a programming language such as Pascal or C++. If you are (and it costs a lot of money and time to do so), then ignore this section and skip to the next one.

Whichever authoring tool you choose, you will be limited in the things you can do with the tool. There is no perfect multimedia environment—just many applications with different strengths and weaknesses. For example, Director is very good to use for animation, but it is much more complicated to script, while HyperCard is good for scripting but only supports a single window, and SuperCard... well, you get the idea.

The authoring environment you choose will affect the look and functionality of the final production. Some applications can be used to create almost

anything, while others may have more limited uses. If you are starting out and trying to decide which program to use (particularly if you are planning to spend time learning the application), then choosing the right program can be daunting. To make a decision, play with other productions created with the program you are considering. Try and find something with a similar look and feel to what you plan to do. And read the section on “Multimedia Environments” (Part IV).

The data

The data is the heart of the production. It's the information you are trying to convey to the user. In the electronic newsletter, it's the articles in the newsletter. For a sales presentation, it's the sales figures and product specifications.

When you begin designing your production you have to ask yourself: “What am I trying to provide to the user?” An online help system is very different from a sales demonstration, which is different from an interactive game. Each has its own characteristics. An online help system may consist of a large amount of information that the user needs to search in a number of different ways. A sales demonstration is usually linear and may run from beginning to end without user interaction. An interactive game may have a linear story that is divided into segments; each segment may consist of many branching choices that the user explores while trying to progress to the next step in the story.

The user interface

It is very important to understand what the data is, and how the user will interact with it. Your first task is to make sure that you fully understand the data. This task can be particularly difficult when you are presented with a new project for which you have to design an interface.

There are two kinds of elements in user interface design: structural and cosmetic. Both are important, but you need to understand their differences. Structural elements include items such as menus, data fields, windows, and buttons. Cosmetic elements include items such as background graphics and the shape of icons.

Structural elements must be defined early in development, because changing these elements involves large changes in a project. Cosmetic elements are usually easy to change, and you can add these elements towards the end of the project. Many elements of a design have both a cosmetic and structural part to them. For example, an icon in a window that accesses a Find command is a structural element; however, the image of the icon is cosmetic. Adding the Find command at the end of the project could be very difficult, but changing the shape of the icon may not be.

But how to get started?

Getting started is difficult, but take heart. As Lionel Trilling said in *The New Yorker*: “Immature artists imitate. Mature artists steal.” Why spend months reinventing the wheel? Spend some time examining other projects that are similar to yours, and then take the elements you like the best and use them in your project!

For the electronic newsletter, I looked through several multimedia projects Apple had produced, as well as some projects from other developers. Apple spends a lot of time and money on their products, so you know you are working with the best when you “borrow” ideas from them!

Note

Note that under copyright law ideas are not protected, only the expression of the idea.

The Macintosh OnLine Reference

The Macintosh OnLine Reference is an electronic collection of tips and hints for Macintosh users, including everything from “How To Turn Off the Trash Warning” to “Setting the Alarm Clock.” This project was created in HyperCard (see figure 3.1).

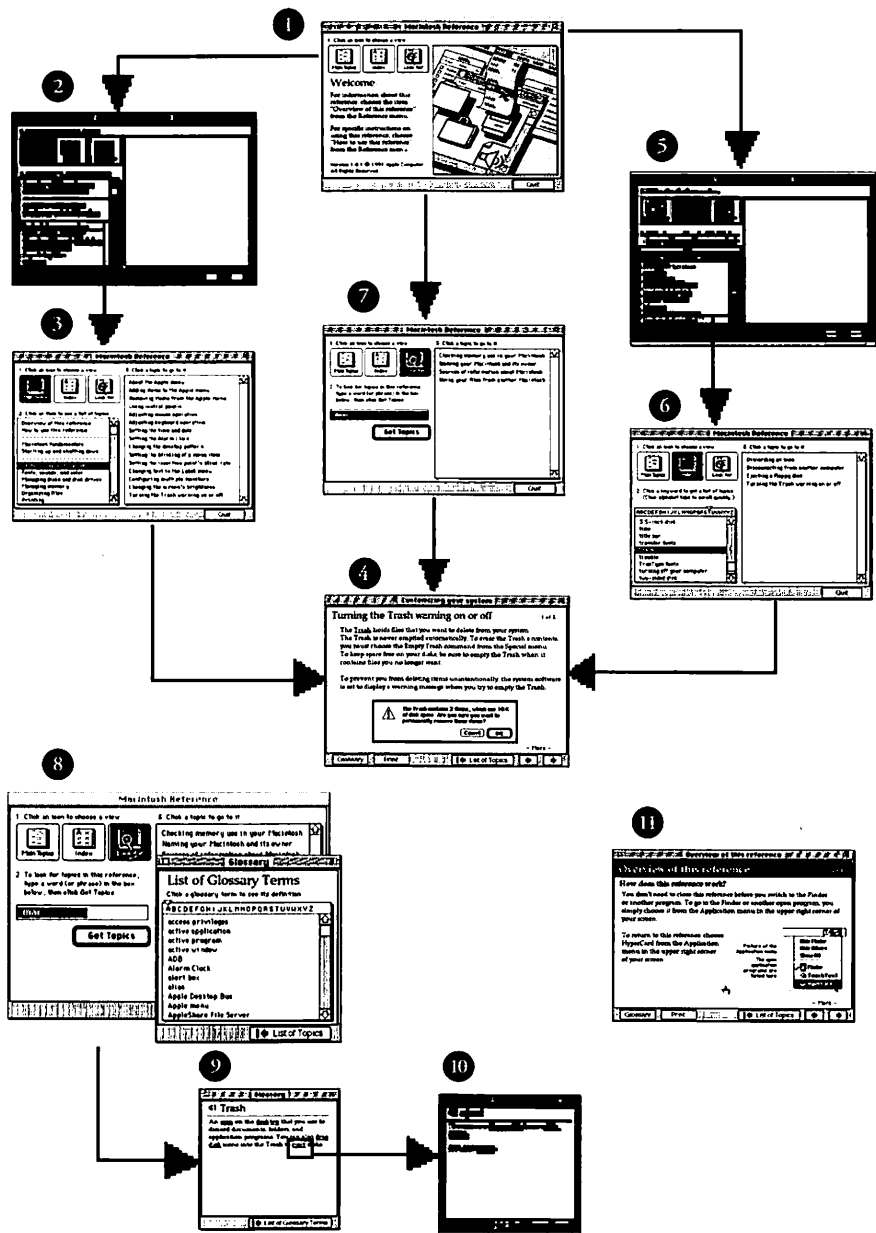


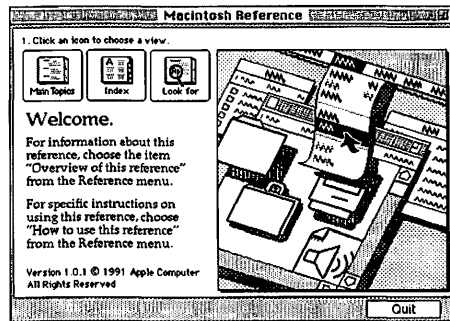
Figure 3.1. The structure of the Macintosh Online Reference. See the text for a description of the functions of each screen.

The structure of the Macintosh OnLine Reference

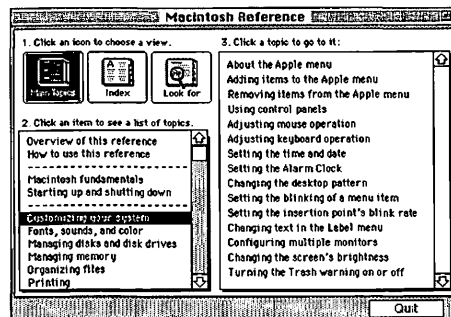
Basically, the Reference is a collection of tips presented on one or more HyperCard cards like card ④ in figure 3.1. Although this information is the heart of the system, there is a whole structure that provides different ways for a user to find the information that he or she needs. This includes an index, table of contents, and a Find command.

The Main screen ① consists of three buttons (top left), a graphic (right), and a brief welcoming description (bottom left). The three buttons provide access to a Main Topics section (essentially a table of contents), an index (an alphabetical list of key words), and a Look For command (a Find command that lists all occurrences of a word).

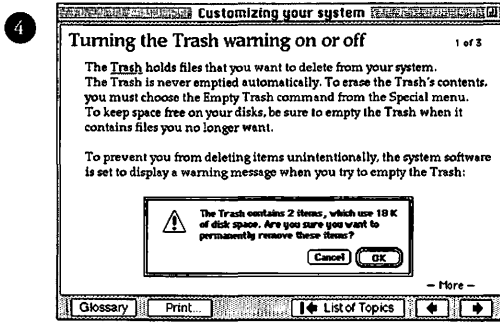
Clicking the Main Topics button in screen ① accesses a list of Main Topics (screen ②). Clicking one of these topics (bottom left of the screen) displays a list of subtopics to the right of the topics, as shown in screen ③. Selecting a subtopic (by clicking it) takes you to screen ④, an actual topic information screen.



Note that in screen ③ you can click a subtopic (the list on the right), choose another topic (the list on the left), or even access the index or the Look For command by clicking the buttons (top left). All these options remain available until you choose an actual topic, at which point the information screen appears (screen ④). From the information screen, you either can return to the list of topics (screen ③) by clicking on the List of Topics button at the bottom of the window, or you can browse through the information cards using the left and right arrows (in the lower right corner). Two

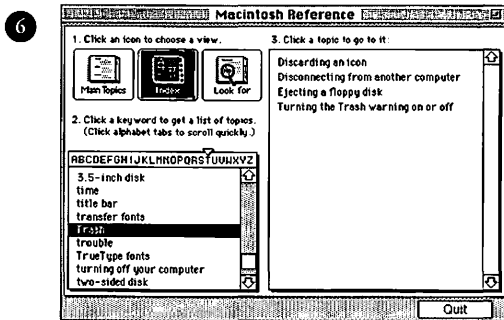


other buttons enable you to print the screen or access the Glossary (described later). To use the index or the Look For command, you must click the List of Topics button to go back to screen ③.

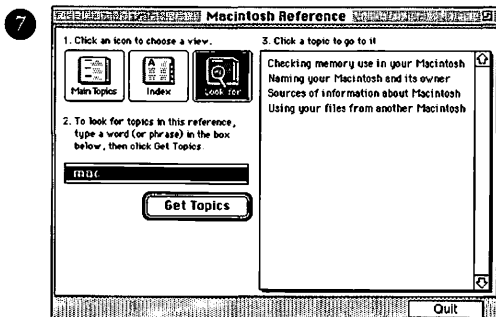


In screen ⑤, clicking the Index button displays an alphabetical index below the buttons. You can scroll through this list, or you can click a letter of the alphabet (above the field) to jump to that part of the index. Clicking on an entry in the index brings up a sublist of the index (screen ⑥).

Note that the index, although it contains different information, works in a manner similar to the Topics list; the major items appear in a scrolling list on the left side of the screen, and the subelements appear on the right side of the screen. Clicking an item in the subindex on screen ⑥ takes you to the information screen (screen ④).



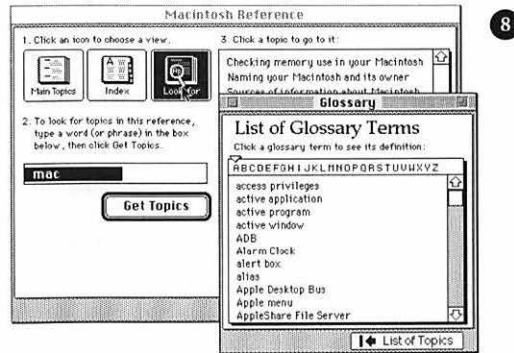
Clicking the Look For command brings up a field that asks you to enter the word for which you are looking. Clicking the Get Topics button brings up a list of matches on the right side of screen ⑦. Clicking one of these topics takes you to the information screen (screen ④).



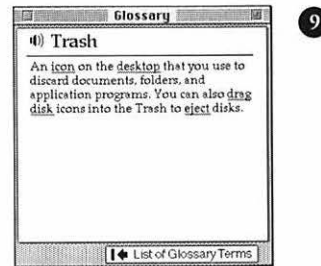
The Macintosh OnLine Reference includes some other

information contained in a glossary. These items are available in a separate window (actually a separate HyperCard Stack) called the Glossary. The Glossary is available from the menu (not shown here) or from the

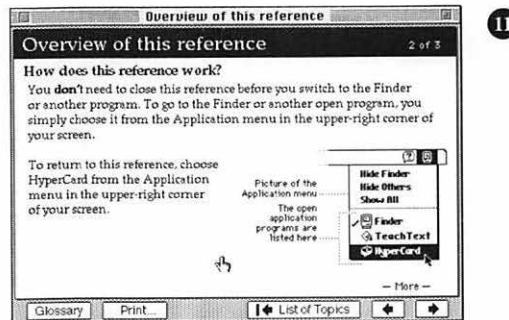
Glossary button that appears on the information screen (screen 4). The Glossary appears as a list of Glossary Terms in a smaller window (screen 8). This list works in the same way as the list in the index; it's a scrolling list with an alphabetical jump-to-letter control at the top of the list.



When you click an item in the Glossary list, the entry and its definition appear (screen 9). Notice the Speaker button next to the title of the entry; when you click the Speaker button, the word is spoken. Some of the words in the definition are underlined. Clicking an underlined word takes you to the entry in the Glossary for that word (screen 10), and so on. You can return to the Glossary list by clicking the List of Glossary Terms button at the bottom of the window.



When you return to the Welcome screen (screen 1), note that the Welcome text suggests a couple of places to go to get more information about how to use the system. Both of these options are available from a menu; choosing one of the options accesses the window shown in screen 11. The first screen in the window is an overview of the reference, while the other screens (not shown) describe how to use it.



Evaluating the Macintosh OnLine Reference

The Macintosh Reference is an interesting example of an electronic information system. However, before we either blindly copy the ideas or look at something else, it pays to spend some time trying to evaluate why

it was designed the way it was. By doing this, we will learn from what Apple did and be able to apply what we learn to our own projects.

- The Table of Contents and Index make the product similar in function to a book.
- There are some nice computer-like features added to the book model. These include the capability to jump quickly from one place to another, the Look For command, and the speaking glossary. The Glossary is linked to itself (key words are marked in the definitions), and the information cards also contain links to the Glossary.
- The Table of Contents, Index, and Look For command all work in a similar way. After users have tried one function, they should have no problem using another.
- The Welcome screen—the first screen the user sees—immediately tells the new user what to do to find out how to use the system, but the user who just wants to experiment can immediately do so.
- Graphics have been merged with the text in the information cards to provide a very pleasant experience. However, note that there is no animation, although it could be argued that animation isn't needed for this particular reference guide).
- Although the two-level system (topic, subtopic) works well for this application, it may not work well for other systems. (What if you have three levels—topic, subtopic, sub-subtopic?)
- Although the electronic reference provides a lot of the features found in regular books, it lacks others. First, it's almost impossible to tell how big the book is! When you look at the information cards, you have no idea how many other cards there are, although it does tell how many cards there are for the current topic (top right of screen 4). When you pick up a book, you know how big it is, and therefore have a rough idea of the amount of information it contains (and the amount of time it would take to read it). This is not the case for most electronic books. For a reference source, this is not so much of a problem (it's unlikely that users will read it from cover to cover), but it may be a problem in other situations.
- Another problem is a lack of bookmarks. You can't mark a place that you have been in such a way that you can get back to it quickly. It would be great (and fairly easy) to add this feature.

All in all, the Macintosh OnLine Reference is a very good example of a multimedia production, and definitely worth looking at.

The Glossary doesn't actually speak; the words were digitized as they were spoken by someone and then played back by the computer. In the future, the Macintosh may be able to speak (see Chapter 11, "Sound").

QuickTime Intro News

When Apple introduced QuickTime, it also pressed a CD-ROM called the Apple CD-ROM Titles Sampler. This CD-ROM contains two multimedia projects: the QuickTime Intro News and the Titles Sampler (both described below). Graphically and functionally, these two products are different; however, in many respects they perform a similar task—both contain a directory of products available from different companies.

QuickTime Intro News runs in HyperCard, just like the Macintosh OnLine Reference; however, visually it is a very different production (see figure 3.2). All the screens have color graphics and a sophisticated look with color backgrounds, animations, and other graphic items. QuickTime is used to play animations as well as small video interviews.

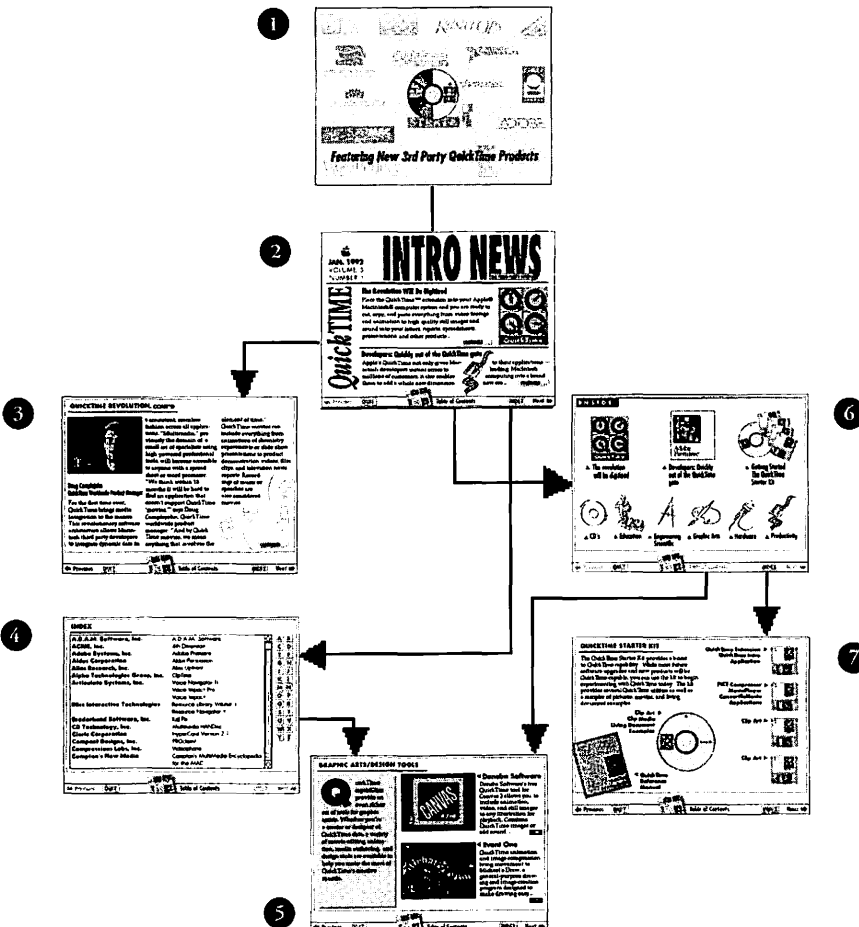


Figure 3.2. The structure of QuickTime Intro News. See the text for a description of the functions of each screen.

The structure of the QuickTime Intro News

When you first run the QuickTime Intro News, an animation is played (screen 1) that displays the title and looks very interesting (the first time

you see it). When the animation finishes, the Intro News screen appears (screen 2). In some ways, this resembles the first page of a newspaper, with the masthead at the top of the screen along with the date of publication. Then there are two "articles" that begin on this screen and continue elsewhere (indicated by the two buttons labeled Continued). Clicking one of these buttons accesses screen 3 which continues the story. Note that screen 3 also contains a QuickTime movie (top left) that you can play.

Notice the Index button at the bottom of each screen. Clicking that button takes you to the index screen (screen 4). You will notice something odd about this index—it's not an index of articles; instead, it's a list of manufacturers. Intro News contains a directory of different products that support QuickTime. To see the information cards, simply click one of the index entries, and you will go to a data card (screen 5).



The data cards are divided up into categories (Graphic Arts, CD's, etc.), and each card contains two or three entries. The entries themselves contain a short text description, and many have a QuickTime movie you can play.

The table of contents (screen ⑥) is available by pressing the button at the bottom of the screen. The table of contents screen contains several elements. The two buttons (top left) point to the two stories that start on the first page—so clicking either of these buttons will take you back to the first screen. The third element (top right) takes you to another screen (screen ⑦, described below). The other elements below these three buttons take you to the master categories in the stack.

Screen ⑦ contains the other article in this publication. This article consists of a short description of the QuickTime Starter Kit (a product that Apple sells) along with an illustration showing you what the QuickTime Starter Kit contains. What you can't see in this illustration is that the screen first appears blank, and then a box appears onscreen. The various elements of the Starter Kit “fly” out of the box and arrange themselves on the screen!

Evaluating the QuickTime Intro News

The QuickTime Intro News is very different from the Macintosh OnLine Reference in structure, form, and appearance. It's almost hard to believe that the same company produced them both. The products were designed to serve different functions—the Reference provides information to a user, while the Intro News seems to be more of a sales tool.

The lessons to learn from this project are:

- Visually, this electronic publication is very appealing, but... .
- When you run this product, you are not exactly sure of what you are dealing with—is it a newspaper or a directory?

- Greater emphasis should have been placed on the table of contents so that users would be able to understand the structure of the production. Also, the two parts (the articles and the directory) should have been more clearly separated.
- More work could have been done to make the directory information useful to the user (much of this was done in the second production, the Titles Sampler, which is described below). Perhaps the Titles Sampler, which is a separate HyperCard stack with its own interface and information, resulted in the developers limiting the functionality of this product.
- Although the animations are entertaining the first time, they are slow and repetitive after that. Keep this point in mind for your own projects. You may want to store a parameter so that a “cute” animation only plays once (or give the user the option of turning off the animation).

To conclude, the QuickTime Intro News is an interesting mix of personalities that can't quite decide what it is. The color, graphical design, and integration of QuickTime make Intro News worth a look, but you should be able to improve upon the basic structure for your design.

Titles Sampler

The Titles Sampler (see figure 3.3) also is included on the Apple CD-ROM Titles Sampler. Just like the Intro News, the Titles Sampler is created in HyperCard, and has a color interface. But the graphic design and the functionality are very different. The Titles Sampler is a directory of products that can be accessed in a number of ways.

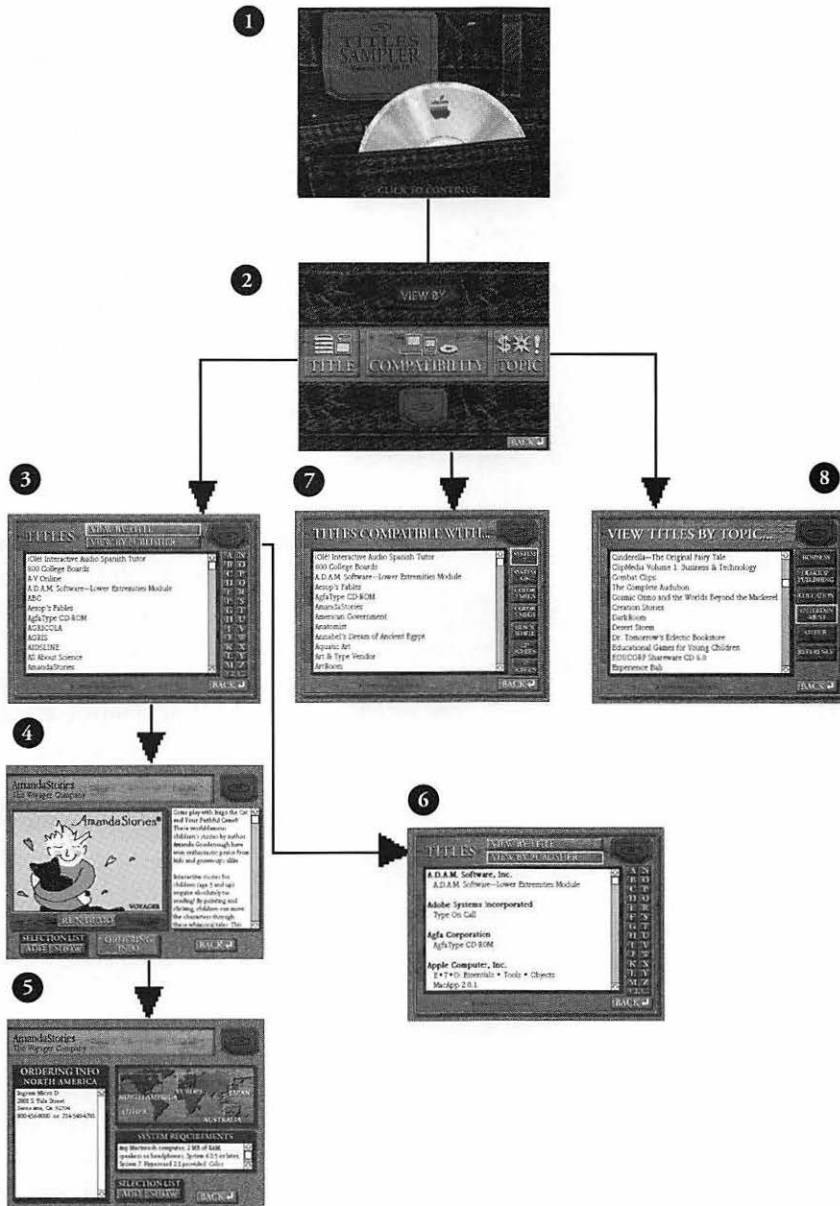
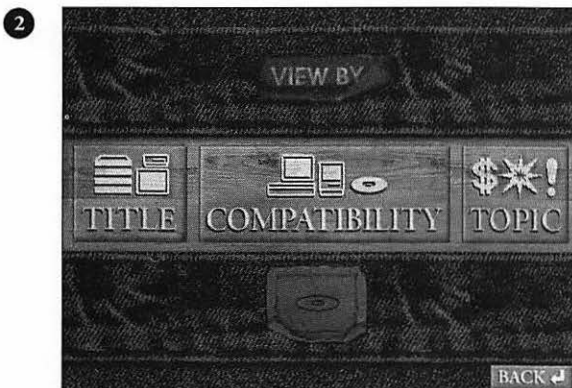


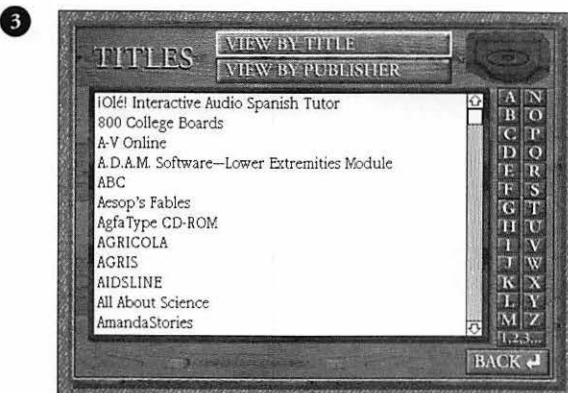
Figure 3.3. The structure of the Titles Sampler. See the text for a description of the functions of each screen.

The structure of the Titles Sampler

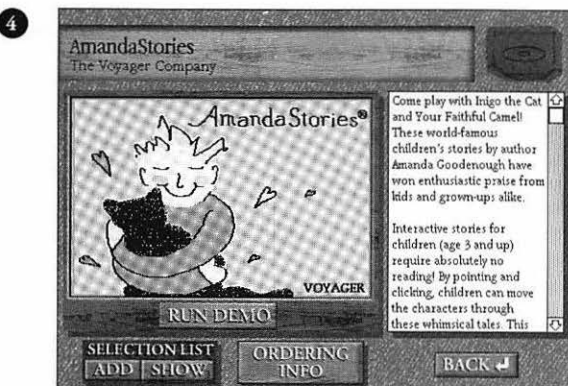
The Titles Sampler begins with animation and a title screen (screen 1). You must click the screen to go to the next screen, which is titled View By



(screen 2). From this screen, you have three options—Title, Compatibility, and Topic. Clicking the Title button accesses a list of titles (screen 3). You can scroll through this list, or use the alphabet on the right side of the screen to jump to a letter in the alphabet.



Although the topics in the list appear in alphabetical order according to product title, it is also possible to list the products according to publisher. Simply click the View by Publisher button at the top of the screen; the screen changes to screen 6.



When you click a title (either in the By Publisher or By Title list), the product information screen (screen 4) appears. This screen includes an image from the product and a short text description. A button under the image will run a demo (if a demo is included on the disk).

In the lower left corner of the screen, you will see the Add and Show buttons. The Titles Sampler can keep a

list of the products in which you're interested. When you click the Add button, the product is added to the list; when you click the Show button, the list is displayed.

Clicking the Ordering Info button on screen 4 (to the right of the Add and Show buttons) takes you to the ordering info screen (screen 5). This screen shows the system requirements for the product, as well as the ordering information. To find the ordering information for different locations around the world, you click

the world map. When a different location is clicked, the ordering information in the ordering info field (on the left side of the screen) is updated.

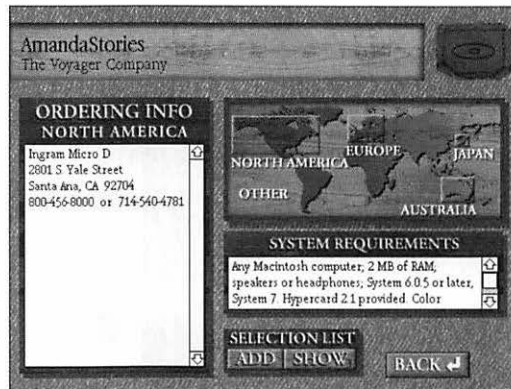
In addition to using the Titles list, you also can access the list of products through a Compatibility list and a Topic list. You select these lists on screen 2, as you do the Titles list. Both of these lists (screens 7 and 8) are similar to the Titles list, but the row of buttons on the right side of the screen changes to reflect either a list of compatibility requirements, such as CD-ROM or Color, or topic categories, such as Business or Desktop Publishing.

Evaluating the Titles Sampler

The Titles Sampler is much easier to understand than the Intro News. The Titles Sampler contains only one type of information, and it is pretty clear to the user what that information is, after only a little exploration. The small number of categories is a limitation, and when you first look at an information card, it is not clear what the product is unless you read the description card. Is it hardware or software? Is it a paint program or a game? Also, the Find command is very rudimentary.

Multiple Media Tour

The Multiple Media Tour, published by the Audio Visual Group, is a CD-ROM containing a large collection of multimedia resources (images,



sounds, and so on). Included on the Multiple Media Tour disc is an electronic catalog, which makes for an interesting design example (see figure 3.4).

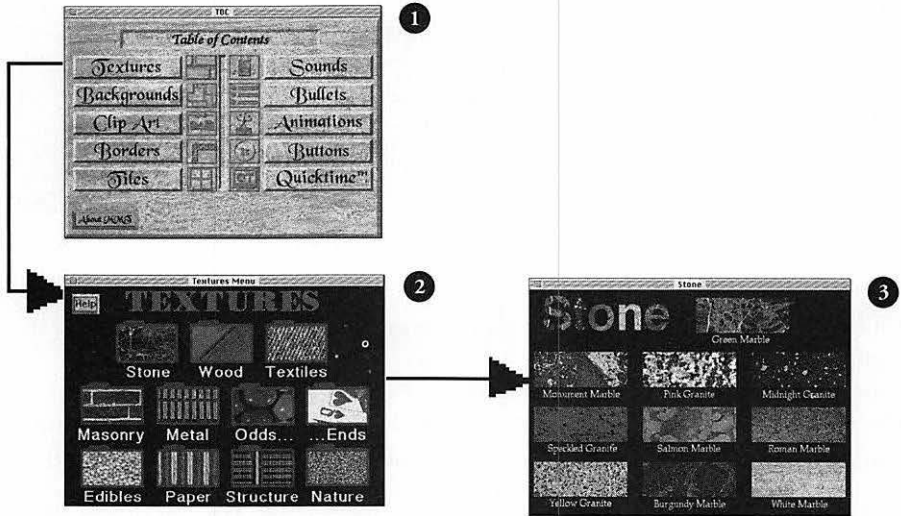
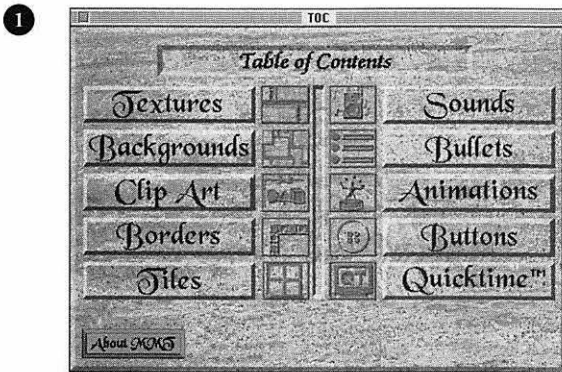


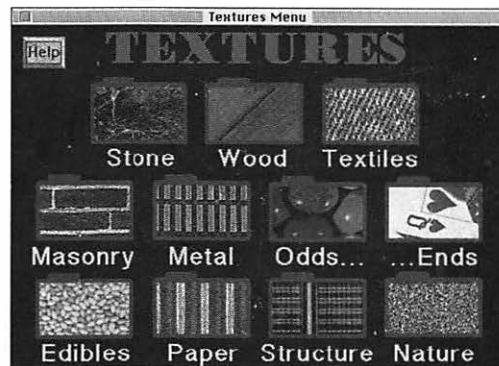
Figure 3.4. The structure of the Multiple Media Tour. See the text for a description of the functions of each screen.



Created in SuperCard, the Multiple Media Tour is a directory of all the multi-media resources on the CD-ROM. The table of contents screen (screen 1) is the first screen you see when you run the program. This screen provides a list of categories. Clicking on a category (in this example, Textures) takes you to a subcategory screen

(screen 2). This screen divides the textures into several sub-subcategories. Clicking one of these sub-subcategories takes you to a screen showing all the materials in that sub-subcategory (screen 3). When you click one of these materials, the screen shows the actual image itself (not shown). You then can copy the material

into the Clipboard and paste it into another program. Not shown is a navigation palette that contains a button you click to return to the table of contents and buttons to move through the cards in the stack.



The Multiple Media Tour is a very simple interface that works for this particular application. It operates in a logical fashion that anybody should be able to use. There are a few minor problems (for example, the Help button is on the category screen, but not on any of the other screens), but these problems don't really interfere with the operation of the product.

Macintosh Human Interface Guidelines Companion

The Macintosh Human Interface Guidelines Companion is another interactive online reference from Apple. Unlike the Macintosh OnLine Reference, this product is in color and was created in SuperCard. (It's much easier to add color in SuperCard than in HyperCard.)

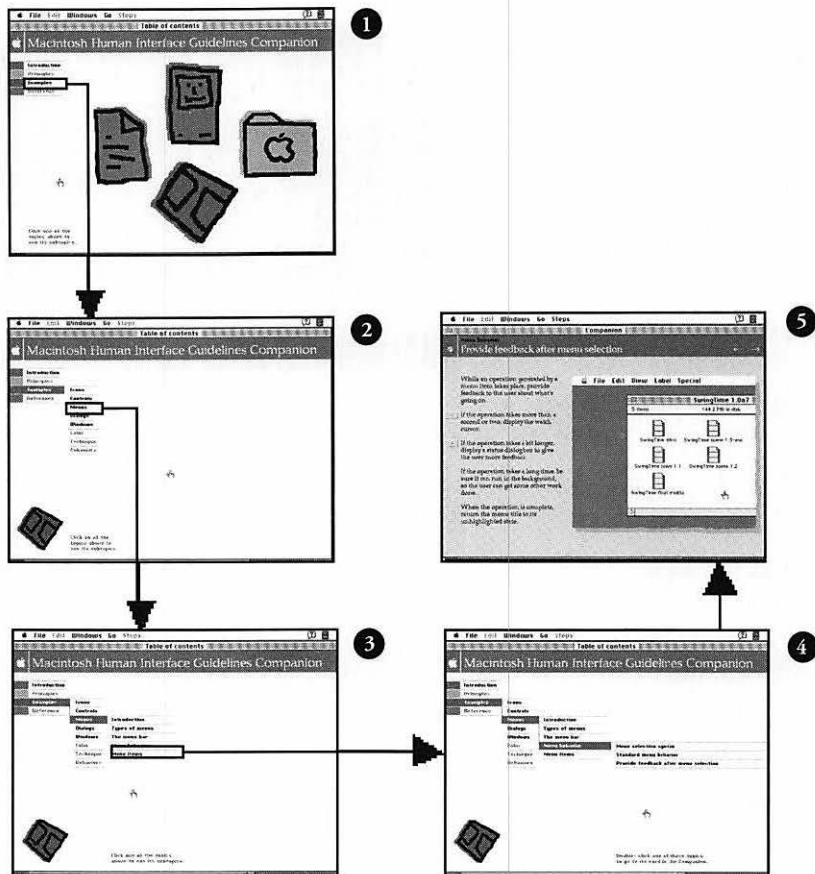
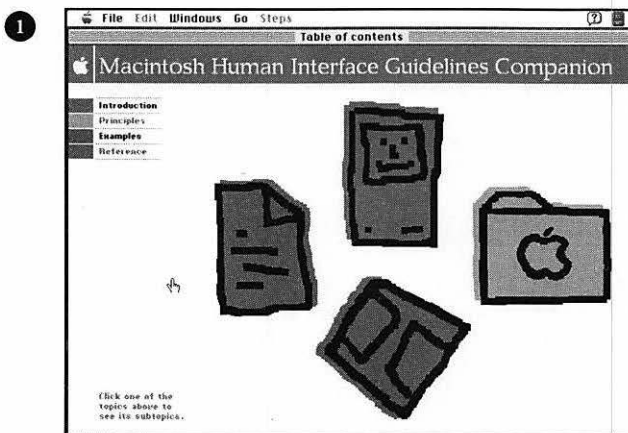


Figure 3.5. The structure of the Macintosh Human Interface Guidelines Companion. See the text for a description of the functions of each screen.

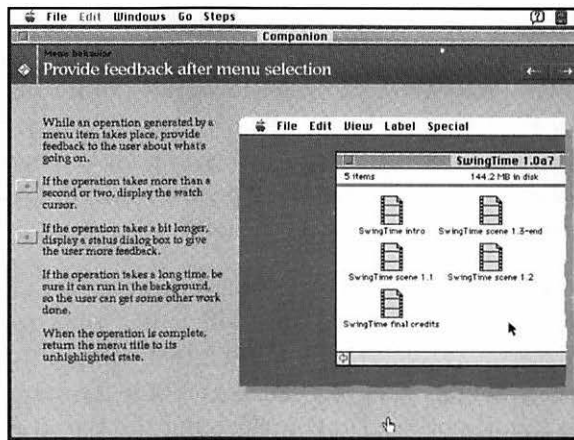


Like the Macintosh OnLine Reference, the Human Interface Guidelines basically consists of a series of cards with information about different topics (screen 5). Because these cards are in color, they are more graphically appealing than the

cards in the Macintosh OnLine Reference. However, the real difference between the products is in the method used to access these cards.

As in the Macintosh OnLine Reference, you are presented with a table of contents that starts with the major headings (screen ①).

Clicking a topic accesses subtopics (screen ②). This process continues down two more levels (screens ③ and ④) until you reach the information screen (screen ⑤).



Visually, the Human Interface Guidelines is much more appealing as well as clear—you always know where you are. You can click any of the parent categories to jump to another topic.

On the down side, although this design works well for this particular product, it will be difficult to add more than one more level to this screen—although it's unlikely that you will have to deal with that many subtopics in a project anyway.

As an implementation criticism (rather than a design criticism), the choice of the 640 by 480 screen for this product means that users with smaller screens will not be able to view it easily. However, Apple obviously intends this product to be used by developers, and developers will more than likely have larger color screens.

It's Time to Start Designing

Because multimedia authoring tools are so easy to use, it's tempting to start prototyping right away—especially with an application such as HyperCard or Director. There is nothing wrong with this approach; however, I recommend that you first create a storyboard, ideally on paper (see figure 3.6).

There are several reasons for using a storyboard. First, it will focus your thinking. I have nothing against building prototypes and then improving them—it's the way I prefer to work—but it's important to start with an idea of the direction in which you are heading. Second, if you are working for a client, a storyboard provides an opportunity for the client to correct all your misunderstandings before you spend countless hours working on the project.

You can create your storyboard in a number of ways. At first, you might just sketch things on paper—it's still a lot faster than drawing with any program available for the Macintosh! After you have an idea of what the project looks like, you can use an object-oriented drawing program such as Canvas or MacDraw to flesh out the layout. These programs are preferable to bitmapped programs such as Photoshop or MacPaint; you will find it much easier to make changes in object-oriented drawing programs. I like to create layouts in PageMaker because I can easily combine graphics and text in the storyboard.

On the left of figure 3.6, blocks represent the screens in a project; on the right, text describes the contents of the screen and anything a reader may need to know about how the project works. Note that interactivity—how you move from one screen to another—is indicated by lines going from one screen to another.

After the storyboard is completed, review it with the client, users, and anyone else you can find. The more input, the better. Also, by this stage, you should have chosen an authoring environment. The authoring environment will have a direct impact on the next phase.

The next step is to consider any interface issues that relate to specific screens (for example, how buttons work, what's contained in fields, and so forth). Depending on your project, this process may require very little, or a lot of thought and work. If the project is reasonably simple, and you are using design concepts used in other applications, then you may be able to skip to the next task immediately. Also at this point, you can start work on some of the cosmetic issues, such as what graphics you need, where they will come from, and what format the data is in.

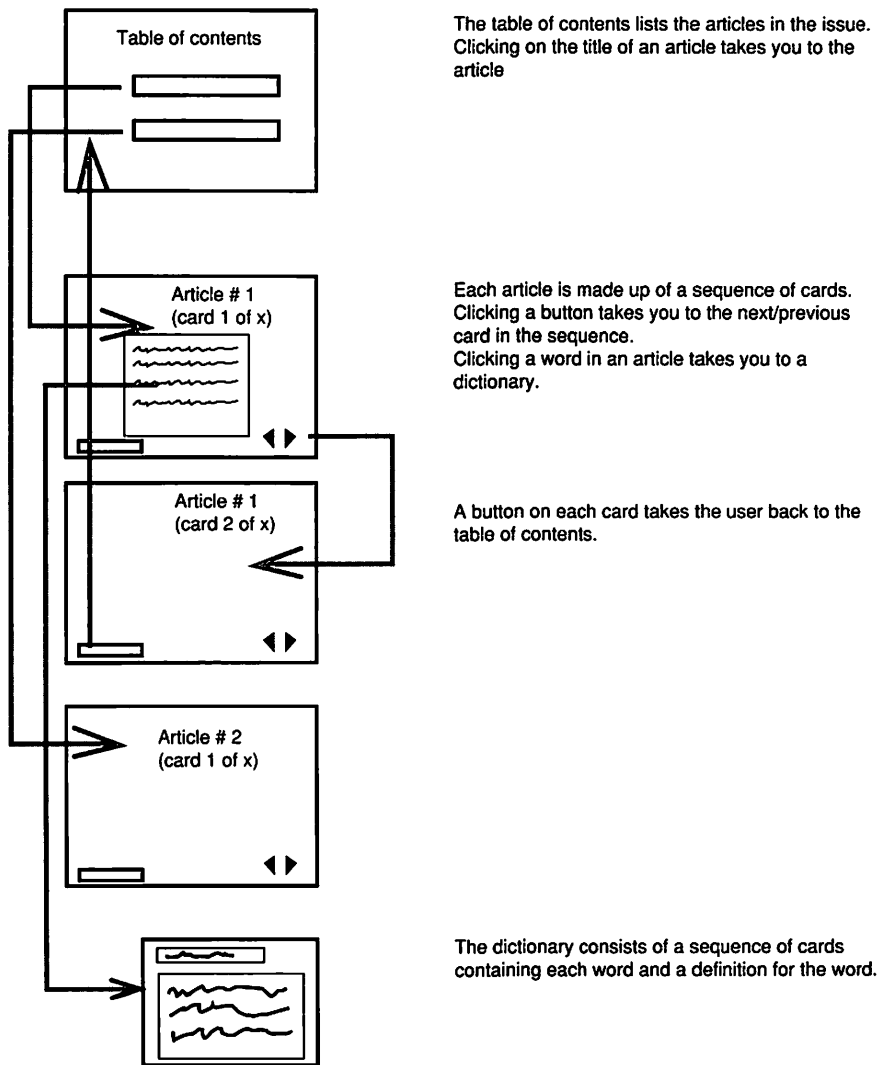


Figure 3.6. A storyboard.

For the electronic newsletter, I created a number of sketches and ended up with the diagram in figure 3.7.

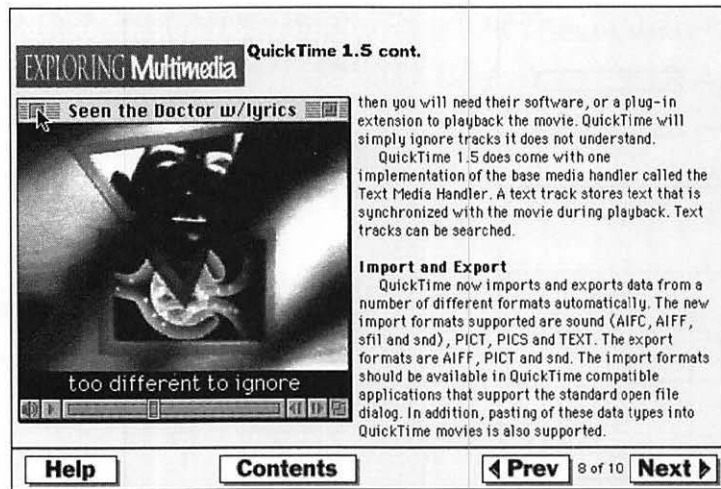


Figure 3.7. Sketch for the electronic *Exploring Multimedia* newsletter.

There were several false starts in the design process. I originally thought that Macromedia Director was the tool to use because it supports QuickTime and many animation effects. As the sketches were produced and I studied the information that the newsletter contained more closely, it became clear that text handling was going to be important. Unfortunately, Director is not as good with text as it is with animation.

SuperCard would have been a good choice because it supports color. It doesn't support QuickTime, but for the prototype that would not be a problem. A problem that was unexpected was that the publisher wanted readers of this book to be able to experiment and build the prototype. That meant that we had to use the development tool that most readers would be able to access. So HyperCard was the final choice. It's not the ideal solution, but it's the one we will have to go with (at least for the prototype).

A lot of time was spent trying to figure out how the article would hang together—would it be spread over several cards, or would the text be in one scrolling field? If the latter example was used, then where would the figures that accompany the article be stored?

The final design consisted of:

- A master card from which each article could be accessed.
- Each article is contained on a sequence of cards. Buttons let the user move forward and backward through the article.

- A dictionary would be attached containing references and all the occurrences of a word in the newsletter.

Other considerations were:

- How are other issues distributed? Are they added to existing copies or are they separate?
- Let the user add bookmarks and keep notes.
- Author, subject, and title indexes.
- Search for any text.

To see how the prototype turned out, see Chapter 16, “Putting it to work.”

Creating the Prototype

Now the fun begins—creating the prototype. One developer has suggested that the best way to create a great product is to engineer a prototype, throw away the complete prototype, and build the final product from scratch! Although this sounds like a great way to produce a product, few have the time or resources to work this way.

As an alternative, the best way to work is in “chunks.” If possible, divide the project into modules that can be implemented independent of one another. Then work interactively; create a rough sketch of the first module—put in the fields and buttons, but don’t finalize the graphics, and don’t feel that you necessarily have to get all of the scripts to work. Rather, you are creating the structure of the project, something you can use to determine if you are heading in the right direction, and then you can come back and add the finishing touches. These elements—the fields and buttons—are the things that everything hangs from, and it makes no sense to start designing graphics before the structure is complete.

The advantage of working this way is that if you spend a lot of time getting the graphics to look right and then discover that the layout of the project has to change because the client wants much larger text fields, then you have wasted time.

After the structure is in place, try to implement specific functions. This method is the best way to work with large projects. Choose functions that can be tested independently of the rest of the project. Test to see what the users think, and then change or add to the prototype.

Why not build the whole thing? If you like to work that way, then certainly do so. But for large projects, or if you are not sure what you are doing, you will be better off if you implement part of the project and see how that goes.

One of the great advantages of authoring tools such as HyperCard and Director is that they make it so easy to change and add things. This is both a strength and weakness. After you have changed something several times—particularly scripts—the scripts can become very complicated and inefficient because you change bits and pieces instead of recreating the whole thing from scratch.

Testing the Prototype

Whether you create just a part of a prototype or the whole thing in one pass, at some point, you will want to show the prototype to others for input and suggestions.

It's a difficult thing to show your baby to others—particularly a prototype that doesn't have all the fancy graphics and sound that you imagine it will have. Much to your frustration, many people will start picking apart the very things you haven't finished yet—"Why aren't there any graphics?" and "Why doesn't that button highlight?"

Unfortunately, there's no way around it: you have to show your prototype to others, and you will just have to deal with what they have to say. Very few people know how to give criticism—many people don't know how to criticize the work rather than the person. ("Why on earth did you do it in *that* color?") Try not to take it personally.

Try to get all the input you can, from as many sources as you can. One of the best ways to test an interface is to show what you have to someone who has never seen the project before. Try not to tell the potential user how to do anything. Resist the temptation! Just sit and watch what the person does.

Developers with a lot of money often hire independent research firms to set up tests that are videotaped. Videotaping has two advantages: you don't have to be in the same room, and you can't interfere and say things such as, "Click the *top* button!" Also, you can view the videotape over and over again.

You don't have to incur the expense of videotaping if you can't afford it, but you will have to work hard to learn how to sit and watch what users do without trying to help. When watching users, there are particular things to watch for. Do they know how to find things? Can they use the index? Do they seem to find everything in the system?

After testing, adjust the prototype. Make any changes, and then test it again. You may have to change something several times before it works. If part of the interface doesn't work, make sure that you get another new user to test it when you make a change. If you show the project to original users, they already will have an idea of how the old way was supposed to work, and that may help them with the new version.

Be prepared to work through several iterations until the project is complete. However, be careful not to lose sight of the objective—it's all too easy to keep going round and round in circles trying to get something just right.

It's also important to keep track of any problems in the prototype that need to be fixed. Keep a list of things you need to fix, and check them as you finish them. Keep a second list of all the things that are done to the project and when. This is particularly important during the final stages of development. The first list is a list of bugs, while the second list is a list of bug fixes, as well as any new features or other changes you decided to make.

Making the Final Delivery

The final delivery is the completion of the project. No matter how stimulating a project is, it's always great to finish something and send it out into the world. For the final delivery, you will have to consider some things you may not have worried about for the prototype, such as documentation and installation. If you want to include documentation, you can provide it on paper or electronically as balloon help or a "read me file." If there are any special installation instructions or processes for the project, include those as well.

All the Things We Forgot

You shipped the final delivery to the client and two weeks later they call you up to tell you that they need some changes made. Don't be surprised—it always happens! Clients will report bugs that weren't found during testing, and new features that need to be added after the project is “completed.”

But it's also important to spend some time looking at the project dispassionately to learn for future projects. Let some time go by, and then go back to the project and try it out. Does it work the way you expected it to? What would you change if you had to do it over? What can you change to make it better? You may never go back and make those changes to this project, but maybe on the next one you will put all you learned to good use.

Things to Watch

No guide to getting started in multimedia production would be complete without some tips on problems to avoid. This chapter concerns designing your project and the conceptual phases of development. Still there are potholes to watch out for. The three biggest ones to avoid are:

1. Not knowing what the purpose of the project is.

Are you doing a sales presentation or an electronic brochure? Different situations require very different design and content. It is possible to create one production that can be used for different situations, but you must budget the extra time to do that. If you are in this situation, try to create one part first rather than doing the whole project.

2. Being too ambitious.

Try to start small and work up to the hard problems! If you are creating electronic manuals for sales training, choose the simplest one first. That way you can try out your ideas without wasting a lot of time. A few successes under your belt will give you the confidence to tackle that killer project at the end.

3. Not having the content nailed down.

You must have the content before you start design and production. If you don't—whether it's because you haven't finalized licensing, or because the clients just can't make up their minds—you are putting yourself in the dangerous position of having to scrap work you have completed because the content is different from what you had expected.

Any one of these things can derail a production, and if your project suffers from all three then you may as well give up now!

Moving on

In this chapter, you looked at the development process in depth, both in theory and for a demo system. In the next chapter, you will review the development process for a particular project.

Dissecting a Multimedia Project

4
chapter

The previous chapter, “Developing a Multimedia Project,” led you through the steps for creating a multimedia project. Hopefully, that chapter will help you organize your project, and give you the courage to get started.

Because the design process still may seem abstract, this chapter explains how a real multimedia project was created. First, a little background information is in order.

Overview of the Project

MarketPlace Business is a CD-ROM database that contains over 7 million U.S. businesses; it is used to produce mailing lists and cold call lists for marketers. MarketPlace users select prospects, using criteria such as type of business and sales revenue. Then MarketPlace generates a list of names that match the criteria. The information is encoded on the disc, but the user can purchase names as he or she wants them. After names are purchased, the user is free to use them for almost any purpose.

MarketPlace uses HyperCard as the front end (the graphical user interface) to the database. HyperCard enables the developers to produce a graphic, simple-to-use interface. MarketPlace also has a multimedia help system that features animated “movies” that show how to use the system. This chapter describes how the movies were developed, and some of the issues encountered during their development.

Note

Even more history: The product was originally developed by Lotus Corporation. When Lotus started development they intended to produce two versions: Lotus MarketPlace Business and Lotus MarketPlace Households. The latter was a database of most of the households in the U.S.

When the Business version was announced, there was such a ground swell of negative reaction to the Household version (even though it hadn’t yet been released) that Lotus ended up abandoning the project.

How I became involved

When Lotus began developing MarketPlace, Rob Lippincott, head of the multimedia department at Lotus, was busy trying to promote the use of multimedia within Lotus. Rob realized that the MarketPlace CD-ROM would provide the space to add multimedia help for users. He advocated and promoted the use of multimedia in MarketPlace with the MarketPlace project team, and used outside contractors to develop some early HyperCard prototypes.

When I joined the project, the prototype design and interface for MarketPlace had been completed. Using the prototype interface design, a multimedia prototype called “Dave goes Prospecting” had been created in HyperCard. This prototype consisted of a sequence of cards making up a “movie.” The movie introduced Dave, who wanted to use MarketPlace to find new prospects for his business. The movie showed some screens from the product that were intercut with pictures of Dave.

The movie had an audio narration, but was very limited in structure; after the movie started, it could not be stopped or paused, and had to play to the end. The animation in the movie also was very limited. Rob felt that Macromedia Director would improve the quality of the animation and offer features that had not been possible with HyperCard alone. That’s where I came in. I was hired to use Director to develop a prototype that expanded upon the ideas in the original prototype. The final prototype was a result of many people’s input, including Rob Lippincott, Marcia Zuckerman, Cindy Null, Lars Jensen, Marian Stern, and Andy Hollinger.

MarketPlace Business was sold to a separate company which offers the MarketPlace Business software. Unfortunately, the movies described here are no longer included. They took 18 MB of space and even though a CD-ROM is very large, after development was completed, it was decided that putting more addresses on the disc was more important than putting multimedia help on it. You win some; you lose some.

Planning the first prototype

Before building a new prototype, Rob and I reviewed the current one and decided what improvements to add. We felt that some means of controlling the movie should be available for the user—something like a VCR control panel. Also, although the narration worked well, we were concerned that it might be inaudible in loud environments, so we decided to add captions. Finally, the first prototype had featured one long story about Dave and how he used the product. We decided to try a different approach: a story with several subparts.

In our new approach, each subpart would demonstrate a particular function of MarketPlace. The help system would consist of one story about Janet trying to accomplish a task using MarketPlace. The movie would explain what Janet was trying to do and then show how she solved the problem using a feature in MarketPlace. The movie would consist of several tasks, each illustrated by accompanying “how to” explanations. A user would be able to view the whole movie, or, if he needed help on a particular topic, the user could view only the segment of the movie that showed how to solve that problem. We referred to this new approach for the first prototype as the Pearls & Swine structure (see figure 4.1).

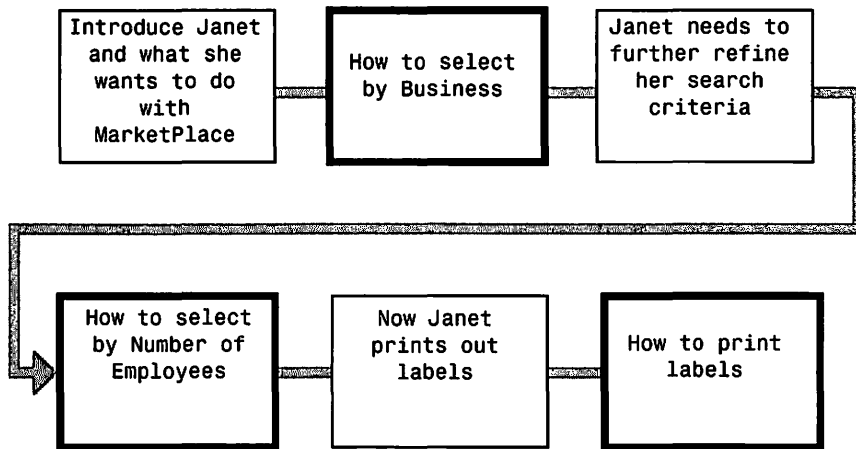


Figure 4.1. Pearls and Swine prototype structure. The entire movie is made up of segments. The “How to” segments, called Pearls, could be played separately to find out how to perform an individual task in MarketPlace.

We eventually ended up with the prototype in figure 4.2. (This is really the fourth version of the prototype, but it is perhaps the best example of the ideas we had.) The prototype featured an audio narration, captions, and animation, as well as a control panel that enabled the user to stop, re-wind, and fast forward.

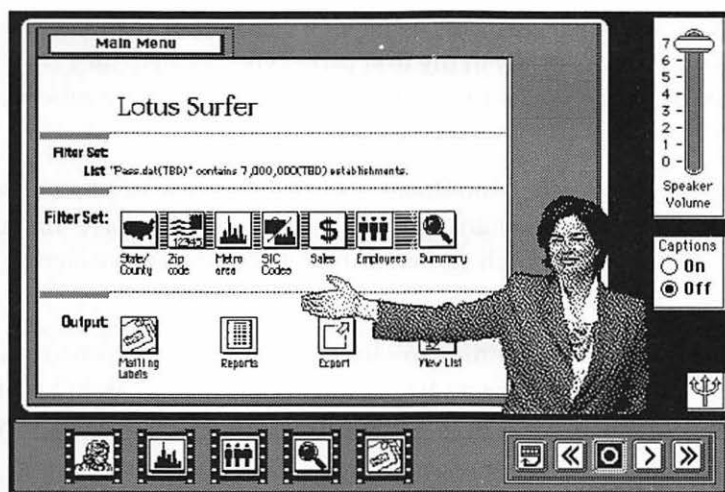


Figure 4.2. The first prototype.

The movie consisted of parts showing Janet and parts showing the product. We even had Janet appear within the screen of the product. This technique was an attempt at an “agent,” and was useful for indicating specific MarketPlace features. For example, Janet would point to where the user should look when the narration said, “Click the Export Button.”

Reviewing the first prototype

In figure 4.2, notice that the actual product screen has been shrunk to make room for the control panel. This change also made it clear that you were looking at a demo, not the real product.

Along the bottom of the screen, you see the control panel. The first icon represented the whole movie. Pressing that button played the movie from beginning to end. The next four icons represented the “How To” parts of the movie. Pressing one of these buttons played only that segment of the movie. The icons in the lower right corner enabled the user to go to the beginning of the movie, rewind, pause, play, or fast forward. Above these controls, you see the exit button—a three-pronged spear (the code name of the product was Trident).

At this point in the process, the prototype was reviewed by the Market-Place documentation team. The movies were considered part of the help system (which included paper documentation and an electronic online

Apple started talking about agents a few years ago. Agents were “intelligent” programs that were designed to perform some task. For example, an agent might read the electronic news services and download only the stories that you would be interested in. In early designs the agent had a human form—a person in a window who would talk to you. There was even a product on the Macintosh (HyperAnimator) which made it possible to create animated people very easily. But the idea never caught on.

There are still software agents under development, but most have an interface of dialog boxes and menus, not something you talk to or who talks to you.

manual that was separate from the movies). The other team members had been involved peripherally in the first prototype because they provided the script for the movie, but up until this point the other members had not seen the prototype.

The team decided to abandon the idea of shrinking the product screen. The small screen was too hard to read, and users would have difficulty following the program. Each screen also had to be hand-tweaked to make it readable—a tedious process.

By this time, we were beginning to question the Pearls & Swine idea. We weren't sure if the scripts were the problem (because we didn't really know how to write a script in this new format) or if the idea itself was wrong.

The next prototype featured a full-size screen and a completely different control panel (see figure 4.3). The control panel was now a palette that only appeared onscreen when the user paused the movie. The icons representing the segments still were displayed in the control panel. Also, in an attempt to make the whole project more cohesive, we cut back on the animation and the screens that contained story segments about Janet. As in the previous prototype, the basic concept remained the same: the movie contained segments that could be viewed together or independently.

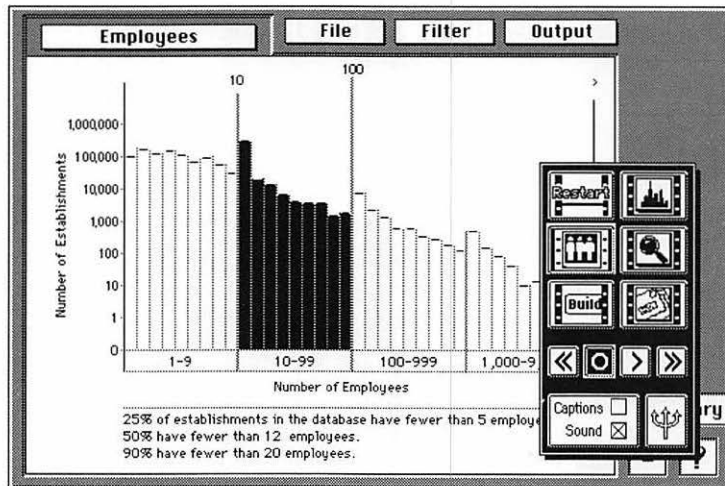


Figure 4.3. The second version of the prototype.

Because the control panel was not visible while the movie was playing (it obstructed too much of the screen), a small box containing the words `Click at any time to stop the movie and bring up the controller` was displayed while the movie was playing. We spent quite a bit of time experimenting with the location of this message and the control panel so that they wouldn't obscure anything important on the product screen.

After this prototype was completed, the team made two decisions. First, we abandoned the idea of segmenting movies that could be watched as a whole. We were unable to make the idea work. The movie was too long and disjointed, and the individual segments weren't in-depth enough. We decided to split the movies into two types: a dozen segments that explained each part of the product, as well as three movies that were introductions to the product and showed someone performing a task with MarketPlace.

Second, we gave up on the idea of Janet acting as an agent. We just couldn't get the concept to work. Whether Janet was on top of the screen or inside a little window, her presence distracted too much from what was going on, and didn't add anything to the experience. We filed the idea away for another project.

Switching development tools

The first two prototypes had been developed in Macromedia Director, and we had planned to use the Macromedia Director Player (a set of XCMDs), which plays Director animations within a HyperCard stack. Remember, HyperCard was being used to develop the rest of the product. But we were beginning to have doubts about using Macromedia Director. At the time we began development, HyperCard 1.2.5 was available; we knew that HyperCard 2.0 would be released soon, and the product had to work with both versions. But we had no idea whether the Director Player would work with HyperCard 2.0. There were also questions about the Player's stability when running under MultiFinder (which was required to run MarketPlace) and the amount of memory that the XCMD and animation required when running.

We also discovered that we weren't using many of the animation effects that Director offered. The early prototypes, for example, contained buttons that grew to 400 percent of their actual size to draw the user's attention. But this animation had been removed; it was becoming clear that we could do nearly all of the animation effects using HyperCard. At this point, we decided to abandon Director and use HyperCard exclusively to make the movies.

The one animation requirement we did have was to simulate in HyperCard the cursor moving across the screen. In our first attempt to simulate this movement, we used a special font that contained a character like the HyperCard hand. This character was placed in a transparent field on the screen, and the position of the field was changed rapidly to animate the field and simulate the cursor moving around the screen. This method (except with a button instead of a field) was used in the final version, and it worked quite well. In fact, on faster computers it worked too well, and we had to add a timing routine to slow down the animation effect.

Because of the way font characters are displayed, at first we reversed the cursor. This was done for two reasons. With this technique, the cursor did not appear transparent—you couldn't see what was under the cursor. We also thought that reversing the cursor would lessen the chance of the user confusing the "real" cursor with the imaginary one—but more on that later.

The final phase

We continued with the control panel, but to make the job of hiding it easier, the control panel was now a field containing font characters (see Chapter 6, "Text"). The prototype had almost reached its final form (see figure 4.4), but there were still many problems to solve.

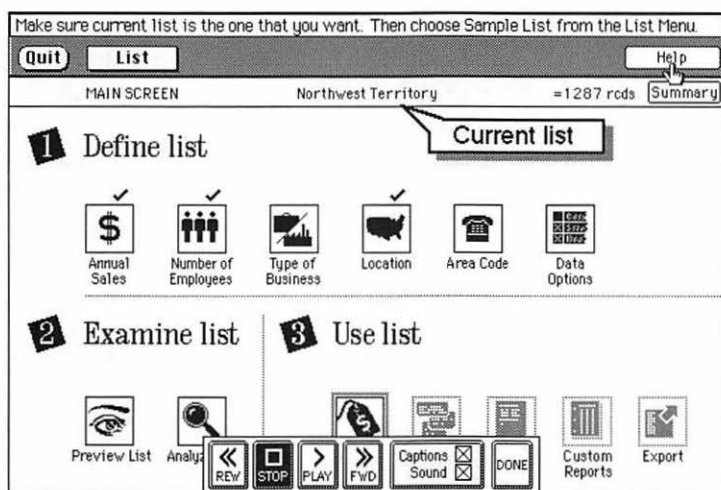


Figure 4.4. The final design for MarketPlace help movies.

In the first prototypes, we experimented with animation effects to attract the user's attention to what was happening onscreen. We created buttons that grew to mammoth proportions so that the user couldn't miss them. We created sliding buttons, blinking cursors, and several kinds of call-out boxes. We even had Janet point at things.

The problem was that although we wanted to attract the user's attention, we wanted to do it in a subtle way—but not in such a subtle way that the user would be surprised when the actual product didn't perform exactly as it had in the demo. We didn't want users saying, "What happened to that box that said 'Click Me'?" After a couple of different attempts, we finally resolved the issue of how to let the user know where to look on the demo screen. We used highlight boxes to point to areas of the screen. These boxes drew the user's attention without interfering with the appearance of the product.

At this point in development, we also were concerned about the amount of disc space the demo would occupy. Although a CD-ROM appears to be almost endless, we discovered that the actual data could easily fill the disc. Each of our movies was taking up 2 MB of space, and we had been allocated only 20 MB (and the project managers were keen to get any of that back).

Part II

User testing is a lot more complicated than this—trying to find out what someone else thinks of a product without applying your own knowledge and biases is very difficult, particularly if you designed or developed it and are hovering over the user saying “No, click *that* button!”

More testing followed. Lotus hired a user interface consultant to test the interface of the product, and he also did some testing of the help system. These tests involved putting people who had never used the product before in front of a computer and asking them to try to use a copy of the prototype. While the new users worked with the prototype, their actions were monitored. Any problems were noted and reported back to the development team.

During this trial period, we discovered our biggest problem was that users confused the movie with the actual product. We wrestled with this problem constantly. How could we make the user aware that it was a movie and not the actual product? (I should add that some of us thought the movie should play within the product, and if the user wanted to actually use the product, then that was fine.) One possible solution was to play the whirring noise of a film projector before the movie started, but would this be too cute and would a younger audience be familiar with the analogy?

Users also sometimes responded to the narration as if they were working with the real product. When the narration said, “Now click the business icon,” some users reached for the mouse and tried to do just that. So we changed the narration to be less directive. The narration now says, “To enter the business options, you would click the business icon.”

The issue of terminology in the narration was very important. Do you “click,” “select,” or “hit” items onscreen? Luckily, as part of the online documentation effort, terminology had already been standardized, and this was used in the movie’s narration.

Another problem was that users weren’t always aware that they could actually stop the movie. The “Click at any time to stop the movie” message now had been moved to the top of the screen because we couldn’t permanently cover up any other part of the screen with the message. The top of the screen wasn’t an ideal place; if captions were turned on, the message was hidden. We also discovered that people never looked there; they were too busy following the animation. When the user did notice the message (usually when it was pointed out to them), the word “stop” suggested to most users that clicking would stop the movie and take them back to the product. Users were too scared to stop the movie because they thought that they wouldn’t see the end of the movie and would have to sit through the beginning all over again. So we changed the wording to “Click at any time to *pause* the movie.”

Towards the end of the prototype process, we realized that the reversed cursor was not acceptable (for an aesthetic reason mostly), so we had to go back to a plain cursor. This process involved switching to an animated button rather than a field. Still the debate about the cursor raged. Some wanted the “real” cursor hidden while the movie was playing to reduce confusion. Others argued that the user would be confused if the real cursor weren’t visible! And, unfortunately, we didn’t have time for more testing.

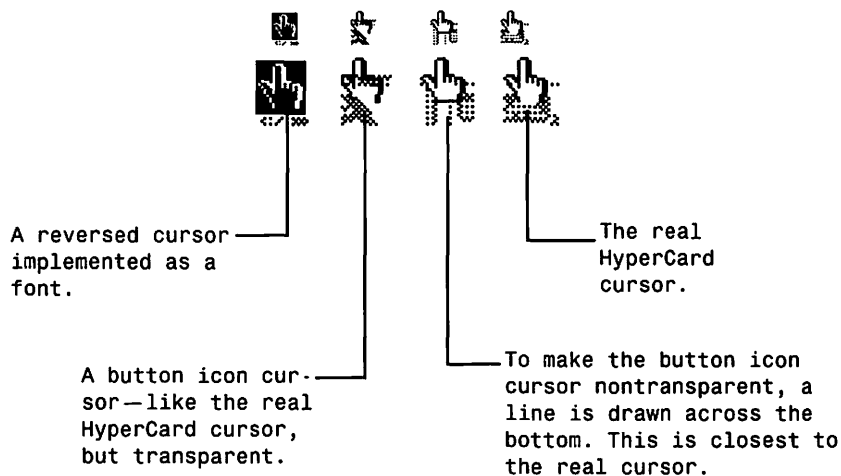


Figure 4.5. Three ways to emulate the HyperCard cursor.

Producing the final version

After the basic structure and a prototype of the movies had been designed and completed, we began writing the scripts and producing the animation. This process was reasonably straightforward.

The documentation team wrote the scripts for the movies and also checked the scripts. To ensure accuracy, the scripts were circulated to other project managers for review. Then, the script writer and movie “producer” read each script in front of a computer, while performing the appropriate functions described in the script. This step helped us find

things that should be highlighted in the movies, and it also provided a thorough last check of the scripts—it was amazing how many minor errors were found in the scripts even after many reviews.

A scratch audio track was recorded using MacRecorder. Each movie was assembled using screen dumps from the product. Some of the screens were improvised because the user interface was not finalized (or even working). Several people reviewed these movies and any errors were fixed. After this first pass, several of the scripts were changed to reflect changes in the product's user interface.

Because we developed the prototype while the user interface of the product was still incomplete, we had to go back and make a lot of changes, which increased the cost of the project. Unfortunately, this was the only way we could get the help system completed in time. The situation was aggravated by the last minute changes the software developers made to the user interface of the product—these changes meant we had to change all our documentation!

The movies were changed or remade to reflect the final product changes, and we hired a professional narrator to record the final sound track. The sound track was recorded at a sound studio, and then digitized from tape using a MacRecorder. We used a sampling frequency of 11KHz, which provided a satisfactory quality playing on the Macintosh internal speaker without taking up too much disk space.

Although the movies themselves were only 3 or 4 MB in total, the sound files took up about 18 MB. We discovered, just as we put the whole prototype together, that HyperCard could not handle more than 16 MB of resources in a single stack. The sounds had to be taken out of the master stack and placed in two resource files that were opened by a special XCMD.

Just before we finished the job, Farallon released MediaTracks, a product that can record and play back actions performed on a computer. Although it came out too late, we considered using MediaTracks instead of HyperCard. I think MediaTracks would have saved some time, but it would have required more time to implement captions and the control panel, and it would not have helped us when we were developing our prototypes. MediaTracks is no longer widely marketed, but there are some new products that capture screen actions to QuickTime movies (see the section on CameraMan and Spectator in Chapter 10, "QuickTime").

Considering the Future

Although I'm pleased with the product we ended up with, I can't help but think about the future. I'm haunted by one prototype we explored, which we dubbed "Obi-Wan." Obi-Wan was a concept for a help system that would be available at all times. Its control panel would always list items the user was most likely to need help with at that moment. To provide comprehensive help, Obi-Wan would link together demonstration movies, an online user guide, and point-and-click information about anything onscreen. Unfortunately, we didn't have the time or resources to fully explore that prototype, but maybe next time!

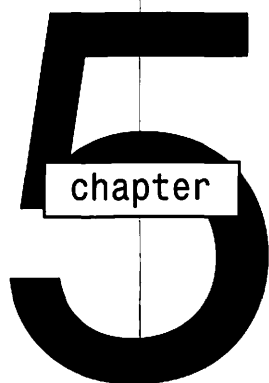
The next section of this book describes the tools for creating and manipulating different media types. The first chapter discusses special hardware that you may need when working with multimedia.

III

PART

Multimedia Tools

Hardware



5

chapter

This chapter is about the hardware that multimedia developers and users need for creating and running multimedia productions. It does not cover all the hardware you'll use, for example, hardware issues relating to specific media topics are covered in their appropriate chapters. See the "Graphics," "3D," "QuickTime," and "Sound" chapters for discussions of hardware that relate specifically to those topics.

Perhaps the question asked most often about hardware is: "What Macintosh should I buy?" That used to be a fairly easy question to answer—back when only three or four Macintosh models were available, and new models were released only once every 18 months.

Those were the good old days—pre-1991! Since then, Apple has been releasing new hardware at a faster and faster clip. Add to that the dizzying array of add-on products, and it's easy to become confused about what to get.

Still, it doesn't help those of you thinking about making a purchase to hear that things are just getting more confusing. So, the first section of this chapter provides a brief history of the Macintosh, and the second section provides a general guideline for buying a Macintosh. Even if you already own a Macintosh, you should read the second section, which includes descriptions of other hardware you may need or want to buy. The final section looks briefly into the future.

A Brief History of the Macintosh

In the beginning, computers were designed for manipulating numbers. After a while, they were used for text handling—database applications and then word processing. Then came graphics and sound, and now computers are being used for video. This evolution of capabilities has made multimedia possible.

The Macintosh has also evolved over the years, though it's interesting to note just how capable the original design was. When introduced in 1984, the original Macintosh had 128K (kilobytes) of memory, a single 400K floppy disk drive, and a 68000 processor. In 1991, Apple was still selling the Macintosh Classic, the case had been redesigned, the motherboard

had been updated, 2 MB (megabytes) had become the minimum memory configuration, and a 1.4 MB high density floppy disk drive was standard, but the processor was still the same 68000. And the basic capabilities of the operating system—text, graphics, and sound—were still the same. It's hard to believe that Apple was able to sell a computer based on the same processor for so long, and it really is an indication of what a good job Apple did with the design of their operating system—or maybe how little progress there was in the rest of the computer field.

The original Macintosh

The original Macintosh had a built-in black-and-white display with 512 by 346 pixels. It had built-in sound support—a speaker with hardware to play 8-bit, 22kHz (kilohertz) sound. It also had a set of standard graphic routines (called **QuickDraw**) that applications used to draw onscreen. This configuration represented a dramatic difference from the PC world, where different graphic display cards required different routines to draw graphics. Macintosh graphics and sound capabilities are more fully explained in Chapter 7, “Graphics,” and Chapter 11, “Sound.”

The original 128K Macintosh was followed by the 512K Macintosh and then the Macintosh Plus. The Plus had 1 MB of memory, an 800K floppy disk drive, and a SCSI (Small Computer System Interface) interface, which made it relatively easy to attach external hard disks and other devices, such as scanners, to the computer.

The Macintosh Plus had many advantages, but it also had many disadvantages. It did not support color, and it was not expandable; you couldn't add plug-in boards to digitize video or display color graphics on a separate monitor. The Mac Plus evolved into the SE and then into the Mac Classic. Both machines were functionally similar to the Plus.

It's important to remember that the Plus, SE, and Classic exist in large numbers. If you are creating a multimedia presentation for mass distribution, you will need to develop one that will run on these models. They support sound, 1-bit graphics, and animation. They do not support color, and cannot currently run QuickTime movies, although this may change (see Chapter 10, “QuickTime”).

The color, expandable Macintosh II

In 1986 Apple released the Macintosh II line, which supported color and had six slots for plugging in boards that used the NuBus standard (a standard for plug-in cards Apple had adopted).

The Macintosh II used the 68020 processor and had new ROMs that included the code for supporting color. Over time, Apple released a number of variations to the basic design—increasing the processor speed and adding models with fewer NuBus slots. Some models lacked NuBus slots altogether, and instead offered a Processor direct slot. These slots can, in theory, be used for similar purposes (for example, a video digitizing board exists for the LC). Since the processor direct slot is not used in as many machines as the NuBus slot, hardware manufacturers have not released many products for those machines.

The addition of color and plug-in cards made the Macintosh much more multimedia-friendly. Plug-in cards made it possible to add video digitizers and high-quality sound boards, while color is necessary for video applications, image retouching, or just to display a photographic image on the screen. The 68020 processor was much faster than the 68000 processor used in the Plus. This extra horsepower was necessary when working with the larger files made possible with color, among other things.

Apple replaced the Macintosh II line with the Centris and Quadra models. At first, the Quadra series was the top of the line Macintosh, using the fastest processor—the 68040. The Centris was introduced as the mid-level series, and featured the 610, 650, and 660AV models. In less than a year, Apple renamed these models as the Quadra 610, 650, and 660AV. There was no difference between the Centris and Quadra models, except the Quadra 610 and 650 featured a slightly faster processor than the Centris models.

The top of the line Quadra models are the 950, 800, and 840AV. These models feature fast processors and several NuBus slots, making them ideal for digital video applications, such as Digital Film from SuperMac or MoviePak from RasterOps.

Most of these models are excellent authoring machines. If you are working with very large images, digital video, or 3D modeling, then you will want to get the fastest machine you can afford (if you want to do full-frame, full-motion video then you need at least a Quadra 650).

Quadra 630

The Quadra 630 is notable because it features a whole new approach to modular design. Whereas previous models supported plug-in NuBus cards and processor direct slots, the 630 features a special slot for a modem or Ethernet card, a slot for video capture (you can spend more and get a capture card with a TV tuner included), and the 630 can be upgraded to a PowerPC with a plug-in card.

The 630 features a 33MHz 68040 chip, making it as fast as all but the fastest Quadra models, yet the 630 is priced very competitively. It is a great multimedia machine—its only drawback is that it does not support full-size NuBus cards, so you cannot use it to capture full-frame video.

In looking at the 68K Macintosh family, one important feature is the existence of a Floating Point Unit (FPU). FPU's are used for intensive mathematical calculations (particularly 3D applications) and some machines do not come with one. While there is a piece of software called SoftFPU, which emulates an FPU, it is slow and does not work in all cases. So if you are planning to do multimedia authoring and buy a 68K Macintosh, then make sure there is an FPU in your model.

The Portable, the PowerBooks, and the Duos

When Apple finally announced a portable version of the Macintosh, there was much excitement among the ranks of Macintosh users. Then they saw the Portable. Weighing about a ton and a half and larger than a VW, it wasn't the kind of thing you wanted to be carrying around.

The original Macintosh Portable used the same processor as the Macintosh Plus (though running at a faster clock rate) and had all the features of a Macintosh (though no color or slots). Perhaps because it was so large, it didn't prove very popular.

Apple got things right when it released a new range of portable Macs called the PowerBooks. They originally offered the PowerBook 100, 140, and 170. The 100 had a 68000 processor and no color ROMs. It was smaller than the other PowerBook models because the floppy disk drive was separate from the unit. The 140 and 170 were similar in shape, and both

were larger and heavier than the 100. The PowerBook 170 had a better screen and a faster processor than the 140. These models were replaced by the 145, 160, and 180 models.

For multimedia developers the biggest difference between the 100 and the other PowerBook models was that the 100 could not run QuickTime because it lacked the color ROMs. For other reasons, the PowerBook 100 was not the success that Apple had expected, and it was discontinued.

Models with a color screen are designated by a “c” after the model name. In 1994 Apple released a new line of PowerBooks, the 500 series. These machines were redesigned inside and out (their shape is reminiscent of 1950’s automobile styling). This series was the first to include the faster 68040 chip (though lacking the FPU), a touch pad instead of the trackballs found in the original models, and two battery bays. Apple claims that they will be upgradeable to PowerPC at a later date (by replacing the motherboard containing the current processor).

The PowerBook is a good portable machine for playing multimedia applications. The latest models have ports on the back for plugging in external color monitors, making it possible to show color presentations if there is a monitor available at the location. Alternatively, you can simply use the PowerBook’s screen—particularly if you have a color screen. Be aware that the PowerBook screens are very different from regular tube displays. The LCD panels tend to be slower at updating, which can affect the appearance of animation and digital video. Also, the dynamic range of some LCD panels is not as great, resulting in less apparent differences between pixels of similar colors. These problems vary depending upon the type of screen; PowerBooks are currently available with two types of LCD panels, Active-Matrix and passive-matrix screens. The Active-matrix screens are more expensive but provide much better performance. The screen resolution of some PowerBook models is not the semi-standard 640 by 480, but a slightly smaller 640 by 400.

As a development platform, one of the biggest drawbacks of the PowerBooks is the trackball and trackpad—it can be difficult to manipulate objects at the pixel level using these devices. You can solve that problem by using an external mouse.

In some ways the Duos replaced the PowerBook 100 because they are as small and light as the 100. But, in reality, the Duos are much more powerful—they have much faster processors than the 100, and they support

QuickTime. When plugged into a base unit, the Duos turn into a desktop computer with a floppy disk drive, NuBus cards, and all the characteristics of a Macintosh II-class computer. Used this way, the Duo might be a better authoring solution than the PowerBooks.

The Duos and PowerBooks do share one minor annoyance for multimedia users. Both product lines, because they run on batteries, feature power-saving features which slow down the speed of the processor when the operating system thinks that the computer isn't doing anything useful. Unfortunately, sometimes when an animation is playing, the operating system may erroneously think that the user isn't doing anything because the user hasn't clicked or typed anything for a few minutes.

There's a couple of solutions to this problem. One option is to move the mouse during the presentation—a little tedious, but it works. The other is to go into the PowerBook control panel and turn off the power-saving option when playing the presentation (refer to the manual that came with the PowerBook to find out how to do so).

The Performas

In 1992 Apple released the Performa line. These are relabellings of existing models and are sold through mass distribution outlets, such as Sears. For playing multimedia these machines are very good, but as an authoring environment only the most powerful of the Performa models are of interest.

The AV Macintosh

In 1992 Apple introduced the AV line of Macintosh computers. Essentially, the 660AV and 840AV were two existing Quadra models, the 610 and 800 with an added DSP chip used to provide 16-bit audio and video-capture capabilities. The DSP is ideal for use in these situations, and while neither machine could capture video at full frame rate because there was no hardware compression built-in, Apple included a slot in the AV's circuitry for adding a third-party, hardware compression board. SuperMac has released a hardware compression board that plugs into the slot.

DSP versus RISC. An argument rages over whether DSPs are needed to perform the work involved in playing audio and displaying video. RISC chip supporters argue that the chips are fast enough that they can be used to do the work; you don't need to have a second processor (the DSP) to do it. DSP supporters argue that there will always be times when the RISC chip will have to do other things, compromising performance. Obviously, there is a savings involved in not adding a DSP, so for the moment it is unclear whether DSPs will reappear on the Macintosh.

When Apple unveiled the PowerPC based Macintosh (described below), the DSP was not used. Instead, the RISC chip was used to provide support for 16-bit audio, and a plug-in board added the video capabilities. (The PowerPC AVs add only video capture and display capability; all PowerPCs support 16-bit audio.)

Processor and clock speed

One factor in choosing a computer is the speed of the machine. A Macintosh Plus is much slower than a Quadra because of the speed of the processor. Existing Macintoshes use either the 68000 (Plus, and so on), 68020 (Macintosh II), 68030 (Ilci and most PowerBooks), or 68040 (Centris and Quadras). The higher the number of the processor, the faster they work—this is generally because the larger processors can work with larger amounts of data. Another variable is the clock rate. This is the speed at which the processor works with information. The higher the clock rate, the more work the processor can perform in a given amount of time. A 68030 running at 33MHz (megahertz) will be much faster than a 16MHz 68030. But a 25MHz 68040 will be faster than either processor.

Which chip do you need? The faster the chip, the faster the machine, and that's a good thing. But more importantly, the operating system routines that support color will not run on a 68000 machine—that's why you don't want a Plus if you want to work with color, or if you want to play QuickTime movies. If you want to work with 3D applications, then the existence of an FPU is just as important as whether the machine has a 68030 or 68040 chip.

Enter PowerPC

In 1994 Apple introduced the Power Macintosh, a radical departure for the Macintosh hardware family. Whereas existing models had been powered by processors based on the Motorola 68K family of processors, the Power Macintosh uses a new RISC chip co-designed and developed by Motorola, Apple, and IBM.

RISC chips work upon the theory that by simplifying the instruction set of the chip (the mathematical and data related operations the chip performs), you can create a much simpler and faster chip. Removing instructions does make the software more complicated, but the theory (which seems to be correct) is that this is more than offset by improvements in chip speed.

But switching processors is a major change. After all, it's the reason that the Macintosh uses a different processor family from the DOS/Windows world (running Intel's 86 family); you can't run a program for the Macintosh on a Windows machine and vice versa. Software developers would have to recompile all their programs, and we wouldn't be able to run our old programs.

Emulation

That would normally be true for the Power Macs, too, but Apple wrote an emulator (a piece of software) which enables the Power Macintoshes to run old applications without any changes. Emulation is slower than native mode, but the Power Macintosh will be acceptable in emulation (unless you were previously accustomed to one of the fastest Quadras). The Power Macintosh has no FPU, so 3D applications won't run unless you get the native application or install the utility SoftFPU, which emulates an FPU. The latter option should only be considered in desperation—performance degrades tremendously.

Native mode (applications compiled for the PowerPC chip) is where the advantages of the processor change becomes apparent. Speed improvements are as much as three times the speed of the fastest Quadra.

Sound

The current Power Macintosh models (the 6100, 7100, and 8100) all support 16-bit audio without any additional hardware. This was previously only available with add-on hardware or on the AV models (see Chapter 11, "Sound"). The AV model Power Macintoshes are really only "V" models; they add video capture capabilities.

Which chip?

The first Power Macintosh models used the 601 chip, running at either 60, 70, or 80MHz. There are already later generations of chips under development which will be even faster than the 601 chip. There also is a 603 chip, which is not as fast, but draws less power. It will probably be used for future PowerBooks.

The future

You can expect faster machines at lower prices—the mantra of the computer industry. Also, expect that the NuBus card standard will be dropped soon in favor of another standard called PCI. The PCI standard will be used by other manufacturers (in theory lowering the price and increasing the market and range of products available). PCI will also support higher throughput rates—nine times the amount of data that can be transferred between the board and the processor or other boards.

Choosing the Right Macintosh

Choosing the right Macintosh can be difficult, particularly when you are just starting out. Often you don't really know what you want to do with the computer, which makes choosing the right one even more difficult. There is, however, one rule that has remained true so far:

It will be cheaper next year.

Whatever computer you buy, it will be cheaper and faster next year. In fact, the computer you buy today very easily could be obsolete tomorrow. If you can, put off buying a computer for as long as possible. This doesn't mean, however, that you should postpone *working with* computers. Sure, if you wait, the equipment will be cheaper, but in the meantime you are missing out on opportunities, and other people are advancing. So, if you can beg, borrow, or steal computer time from other sources, then postpone making a purchase. But at some point, you will have to jump in and take a chance.

The multimedia user's Macintosh

Although it would be nice if a multimedia user could buy any Macintosh and be able to run any available multimedia production, this is unfortunately not the case. Many current multimedia productions have very specific requirements:

- Color
- 13-inch or larger monitor
- 5 MB or more of System memory

If you have a Macintosh that meets all these requirements, then the chances are very good that any multimedia production can run on your computer. But, just in case it can't, be sure to check the requirements of a multimedia product before making a purchase.

Of course, these requirements do not necessarily apply to special multimedia productions. For example, if a museum is planning to build and install an interactive system, it can specify the minimum requirements and build its multimedia presentation to match those.

The multimedia developer's Macintosh

Do not despair that the constant changes make it impossible to keep up with the latest hardware available. Just as you can't afford to keep buying new equipment, neither can anyone else!

If you are buying a new machine today, it makes sense to buy a Power Macintosh because that's the direction that Macintosh hardware seems to be taking. Most multimedia applications are available in PowerPC versions.

The next question is which one? Today the question becomes how many slots do you need? This question is made more complicated because Apple plans to switch from the NuBus plug-in board standard to PCI.

- Color (at least 16 or 24 bits)
- 13-inch or larger monitor (and/or multiple monitors)

- 8 MB or more of System memory (about 20 is becoming the minimum)
- At least 200 MB of free disk space after you have installed applications

Other Hardware

After you have purchased your Macintosh, you might think that you have all you need to start working. Sadly, this is not the case; you may need to consider buying several other products to improve or augment the performance of your base computer.

Graphics display boards and graphics accelerators

All the Macintosh models available have some type of built-in video support. You still may need to buy a separate video graphics card, however. Third-party cards offer several benefits, including support for alternative bit depths and built-in graphics acceleration.

For example, the PowerMac 7100's built-in video card displays only 16-bit color, but you could buy a 24-bit color card and use that if you need to view graphics in 24 bits. What bit depth do you require? For photographic retouching work, you probably will want 24 bits. For video work, 16 bits is often preferred because the quality of the image is similar to 24 bits, but the data rate is reduced. The bit depth, of course, is primarily a consideration when creating the video. You can display a 16-bit movie on a 24-bit screen and vice versa, but if you view only 16-bit movies on a 24-bit monitor, you aren't going to notice any improvement, so why pay extra? For 2D animation work, 8 bits is often all that you need. For most 3D renderings, 24-bit color will enable you to see the best-looking images, though you don't need a 24-bit display to render the images.

Storage devices

Multimedia productions, particularly those that involve sound or QuickTime (video), quickly fill hard disks and any other storage medium you may have available. Unfortunately, there is no hard and fast formula for predicting how large your productions will be. You should buy the biggest hard disk drive you can afford, with 300 MB as a bare minimum. This number does vary depending upon the other types of storage you might have available. If you have a removable disk of some kind, then you may be able to work with less hard disk space.

Hard disk drives

Two important numbers measure the performance of a hard disk drive. The access time, or seek time, is the time it takes the head of the disk drive to get to where the data is stored on the hard disk. This number is important because it affects the time it takes for things to start happening (such as for a movie to start playing). More importantly, if you are accessing a lot of information stored in different places, then a slow access time will affect performance. Most hard disk drives available today have access times in the range of 10 to 17 ms (milliseconds), which is the recommended range.

The second number is the sustainable transfer rate—the amount of information that can be transferred from the hard disk to the computer. If you are recording or playing QuickTime movies, then you will want a drive with the best performance possible. Unfortunately, this number is not dependent just upon the disk drive. Many Macintosh models cannot support transfer rates higher than around 1,500 kilobytes per second because of the limits of the SCSI bus in those models. Some hard disk manufacturers have bypassed this problem by using a NuBus card that plugs into the Macintosh and transfers information through the NuBus hardware rather than through the SCSI hardware. As one example, Storage Dimensions sells a 1G (gigabyte) 2FAST SCSI disk drive that has a transfer rate for writing data of 2,097 kilobytes per second and a read rating of 2,419 kilobytes per second when used with a Macintosh IIfx. This is very close to the rating of a Quadra.

Some hard disk manufacturers sell special AV model hard disks. These drives don't perform the automatic recalibration which some drives perform. Drives which automatically recalibrate are checking to see if the

disk has expanded or contracted due to changes in temperature—such changes can affect the reading of the tracks on the disk. While the recalibration is important, if it happens while you are trying to access the disk then performance is seriously jeopardized. This is particularly true if you are using a streaming application (such as QuickTime), but not so important if you are working with a program like Macromedia Director, a graphics program, or a 3D application.

Removable media

Several removable media options are available. The most common ones in use today are SyQuest, Bernoulli, and 128 MB optical disks.

SyQuest drives originally stored 44 MB on a cartridge, but then came models supporting 88 MB. The 44 MB and 88 MB drives are the same physical size and shape—about five inches square and three-fourths of an inch thick, but the older drives cannot read the 88 MB cartridges. Some of the 88 MB drives can only read the 44 MB, some can read and write to them, and it has been reported that some can't read or write to the older size drives.

SyQuest also has released a 3.5" format disc which holds either 105 MB or 270 MB, and a 200 MB unit in the 5" format.

SyQuest drives have two advantages. Because of their popularity, both the drives and the disks are inexpensive compared with other removable media. The disks, however, are more fragile and more subject to errors than the other removable media mentioned here. If you need to send large amounts of material to service bureaus or customers, you may want to buy a SyQuest just because your customers might have a SyQuest drive.

OEM: Original Equipment Manufacturer. For a curious reason, an OEM company simply puts its label on a unit made by another company.

Bernoulli drives and disks are made and sold by only one manufacturer, unlike SyQuest drives, which are made by one company but have several different companies which OEM these drives. Bernoulli disks are available in two sizes, similar to those of SyQuest disks (both in physical size and storage capacity). These drives are not based on the Winchester technology used in SyQuest drives, but on a different design that is much more resistant to disk crashes and other media problems. If you are constantly moving your drive (even while it's running), you should consider a Bernoulli drive. They are slightly more expensive than SyQuest drives, however, and not as widely used.

The 128 MB optical disk drives have become popular over the past couple of years, particularly as the price has come down to less than \$1,000 for the drives. The disks are about the size of a 3 1/2-inch floppy disk, but double the thickness. The initial costs are higher than a SyQuest or Bernoulli system, but the cost per megabyte is lower for these drives. Double-sized drives (256 MB) are now available, and you can look for even lower prices over the next few years.

Some other removable drive options are available, including Flopticals (which hold only 20 MB) and 600 MB opticals (which tend to be slow and expensive). Neither one is as popular as the SyQuest and Bernoulli drives, and they are not recommended, primarily for that reason.

CD-ROM

The CD-ROM medium stores large amounts of information (600 MB) on a distribution medium that is cheap to reproduce (costing less than \$2 per disc when ordered in quantity). It has taken several years for CD-ROM to establish itself, but there now seems no doubt that CD-ROM can be used successfully for distributing commercial products. The cost of CD-ROM drives has come down, and Apple now includes a built-in CD-ROM in most models.

The slow transfer rate and access times of CD-ROM drives compared with hard disk drives does limit their application. Double-speed drives and disk caching software can improve performance considerably, and careful attention to design by the multimedia producer can mask some of these performance problems, but it's unlikely that CD-ROM drives will approach the performance of hard drives soon.

Creating a CD-ROM is surprisingly simple, particularly if you are creating a Macintosh HFS (Hierarchical File System—the system used for organizing and storing files on a Macintosh) format disc. All you have to do is place the files on a hard disk, arrange them how you want them to appear, and then send them to a CD-ROM pressing plant. The pressing plant creates a master (costing around \$1,500 to \$2,000) and then presses as many discs as you want for about \$2 each. Prices vary from company to company and differ depending upon packaging and label printing options. All manufacturers create a check disc, which is a single copy of the CD-ROM that you can use for testing purposes. This disc costs around \$300 and is well worth the money to make sure that the CD-ROM works the way you thought it would.

Caching: The software saves extra information read from the disk into a piece of memory. This often saves time if the user wants to read something that has already been read, and seems to improve the performance of the drive, even though the speed of the mechanism itself has not changed.

Creating a CD-ROM that is compatible with both Macs and PCs is also possible. Several commercial games and educational products are already available for both computers on a single CD-ROM. The CD-ROM pressing plant will provide you with specifications for preparing the disc and associated artwork.

A short list of CD-ROM plants

3M Optical Recording Department
3M Center Building 223-5N-01
St. Paul, MN 55144-1000
(612) 733-2142

Nimbus Information Systems
Box 7305
Charlottesville, VA 22906
(804) 985-1100

DMI
1409 Foulk Road, Suite 202
Wilmington, DE 19803
(302) 433-2500

Optical Media International
485 Alberto Way
Los Gatos, CA 95032
(408) 395-4332

METATEC
7001 Discovery Boulevard
Dublin, OH 43017
(614) 761-2000

SONY Electronic Publishing
1800 N. Fruitridge Avenue
Terre Haute, IN 47803
(812) 462-8160

CD-ROMs are usually pressed; however, systems are available that “burn” information into special blank discs to create one-off CD-ROMs that can be used in regular CD-ROM drives. These devices are expensive (up to \$6,000), but there are low-priced models (under \$3,000). The differences seem to be in features and software functionality. Blank discs cost around \$30 each.

Some service bureaus can create these discs for you (essentially they are offering the check disc service offered by the disc pressing plants but without the option to mass-produce the discs).

Laser discs

Although laser video discs may appear to be little more than oversized CD-ROMs, they are actually a completely different format for storing information. Laser video discs store only video and sound and provide that information not in a digital format, but in an analog format suitable for playback on a television monitor.

Laser discs have been used with computers to create interesting presentations. Professional laser disc players have a serial port that can be used to control the device. With the appropriate cable and software, the Macintosh can start, stop, and advance the player to a specific frame on the disc. Usually an external monitor displays the signal coming from the laser disc, which means that the user has to deal with two monitors—the Macintosh and the television. Figure 5.1 shows how a Macintosh, a laser disc player, and a television monitor work together.

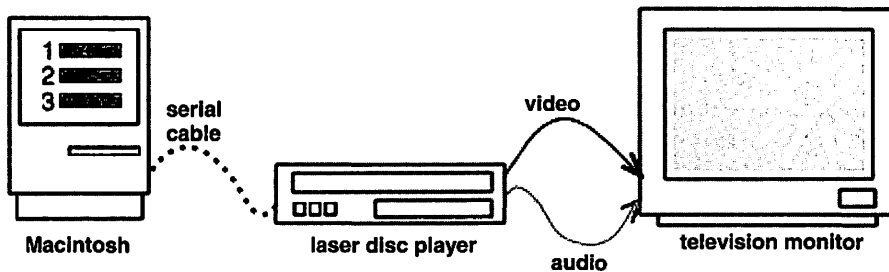


Figure 5.1 When using a laser disc connected to the computer, the user tells the computer what to do and then watches the result on a television monitor.

It is possible to use a video display board for the Macintosh which can display video on the Macintosh screen. This configuration creates a seamless production, but it increases the cost of the equipment needed to play back the production because you need a Macintosh with a color monitor and a special card that will display NTSC (the standard used for television transmission in the US) video on the Macintosh screen.

Using laser discs has both advantages and disadvantages. Access time from one end of the laser disc to the other can amount to several seconds, making it impossible to jump seamlessly from one part of the video to another. Integrating a laser disc with your production is much more complicated than working with hard disk drives or CD-ROM drives because of the extra software, hardware, and cables necessary to get the thing working.

The quality of the sound and video, however, is much higher than is currently possible with QuickTime (currently the only alternative if you need video in a multimedia production). Also, a laser disc can be used as a source for a large number of still images. Each frame of the video can be

APDA telephone numbers: F
rom the United
States call
1-800-282-2732;
from Canada call
800-637-0029;
from other countries
call 408-562-3910.

accessed independently, and an hour of video is made up of more than 100,000 still images. You can have a laser disc made for \$300—there are a few service bureaus that do this, check in the back of *MacWeek* or *Macworld*—which makes it quite an attractive medium for special installations.

The HyperCard VideoDisc Toolkit is a collection of XCMDs for controlling laser discs from within HyperCard. It is available from APDA for \$40.

Still other hardware

Chapter 7, “Graphics,” contains information about scanners and drawing tablets. Chapter 9, “3D,” discusses rendering acceleration options. Chapter 10, “QuickTime,” contains information on video digitizers and hardware compressors. Chapter 11, “Sound,” discusses sound digitizers.

The Future

Apple is working on versions of its System software that will run on other platforms. There is rumored to be a version of the System under development that will run on PCs. Apple is also moving in other directions with Kaleida, Taligent, and Newton.

What does this all mean? It's too soon to tell where the Macintosh operating system will go—whether it will continue to evolve or become obsolete. Even less clear is what these newest projects mean for multimedia developers who currently work on the Macintosh. Still, it's too early to start panicking. Apple is strongly committed to the Macintosh and will remain so for the foreseeable future.

Text

6

chapter

You may wonder why a book about multimedia includes a chapter about text. Text is an important part of multimedia presentations and programs. Text is often the most concise and compact way of communicating a large amount of information—at times, it's the only way. Multimedia developers sometimes get carried away by the bells and whistles of sound and video, and forget that text may be conveying the most important information in the production. This chapter is here to remind you not to make that mistake.

When you discuss text, you also think of Hypertext—the linking of related information. This chapter explores hypertext, including an unusual new application of Hypertext on the information superhighway (Internet).

The chapter begins with a brief history of text on the Macintosh—particularly how the Macintosh stores and displays text. If you already are comfortable with terms such as bitmapped font, point size, PostScript, TrueType, and ATM, you can skip the first section and go straight to the sections on what text says and how it looks. The following sections describe how hypertext (linked text in electronic documents) works, provide tips for using text in productions, and end with a rather unusual use for text in your productions.

The chapter concludes with a description of print-to-disk utilities—a method of distributing materials that is usually, but not necessarily limited to, textual documents.

A Brief History of Text on the Macintosh

The first Macintosh was rather unique in the personal computer market in that it used **bitmapped fonts** to create text. When a computer uses bitmapped fonts, it stores each text character as a small graphic file. The character is defined by the arrangement of pixels onscreen. The computer stores a separate graphic for each **point size** (height) of each character in the font. The larger the point size of the font, the larger the size of the file that contains it.

To limit the amount of space required for storing bitmapped fonts, early Macintosh computers offered a limited number of font sizes—10, 12, 14,

18, and 24 points. Users could display text in other sizes, but the computer simply scaled one of its existing font sizes, so the user didn't always get the desired result.

Early Macs had a screen resolution of 72 pixels per inch (ppi) and stored the fonts at that resolution. (Most Macintosh monitors today have screen resolutions between 72 and 80 ppi.) The fonts looked great onscreen, but didn't look good when printed.

At that time the only printer that worked with the Macintosh was a dot-matrix printer that printed at 144 dots per inch (dpi)—a dot on a printer is like a pixel on the screen, so you can compare ppi to dpi. The problem with printing a 72 ppi font on a 144 dpi is that when the 72 ppi character gets to the printer it has half the information (or pixels) that the printer is capable of printing. The printer will print the character by scaling it. The character will look okay, but it won't look as smooth as the printer is capable of printing it.

To improve the resolution of printed type, the Macintosh sent characters of double the size being printed to the printer. If you were using 12-point type, for example, the computer sent 24-point (double the size) type to the printer. The printer scaled the 24-point type to 12-point type, which created better-looking printed characters (because the 24 point, 72 dpi character translated into a 12 point, 144 ppi character on the printer—see figure 6.1). This arrangement worked fine for text in 12-point type. But if you wanted to print 24- or 48-point type, then you needed to use a font of double the size to print, otherwise you wouldn't get the desired result.

Shortly after introducing the Macintosh, Apple entered into an arrangement with Adobe, a small company that had been developing a page description language called **PostScript**. Apple released a laser printer that printed at 300 dpi and used PostScript. Bitmapped fonts were of inadequate quality for use with this printer. Further, PostScript had its own font description that consisted of a mathematical description of each character in a font. PostScript stored this information in a PostScript font file that existed in ROM on the laser printer or on the Macintosh (in the latter case, the computer must send the font to the printer each time a document needs the font). Figure 6.2 illustrates the process of printing a PostScript font with the Macintosh. In order to use a PostScript font in a document, however, you must still install a bitmapped version of the font in the Macintosh System.

A Page description language is a computer language that tells a printer what to print on a page. Rather than working out which pixels to turn on and off and then sending all that information to the printer (which takes a long time), the computer simply sends a description of what to print by using the page description language. For example, the computer tells the printer to draw a circle of a certain size at a specific location.

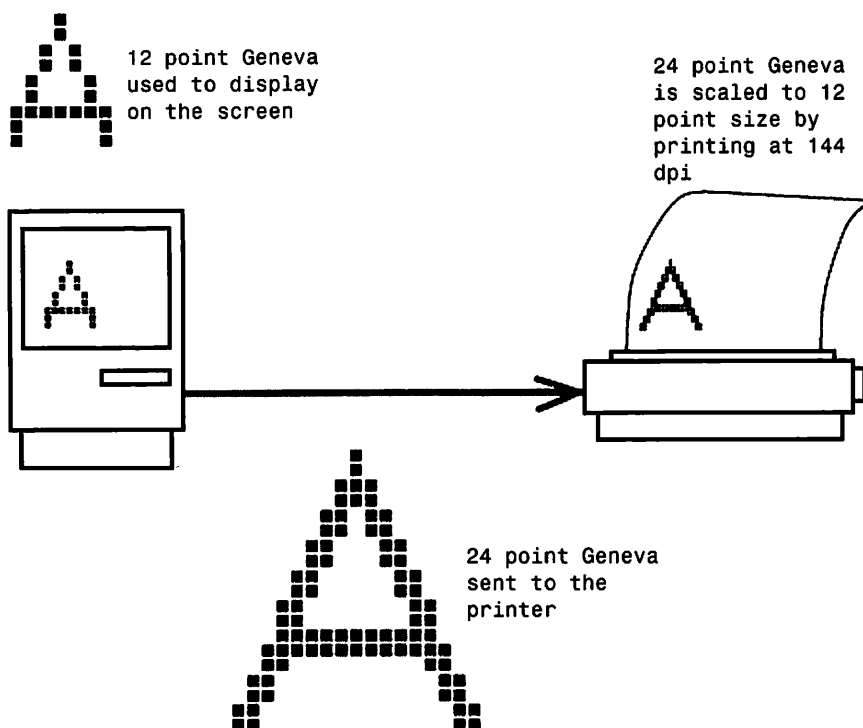


Figure 6.1. Printing bitmapped text to a dot-matrix printer.

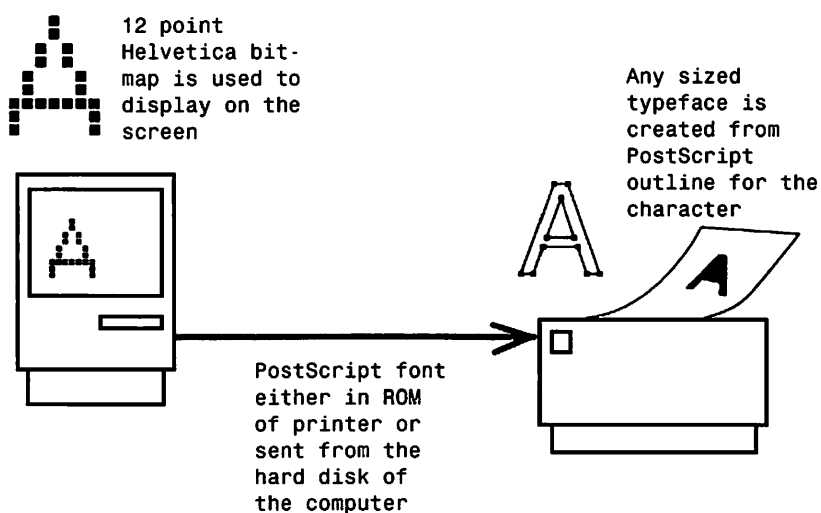


Figure 6.2. The relationship of bitmapped and PostScript fonts.

Note that figure 6.2 shows a laser (PostScript) font, called Helvetica, while the previous figure (6.1) shows a bitmapped font called Geneva.

Although the first PostScript fonts looked great when printed, their onscreen display was still limited by the bitmapped sizes because the Macintosh system still used the bitmapped fonts when displaying text onscreen. To overcome this limitation, Adobe developed Adobe Type Manager (ATM). This product used the PostScript font printer files to create any size font for display on the Macintosh screen. ATM was particularly helpful for viewing large fonts onscreen.

At about the same time that Adobe introduced ATM (perhaps there was a connection!), Apple announced a new type technology called **TrueType**. In a sense, Apple designed TrueType to replace the bitmapped and PostScript files with a single file that displayed a font at any size. TrueType entered the market as a competitor to the existing standard of PostScript fonts, and the publishing world was uncertain what effect the two products would have on each other's sales and popularity.

At this writing, both products appear to hold their share of the marketplace. PostScript continues to be very popular in desktop publishing applications, and TrueType is distributed with many of Apple's low-cost printers. The introduction of QuickDraw GX promises the addition of even more capabilities to TrueType. Many fonts are available in both TrueType and PostScript and, as a further benefit, you can use TrueType and PostScript fonts at the same time.

QuickDraw is the set of routines that applications use to draw graphics (including text) onscreen. QuickDraw is very similar to PostScript (it can be thought of as a page description language, but whereas PostScript is resolution independent, the original version of QuickDraw was limited to 72 dpi—the resolution of the Macintosh screen). The new version of QuickDraw, QuickDraw GX, upgrades QuickDraw so that it can be considered as a full-page description language.

Tip

The price of fonts has fallen dramatically in the past few years. Adobe sells a number of type sets that contain a selection of different display fonts at a very reasonable price. Adobe also sells Type On Call, a CD-ROM that contains all Adobe fonts. You must buy access to each PostScript font, but you get a bargain deal on one font. Best of all, the disc contains all the bitmapped versions for all Adobe fonts, and you can use the bitmapped versions at no charge.

Several other companies offer collections of fonts as well, so most users can put together a reasonable collection for very little money.

What Text Communicates

As a multimedia producer you need to consider more than just the information text represents. You need to consider electronic features that can be added to text—like hypertext—as well as more down-to-earth considerations such as, “Will the user have the font I am using in my presentation?”

Text

Text might appear to be dull and boring. It doesn’t have the excitement and appeal of sound and video, but it *is* important, and computers add new power to text that it never had before—the capability to link text together so that the user can quickly and easily find related information.

Hypertext

Hypertext is a term coined by Ted Nelson (a pioneer in the field of computer interactivity) to describe the linking or threading of text to help communicate knowledge. In some respects, hypertext is like a computerized footnote. By simply clicking a word, you can see more information about the word (see figure 6.3). But why limit the link to words and their description? Perhaps the link may be made to every other occurrence of the word in the book, to other books, or perhaps to a bibliography. The possibilities are unlimited.

That is why it’s so frustrating that creating hypertext documents on the Macintosh is still amazingly difficult. There is no program that makes it easy to do. About the only applications that you can use are HyperCard and SuperCard, and while you can create hypertext links in these programs, it’s not as easy as it should be. HyperCard is not hypertext. (For more information on HyperCard, see Chapter 15, “Authoring Environments.”)

The HyperCard programmer can find which word in a text field the user clicked on, but the programmer has to then use that word to search for related topics. In other words, it’s up to the programmer to create the

links. HyperCard has an added feature called Groups, that enables the developer to group two or three words as one term. When the user clicks on one of the words in the term, HyperCard returns all the words in the Group. This makes it possible to provide a link to the term, for example, “Ted Nelson,” without worrying about whether the user clicks the word “Ted” or “Nelson.”

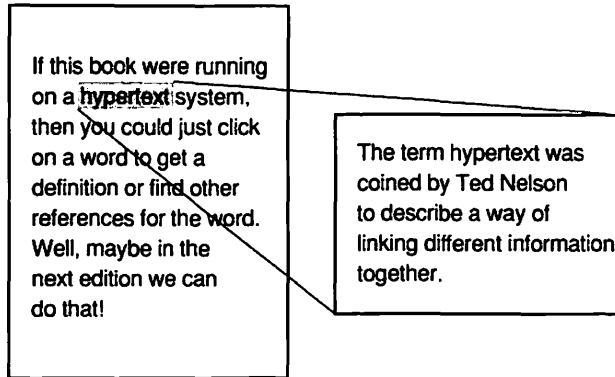


Figure 6.3. The basic idea of hypertext.

When you create a text-rich multimedia production, try to imitate the features of a book in your production. Include, for example, a table of contents, an index, and a bibliography. In addition to those features, you might consider adding a dictionary or appendix with definitions for selected terms. As a final step, you can add links between words.

Some hypertext elements are logical design choices. Creating a table of contents and an index as separate hypertext system elements that users access through a tool palette or menu command is a logical—and obvious—design decision. Other elements of designing a hypertext system are less obvious. In designing the interface for a hypertext system, determining how to enable the user to interact with the features is the biggest, and most frequently experienced, design problem.

You must deal with a number of issues when designing a hypertext system. The following sections describe the two most important issues.

Getting around in hypertext

All hypertext designers must determine how to implement links within the hypertext system. An example of a simple hypertext system link is as follows: If you encounter a word about which you want more information, you click on the word and the information appears. If the link has only one destination—for example, a dictionary definition—then a simple hypertext link can display the information when you click a word. But what if the information is in more than one place? What if you need to create an additional link in the text? If a word is referred to only in one other place in the text, then perhaps you can link to that place. But what if there are other references (see figure 6.4)?

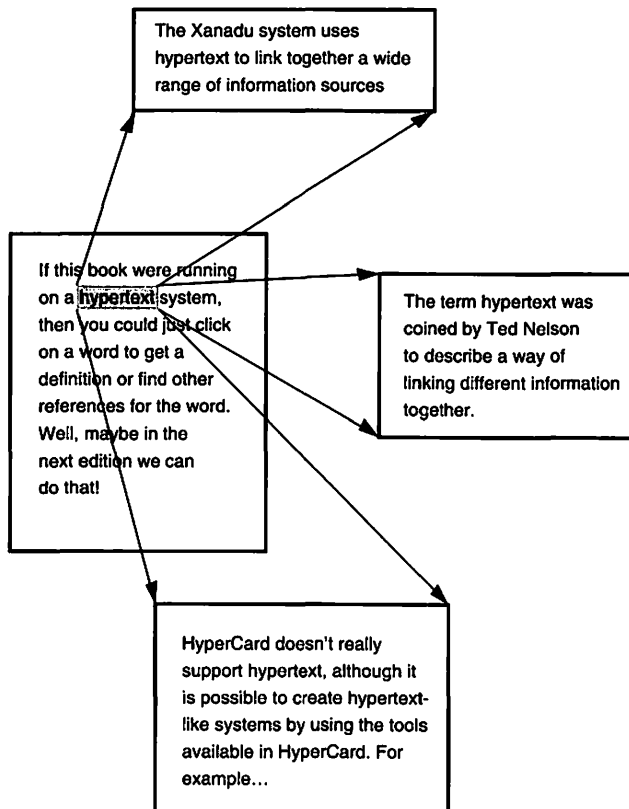


Figure 6.4. Hypertext poses the problem: Where to go from here?

You can choose either of two solutions to this problem. One solution is to hold down a special key for each type of destination. For example, Command-clicking may go to the dictionary, and Option-clicking may go to a reference elsewhere in the text. The other solution is to list all possible links in a palette that appears when the user clicks on the word.

None of the linking solutions is completely satisfactory. Pointing and clicking is the easiest linking method, but inadequate for multiple destinations. Option- or Command-clicking to take you to a specific place requires that you know which key to use to get to that place. The palette provides an easy way to get to multiple places and shows you all the choices, but it requires a lot of extra clicking if the vast majority of links are single links to the dictionary (you must click the word, and then click the entry in the palette). The data in your hypertext system may help you decide how to implement linking features.

Getting back to where you were in hypertext

The next problem with hypertext is returning to the original position after the link is completed. If you click on something and are taken to another part of the text, you must have a way to get back to where you were. HyperCard provides two features for achieving this return. The most commonly used method is to place a button on the card which, when clicked, takes you back to the previous card. This button uses a command that is available from the Go menu in HyperCard to take you back to the most recently viewed card. The other feature is the Recent window, which shows thumbnails of where in the text you have been (see figure 6.5). You can use the Recent window to jump back to a previous card.

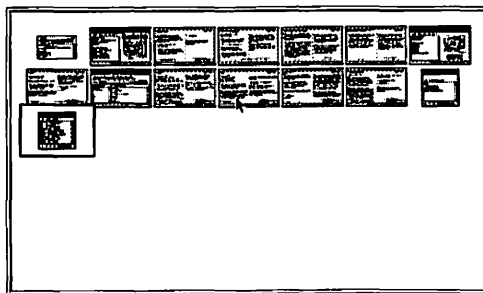


Figure 6.5. The Recent window in HyperCard.

Both of these methods of returning to a link's starting point are helpful to the user, and you should provide access to them. The most important rule is to provide a clear way for the user to get back to where he was located in the text. If your hypertext system does not do this, the user wastes a lot of time making the return or, even worse, gets hopelessly lost in the effort.

Using Text in Productions

The discussion of computer text and type uses some typesetting terminology in ways uncommon to other print media. The following definitions, therefore, may eliminate any confusion over the use of those terms in this chapter.

In strict typesetting terminology, a **typeface** is a specific style (for example, Helvetica Bold), and a **font** is a specific character size of a specific typeface (for example, 14-point Helvetica Bold). A **type family** is a collection of similar typefaces (for example, Helvetica, Helvetica Italic, and Helvetica Bold make up a type family). The Macintosh documentation, however, uses the term "font" to mean type family (for example, Helvetica is a font, and bold or italic are styles).

When using fonts in multimedia productions, you will often encounter the same questions that traditional graphic designers consider: Which font communicates the information? Which font looks best? In short, which font should I use?

Multimedia designers have an additional problem because they have to design for the computer screen. This means accommodating the smaller screen area (compared to a standard piece of paper), as well as the lower resolution of the screen. (The lower screen resolution means that you usually have to use slightly larger fonts than would be possible on a printed page because printers usually have much higher resolutions.)

Fonts

The multimedia developer should have access to a wide range of fonts. As a designer, you must decide whether to use bitmapped, PostScript, or TrueType fonts. When selecting a font, remember that some fonts are common to almost every user's System. Those fonts are: Geneva, New York, Chicago, Times, Helvetica, Palatino, and Zapf Chancery. You can use

these fonts freely in your multimedia project designs without worrying that a user's System doesn't support the font you have selected. If you use almost any font not included in the previous list—whether it is bitmapped, PostScript, or TrueType—you face the potential problem that users might not have the font installed in their System.

If a user runs a project that includes a font that she doesn't have, another font will be substituted (often Geneva). The user can see and read the type, but the substitution can often make the presentation difficult to read.

How can you avoid this problem? You could include the fonts you are using as a separate file with the presentation. The problem with that is that most fonts are protected by copyright. If you want to give a copy of them to someone you must license the copy.

If you decide to include a copy of a bitmapped font, you must determine how to distribute the font with your production. Apple maintains that the best way to accomplish this distribution is to have the user install the font into his System file. This solution, however, is not always ideal because it's an extra step the user has to perform.

An alternative solution is to install the font into the player application; when that application is open, the font is available. Apple does not recommend this procedure, primarily because of possible font conflicts. Apple also may fear that this method might not work in a future version of the System. This method also causes problems if you try to print a document using a print spooler (when the spooler tries to print the document, the font may no longer be available).

Tip

If your production includes a large amount of text, I recommend that you distribute a copy of the font. This advice is particularly appropriate if you need only the smaller type sizes, in which case you can use a bitmapped copy. For larger headlines or display type, consider converting the text to a graphic. Although this technique consumes more memory (particularly if you are storing a lot of text this way), it does result in text that always looks right.

The issue of font protection is complex because a typeface currently cannot be protected under copyright laws. Some of the program code used to generate PostScript fonts, however, has been granted protection. As a result of this protection, someone can create a font that looks like a protected PostScript font, but can't simply copy the PostScript file.

Bitmapped font files are a different story. Although a company may want to claim a copyright on a bitmapped font, most companies (like Adobe) have allowed users to copy the bitmapped versions freely, so the users can create layouts at home and then go to a service bureau to print the final copy.

Some production designers may want to create their own fonts. You can create fonts in a couple of ways. If you need only a few bitmapped font characters, you can create them using ResEdit, Apple's resource editing program. ResEdit is available from user groups, online services, and APDA (Apple Programmers and Developers Association). ResEdit is not a simple program to use, and very little documentation exists for it. A good reference is the *ResEdit Reference*, published by Addison-Wesley. You can use ResEdit to customize icons and cursors. Figure 6.6 shows the creation of a special character in a font using ResEdit.

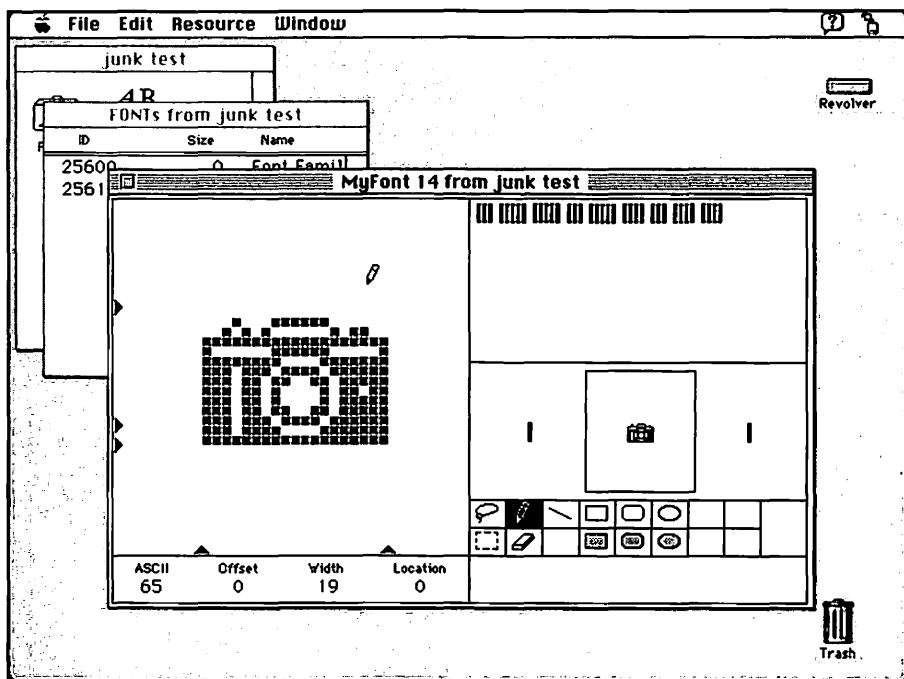


Figure 6.6. Editing an iconic character in a font, using ResEdit.

To create a PostScript or TrueType font, you must use a program such as Fontographer or FontStudio. These programs are designed for creating fonts, but also provide sophisticated drawing tools that rival those in Adobe Illustrator and Aldus FreeHand.

Hyperfonts?

You can use fonts to create unexpected things—such as buttons and icons. You can put just about any kind of graphic into a font, including signatures or other items that you may use many times.

A designer can create a control panel, for example, made up of a text field with a special font for the control panel's buttons (figure 6.7 shows an example I created in a HyperCard stack). When the user clicks a “button” in the field, a script determines the character pressed and then briefly substitutes a reversed character to provide a highlighting effect.

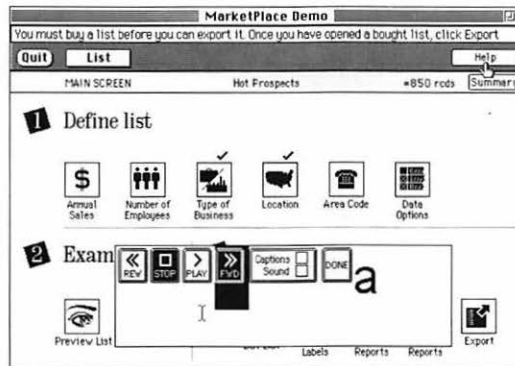


Figure 6.7. A font used to create buttons in a control panel.

As the example in figure 6.7 above shows, there's more to fonts than just text!

Internet HyperText

The information superhighway may just be a dream or marketing hype, but there seems general agreement that the precursor of the superhighway exists—the Internet.

The Internet is of interest here because of a new feature of the Net: the World Wide Web. The Web is really just different servers (computers) on the Net that have documents using the HTML (Hypertext Markup Language). These are documents which contain links to other documents,

graphics, or other objects on the network. The interesting feature of HTML documents is that they can point to documents on other systems—this creates an international collection, and connection, of information. In many respects it may be the first step to Xanadu.

HTML documents, when viewed in a text editor, are uninteresting; they contain the text of the documents, as well as indexes that point to the other documents or graphics on the network (see figure 6.8). They become interesting when you use a program such as Mosaic, or MacWeb, which are able to display the documents and interpret the indexes. These programs hide the links, providing a document that you simply click on to display another document (which may be contained on a computer in another country)!

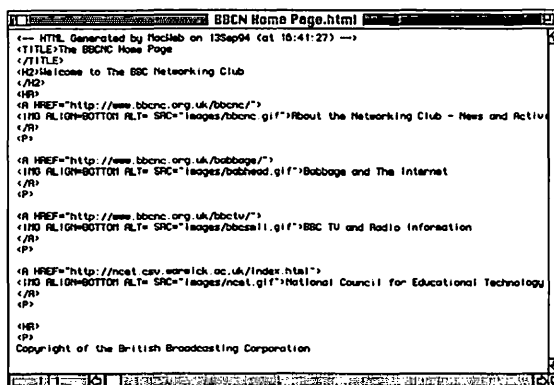


Figure 6.8. An HTML document viewed in a text editor.

HTML is a recent development. There are currently no programs that enable you to create an HTML document; you have to do that manually. Most of the software to access the World Wide Web (Mosaic, MacWeb) is public domain or shareware (see figure 6.9).

To run these programs you need a copy of MacTCP (routines that talk TCP/IP (Transmission Control Protocol/Internet Protocol) and either a SLIP (Serial Line Internet Protocol) or PPP (Point to Point Protocol) connection and software drivers that support the chosen protocol. Getting all this stuff together and working is beyond the scope of this book. The *Internet Starter Kit, 2nd Edition* published by Hayden Books not only covers this topic, but includes most of the software mentioned above.

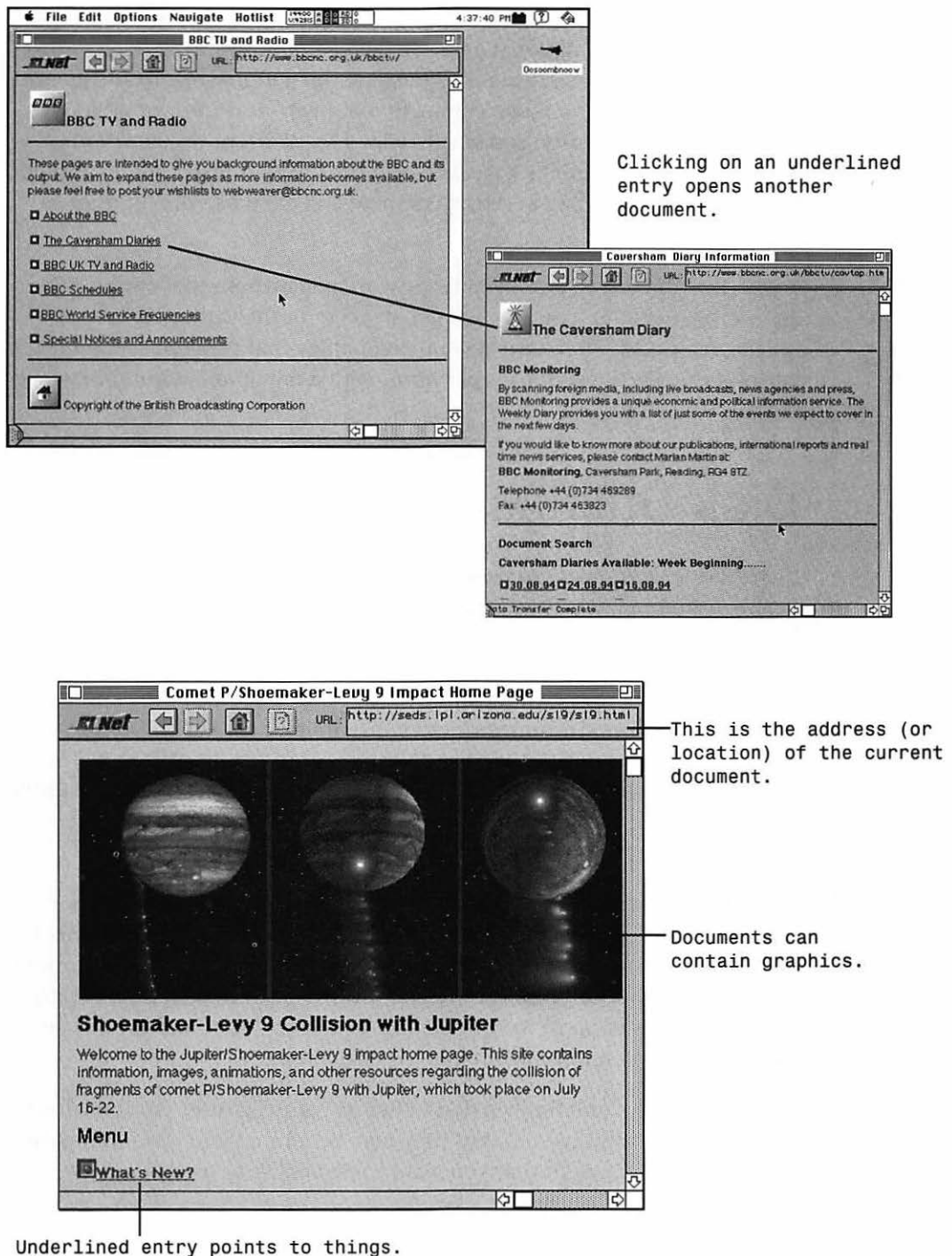


Figure 6.9. Viewing an HTML document in MacWeb.

The future of the World Wide Web is dependent upon the Internet. Originally conceived as a national network between universities and government institutions, the Internet has become the national electronic meeting place. The Internet is a loose connection at best—it is not run or controlled by a single entity, and is subsidized by different organizations. This may be a strength or a weakness in the future. Certainly there could become conflicts as pay services compete and even use the Internet to provide services.

If you don't belong to a university, or work somewhere that can provide an Internet account, you can still get an account through a number of different sources. There are several companies that provide accounts at a price—typically a few dollars per hour, with a minimum per-month usage fee.

Disk Utilities

Distributing documents to users who didn't have the application created problems in the past. That's where print-to-disk utilities are useful. Essentially, these programs act in a similar manner to a virtual printer. Your application sends information to the utility and it creates an electronic version of the document that is saved in the Print-to-Disk utilities own format. These documents can be distributed; usually there is a Reader utility that can read these documents but not create them. The Reader can be distributed either for a fee, or for free (depending upon the licensing agreement). These utilities are very useful for distributing electronic documentation.

One of the most important features is that they are all cross-platform; there are player applications for Windows that enable you to open documents created on the Macintosh. Of the three utilities available—Acrobat, Common Ground, and Replica—Acrobat is the most powerful and feature-rich. It includes support for QuickTime, provides thumbnails of the pages in the document, and even offers a hyperlink feature.

Common Ground and Replica don't offer as many features, so they don't come close to "multimedia," but they may be all that you need to create an electronic document that you want to distribute to others.

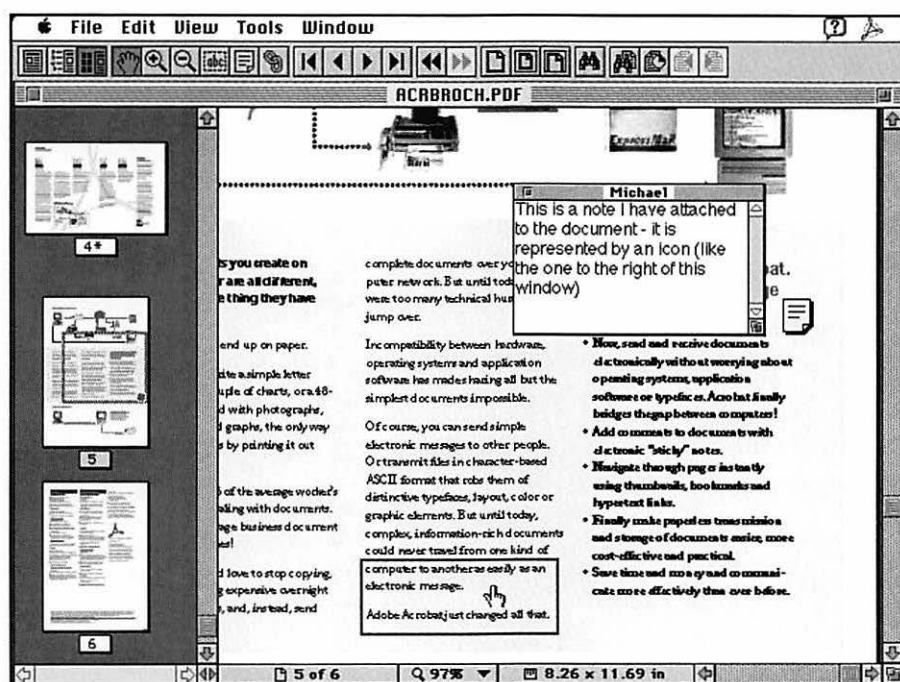
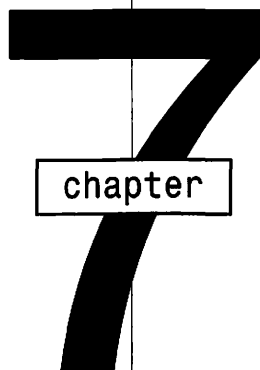


Figure 6.10. Viewing an Adobe Acrobat document.

The next chapter discusses graphics. It has been said that a picture is worth a thousand words, so it is perhaps appropriate that the graphics chapter follows this chapter.

Graphics



chapter

Creating graphics on a Macintosh is a simple process. You grab the mouse and “draw” or “paint” with it just as though you were holding a pen or brush. As you drag the mouse, a “pen” or “paintbrush” moves onscreen and produces the graphic. Creating graphics on a Macintosh is so simple and obvious that it’s hard to believe this chapter is necessary.

Well, of course, creating graphics on a Macintosh isn’t *that* simple. It took a long time to get to where we are today, and there are a lot of choices when it comes to creating graphics on the Macintosh, including what format to save the graphic in, and (more importantly) which of the many programs to use to create the graphic.

A Brief History of Graphics on the Macintosh

In the beginning was the screen. (And if you look closely, you will notice that the screen is made up of individual “points” called **pixels**, which stand for “picture elements.”) It didn’t take long for someone to realize that you could create graphics by changing the color of the pixels. And so computer graphics were born. By the late 1950s and the 1960s, some reasonably sophisticated graphics programs were available for mainframe computers, but it took the arrival of the personal computer in the early 1980s to bring computer graphics into the mainstream.

Graphics programs can be split into two broad categories; bitmap-based and object-oriented (the latter are also sometimes called “drawing” programs just to confuse things). With bitmap-based programs, an illustration is created on a “page” that contains a predefined number of pixels (the resolution of the illustration). The graphic is created by changing the color of individual pixels. This provides a lot of flexibility, but it is difficult to change elements once they have been added.

Object-oriented programs use mathematical equations to describe individual elements (such as a circle or rectangle). An illustration is created using these elements. It’s easy to edit elements after they’ve been created, but it’s more difficult to fine tune your illustration with an object-oriented program.

The first Macintosh graphics program was MacPaint. The first version of this program was designed for use with the early Macintosh computers, on whose screens pixels could be only black or white. MacPaint provided tools for drawing circles, rectangles, and lines, as well as paintbrush, airbrush, and eraser tools. Figure 7.1 shows MacPaint 2.0, the most recent version of this program.

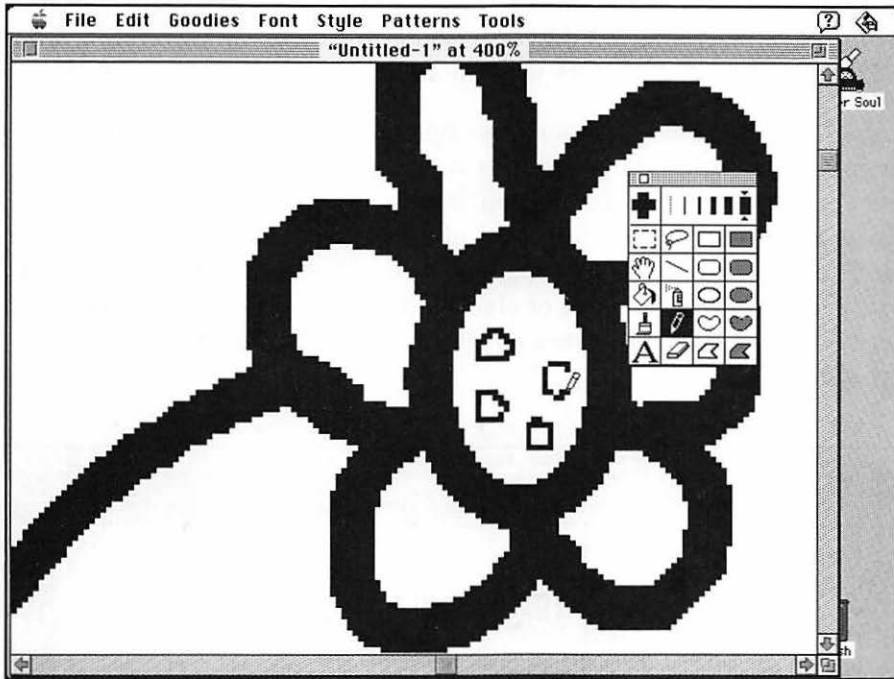


Figure 7.1. MacPaint 2.0, the most recent release of the original Macintosh paint program. The pencil tool is being used to add details to the graphic.

MacPaint is easy to use, but very limited. The program creates graphics at 72 dots per inch (dpi) and a fixed size of 576 by 720 pixels, which produces a graphic slightly smaller than an 8.5- by 11-inch sheet of paper when printed at 72 dpi. The 72 dpi resolution matches the resolution of the Macintosh screen, so the graphic prints at the same size it appeared onscreen. This is fine when printing with the ImageWriter printer (the first printer for the Macintosh) because the resolution of the ImageWriter is only 144 dpi. But the low resolution becomes noticeable when MacPaint images are printed on a high resolution printer such as a laser printer (300 dpi).

If you have a Classic, Plus, or similar Macintosh model, then each pixel onscreen can be one of only two colors—black or white. If you have a color-capable Macintosh and a color monitor, then each pixel can be black, white, or any other color (although the total number of colors that your Mac can display onscreen at one time varies depending upon your equipment).

MacPaint and other bitmap-based graphics programs have other problems. After you create a graphic, it is very hard to make changes. If you draw a line and then decide to move it just a bit, you have to erase the first line, pixel by pixel, and then draw a new line in the correct location. This is a very tedious procedure. Another problem is that MacPaint's graphics are limited to the resolution at which you create them—in fact, all bitmapped graphics are limited to the resolution at which you create them. If you create a graphic at 72 dpi, you cannot achieve better results by printing the graphic on a 300 dpi printer than you can by printing it on a 72 dpi printer. You're stuck with the 72 dpi resolution.

Enter MacDraw. This program uses **objects** (circles, lines, polygons) to create graphics. At first glance, this seems to be what MacPaint offered, but the difference is that when you draw a circle in MacDraw, the circle is saved as a coded description of the object. The object is defined as a circle of a fixed size in a fixed location rather than as a collection of pixels. You can easily move, delete, or change the size of the circle after you create it. Figure 7.2 shows how an ellipse can be manipulated after it is created.

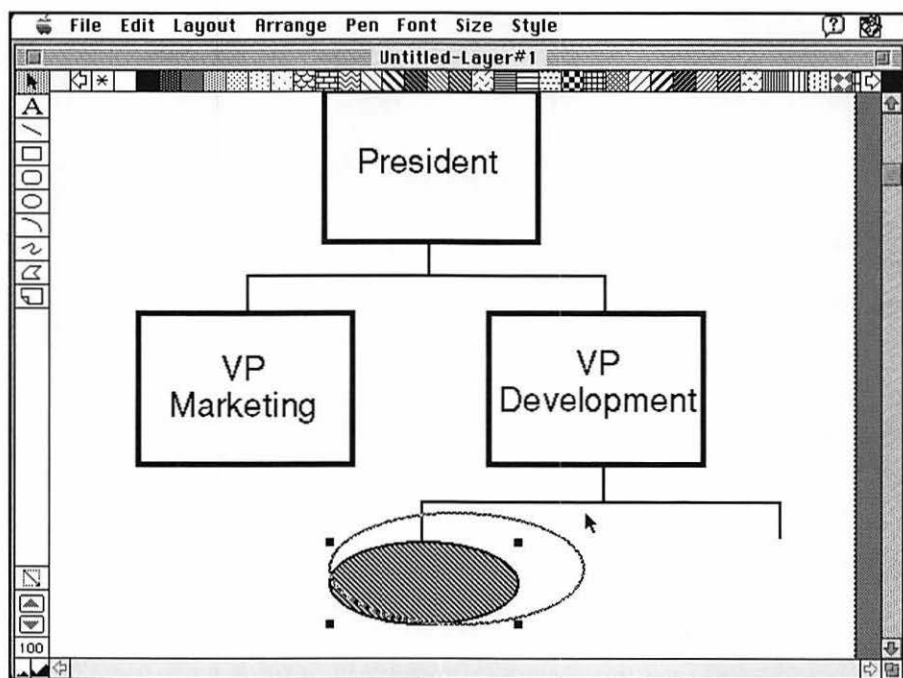


Figure 7.2. Resizing an ellipse in MacDraw.

MacDraw supported a new file format, called **PICT** (for **PICTure**), in which you could save graphics that contained objects as well as bitmapped illustrations.

Creating a graphic with MacDraw is very different from creating a graphic with MacPaint. In MacDraw everything is based on a mathematical description—a line, a curve, or a polygon. This can constrain the effects that you can create. And you can't manually turn pixels off or on in the graphic. Also, the objects in a MacDraw graphic are stacked on top of one another. Although you can select an object and move it in front of or behind another object at any time, you constantly must deal with the relative location of objects.

But MacDraw's object graphics are easier to edit than MacPaint's bitmapped graphics, and have other benefits as well. Because MacDraw saves the objects as coded information rather than as arrays of pixel values, MacDraw files are much smaller than MacPaint files. Even better, the resolution of the MacDraw objects depends only upon the device used to display or print the graphic. Onscreen, a MacDraw graphic looks as though it were drawn at 72 dpi. On a laser printer, the graphic is “drawn” at 300 dpi—the computer or printer simply adds more pixels based on the description of the object.

You may think that object-oriented graphics programs would have spelled the end of bitmap-based graphics programs. This was not the case (there is even a program that combines the features of a bitmap-based graphics program and an object-oriented graphics program—SuperPaint). First, no object-oriented program has yet been able to imitate traditional “painterly” effects as well as a bitmap-based painting program. Second, scanners create only bitmapped graphics, so if you use a scanner and need to manipulate the scanned graphics, then you will need a bitmap-based graphics program. Finally, it is almost impossible to recreate photographic quality artwork using object-oriented graphics (short of simply recreating each pixel with an object, which is hardly useful).

MacDraw and object-oriented graphics were a major step forward for some people, but others still struggle with pixels. Specifically, anyone with a scanner needs to save and edit scanned images. The original PICT format was not suitable for this task. (The original PICT format did not support the quantity of information that scanners produced. PICT has since been updated and now can be used to save scanned images.) To solve this problem, a group of software and hardware companies got

together and created an informal standard graphics format for large, multiple bit-depth images. This format, **TIFF** (Tagged Image File Format), is still widely used for scanned images.

Our history of Macintosh graphics continues...

Shortly after MacDraw appeared on the scene came the introduction of the Apple LaserWriter. The LaserWriter holds an important place in history—by enabling the desktop publishing revolution to occur, the LaserWriter literally saved the Macintosh from oblivion. The LaserWriter also introduced PostScript to the Macintosh graphic artist.

PostScript is a page description language. It is the language the computer uses to tell the printer what to print. The Macintosh uses its own language to draw onscreen—**QuickDraw**. The printers used with the Macintosh before the LaserWriter were thus QuickDraw printers, and the Macintosh used QuickDraw to create the pattern of pixels it sent to the printer. QuickDraw was good for screen printing and low resolution printing, but wasn't as sophisticated as PostScript. One important difference was the basic drawing element of each language. QuickDraw used polygons to create complex shapes. PostScript used **Bézier curves** (a mathematical description of a curve based on two anchor points and two “handles,” which you move to change the shape of the curve). The Bézier curves not only were the basis of the fine resolution possible with the LaserWriter, but they also became the basis of a new generation of drawing programs—the PostScript drawing programs. (See Chapter 6, “Text,” which also discusses PostScript and QuickDraw as they relate to text.)

Not long after the success of the LaserWriter, Adobe, the creators of PostScript, released a program called Illustrator. Illustrator was both similar to and very different from MacDraw. Like MacDraw, Illustrator was an object-oriented program. Anything that was created in Illustrator was treated as an object that retained its identity and could be edited at any time. But whereas MacDraw was based on QuickDraw and used polygons to create shapes, Illustrator used Bézier curves (see figure 7.3).



Included on the disk is a demo version of Illustrator.

Although Illustrator is more complicated to use than MacDraw, it is much easier to create very sophisticated graphics with Illustrator. It has become (along with the competing product, Aldus FreeHand) the standard graphics program for graphic designers.

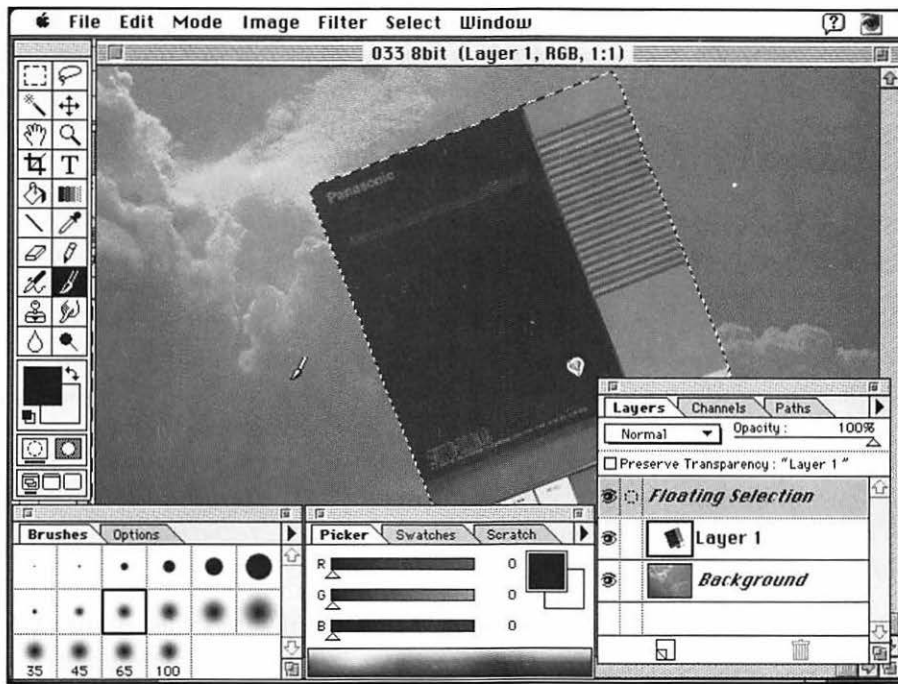


Figure 7.3. Adjusting a Bézier curve in Illustrator.

One particular advantage Illustrator and FreeHand have over MacDraw (and similar programs based on QuickDraw) is that they are more accurate when printing to PostScript printers. QuickDraw code has to be converted to PostScript code in order for a QuickDraw graphic to print on a PostScript printer. This conversion often causes slight problems (for example, lines that don't quite align the way you expect them to align). These problems don't occur with PostScript drawing programs. (Of course, if you are creating a graphic for display onscreen, then this is not a problem.)

Both Illustrator and FreeHand save the graphics they create in **EPS** (Encapsulated PostScript file) format. A PostScript file is simply a file containing the PostScript code needed to print a page or pages. This file can be sent directly to the printer using a utility, and the printer prints a page containing the graphic. Alternatively, the graphic can be imported into a page layout program and then printed as part of the layout. The difference between a PostScript file and an EPS is that the EPS contains a low resolution bitmapped version of the PostScript file (that is, what the graphic looks like). This feature is used in page layout programs to provide previews for positioning the graphic, but they are comparatively low resolution representations of the graphics.

PostScript files are principally useful for printing on PostScript printers. They are virtually useless for printing to non-PostScript printers. Many people like to use Illustrator or FreeHand to create graphics, however, and there's no reason why you shouldn't use these programs to create an illustration that will be seen only onscreen.

In the last few years, a new file format has become popular for saving bitmapped graphics. **JPEG** (Joint Photographic Experts Group) is a file format specifically designed for compressing photographic images. Unlike some compression formats that reduce the size of a file but always recreate the original file exactly, JPEG is "lossy." A **lossy** compression format throws away some information; however, if you use the format correctly, you will not notice that anything has been removed. JPEG provides a sliding scale of compression, making it possible to increase the compression factor (which reduces the quality of the image and the size of the file) or reduce the compression factor (and increase the quality of the image and the file size).

JPEG has its detractors. Some people get upset at the very idea of throwing away any information. But I find it impossible to tell the difference between the original and JPEG-compressed files, at least when using the minimum compression settings (and even at the minimum settings, the space savings are considerable). Describing the JPEG compression process in depth is beyond the scope of this book, but there are other references available for those who are interested (including the *QuickTime Handbook*, published by Hayden Books).

Note

Apple included a JPEG compressor in QuickTime. If your computer can run QuickTime, then you can use QuickTime to compress images in the JPEG format. This is particularly useful if you have to send files over a modem.

Tools of the Trade

Today MacPaint has been replaced by several programs that are designed for manipulating scanned images or creating "painterly" effects. Object-oriented graphics programs have split into two camps. On one side are programs such as ClarisDraw and Canvas, which provide the features

found in the original MacDraw and some additions. On the other side are the PostScript graphics programs Adobe Illustrator and Aldus FreeHand.

Which object-oriented graphics program should you get? If you are doing screen graphics or simple printed graphics, then you can choose any or all of the applications available, since the final output is at comparatively low resolution. If you are a serious desktop publisher or graphic designer who is typesetting your output, then you should consider Illustrator or FreeHand, though you will probably end up with several other applications as well.

The following sections summarize the types of graphics applications available, as well as the different file formats that they use.

Graphics editing programs

For manipulating bitmapped graphics (such as scanned images), the most popular program is Adobe Photoshop (see figure 7.4).

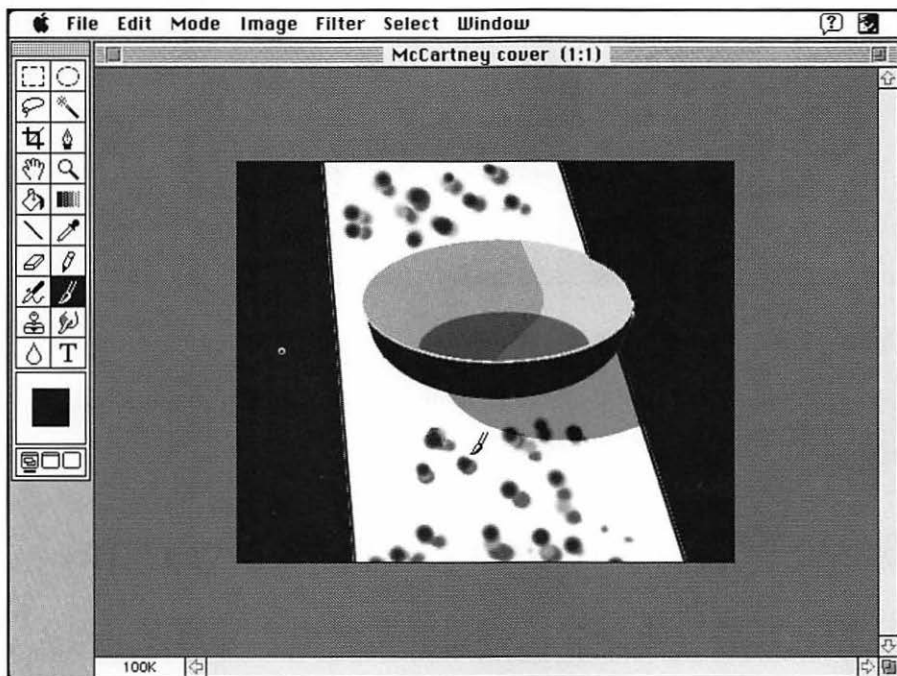


Figure 7.4. Adobe Photoshop.

If you want to create “painterly” illustrations, then you should look at Fractal Design’s Painter (see figure 7.5). This program does a very good job of imitating traditional painting tools such as chalk, charcoal, and different paintbrushes. It’s an amazing program, particularly if you have a pressure-sensitive drawing tablet (discussed later in this chapter). The latest version (3.0) includes tools for recording and repeating sequences of painting operations.

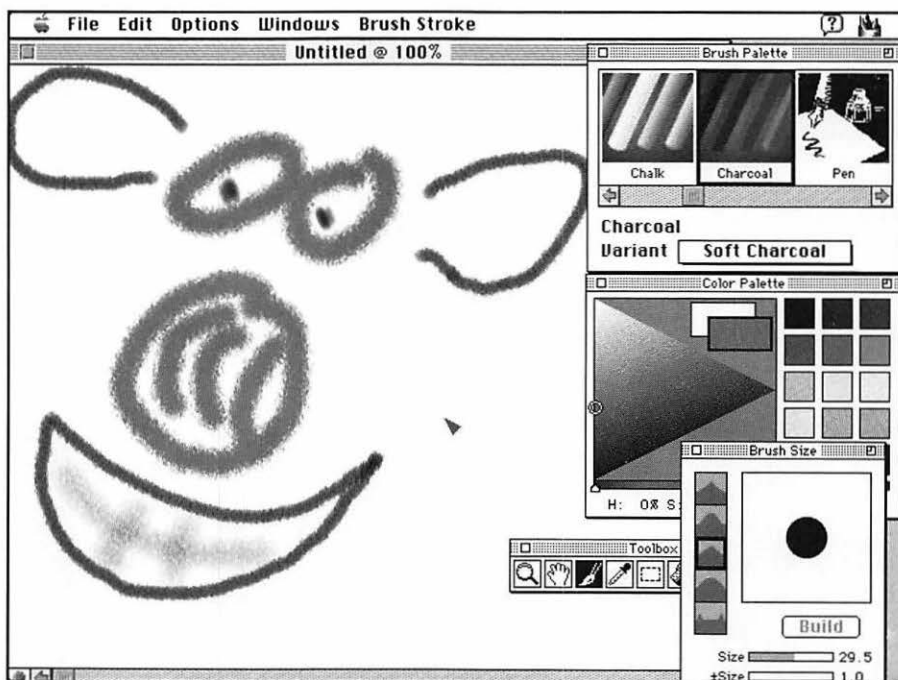


Figure 7.5. Fractal Design’s Painter.



Included on the disk are demo versions of Photoshop and Painter.

Basic object-oriented graphics programs

For object-oriented graphics, two of the most popular programs are Canvas from Deneba and ClarisDraw from Claris. There’s also SuperPaint from Aldus, which supports both a bitmapped and an object layer. This

feature provides a “best of both worlds” combination, except that the individual functions are not as sophisticated as the dedicated applications. If you are working only occasionally in either medium, then SuperPaint may be a good choice.

PostScript graphics programs

The two most popular PostScript drawing programs are Adobe Illustrator and Aldus FreeHand. Both offer a similar selection of tools and functions, and it often comes down to a personal preference choice. I may prefer the way the pen tool “feels” in Illustrator, and you may prefer the way layers containing objects can be built up in FreeHand. Both programs have been updated frequently, and are now very powerful tools. Unfortunately, Aldus and Adobe have recently become one company. Freehand was actually developed by Altsys (the makers of Fontographer) and only marketed by Aldus, so at this writing it is unclear what the future of FreeHand will be.

Graphs or graphics?

There are some other programs that you may use to create graphics which aren’t really graphics programs at all. In particular, spreadsheet programs like Microsoft Excel provide charting functions for creating very sophisticated charts. The latest version creates all kinds of 2D and 3D charts. The only problem I find with Excel is that you have to be pretty sophisticated yourself to figure out how to use it! However, a program like Excel can be very useful if you need to create charts and graphs, because even if the chart it produces isn’t exactly right, you can use the basic charts Excel creates as the basis to create and manipulate the charts in another program.

File formats and applications that work with them

There are a lot of graphic file formats out there. Fortunately, though, most programs support several formats. You shouldn’t find exchanging graphics with another person too difficult—even if that person has a graphics program that is different from yours.

MacPaint

The MacPaint format is still widely used, although no program today supports only the MacPaint format. Many bitmap-editing applications, such as Photoshop and Studio/32, read MacPaint format, as do all desktop publishing programs (though none of the latter group will let you edit MacPaint graphics).

PICT

PICT is widely used for bitmapped and object graphics. It no longer has the limitations in color, resolution, and size that the original format had.

TIFF

The TIFF format was originally created for scanned images. The most common programs for manipulating TIFF images are Photoshop and ColorStudio.

Tip

Should you use the TIFF format? Although TIFF is still widely used, it has some disadvantages. The greatest disadvantage is that TIFF does not compress files in any way, resulting in files much larger than those saved in PICT format. (A compression option that considerably reduces the size of a file has been added to the TIFF standard, but the option is rather slow and increases the time it takes to open and close the files.)

If you are working in desktop publishing, then you may want to use TIFF. In the past, I have found that some desktop publishing programs (such as Aldus PageMaker) handled the TIFF format much better than they handled PICT; however, this is not so much the case today. If you aren't involved in desktop publishing, then it's probably better to use PICT. Most of the advantages of TIFF have disappeared as a result of improvements in the PICT format.

RIFF

This is a compressed bitmapped format first used by the grayscale editing program ImageStudio; RIFF is also used in ColorStudio. The RIFF format is not widely used, although several programs support it.

EPS

The EPS format is supported by Illustrator and FreeHand. You also can save bitmapped graphics in EPS format (but see the tip later in this chapter).

Remember that EPS is designed for sending files to a printer, not for exchanging files for editing. It is generally not possible to use Illustrator to edit drawings created in FreeHand and vice versa.

Tip

When should you use EPS? Although the above discussion may imply that EPS files are only for object-oriented graphics, you can save bitmapped graphics in EPS format. Saving bitmapped graphics in EPS format creates large files that are difficult to work with, however, so you should use the EPS format for bitmapped graphics only when you have a specific purpose in mind—for example, for printing on a PostScript printer (and even then, using the EPS format is not always necessary). Saving a bitmapped illustration in EPS format will *not* enable you to edit the file in Illustrator or FreeHand.

Targa

The Targa bitmapped format comes from the PC platform, and though several programs support this format (including Photoshop), Targa is really just for those moving files from the Mac to the PC and back again.

GIF

GIF (Graphics Interchange Format) is a compressed file format for bitmapped images. CompuServe developed it for distributing graphics over modems. It has become fairly popular, and several programs support this format, although it does not support bit depths greater than 8 bits.

BMP

The BMP format is a Windows format for graphics. It is included here only because of the increased interest in cross-platform development. Several Macintosh programs (including Photoshop and Painter) can read and save in BMP format.

JPEG

The JPEG format was developed by an independent group. Although it is now “standard,” in the early days (only a couple of years ago!) a program that read JPEG files could not always read the JPEG files created by another program.

When JPEG compresses photographic images, it throws away some information. This loss, however, is usually not noticeable. Because JPEG enables you to adjust the compression factor, you can control the loss to some extent. If you compress the file to the maximum amount, you reduce the quality of the image more, but if you reduce the compression factor, you retain more of the image quality.

The JPEG compressor that Apple includes with QuickTime is available to anyone who can run the QuickTime extension. Included in the QuickTime Starter Kit (a collection of utilities sold with QuickTime by Apple) is a utility called Picture Compressor, which can be used to compress images.

Photoshop offers its own options for JPEG compressing an image. These are available by either choosing JPEG as the file type when saving the file, or when saving in the PICT format (however, JPEG is only offered as an option for RGB (32-bit) images; 8 bit images cannot and should not be saved with JPEG).



Included on the disk is a copy of QuickTime 1.6

As shown in figure 7.6, the first step is to open an image in Picture Compressor. The next step, shown in figure 7.7, is to choose the compressor (Photo-JPEG) and the compression quality. Selecting Compress from the Compression menu starts compressing the image, and a new window, shown in figure 7.8, displays the resulting image.

As you can see in this example, a 1M PICT file was reduced to 100K by using the Normal compression quality. Choosing Most would have increased the savings even more (though at the expense of image quality). After an image is compressed, it is saved on a disk. Other applications recognize the new file as a PICT file, so any existing application that can open a PICT file can open this compressed file. (When the application asks the System to open the file, the System recognizes that the file is compressed and decompresses the file before providing the data to the application.)

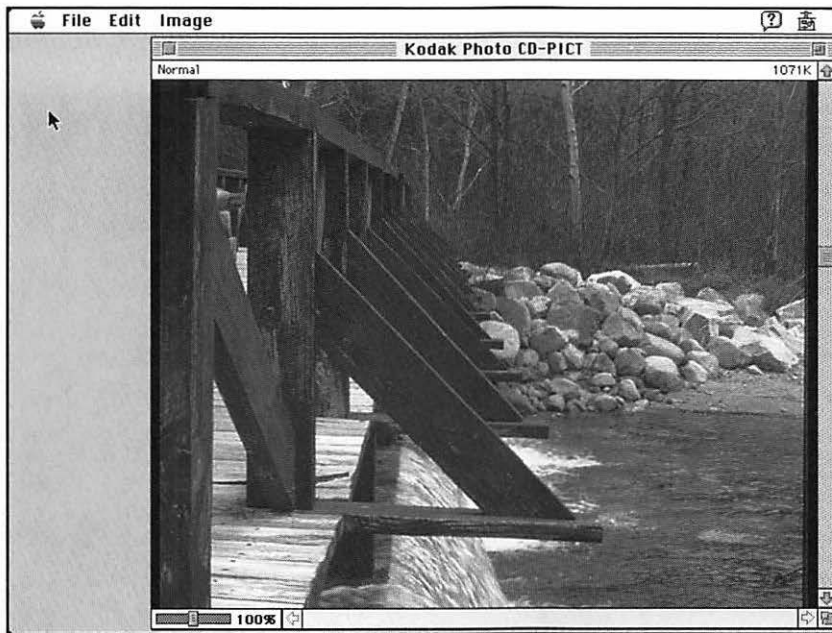


Figure 7.6. Compressing using Picture Compressor.

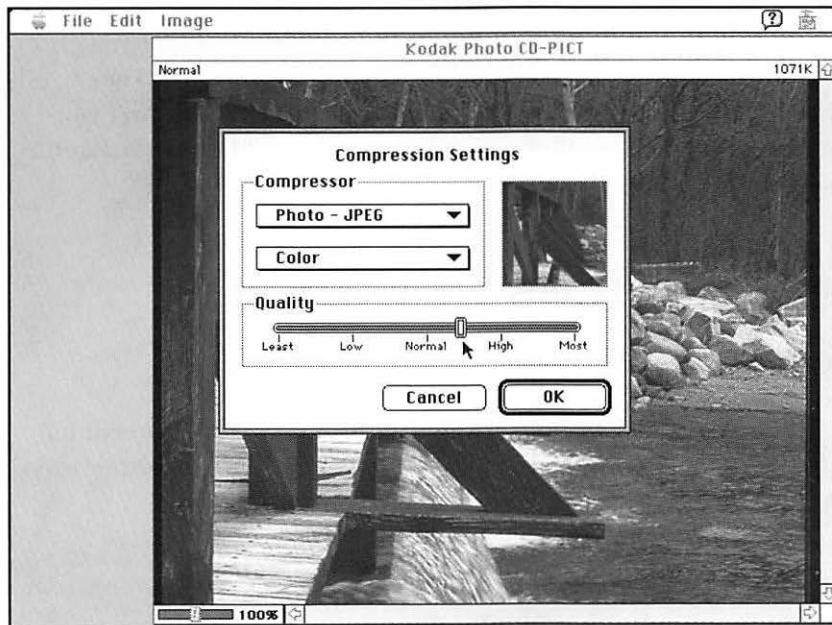


Figure 7.7. The Compressor and Quality options.

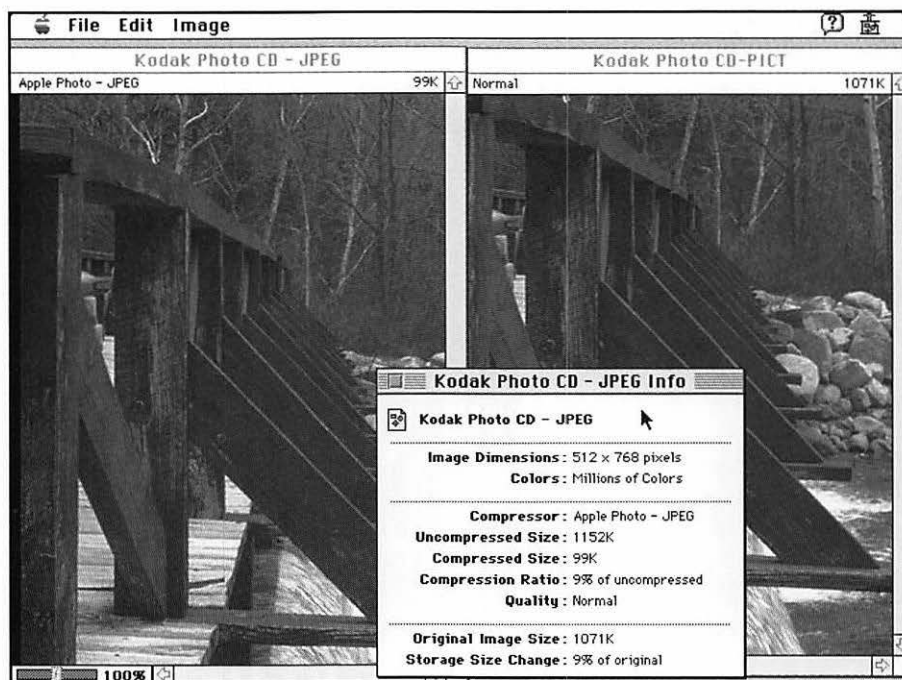


Figure 7.8. The resulting image.

Existing applications can read QuickTime-compressed images (though they will not be able to create them), but you must have QuickTime to read these files. Also, the files are not in a strict JPEG format, so if you want to send the file to another computer system, additional translation work is required.

Other Sources of Graphic Images

So far, this chapter has focused on applications for creating and editing bitmapped and object-oriented graphics. There are, however, other ways of creating graphic images for your multimedia projects.

Video images

If you have (or are considering obtaining) a video camera for recording source material for QuickTime movies, you might consider using that camera as a source of still images. All that is required is a video digitizer, and you probably have (or are considering purchasing) a digitizer for capturing the video anyway.

Be aware that some video digitizers are better than others. For example, the VideoSpigot from SuperMac is great for capturing small images (up to 320 by 240 pixels at 72 dpi resolution), but cannot be used for digitizing larger frames unless the source image is still. This is because capturing a large image takes several frames to complete. The digitizers from RasterOps (the 24STV, 24 MXTV, and so on), as well as Radius' VideoVision, are all capable of digitizing a large single frame. RasterOps includes a plug-in extension for Photoshop that can be used to capture images and import them straight into Photoshop for editing (see figure 7.9).

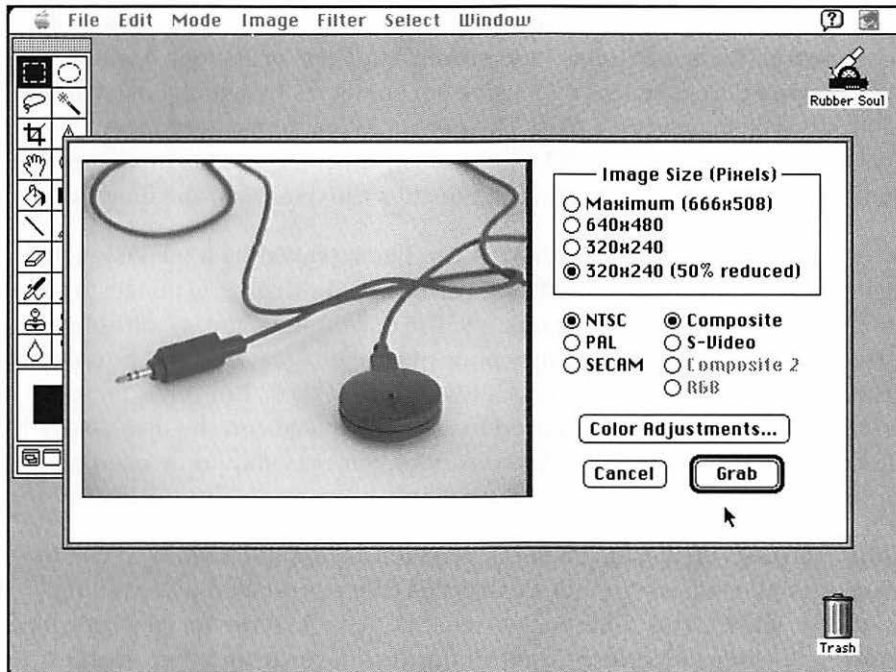


Figure 7.9. Grabbing an image with a RasterOps digitizer using a Photoshop plug-in.

If your digitizer can't digitize a single frame, you still may get good results if your video source has a clean pause (that is, the video image remains stable when the source machine is placed in pause mode).

Although a video camera can be a quick and convenient tool for grabbing still images, it has certain drawbacks. For example, nearly all cameras require a second or so to start and stop recording, which makes quick "grab" shots almost impossible. When digitizing an image from videotape, you need several seconds of the subject to cue yourself to click the "Grab" button. For this reason, you may want to use other sources for digitizing a large number of images.

Still video images

Still video cameras have been available for several years. They use a CCD (charged couple device) to capture an image, much the same way camcorders capture a frame of video. The images are stored on a small magnetic disk, however, rather than on videotape.

Two versions of the cameras are widely available. The professional units cost several thousand dollars and store 25 images on a single 2.5-inch video floppy disk. The less expensive consumer units, which cost around \$500, store 50 images on a disk. The consumer cameras store more images by halving the vertical resolution of the images. This increases the number of pictures per disk, but diminishes the quality of the images.

After an image has been captured, it can be displayed on a television, or digitized by using a video digitizing card (just like digitizing images from a video camera or tape). Unfortunately, the consumer cameras, although attractively priced, produce very poor quality images. This poor quality is primarily because of the low resolution of the images, but other factors such as the encoding system used to save the images on the disk also affect the quality. Generally, the consumer cameras should be used only for images of 320 by 240 pixels or smaller.

Although the professional cameras produce acceptable results at 640 by 480 pixels, the images are not as sharp as those produced by scanning photos or slides. This difference is most noticeable if the images captured with a still video camera are used alongside images from other sources like a slide or flatbed scanner.



Figure 7.10. The Sony Mavica still video camera.

Some professional still video cameras do not store their images on the 2.5-inch floppy disks. Rather than convert the image to the analog signal used on the 2.5-inch disks, these cameras store the data digitally, which improves the quality of the image. These cameras also offer improved resolution thanks to higher-resolution CCDs. Most notable among these are Kodak's DCC-200 series cameras. These cameras are based on the Nikon 8008S camera body, and include a special back that contains the CCD and a hard disk for storing images. After images are captured, the camera is attached to a computer via a SCSI cable, and the images are downloaded to the hard disk.

The DCC-200 series cameras produce high quality results, but they are extremely expensive (around \$10,000). If you are taking more than five images per day, the camera will pay for itself, but if you do not generate that number of images, you will probably be better served by other methods.

QuickTake 100

Apple's QuickTake 100 is a digital camera that was developed with the assistance of Kodak. This camera is inexpensive (about \$600) and is purely digital. (Unlike cameras like the Xapshot and Mavica which store the image on an analog disk, the QuickTake stores the images in memory.) A special compression algorithm is used to store the images; unfortunately, the effects of either this, or the low resolution of the camera itself, can often be seen in the images. They are marginally better in quality than the Xapshot, but the 640 by 480 images are nowhere near as clean as the images you can get from a Photo CD or from a scanner. Also, the camera can only take 8 large (640 by 480) images, or 32 small (320 by 240) images, and there's no removable media for storing images—you have to download them to your computer.

Apple has provided some very good software for working with the QuickTake. You plug the QuickTake into a cable that you connect to the Macintosh serial port. You can then run the QuickTake software, view thumbnails of the software, and download the images to the computer disk drive. Or you can mount the camera on the Macintosh desktop and treat it as though it were an external disk drive.

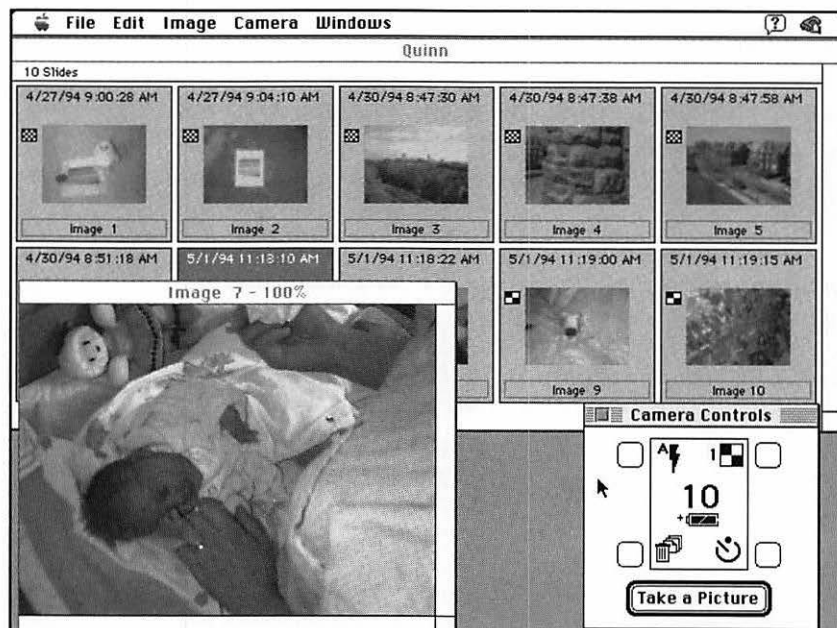


Figure 7.11. QuickTake 100 software.



Sample images taken with the QuickTake camera are included on the CD-ROM.

Scanned images

Flatbed scanners and slide scanners are the most common choices for digitizing still images. The prices of both pieces of equipment have come down considerably in the past five years. Devices that used to cost around \$10,000 are now available for less than \$2,000.

It is possible to get a single pass, 24-bit color flatbed scanner for less than \$1,000. This is amazing. In selecting one of these scanners, the decision often comes down to features and software. Several scanners include a version of Photoshop with a plug-in for Photoshop so that you can drive the scanner from within the Photoshop application. This feature makes scanning and manipulating the images very convenient. Other scanners, such as the Hewlett-Packard Scanjet IIc, have their own applications. A good third-party application for driving scanners is Ofoto from Light Source, Inc. (see figure 7.12). This program not only drives the scanner, but also provides several controls for adjusting images and functions for calibrating the scanner. If you scan a lot of color images, you may want to consider Ofoto.

Some professional still video cameras do not store their images on the 2.5-inch floppy disks. Rather than convert the image to the analog signal used on the 2.5-inch disks, these cameras store the data digitally, which improves the quality of the image. These cameras also offer improved resolution thanks to higher-resolution CCDs. Most notable among these are Kodak's DCC and DCS series cameras. These cameras are based on the Nikon 8008S and N90 camera body, and include a special back that contains the CCD and a hard disk for storing images. After images are captured, the camera is attached to a computer via a SCSI cable, and the images are downloaded to the hard disk.

These cameras produce high quality results, but they are extremely expensive—while some models are available for around \$10,000, the DCS 460, which has a resolution of 3060 by 2036 costs almost \$30,000.

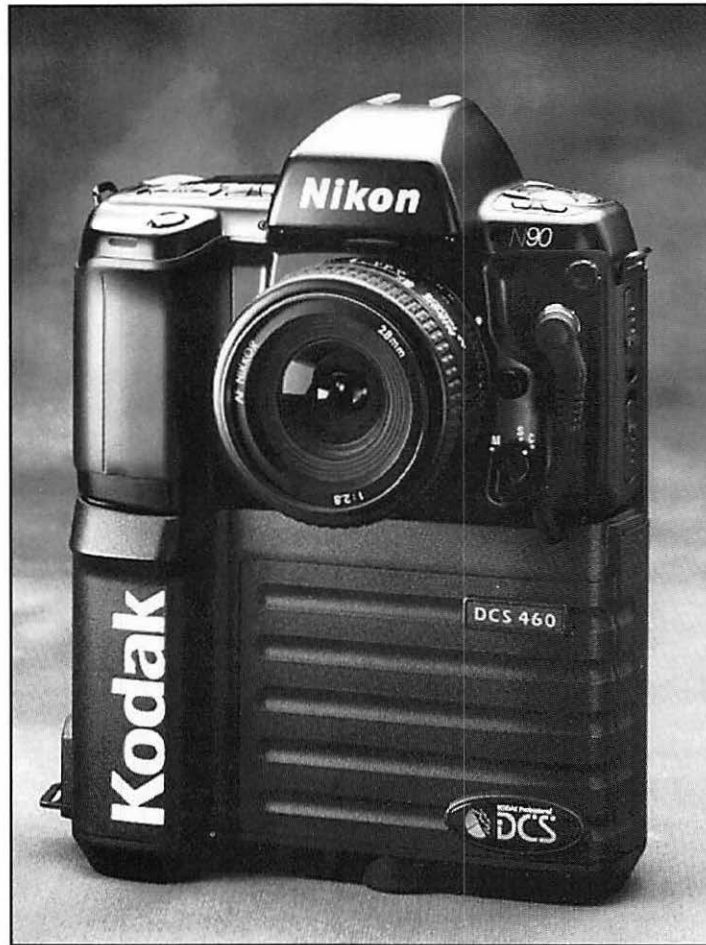


Figure 7.12. The Kodak DCS 460 digital camera.

If you are considering purchasing a slide or flatbed scanner, you should first look at Kodak's Photo CD system (discussed in the following section). The Photo CD system may be more cost-effective than a scanner if you do only a few scans per week, and it probably will provide images of much higher resolution than a scanner (although the resolution is not as critical for most multimedia developers because they are generally working only with screen resolution images of 640 by 480 pixels).

Kodak Photo CD images

Last year Kodak unveiled the Photo CD system, with the primary intention of selling it in the consumer market. Photo CD also has received a lot of attention from desktop publishers, graphic artists, and multimedia producers.

What is Photo CD?

Back when still video cameras first appeared, Kodak began to worry that this type of camera would spell the death of film. So, the company set about creating a system that provided the features of a still video camera (your pictures on TV) while preserving film.

How does Photo CD work?

You take a roll of film and expose it in the traditional way—using your camera. Then you take the exposed film to a photo processor and ask the processor to develop the film and provide a Photo CD. The photo lab develops the film, and then digitizes the images and stores them on a CD, much like an audio CD or CD-ROM (except that the information is in a different format). Up to 100 high-resolution images can be stored on one Photo CD. The photo processor also can print traditional photographs from the film negatives.

When you come back to pick up your photos, you receive the Photo CD and film negatives. You take the Photo CD home, insert it into a special CD player (which also plays regular audio CDs) and watch the images on your television. It's simply marvelous, and it's simply hard to believe that Kodak thought a lot of people would want to look at their photographs on television.

Still, the Photo CD contains all these images digitized at 3072 by 2048 pixels, and Kodak has released programs for reading Photo CDs on CD-ROM drives attached to computers. This is where Photo CD has found its biggest market. For between \$1 and \$3 per image (depending upon where you go), you can obtain images scanned at high resolution and stored on an easy-to-handle medium—a CD.

Apple has even licensed the Kodak decompression routine and added it to QuickTime. When you insert a Photo CD into a computer running QuickTime and the Apple Photo Extension, the images on the disk appear as thumbnails (small graphic representations of the images); they can be opened and viewed using any application that reads PICT files. Figure 7.13 shows a Photo CD displayed in the Finder. The bottom left window is a QuickTime movie that shows all the images on the disk.

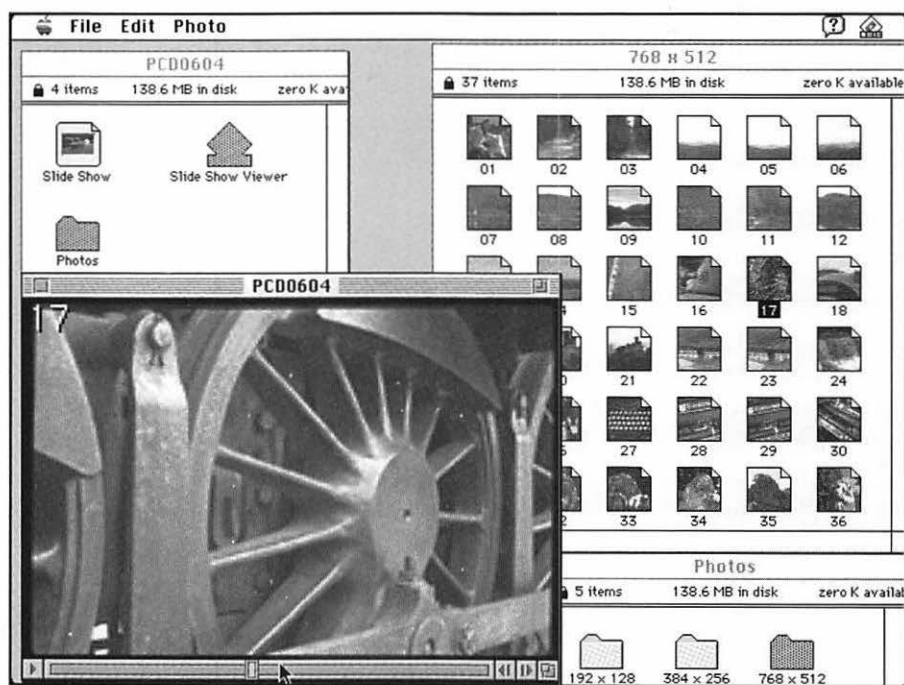


Figure 7.13. A Kodak Photo CD.

How do single-session and multi-session drives differ?

If you already own a CD-ROM and want to use Photo CDs with it, then you need to know about single-session versus multi-session discs. The Photo CD format supports multiple sessions. You can add one roll of film to a Photo CD, and then come back two weeks later and add a second roll of film to the same Photo CD. You can continue to do this until the disc is full. When the first roll of film is put on the disc, that Photo CD is a single-session disc. If you add another roll of film at a later date, then that Photo CD becomes a multi-session disc.

Nearly all CD-ROM drives sold before 1992, and some still being sold, are only “single-session compatible.” If you insert a multi-session Photo CD into the CD-ROM drive, the drive will read the images from the first session, but it won’t read the additional images.

This limitation is not a major problem as long as you don’t create a multi-session Photo CD when you have only a single-session-compatible CD-ROM drive. That’s why it is important to check your drive. Of Apple’s CD-ROM drives, only the 300 series drives support multi-session disks. The 150, SC, and SC Plus drives do not support multi-session Photo CDs. To find out about other manufacturers, refer to the instruction manual or contact the manufacturer.

Clipped images

For those who don’t have the time or talent to create their own graphics, a wide selection of clip art is available. Refer to Chapter 12, “Clip Media and Copyright,” for more information on this source of graphics.

Unusual Graphic Tools

There are several graphic programs that don’t quite fit the categories already explored. Here are a few of the most unusual (and useful).

TextureScape from Specular (makers of the 3D program Infini-D) is a tool for creating textured images that are useful as backgrounds in your presentations (see figure 7.14). TextureScape enables you to choose a basic shape and then specify coloring, how often it is repeated, and the lighting applied to the object. By building up layers of these objects, very complex patterns can be created. You can even create the basic shapes using Illustrator and import them into your texture files.

TextureScape actually uses a 3D rendering engine to create the image. Each of the objects in a texture can have a bevel applied to it; that’s how lighting effects are calculated. Included with TextureScape is a library of basic surfaces which you can modify.

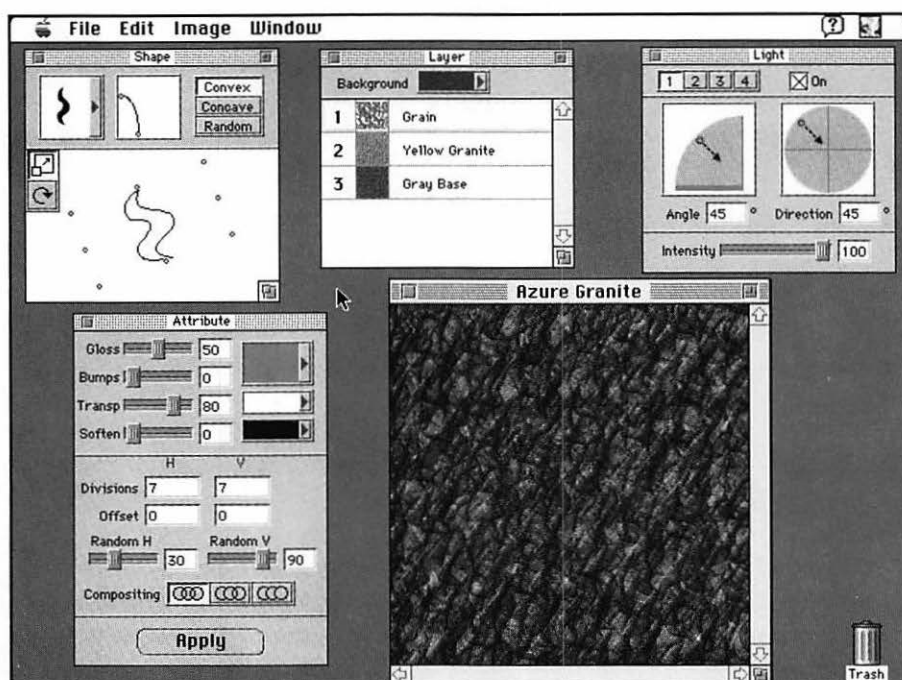


Figure 7.14. Defining an image in TextureScape.

Collage, also from Specular, is—as its name suggests—a tool for creating collages of multiple images (see figure 7.15). You can import several images and rotate and matte the images, as well as add text and drop shadows. While you can do all this in Photoshop, Collage has two advantages. When working in Collage, you aren't working with the whole image, instead it uses proxies—low resolution representations of the actual images. This makes it quick and easy to manipulate very large files. The other advantage is that it is very easy to change things because each object retains its identity; you can click and move it at any time.

Certainly, Collage doesn't replace a program like Photoshop or Painter, but it is a useful tool if you like to create collage types of illustrations.

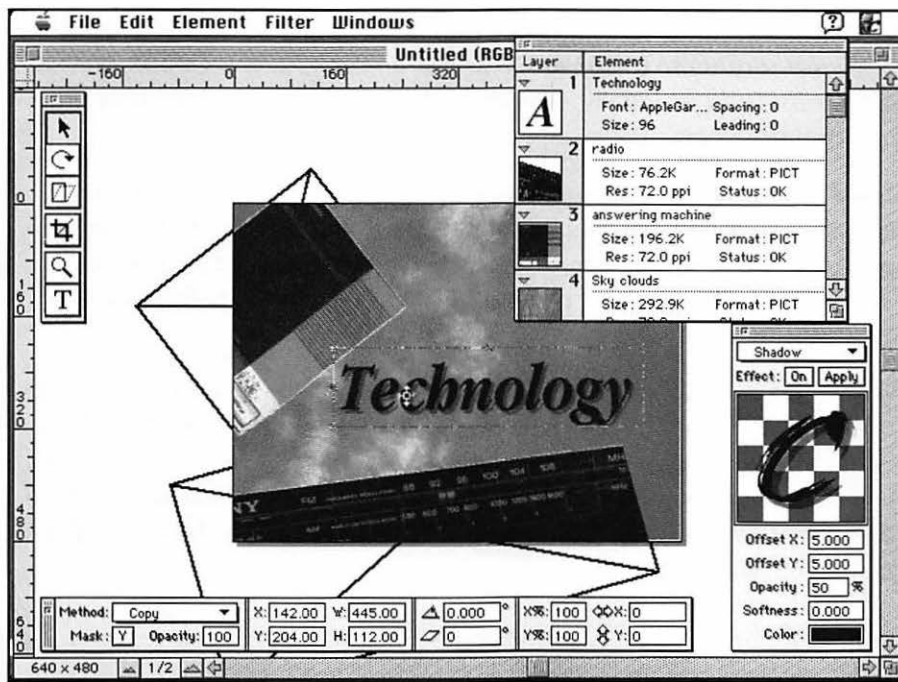


Figure 7.15. Collage.

Jag II, from Ray Dream (makers of the 3D program Ray Dream designer), is a specialized processing tool. Often bitmapped images suffer from jaggies—the stair-step effect seen where one solid color borders another. This effect can be minimized by using anti-aliasing; inserting pixels of a transition color in the stair steps. When viewed from a distance the stair-stepping is markedly reduced (see figure 7.16).

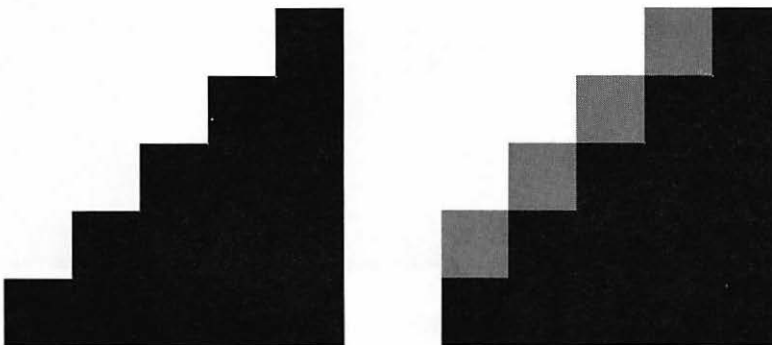


Figure 7.16. Anti-aliasing to reduce the stair-step effect visible in bitmapped graphics.

Jag II is a utility that takes an image and automatically applies an anti-aliasing algorithm to the image. This can be useful as a post-processing effect for images which you would like to smooth, particularly those exported from object drawing programs such as ClarisDraw.

DeBabelizer is an automated image processing tool. It is particularly useful if you have a large number of images that you want to translate from one format to another, or perform some other function such as convert to one customized palette. DeBabelizer, from Equilibrium, is scriptable and supports Photoshop filter plug-ins.

Need an out-of-this-world image? **KPT Bryce** is a landscape and scenery generator which can produce just that (see figure 7.17). Based on 3D modeling, Bryce enables you to choose a terrain, adapt it to your heart's content, choose coloring, and add the sky and atmospheric effects. It's a great way to spend a few hours, and who knows? You might come up with something useful!

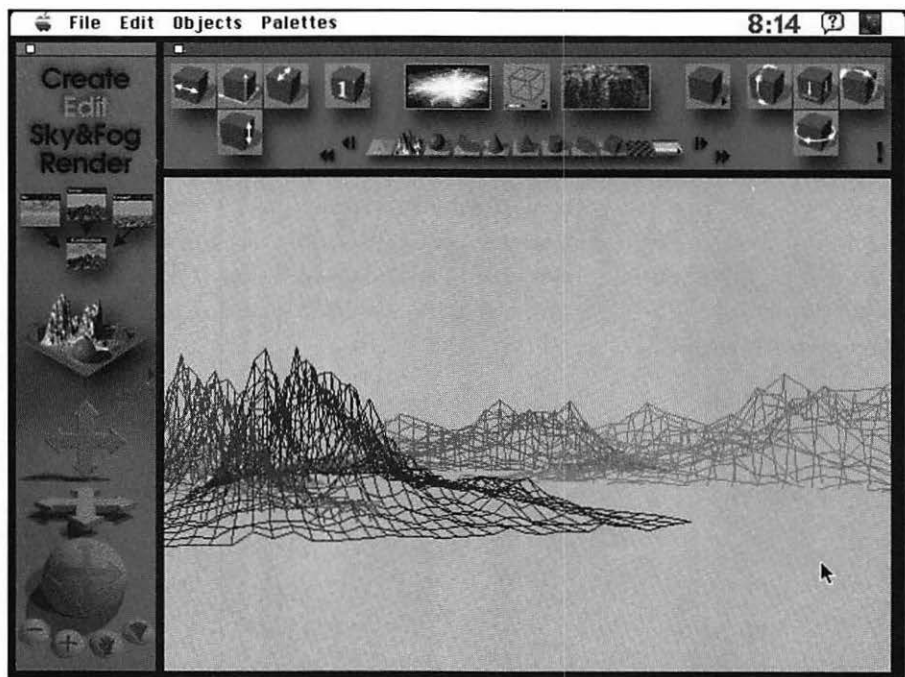


Figure 7.17. Creating your own world in KPT Bryce.

Pressure-sensitive drawing tablets

Pressure-sensitive drawing tablets don't really fit in this section, but I had to squeeze them in somewhere! If you have a talent for drawing—or even if you just like doodling with a paint program—then you should definitely consider purchasing a pressure-sensitive drawing tablet. Several programs, including Fractal Design Painter, support these tablets. When you draw with a pressure-sensitive tablet, you can adjust the width or the flow rate (or some other characteristic of the line you are drawing) by varying the pressure with which you draw. The pressure-sensitive tablet makes it possible to imitate traditional drawing media much more closely than is possible by using just a mouse.

I highly recommend using a pressure-sensitive tablet. Be warned, though—it helps to have a fast computer to work with these tablets and the paint programs that support them. The computer must perform extra processing to create the complex lines that you draw.

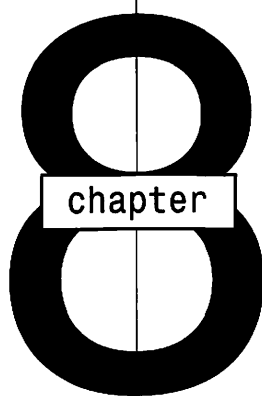
The Future

In the future it will become easier to sample and manipulate the world in which we live.

As computers become more powerful, video and still images will become harder to distinguish. A need for still images will remain, but motion will tend to swamp other forms of media. Digital effects will become more sophisticated, but designers will continue to search for that new look. Nothing gets easier; it just changes.

Now that you have mastered the world of single images, the next chapter explores a world filled with hundreds and thousands of them—animation.

Animation



Until the arrival of digital video and QuickTime (see Chapter 10, “QuickTime”), animation programs were the only method for creating motion on the Macintosh. For many purposes, animation programs remain the best way to create motion. From animated diagrams to cartoon movies, animation tools are an important part of the multimedia producer’s arsenal.

Today's Macintosh users can choose between any one of several animation tools, but this was not always the case. For the first few years of the Macintosh, the only available animation program was a product called VideoWorks, which eventually became Macromedia Director (see figure 8.1). Now there are several different programs available, from the low-end animation packages such as PROMotion and Animation Works, to the animated presentation programs Cinemation and Motion Works. Director remains the preeminent animation package on the Macintosh—and is described in much greater depth in Chapter 14, “Interactive Presentations”—but it now has some stiff competition.

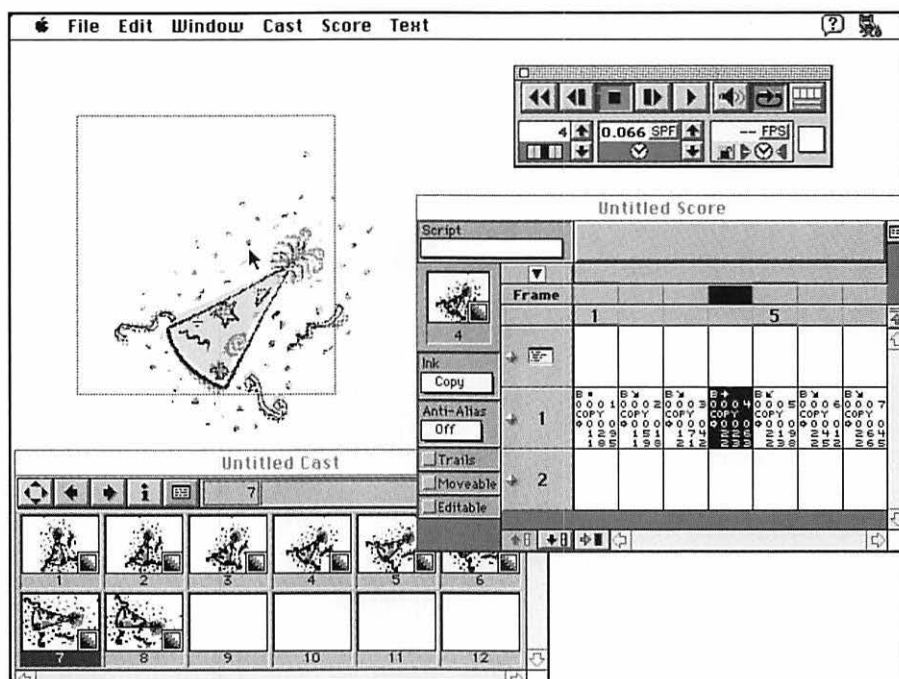


Figure 8.1. Creating animation in Director.

This chapter is devoted to an overview of the techniques and problems you may encounter when working with animation; it also provides an overview of the many animation packages available for today's Macintosh users.

Working with Animation

Working with animation on the Macintosh (or on any computer, for that matter) involves becoming familiar with the techniques and problems of animation. When you understand these techniques you will be better prepared to solve the problems presented by the animation process. The information in this section outlines some animation techniques and some of the problems you must solve in order to make full use of your animation application. This section also discusses formats in which you can store animation files.

A few principles of animation

Two divergent schools dominate computer-based animation. Members of the first school are those who use the 3D modeling and animation tools to create 3D animation (see Chapter 9, "3D," for more information on 3D modeling). Most animators who use those programs strive to create the most realistic animation possible. To achieve that realism, the motion of animated objects must closely model the movements of real objects.

The second school—the traditional animation school—is rooted in the long history of cartoon animation. It is this second group of animators who will be most interested in the programs discussed in this chapter. As the sophistication of the 3D programs increases, these two schools will diverge even more, because the ultimate goal of 3D modeling is to recreate real life using a computer. If you are interested in 3D animation you should skip to the 3D chapter.

Several principles govern cartoon animation, most of which center around the techniques that use exaggeration. Movement, actions, and reactions in cartoons are always exaggerated because cartoons themselves are exaggerations.

Anticipation is a specific application of exaggeration. Before moving in one direction, a cartoon character draws back slightly in the opposite direction. This backward movement makes the character look as though he is gathering himself together before he starts off, and results in a much more realistic movement than if the character simply started forward.

Squash and Stretch are used to show the effects of acceleration on an object. When a ball hits the ground it squashes down; it then stretches out as it rebounds. Objects do this in real life, though we rarely notice it. Figure 8.2 shows a party hat demonstrating the effects of Squash and Stretch.

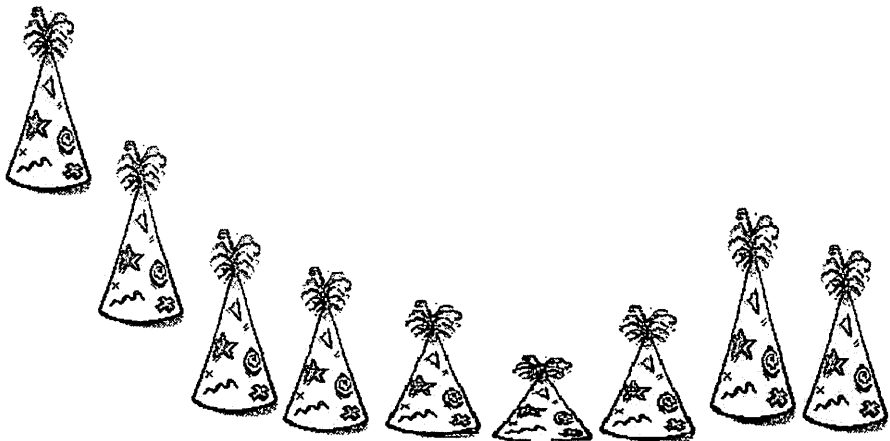


Figure 8.2. An object squashing and stretching.

A number of books are available for those interested in learning classic animation. One book I like is *The Animators Workbook*, by Tony White. This book's tutorial format leads you through animation principles and shows several examples. There also are a number of books on Disney and Warner Brothers cartoons which would serve as good introductions to cartoon animation.

As a multimedia producer, you may use animation programs frequently, and yet never use any of the classic animation techniques. Often, animation in multimedia presentations involves very simple actions such as animating a cursor in a program demonstration, or animating bullet points onto the screen during a presentation.

However, even in those cases you may learn things from the classic animation styles. For example, rather than have a bullet point simply slide off the screen, you could use the anticipation technique of moving the item fractionally in the opposite direction before the object starts sliding off the screen.

You also should study the motion of objects in the real world to try and recreate fluid motion in your animations. Many animation programs enable you to determine the starting and ending points of an object; the animation program will then calculate the in-between steps to create the movement of the object across the screen. There are two problems with this. First, the movement is in a straight line from point A to point B. Second, the object will instantly begin moving and will instantly stop. There will be no apparent acceleration of the object. This makes the movement of the object appear very unrealistic.

Instead, you should try to vary the path and speed of the object. Some animation programs enable you to do this. Director provides a special command for doing this, but you have to explicitly use these functions. Usually the default is to create the simple movement.

Computer-based animation can result in beautiful—and quite complex—productions. As with any computer tool, the more capabilities a program has, the more you need to learn in order to use the program productively. The following section discusses some problems unique to computer-based animation and offers some solutions to those problems.

A few problems of animation

Synchronizing sound with animation is one of the most difficult tasks facing any animator. Even if you create animation in the traditional method and record the results to film, you must spend a lot of time and effort in order to synchronize the sound with the animation.

You may think that this task is easier on the computer, but it isn't. Unfortunately, the computer often exacerbates the problem rather than solves it. The problem with sound synchronization on the computer is that the computer is sometimes unable to play the frames of the animation at the correct speed to maintain synchronization with the sounds. This is

particularly true when animations created on faster machines are played on slower machines. You can also create an animation that is too complex for your hardware to play back at the frame rate you desire. For example, you can create an animation in Director that you want to play at 30 frames per second, but if the animation involves a lot of objects moving around the screen, then the animation may only play at 10 or 15 frames per second.

If you try to synch a sound track to the animation that you created at 30 frames per second, you will have a problem because the animation will take twice as long to play as it should. You then either have to increase the length of the sound track or remove frames from the animation in order to synchronize the animation to the sound track.

You need not record the sound track as one long piece. Instead, you can cue individual sound bites to specific events in the animation. This approach reduces some of the synching problems, but not all of them. It's not feasible, for example, to cut what a character speaks into individual words and then cue the words to the opening and closing of the character's mouth!

A second problem in computer-based animation is memory management. Before most animation programs play an animation, they load all the elements used in creating the animation into memory. With all the elements loaded in advance, the program has immediate access to all the animation's graphics and can quickly display them as they occur during playback of the animation. The problem with this arrangement is that your animation must fit within your computer's available memory; further, if you link two animation files together, the program will spend several seconds loading the second animation before it begins playing.

Apple's QuickTime software has effectively eliminated the animator's synching and memory problems. QuickTime (described in Chapter 10, "QuickTime") provides a mechanism for maintaining synchronization between animation and sound, no matter what computer is playing the animation. QuickTime doesn't suffer from memory problems because it can read from disk while playing animation onscreen.

QuickTime has its limitations, however. Currently, QuickTime does not fully support interactivity (while several animation programs do). Further, because QuickTime saves each frame individually, its files can be much

larger than animation program files (which save only individual animation elements). And finally, slower computers may not be able to play back QuickTime animations that fill the entire screen.

File formats

Prior to QuickTime's release, only two formats for moving animation information from one program to another were in common use: a folder of PICT images and the PICS format. When saving an animation as a folder of PICTs, each frame of the animation is saved with the same name plus a unique sequence number at the end of the filename (for example, TEST 0001, TEST 0002, TEST 0003, and so on).

The PICS format was an attempt by several developers to manage the problems inherent in shifting animation around as separate images. Apple didn't develop the PICS format, but PICS became a de facto standard for the Macintosh because it provided an easy method of linking several PICT files into a single file. Nearly all programs that create animation can save the animation in PICS format.

The PICS format includes an optional method for reducing the size of the files. This option enables the program that creates the PICS file to save only the changes that occur from one frame to the next (see figure 8.3). This compression can considerably reduce the size of the file. Any program that reads PICS files should be capable of recreating the original frames.

QuickTime may soon replace the PICS format as the standard for exchanging animation information. Many programs, however, include both PICS and QuickTime support. This approach makes sense because many users still can't run QuickTime.

The QuickTime Animation compressor is provided specifically for compressing animations. The compressor uses **run-length encoding** to compress animation. Run-length encoding essentially looks at an image and stores the number of duplicate pixels as a count (for example, 20 pixels with the same color begin in row 5 at pixel 33), rather than as separate values. This form of compression works well with animation, which is frequently made up of large areas of the same color.

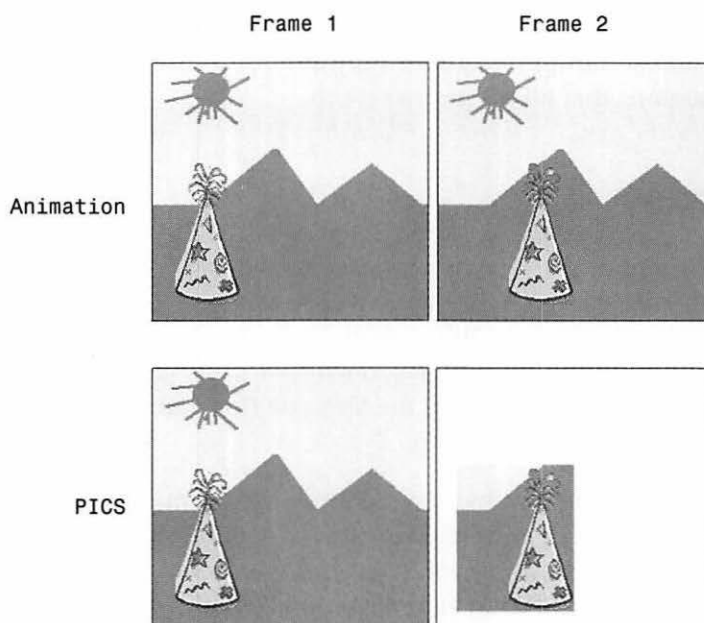


Figure 8.3. Reducing the size of PICS files by saving only the frame difference.

Some Animation Tools

In the last couple of years, several new applications for creating animation have appeared on the market. These applications offer different tools and methods for creating animation, and several are a lot cheaper than Director. Although Director still may be the best choice for some people who want to create animation on the Macintosh, it is no longer the only choice. This section provides a glimpse at some of the animation applications currently available for Macintosh users.

Life Forms

Macromedia's Life Forms is the most unusual of the tools listed here because it is designed for animating one thing only—the human body. Life Forms provides a wireframe model of the human body (see figure 8.4). You animate this form on the screen to create a sequence of movements. To create the sequence, you set the form in one position, advance the animation frame counter several frames forward, and then adjust the

positions of the form by clicking and dragging. You don't have to set the position in every frame; Life Forms creates the in-between frames to complete the motion. This technique is called **key frame** animation (see Chapter 9, "3D," for more information about key frame animation).

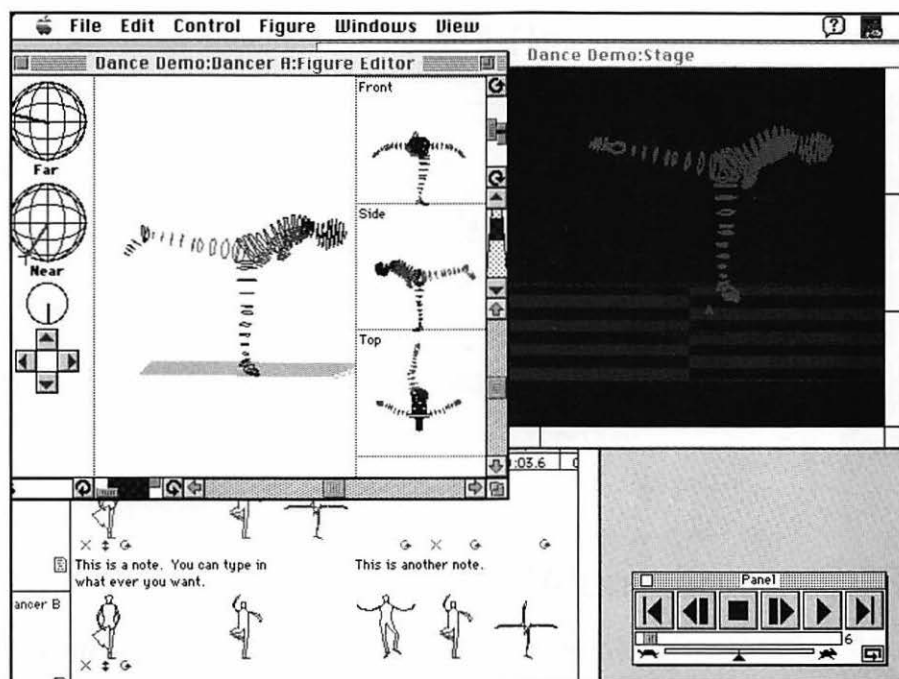


Figure 8.4. Life Forms.

Because Life Forms animates the human body only, the parameters of movement for each part of the body are built into the model—you can't move the body into impossible positions. Although some may consider these built-in parameters a limitation, the parameters make creating the animation sequences much easier!

When you complete a Life Forms animation, you can export it to a 3D program for rendering. Programs supported by Life Forms include Swivel 3D, Electric Image, and Macromedia Three-D. You also can save the animation as a PICS file for importing into other animation programs. If you use this option, you can add other elements or redraw Life Forms' simple outline output. For example, the juggler shown in figure 8.5 was created in

Life Forms, the globes were added in Director, and the completed animation was then exported to QuickTime. Life Forms also includes an XCMD for playing back Life Forms animations within HyperCard, Director, or any other program that supports XCMDs.

The QuickTime movie Juggler (described in the text above) was created using Life Forms and is on the CD-ROM.



Figure 8.5. Juggler, a QuickTime movie created in Life Forms, with globes added in Director.

PROMotion and ADDmotion

Developed by Motion Works, PROMotion and ADDmotion are essentially the same program—PROMotion is a stand-alone application, and ADDmotion is a set of XCMDs that run within HyperCard. The XCMDs enable you to create and play animations within HyperCard stacks, thereby adding the animation capabilities of ADDmotion to the scripting capabilities of HyperCard. ADDmotion is currently bundled with HyperCard 2.2. The programs differ slightly—PROMotion can save animations as stand-alone documents and has a print-to-video feature which blanks the rest of the screen while an animation is playing (useful when recording to videotape). Both programs, however, provide the same tools for creating animation.

Both PROMotion and ADDmotion use paths and actors to create animation. You use either program's built-in paint program to create an **actor** (a graphic that is to be animated), or you import the actor from PICT or PICS files. An actor can consist of several different **cels** (frames). The cels may show the actor in a series of different positions—for example, running.

Actors are animated over backgrounds—still images which are imported into the program and placed onscreen. You place the actor on the screen, and then animate the actor by drawing a line or path on the screen using special tools. You can alter this path at any time. During playback, the actor cycles through its cels while it follows the path. Figure 8.6 shows manipulation of the path an actor moves over in PROMotion.

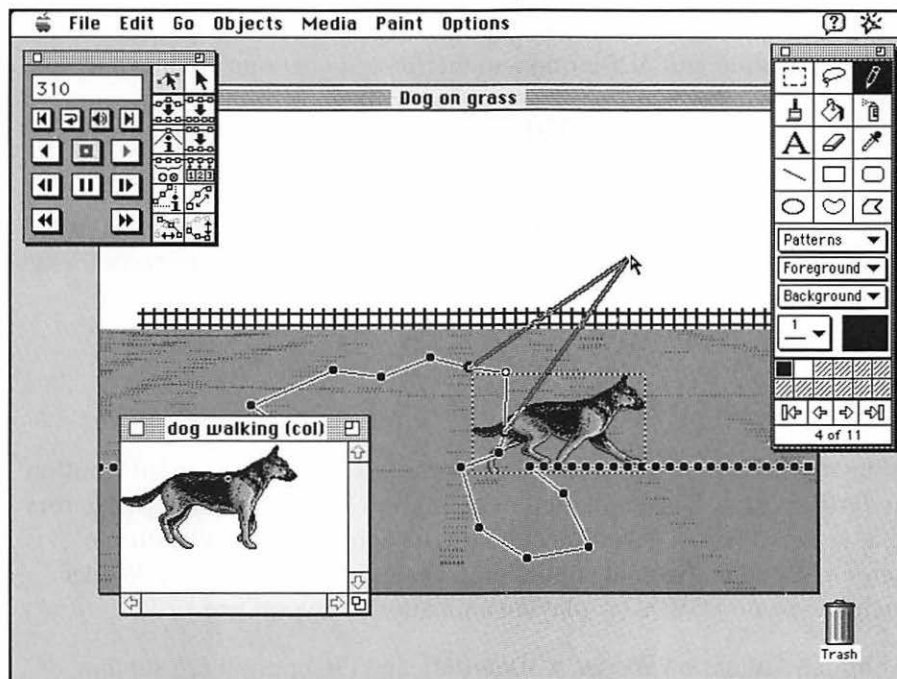


Figure 8.6. PROMotion.



A demo version of PROMotion is on the CD-ROM.

Because ADDmotion (the original program) is designed to work with HyperCard, its animations can be interactive. The animation can stop at different points and pause for a specified amount of time, or until you click an actor. Clicking also can cue events. In ADDmotion, this cue sends a call-back to HyperCard (in other words, when you click on an actor, you issue a message to HyperCard that tells it to do something). PROMotion supports AppleEvents—a new capability added to the Macintosh system, which allows one program to send messages to other applications, telling

them to perform specific functions. You can control a wide range of functions from a PROMotion animation when you use it with a program such as UserLand's Frontier.

You can export animations that you create in PROMotion to other applications. PROMotion includes support for QuickTime movies, an After Dark format file, and a self-playing movie format. You can distribute these stand-alone animations free of charge.

The PROMotion and ADDmotion animation applications are inexpensive and provide many features. The biggest problem with both programs is that creating animations with them is rather slow—saving and opening animations and actors takes several seconds on all but the fastest Macs.

While Motion Works still sells PROMotion, they will soon release a new product called Multimedia Utilities. This is a collection of six tools including an animation program.

Animation Works

Animation Works, from Gold Disk Software, is very similar to PROMotion and ADDmotion. The animation in Animation Works is made up of actors that are made up of cels. You animate the actors along paths on the screen to create the final animation (see figure 8.7). Animation Works includes some XCMDs for playing animation in HyperCard.

Although Animation Works, ADDmotion, and PROMotion are similar programs, each has benefits over the others for specific uses. If you are interested in using animation with HyperCard, then you may prefer ADDmotion because animations can be used to control HyperCard functions. If you want to control AppleEvents, then PROMotion might be your choice because of the built-in AppleEvent support. If you want to create path-based animation only, Animation Works may be the best choice because its performance is better when editing and manipulating animations.

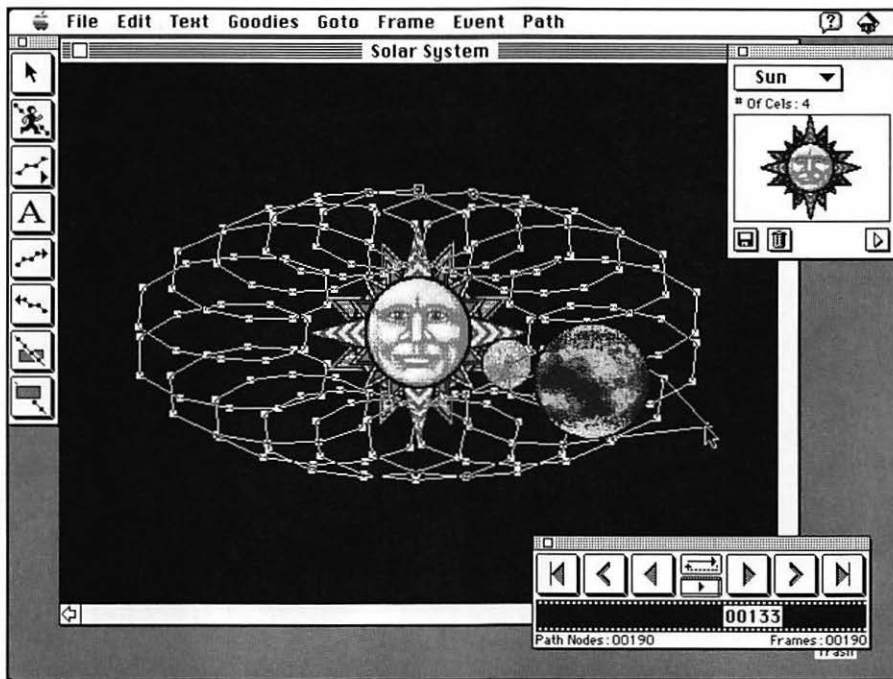


Figure 8.7. Editing path-based animation in Animation Works.

Cinematic

Cinematic is an interactive animation environment that includes several tools to help you create animations quickly and easily. You can drag an object around the screen, and Cinematic records the movement and then plays it back. You also can specify key frames and have Cinematic create the in-between frames. Cinematic also can use a loop of an animation, such as a figure walking, to create an animation sequence.

Cinematic was designed to animate presentations. You can import a PowerPoint or Persuasion presentation into Cinematic and then add animation effects, such as sliding bullets, to the presentation. You can export Cinematic movies to QuickTime or PICS files.



Cinematic has two primary windows. You draw and manipulate objects in the Movie window, which also displays the animation. The Filmstrip window shows small thumbnails that represent the different frames of the animation; you can click a

thumbnail to advance to the frame it represents. Figure 8.8 shows the Cinemation screen. In the figure, a globe is in the Movie Window and a series of thumbnails of the globe is in the Filmstrip Window. A demo version of Cinemation is on the CD-ROM.

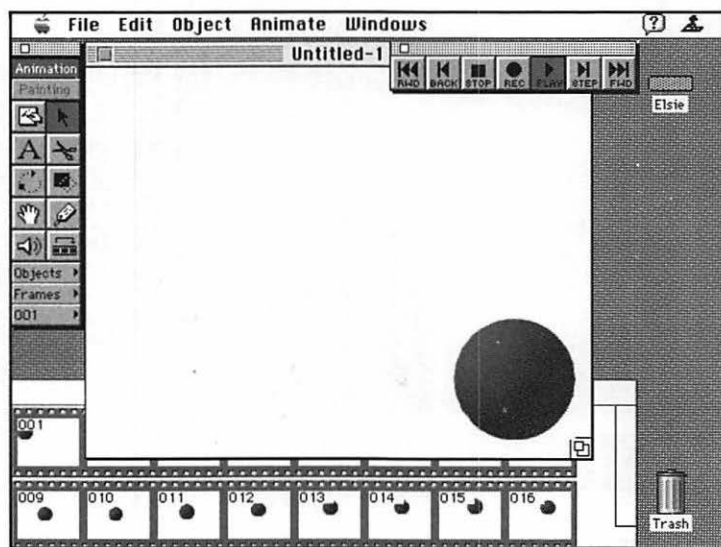


Figure 8.8. Cinemation.

MovieWorks

MovieWorks is also designed for creating animated, interactive presentations. MovieWorks is actually four programs combined into one animation package. You use the Text, Sound, and Paint Editors to edit the separate media types. You use the Composer application to combine these elements to create QuickTime movies or interactive presentations. Composer enables you to integrate the text, sound, and paint materials, and to specify timing relationships between the presentation's elements.

To create a presentation you must import the presentation elements into MovieWork's Mediabase Window and then arrange them on the Stage. The tools provided in the Tool Palette are used to create paths along which the different elements of the animation move. Each element of a movie is contained in a separate track. You can view the tracks in the TimeView window, which also shows where you have applied effects. Figure 8.9 shows the manipulation of a path used to animate an object in MovieWorks.

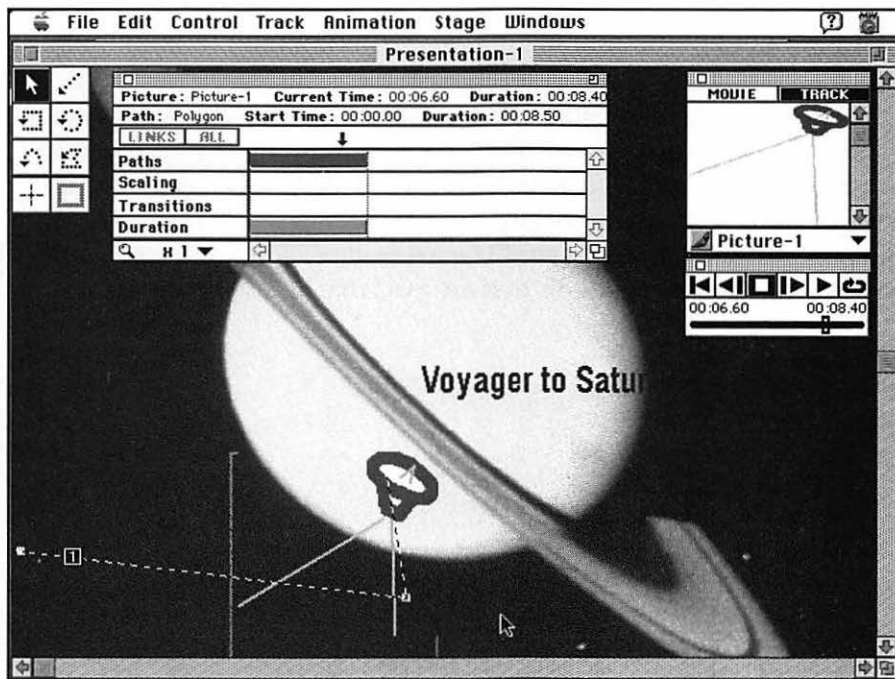


Figure 8.9. MovieWorks.

Other animation packages

Some other animation products are worth mentioning. Animation Stand, from Linker Systems, is an animation tool that operates on the traditional cel-based principles of animation. In Animation Stand, you record the position of every object in every frame on a spreadsheet (traditional animation calls this the **dope sheet**). This program is expensive, and is really designed for the professional who outputs animation to film or videotape.

Finally, don't forget Director, which remains a very popular and powerful package that can be used to create interactive animations. You should read Chapter 15, "Authoring Environments," for more information about Director.

The Future

The future of animation on the Macintosh will be exciting—a safe prediction now that QuickTime provides a very good medium for distributing animations. Interest and demand for 3D tools that create animation is growing—probably because most of us can't draw well enough to create animation in the traditional way! The following chapter discusses some 3D modeling and animation tools that are available for use on the Macintosh.

3D

9

chapter

It took a long time for the Macintosh to find a niche in the 3D world. Until the Macintosh II arrived, 3D and the Macintosh warranted little consideration—the hardware wasn't powerful enough and the screens were only black and white. You couldn't use the Macintosh to create photorealistic images. Now Macintosh users suffer from an embarrassment of riches—more than a dozen different applications perform some kind of 3D modeling, rendering, or animation.

If you work in multimedia, then you need to know at least the principles of 3D programs, if only so you know what's involved in creating a 3D logo for your presentation. Once you decide you need to create 3D images, you'll discover that a vast number of 3D programs are available, and you should have little trouble finding a product that suits your needs.

This chapter begins with a discussion of the basic functions performed by 3D applications and the features these programs offer. The following section discusses how to choose between the many programs available, and the chapter finishes with an overview of several popular packages.

The Three Functions of 3D Modeling

You can perform three functions with 3D applications: modeling, rendering, and animation.

- **Modeling** is the process of creating the different objects that make up a scene.
- **Rendering** is the process of reproducing the scene in a photo-realistic manner.
- **Animation** is the process of creating a sequence of frames that show the objects moving within a scene.

When choosing a program, it's important to realize that some programs perform all three functions, and others perform only one or two. For example, Ray Dream Designer enables you to create models and render a single image, but has no facility for creating animation. Pixar's ShowPlace is an animation and rendering package that provides no modeling tools; you must import models that are created in other programs. Pixar's MacRenderMan is a rendering program that does nothing but render

images from scenes created in other programs. Specular International's Infini-D and Strata's StrataVision provide all three modeling program functions.

Should you buy a program that provides all three 3D functions (and save some money), or should you buy two or three specialty programs? The answer to this question depends upon what you want to do with 3D, as well as the functions provided by the different programs. For example, although Infini-D and StrataVision are good all-purpose tools, you may prefer Macromedia Three-D's animation process. Alternatively, you may not want to create animation, and you may find that Ray Dream is all you need. The modeling tools offered in Sketch! and MacroModel may be more to your liking than those offered in the other programs. The section "Considerations when Choosing a 3D Application" will help you in making these decisions.

The following sections detail the three functions of 3D applications in more detail.

Basic modeling

All 3D modeling programs contain tools with which you can create some basic objects, such as blocks, spheres, pyramids, and cones. You can use these objects to create simple scenes only; you cannot use them to create complex scenes. Figure 9.1 illustrates the basic object tools available in Infini-D.

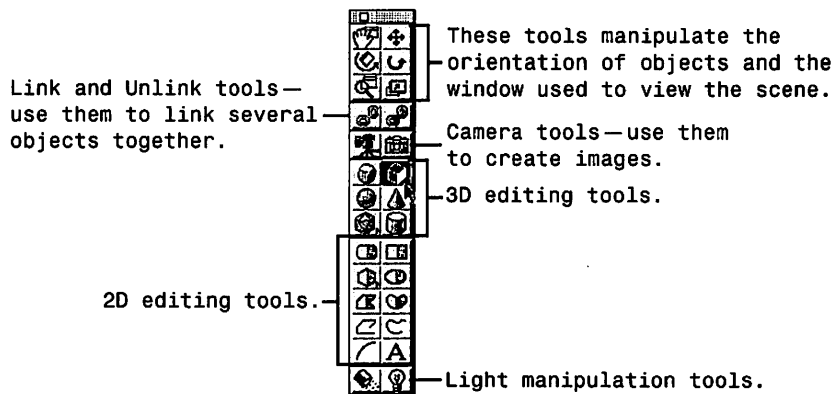


Figure 9.1. The basic object tools in Infini-D.

Extrusion

The simplest 3D modeling method is extrusion. When you create an **extrusion**, you draw a two-dimensional shape and then extrude or push out the shape in the third dimension to create a 3D shape (see figure 9.2).

The complexity of the extruded shape depends upon the tools the program provides for drawing the 2D shape. Some programs provide only a polygon drawing tool; others can draw smooth curves. Several 3D programs enable you to import a PICT file and use it as a template for the 2D shape.

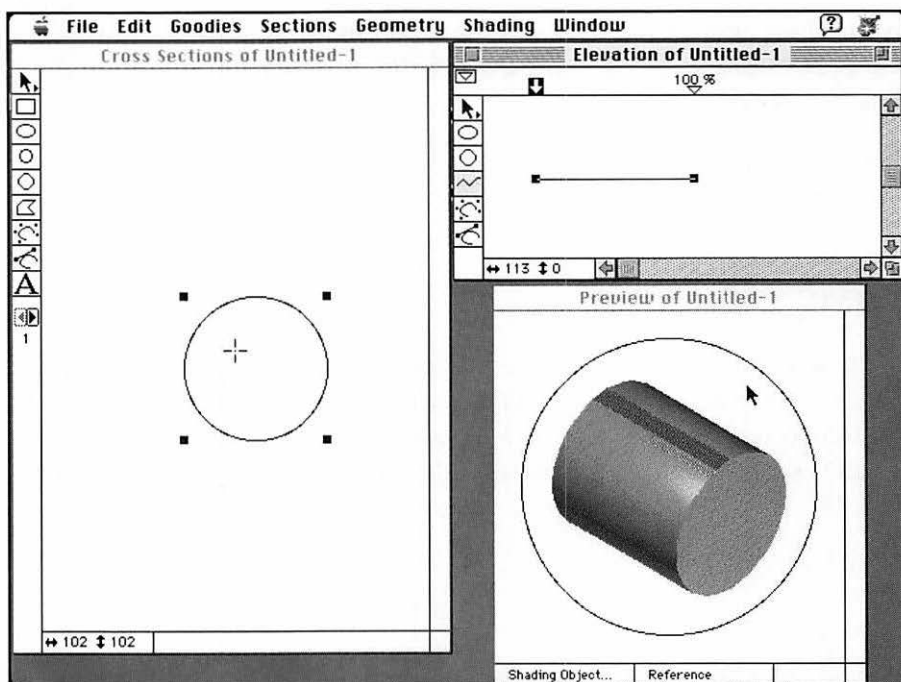


Figure 9.2. Extruding a 2D shape in Ray Dream Designer.

Extrusion is particularly useful for creating logos. Several programs, such as Pixar's Typestry, enable you to add bevels to extruded objects to further enhance their 3D appearance (see figure 9.3).

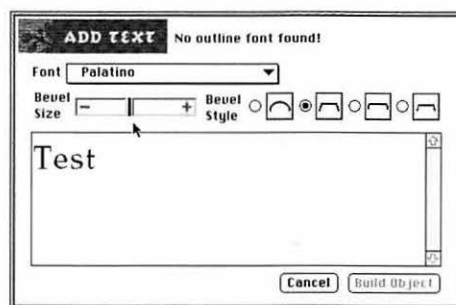


Figure 9.3. Adding bevel size and style to text in Typestry.

Lathing

You create a **lathed** object by sweeping a 2D shape around an axis of rotation to create a 3D shape (see figure 9.4). Lathing results in much more complicated objects than those you can create with extrusion. Lathing is particularly useful for creating objects such as bottles, cups, light bulbs, or any other shape that is symmetrical around an axis.

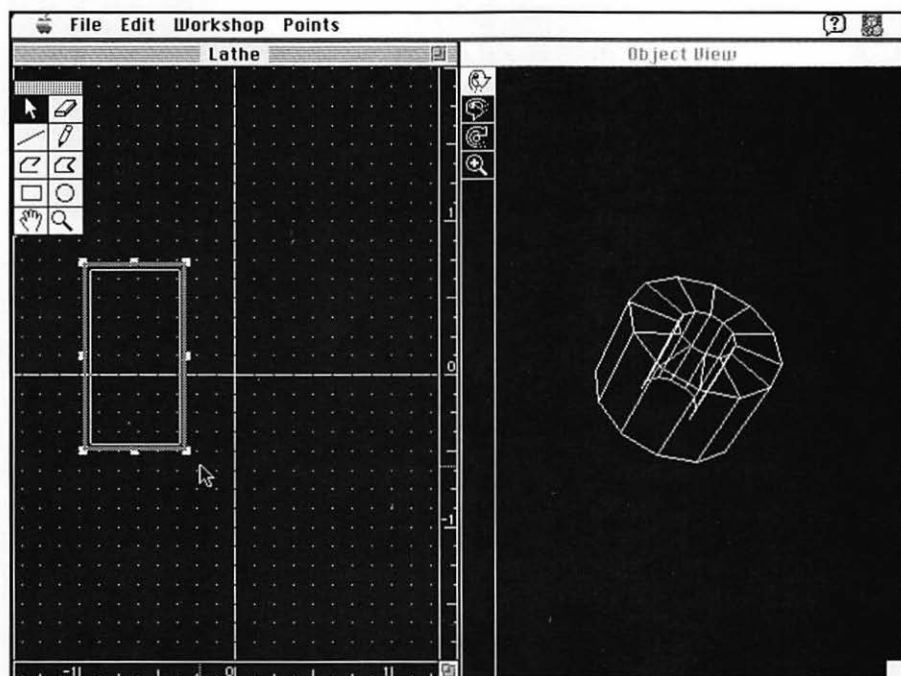


Figure 9.4. A lathed object in Infini-D.

Sweeping

Sweeping (sometimes called “lofting”) is a hybrid of the lathing tool. Sweeping enables you to scale the 2D shape as well as move the shape up or down the axis of rotation as you create the 3D object. The nautilus shell shown in figure 9.5 is an example of an object created with sweeping.

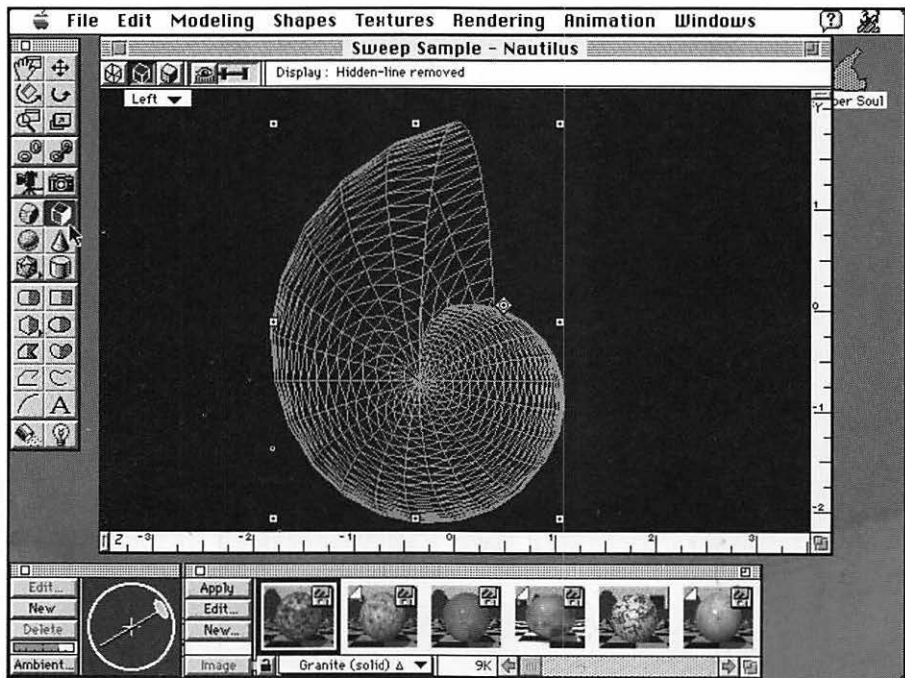


Figure 9.5. Sweeping a shape in StrataVision 3D.

Freeform

Although you can use extrusion and lathing to create a wide variety of shapes, some shapes are impossible to create using those tools. Nearly all 3D programs provide some kind of freeform modeling that enables you to manipulate an object in three dimensions.

The concept of freeform modeling can be difficult to grasp. The processes of extrusion and lathing are reasonably easy to visualize, but the concept of working in three dimensions can be confusing—particularly since the computer screen has only two!

The “traditional” method for freeform modeling is provided in programs like Infini-D. These programs provide three **planes** (views) of the object—usually top, side, and front—and enable you to manipulate the object in each of the planes. The results of your manipulations are shown in an **isometric** (three-dimensional) view. Infini-D’s freeform modeling screen is shown in figure 9.6.

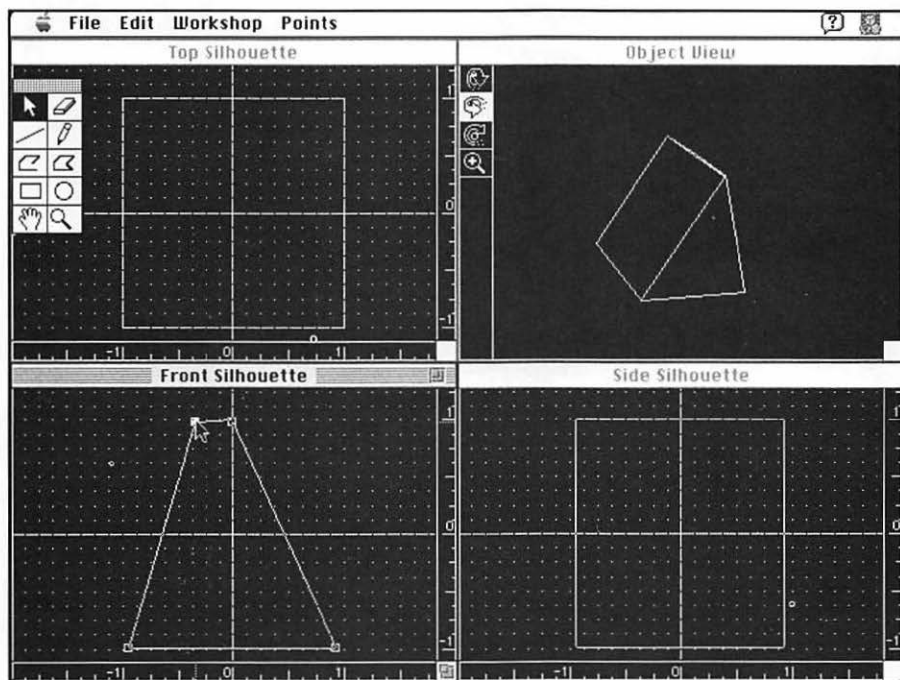


Figure 9.6. Creating a free form object in Infini-D.

Advanced Freeform

Even for the most practiced user, the modeling tools discussed so far have limitations. You can use them to manipulate only certain edges of the object—you can’t just grab part of the object and pull or push it to make the shape you want. To overcome that limitation, you need a tool that simulates clay modeling.

Enter Sketch!. Sketch!, from Alias Research, uses a different method of creating shapes. Each shape is made up of many control points; you can grab, drag, or move any one of these control points to change the shape. Figure 9.7 shows a shape being created with the Sketch! 3D modeling program.

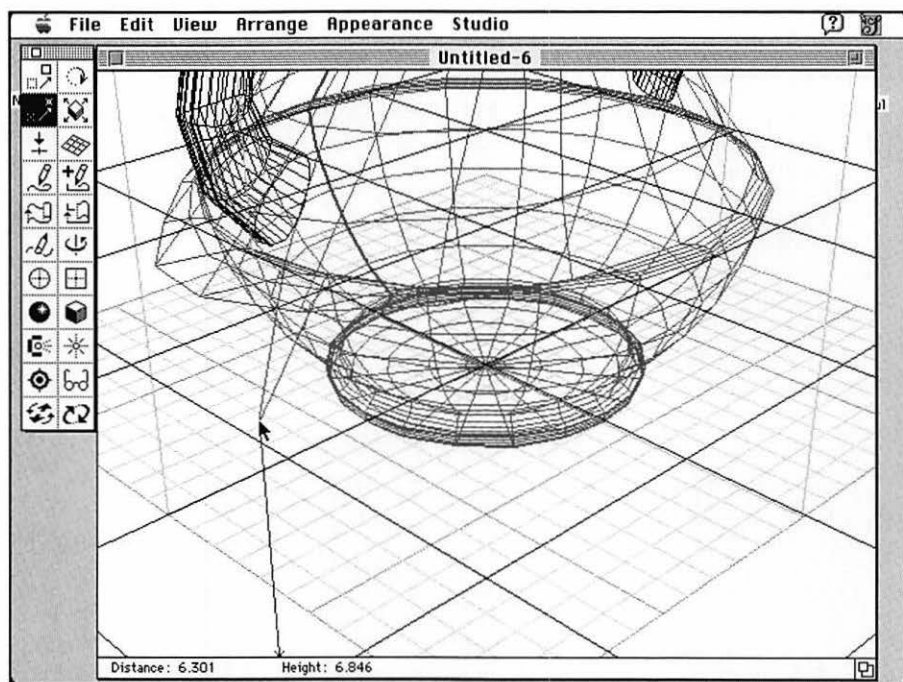


Figure 9.7. Manipulating a shape in Sketch!.

Another program which offers similar functionality is Macromedia's ModelShop. Although Sketch! and ModelShop enable you to create very complex shapes, their functionality comes at a price. They both require a lot of memory and a fast processor.

Exchanging Models—DXF format

Even if one program doesn't create the shapes you need, you may be able to create the shapes in another program and then import them. While all 3D modeling programs use their own proprietary format for saving 3D information, one standard format does exist for interchanging 3D information—DXF. This format was originally designed for transporting files

between CAD applications. DXF supports both 2D and 3D scene descriptions, but the general nature of the format means that various companies have implemented the format in slightly different ways. Just because two programs both say they can save and read DXF format does not mean that you can move models quickly and easily from one program to the other. Further, because DXF files tend to be much larger than models saved in a program's native format, you may create a model that is so large and complex that the importing program cannot read and import it. To avoid this problem, test to determine whether or not you can transfer models between programs before you begin working on a project where this is a requirement.

Rendering

3D modeling programs with rendering capability use mathematical algorithms to create lifelike 3D objects. Most of these programs provide some kind of high quality photo-realistic rendering algorithm, and nearly all support several different algorithms. Some are fast but low quality, while others are slow but produce much better results. These algorithms split into two camps—shading algorithms take into account the surface color of an object and other variables to work out what color should be applied (shaded) on an object. Ray Tracing algorithms trace the path of light in a scene, creating color on objects as the path of the light rays is calculated.

Images created with one program may have subtle visual differences from images created with another program—even when both programs offer similar rendering quality.

One criticism leveled by some potential users of 3D applications is that the renderings look “too perfect” and too obviously computer-drawn. Even the most sophisticated programs suffer from this problem; it may be some time before computers can produce images that are indistinguishable from photographs.

When deciding which rendering capability you need in a 3D modeling program, you may want to consider how much time the program requires to render an image. Infini-D, for example, can create stunning images with Ray Tracing but takes a long time to do so. You could, of course, also use Phong Shading in Infini-D, but that results in lower-quality images. Luckily,

you can speed the rendering process in some programs—including Infini-D—by using either network rendering engines or plug-in accelerators (these devices are discussed later in the “Network and accelerated rendering” section of this chapter).

Shading

The shading algorithm displays the rendered image; the algorithm used determines the time required to render the image, as well as the image’s quality.

Wireframe

When you edit and manipulate models, most programs display the objects either as a **bounding box** (a rectangle representing the size of the object) or a **wireframe** (an outline of the object’s basic shape). Figure 9.8 illustrates an Infini-D wireframe display. Using these display styles speeds redrawing and improves the program’s performance while you work with the model.

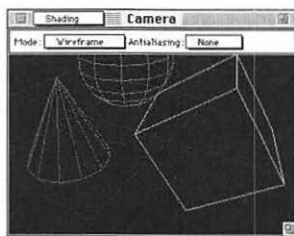


Figure 9.8. Wireframe representation of a scene created in Infini-D.

Flat shading

With **flat shading** the program renders each facet (or polygon) that makes up the surface of the object with a single flat color. This shading technique results in sudden color changes from surface to surface and accentuates the individual surfaces that make up the object (see figure 9.9). Flat shading is very fast, and you can use it to give a quick indication of the colors in a scene.

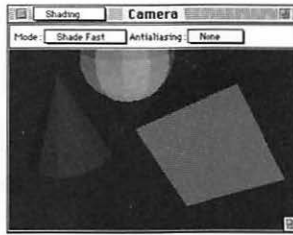


Figure 9.9. Flat shading representation of a scene created in Infini-D.

Gouraud shading

Gouraud shading is a technique in which the 3D rendering program calculates the colors of each surface at the surface's vertices and then interpolates the colors across the surface. This shading technique produces a smearing of color across each surface to connect subtly with the color of the adjoining surface (see figure 9.10). Because the surface colors do not change abruptly, Gouraud shaded objects look much more realistic than flat shaded objects.

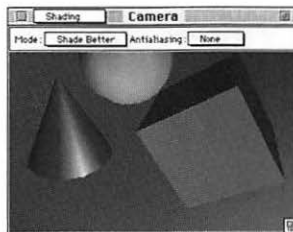


Figure 9.10. Gouraud shading representation of a scene created in Infini-D.

Phong shading

Phong shading is a smooth shading algorithm developed by Phong BuiTuong. Phong shading takes into account three characteristics of a surface: diffusion, specularity, and shininess. Using information about these characteristics and about the scene's lighting, the algorithm calculates the color value for each pixel of the surface. Phong shading can create very smooth surfaces that have reflectivity, shadows, and transparency (see figure 9.11).

A scene that would take a minute or so to render in Gouraud shading would take several minutes with Phong shading.

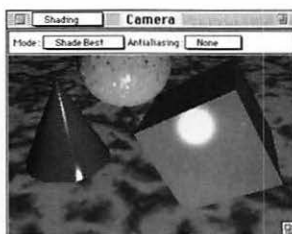


Figure 9.11. Phong shading representation of a scene created in Infini-D.

Ray Tracing

Ray Tracing calculates the path of light rays from their sources in the image to the viewer's eye. Ray Tracing figures in reflection and transparency, and creates extremely realistic scenes (see figure 9.12). Rendering a scene with Ray Tracing is a slow process. A scene that is rendered in 10 minutes with Phong shading may take 100 minutes or more to render with Ray Tracing.

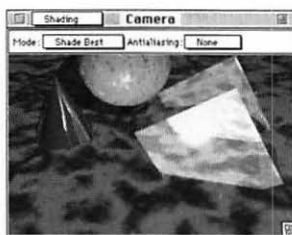


Figure 9.12. Ray-traced representation of a scene created in Infini-D.



These example images are included on the CD-ROM that comes with this book in the folder Rendering Images.

Surfaces

While the rendering algorithm is largely responsible for determining the quality of a final image, another determining factor is the software's surface modeling capabilities.

Procedural surfaces

Nearly all 3D programs offer some kind of procedural shader. A **procedural shader** is a mathematical description for a surface that defines such properties as the shininess and transparency of the surface. The advantage of procedural shaders is that they can accurately illustrate an object that is made from a solid material—for example, a sphere carved from a piece of wood appears to have grain that runs realistically through the sphere.

The problem with using procedural shaders is that, in most cases, you cannot write your own shaders and thus are limited to those that come with the package. Luckily, most shaders have parameters that enable you to adjust the qualities of the surface (for example the frequency and color of the grain in the wood) to create a range of different surfaces. You can write your own shaders for MacRenderMan, but you must first learn the RenderMan render language (which resembles C code).

Texture mapping

Texture mapping enables you to import a flat, two-dimensional image (for example a PICT file) and wrap it around an object to create a surface. With this rendering technique, you can add just about any surface to an object simply by scanning a picture or drawing a surface in a paint program. Programs usually enable you to control the orientation of the surface as you wrap it around the object. This capability is particularly helpful when you want to apply a specific pattern to a surface (like a logo on the side of a spaceship). Figure 9.13 shows texture mapping being applied in the Ray Dream Designer program.

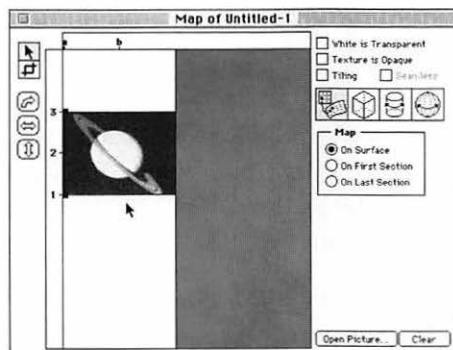


Figure 9.13. Applying a texture map in Ray Dream Designer.

You also can use texture maps to create bump maps, reflection maps, and transparency maps (see figure 9.14). A **bump map** creates the illusion of raised areas on an object's surface. You can use a bump map, for example, to depict continents on a globe, as illustrated in figure 9.15. The figure shows two renderings of the same object (a globe). Both renderings use the same texture map, but a bump map is applied to the object on the right. Both renderings in this figure were done in Macromedia Three-D.

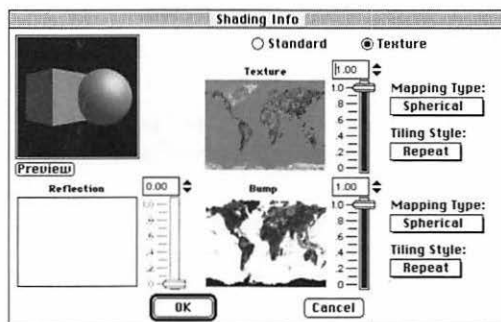


Figure 9.14. Specifying the texture map and bump map in Macromedia Three-D for figure 9.15.

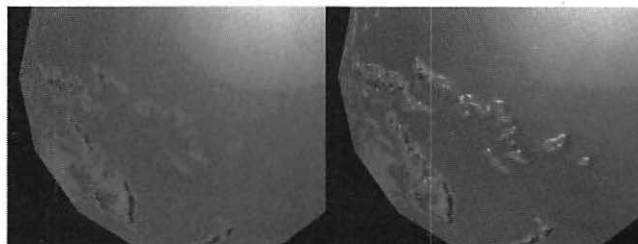


Figure 9.15. Two renderings of an object using the same texture map, but with a bump map applied to the rendering on the right.

The raised areas shown on the globe's surface don't actually exist on the model, but are effects added by the bump map. Most bump maps are merely grayscale images. The whiter the pixel in the image, the higher the "bump" on the surface (you can reverse this effect in some programs).

Reflection and transparency maps work in a similar fashion. The reflection map alters the reflectivity of the object's surface (one part could be highly reflective, while another part would be less reflective); transparency maps perform the same function for the transparency of an object.

Lighting

Lighting plays a major role in the realism of scenes—a fact often overlooked by those starting out in 3D. Creating realistic lighting is not a trivial exercise, as anyone who has tried to light a real-life scene for photography or videography can tell you. You often must add several lights to a scene to simulate daylight.

Perhaps the biggest problem in learning how to light a scene is that you must render a scene at the highest quality to see the lighting's complete effect on your scene. This can take a long period of time, and makes it difficult to experiment. Beginners should experiment with lights in a simple scene that can be rendered quickly so that the effect of lighting changes can be seen easily.

Three-D programs usually offer three types of lights: **point lights** cast an even light in all directions; **spotlights** cast light in one direction only; and **ambient** light provides an overall level of light in the scene. You can light your scene with ambient light, but because ambient light usually does not cast shadows, you will have to add extra lights to make your scene look natural. Figure 9.16 shows the three types of lights as they affect a scene.

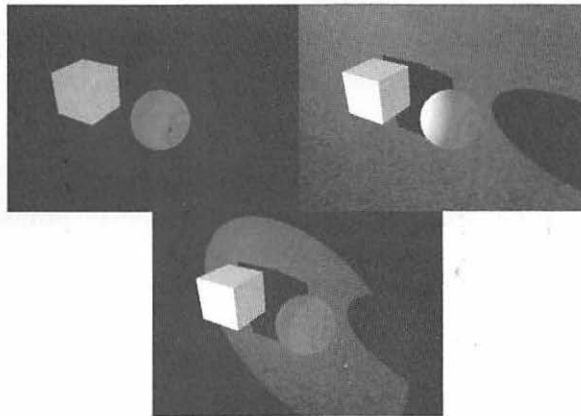


Figure 9.16. The same scene rendered with different lighting. Top left has ambient lighting only. Top right has point lighting with ambient lighting, and the bottom image has spotlighting with ambient lighting.

Positioning spotlights in 3D scenes can be tricky. Some programs provide a “point at” feature that enables you to point an object at a scene. This feature can be useful for positioning spotlights (see figure 9.17).

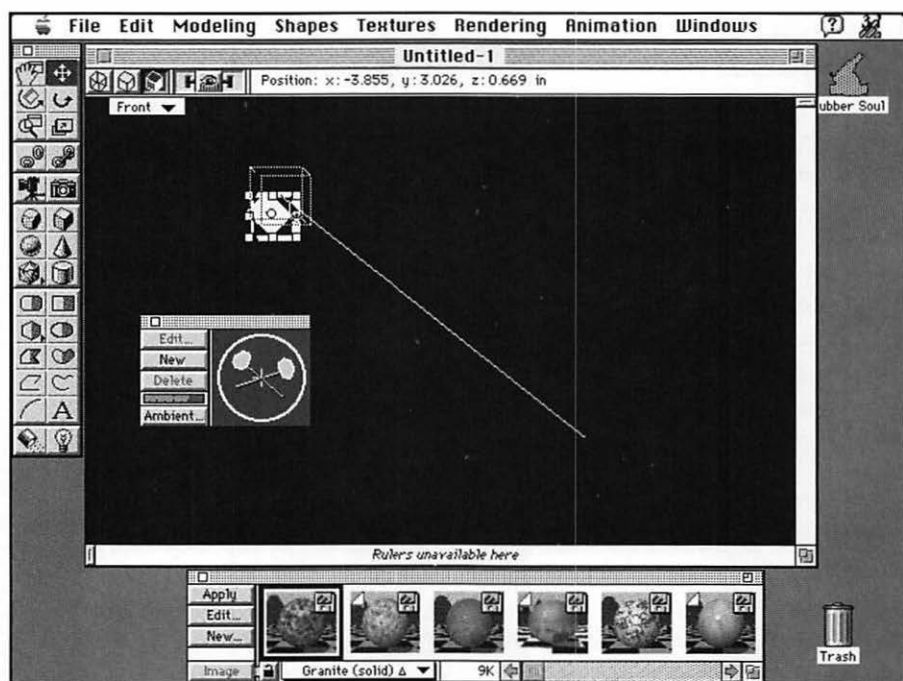


Figure 9.17. Adjusting a light in StrataVision. As you drag the light around the screen, a line from the light indicates the direction the light is pointing.

RIB format and RenderMan

As mentioned earlier in the chapter, some 3D modeling programs do not include a rendering function. When you create objects and scenes with such programs, you use other programs—called **renderers**—to render the scenes. You must save the scenes in a format that can be used by the renderer.

A de facto standard format for saving scenes to be rendered by a renderer is the RenderMan RIB format (a rendering package developed by Pixar). RenderMan has its own language for defining a scene, as well as for writing procedural shaders. A modeling application saves a RIB file for rendering by a RenderMan-compatible renderer. Pixar sells MacRenderMan, a Macintosh version of the RenderMan renderer, although they also sell rendering packages for several other platforms.

The quality of the RenderMan renderer has helped make RIB a standard rendering format of the 3D world. Several programs save files in RIB

format, but many others include their own high quality rendering engine and do not support RIB. A lack of RIB support does not mean a program can't be useful in its own right.

Network and accelerated rendering

The time required to render a scene has deterred many users from undertaking large tasks using Macintosh 3D packages. Several developers have recognized this fact and have come to the rescue with network rendering solutions. Network renderers send out parts of a scene's data file to several computers on a network. Each computer renders a portion of the image and sends the results back to the master rendering program. The master rendering program puts the parts together to create the final image. Most network renderers break an image into small squares; as a rendering engine finishes a square, the network renderer sends it another, until all the squares in the image are complete. Network rendering greatly speeds the process of rendering a scene.

Network rendering engines are available for Ray Dream Designer, Infini-D, and StrataVision. The engines must be purchased separately for each machine on the network. Figure 9.18 shows network rendering in DreamNet, the Ray Dream Designers network renderer.

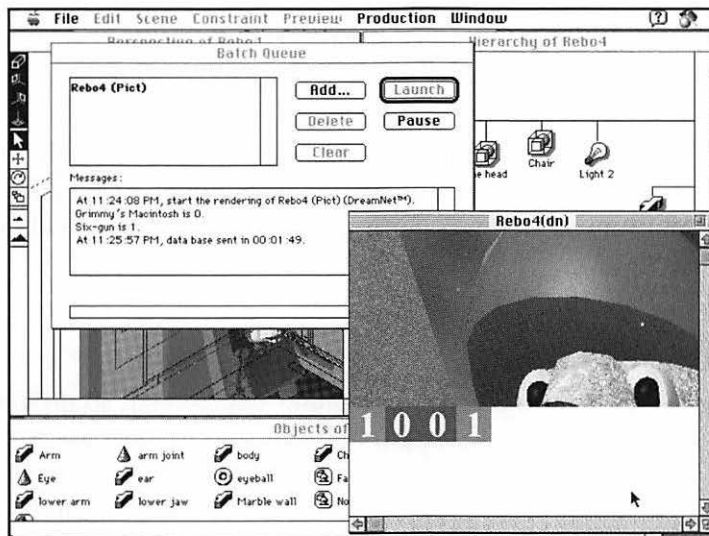


Figure 9.18. Network rendering using DreamNet.

Another way to speed up rendering times is to use a hardware accelerator. The YARC (the reverse spelling of “Cray”) company makes a plug-in accelerator board that works with some software products to accelerate rendering times. Infini-D and MacRenderMan both work with the YARC board. Plug-in boards like the YARC board use a special processor and require that the developer rewrite the software to work with the board. Another way to decrease rendering times is with an accelerator card, an example of which is the Radius Rocket. These products contain a faster version of the Macintosh chip. Such boards speed up all Macintosh functions, not just rendering times. However, a YARC board can be three or more times faster than a single Rocket at rendering a scene.

Power Macintosh

By converting their applications to native Power Macintosh applications, developers have been able to speed up their programs by 300 percent! That’s the good news. The bad news is that if you have a 3D application that hasn’t been converted to Power Macintosh, and you are running it on a Power Macintosh, then you’ll probably need to use SoftFPU, an FPU emulator. This is because most 3D applications require an FPU, and there’s no FPU in the Power Macintosh. Unfortunately, while using SoftFPU will enable you to run the application (though some programs may not work), it will run very slowly, so it’s not recommended.

Animation

Several programs provide animation features, though some make it much easier to create and manipulate animation than others. The following sections discuss some animation features and the methods used to create animation.

Creating animations

There are two primary methods for defining animation in 3D programs. The simplest method is called **key frame** animation. In key frame animation, you choose a particular frame (say, frame 10) in an animation, and instruct the program to record the position of all the objects in the frame. You then choose another frame (say, frame 25), adjust the position of the objects, and have the program record the position of the objects in the

new frame. The program animates the scene by calculating the in-between positions of the objects from one key frame to the next.

The advantage of working with key frame animation is that it is easy to create, and the program can easily calculate and generate it. The disadvantage is that you can have difficulty editing or making subtle changes to the animation. Simulating acceleration of objects over time is particularly difficult in key frame animation.

Several programs now offer **event-driven** animation. With event-driven animation, you choose any object at any frame in an animation and move the object to a new location. This creates a new event that is specific to that object. Adjusting this event for this object does not alter the events or movement of other objects. Event-driven acceleration also makes it possible to add complex motions and change them easily. Figure 9.19 shows event-driven animation being created in the Infini-D program.

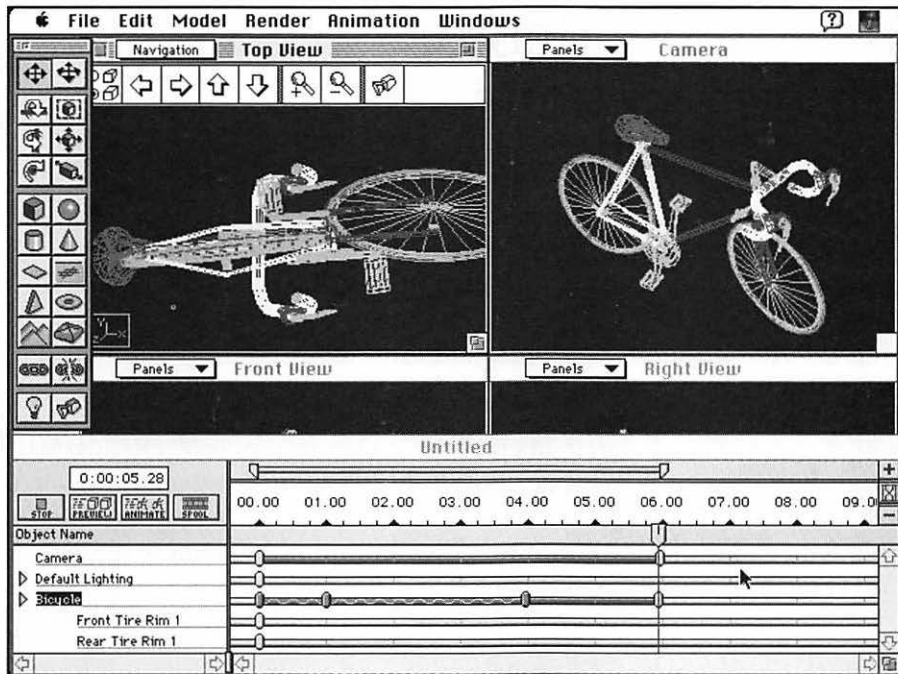


Figure 9.19. Event-driven animation in Infini-D.

PICT, PICS, and QuickTime

Three-dimensional animation programs export their results in a number of formats. The most popular formats are a series of PICT files, a QuickTime movie, or a single PICS file. The PICT files format enables you to edit the images in a graphics program, while QuickTime is perhaps the best way to distribute and play animations. With the advent of QuickTime, the PICS format has become less widely used, although many programs still support it.

Now that you are familiar with the features and functions available in 3D applications, you are better able to determine which program will be useful for your work. Each 3D modeling application is a unique blend of the functions and capabilities described in this section. The following section provides more information to help you choose an application that fits your needs.

The Categories of 3D Applications

To make the task of choosing a 3D application easier, it helps to first divide the many 3D applications into three broad categories: architectural modeling 3D applications, simple 3D extrusion applications, and general purpose 3D applications. A fourth category—Virtual Reality—is separate primarily because of the way it is used, rather than the features of the program. By categorizing applications in this way, you can at least eliminate some programs as you try to decide which program to buy.

Note that not every program fits into this categorization—MacRenderMan does not fit because it is a special purpose rendering program. However, if we created enough categories to completely enclose every product, it would not make sense to divide the programs into categories at all! 3D CAD programs are not included in this discussion.

Architectural modeling 3D applications

Architectural modeling applications are specially designed to simplify the task of creating models of buildings and cities. Two such applications, Macromedia's ModelShop and Dynaware's DynaPerspective, provide

precise dimensioning of the objects that make up a 3D model. Both programs also provide functions useful for architectural models, such as the capability to create shadow studies (display where the light falls when the sun is in a particular location). Although neither application has very sophisticated rendering support, you can export models to another program to render the final scene. Both applications can create animated fly-thrus—an animated movie where the camera appears to fly through the building—for export to QuickTime or an animation program. Figure 9.20 shows the DynaPerspective program in use.

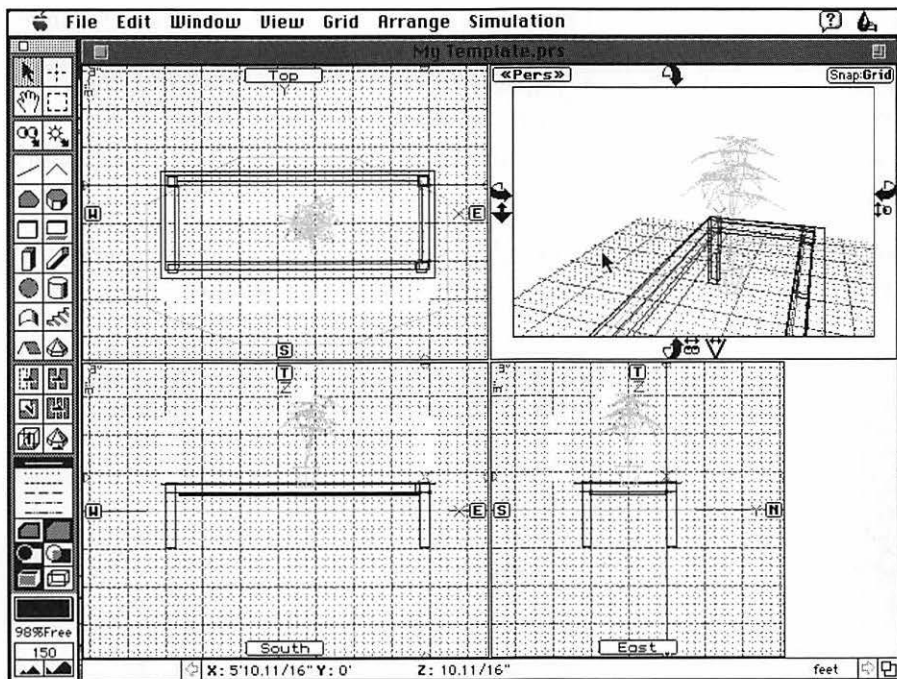


Figure 9.20. Designing a table with a plant in DynaPerspective. Elements like this are used as building blocks to create final scenes.

Simple extrusion 3D programs

Simple extrusion programs make up a relatively new category of 3D applications. These applications provide only the extrusion method of modeling (see the section entitled “Extrusion” that appears earlier in this chapter). You draw a two-dimensional object, and then extrude it into the third dimension to create a three-dimensional object. Most of these

programs are primarily designed for working with type, but some enable you to extrude other shapes as well.

Extrusion modeling is easy to understand and work with (the programs are simple for developers to write, as well). And since many people use 3D applications for the sole purpose of creating 3D logos, these programs are just right for that purpose.

Even though applications in this category provide extrusion options only, the quality of their renderings can equal those of general purpose 3D applications. Pixar's Typestry program includes a built-in version of the MacRenderMan rendering engine (see the discussion of this Pixar rendering software earlier in the chapter). MacRenderMan produces extremely high-quality results. Typestry also can create a simple flying logo—3D letters and company logos that fly across the screen—and export the results to a QuickTime movie. Figure 9.21 shows a scene being created in Typestry.

Competing products include LogoMotion from Specular, Dimensions from Adobe, and addDepth from Ray Dream.

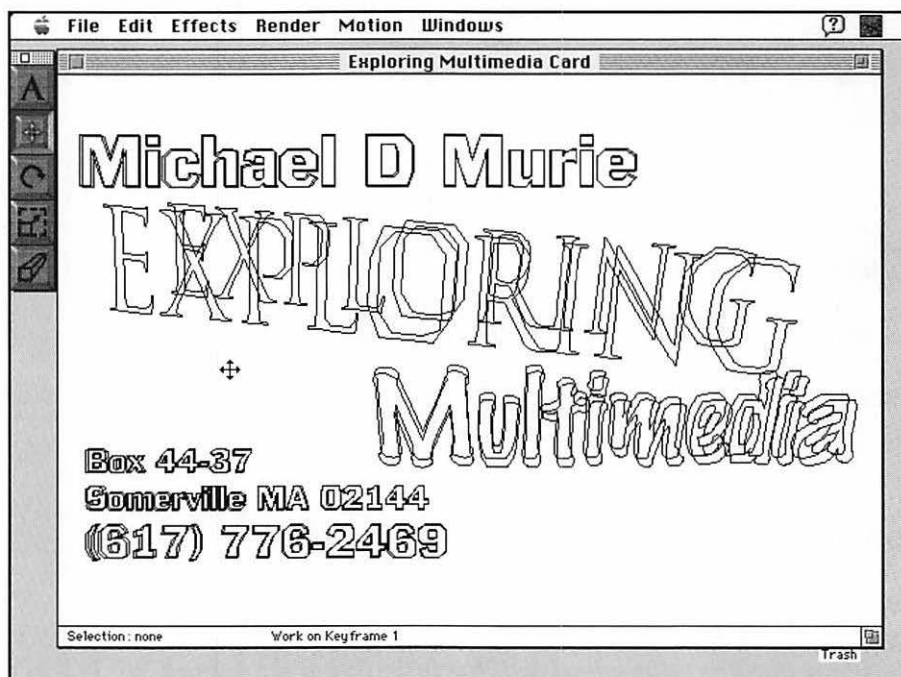


Figure 9.21. Creating a scene in Typestry.

General purpose 3D applications

In terms of categorization, the general purpose 3D applications are all the programs that don't fit into the other two categories. Not to confuse the issue further, you could say that all 3D applications are general purpose tools, and that the other two categories of 3D applications are merely subsets of this category. Examples of general purpose 3D applications are Infini-D, Ray Dream Designer, StrataVision, and Macromedia Three-D. These programs—and many others like them—have functions beyond simple extrusion and are not designed specifically for architectural modeling. They can therefore be used as general purpose 3D modeling tools.

Virtual Reality

Virtual Reality is the simulation of an environment using a computer. The idea is to provide an artificial experience as realistic as possible. How is this different from a “virtual reality” presented in a game? One of the goals of VR is to provide as much realistic stimulation to the user as possible. This is often done with stereoscopic vision systems, pressure sensitive input devices, and complex sound systems.

Virtual Reality is a new field, and there are software and hardware tools available for the Macintosh. There is one 3D application which makes it possible for the user to explore a three-dimensional world. Virtus' Virtus VR has most of the same characteristics of the 3D architectural modeling applications. Virtus VR enables you to create models in a way similar to ModelShop and DynaPerspective, but it offers one spectacular advantage to those programs. With ModelShop and DynaPerspective, rendering a fly-thru animation takes a long time; the programs can't produce a fly-thru in real time. Virtus VR can render the scene fast enough that you can change the view of the scene and immediately see how things change. The program can provide the illusion of walking through a scene (see figure 9.22). QuickTime movies can be applied to surfaces—for example, the user approaches a wall and sees a TV screen. The quality of the rendering does not rival that offered by the other 3D modeling applications, but the real-time feedback provides other benefits.

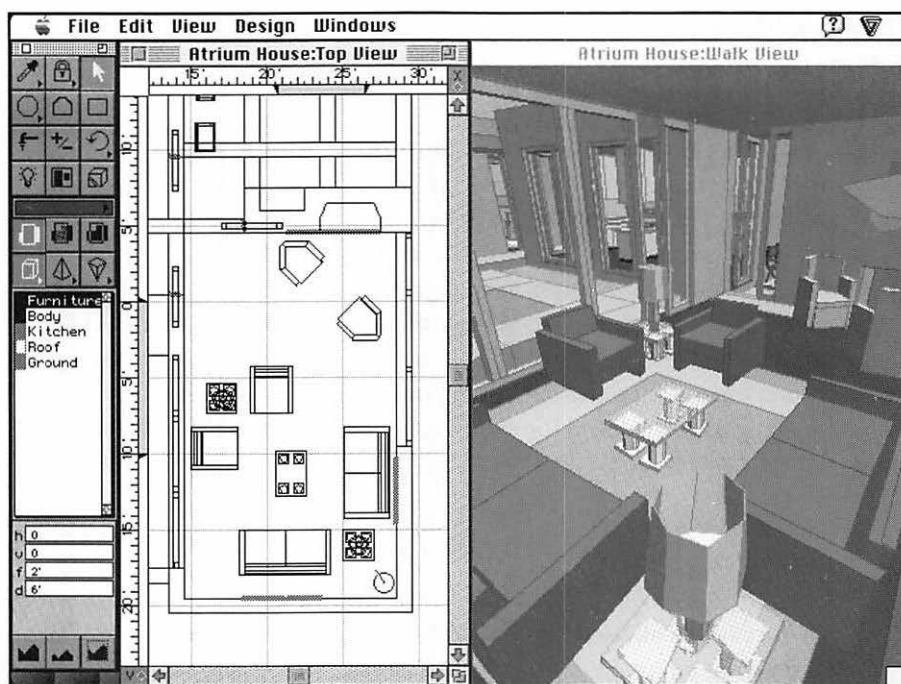


Figure 9.22. Virtus VR. The window on the left shows the model being explored, while the window on the right is the rendered view of the scene.

You can use Virtus to create stand-alone versions of models for distribution to others. Other users can view and explore the models, but cannot change them.

Virtus continues to improve the program. A new version is expected with improved rendering quality. If they add buttons that initiate other actions—jump to another part of the models, or display a graphic—then Virtus would be particularly useful to multimedia developers.

Considerations When Choosing a 3D Application

Several 3D applications currently share the market, and each has its own strengths and weaknesses. Choosing the right application can make the difference between owning a productive software tool and owning a box of software that sits untouched on the shelf.

Even within each category of 3D applications you are faced with a number of choices. Although the programs vary in the ways they enable you to create models (how they are manipulated, and in the quality of final renderings), the fact remains that you can create the same scene in many of the programs and achieve similar results. Choosing a program may come down to personal preference based on the kind of application you have in mind, or some particular feature that you need.

For beginning 3D users, choosing a program can be particularly difficult. Take the time to consider the uses to which you may put the program. Do you intend to use animation? Do you plan to animate logos only? Are you looking for a neat effect, or a serious 3D modeling tool? If you can answer those questions, you may be able to narrow the field, from a dozen or so possibilities to a few strong contenders.

Other good sources of information are trade shows and colleagues. Approximately once a year one of the Macintosh magazines has a comparison of several 3D applications. Be wary of computer stores, where the salesperson probably won't be familiar with all the features that are available for each product.

The following section provides details about some of the 3D modeling applications currently on the market. The information in these sections can help you further understand the capabilities of 3D modeling, as well as the individual strengths and weaknesses of the listed programs.

A Look at Some 3D Applications

To help those starting out in the world of 3D, the following sections briefly describe some of the programs currently available and highlight some of the advantages of each program.

Ray Dream Designer

Ray Dream Designer is a good all-purpose tool for creating still images. It does not support animation, but you can purchase network rendering software for speeding up the process of rendering. Designer does not support Ray Tracing.

Designer is a two-application set consisting of Light Forge, in which you create objects, and SceneBuilder, in which you arrange the objects into a scene that is then rendered.

Although Light Forge doesn't have the freeform editing capabilities offered by other programs, because of the way Light Forge displays objects as you create them, the application is much simpler to use than most other three-dimensional modeling programs.

Figure 9.23 shows an object being extruded in Light Forge. In the figure, you see the 2D shape drawn in the left window, and an extrusion line drawn in the upper-right window. The result of the extrusion is shown as an isometric 3D shape in the bottom-right window. You can click and drag on the 3D view to change the orientation of the shape; this enables you to view the results of your modeling from different angles.

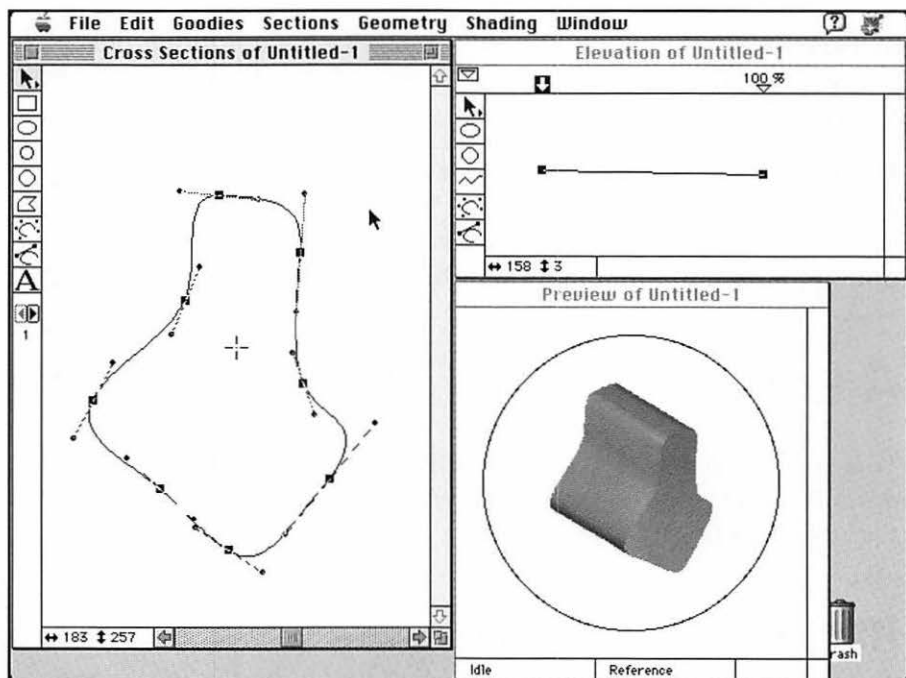


Figure 9.23. Creating an object in Light Forge.

You can create more complex objects by defining a series of different shapes along the length of the extrusion; Light Forge then creates a “skin” over those shapes (see figure 9.24). You step through the shapes in the left window using the two button arrows in that window’s tool bar.

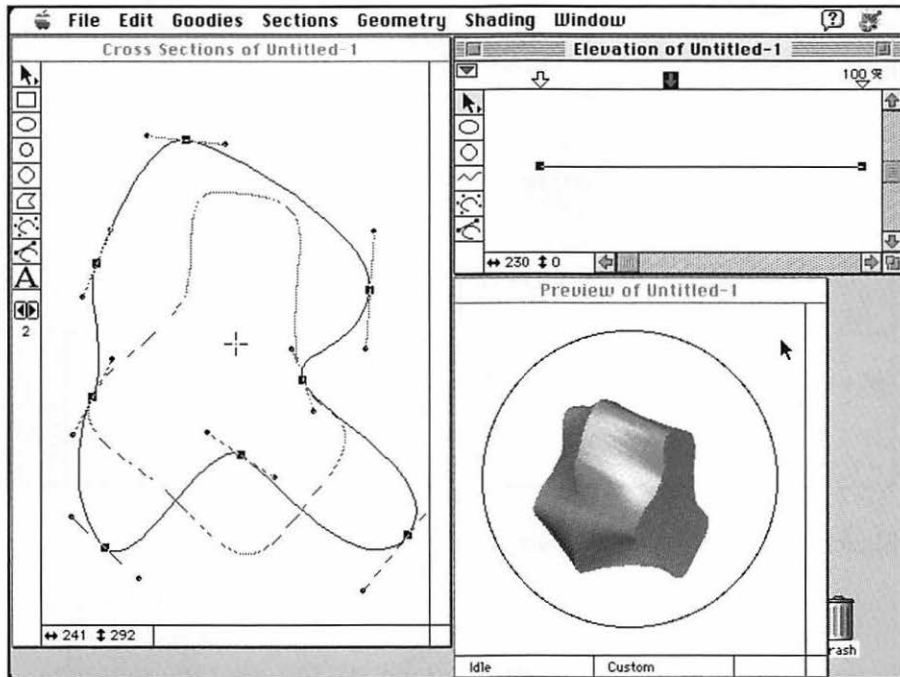


Figure 9.24. Complex objects in Light Forge.

You define the object’s surfaces in Light Forge, and then save the object and import it into the SceneBuilder application.

In SceneBuilder, you arrange the objects you created in Light Forge into a scene (see figure 9.25). The objects appear in an isometric view. Although moving objects in a single view can be difficult (considering the mouse moves in two directions only), Designer uses projections of the images on the three walls around the scene to help you control on which axis the movement occurs—you simply click on the projection of the object on one of the walls to move it along two axes. For example, clicking on the projection of the object on the back wall enables you to move the object up and down and side to side, while clicking on the projection on the floor enables you to move the object forward and backward and side to side.

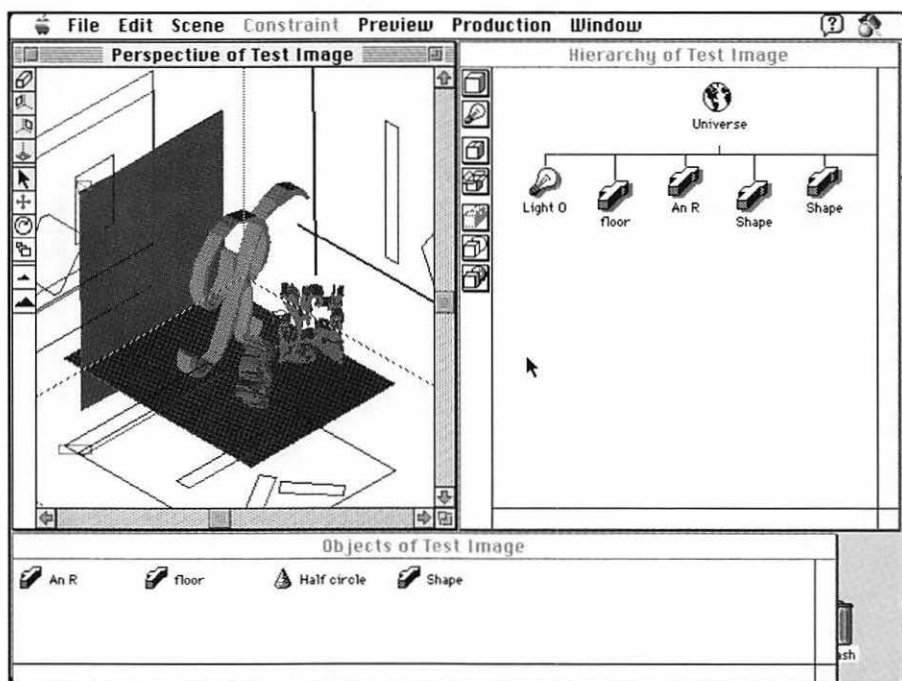


Figure 9.25. Arranging a scene in SceneBuilder.

Ray Dream Designer is a reasonably powerful program, and its low price (around \$300 by mail order) makes it a very good value. Although Ray Dream doesn't provide freeform object editing, the other tools make up for this omission. The program's rendering quality cannot match that of a program such as Infini-D, but it is certainly acceptable for most illustration purposes.

If you are new to 3D work, Ray Dream's simplicity of use and low cost makes the program worth your consideration.

Specular International Infini-D

Infini-D from Specular International is a good all-purpose 3D application with general purpose modeling tools, a high-quality rendering engine, and powerful animation features. If you want to buy only one program, then Infini-D may be the one for you.

Specular has released a network rendering engine (called BackBurner) that also supports the YARC co-processor board. You can create beveled text in Infini-D, and the latest release has special “assistants” that you use when creating animation effects.

Infini-D includes several procedural shaders and access to all of the shader parameters through a standard dialog. You can use PICT images as a surface map, or even use a QuickTime movie mapped to a surface. When you map a QuickTime movie to a surface, Infini-D uses the successive frames from the movie as a surface on the object in each successive frame of the animation. Figure 9.26 shows the Infini-D interface.

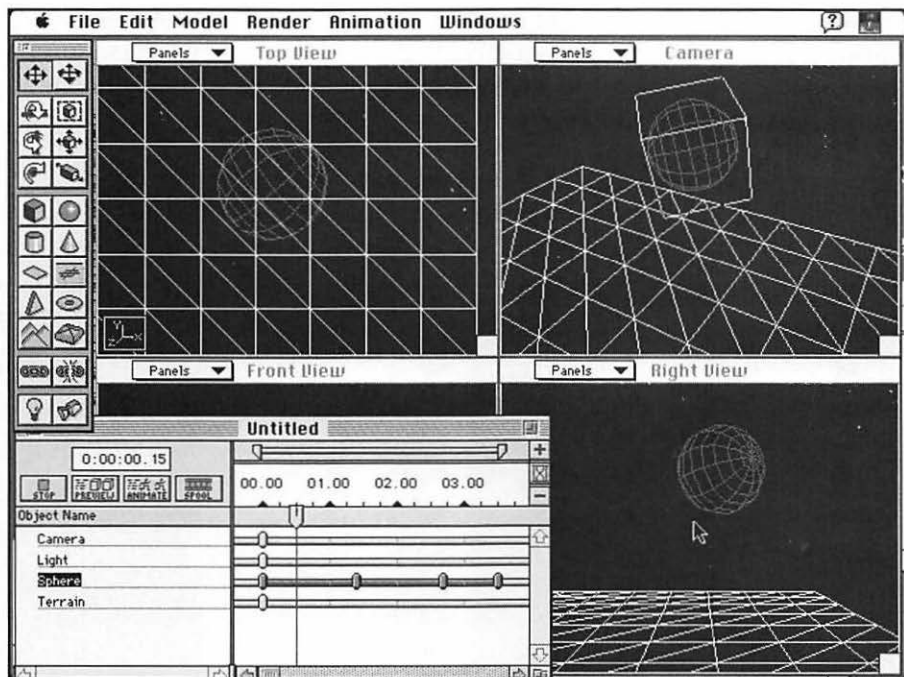


Figure 9.26. Infini-D. The top left window shows the side view of the scene, while the top right window is the camera view. The bottom right window shows the side view of the scene, while the bottom left window is the animation sequencer—used to animate the scene.

Strata Inc's StrataVision 3D

StrataVision 3D's modeling, rendering, and animation features rival those of Infini-D. Although Infini-D probably has the edge in animation, some users prefer the quality of the rendering produced in StrataVision. The program interfaces are very different, and you may prefer one program's interface over that of the other.

Other differences between Strata and Infini-D are that Strata offers a sweep tool and a variation of the Ray Tracing algorithm, called Raydiosity. Raydiosity takes into account inter-reflections among objects and is especially suited to scenes with indirect lighting. Raydiosity renders even more slowly than Ray Tracing, however. Strata has released a more powerful (and expensive) program called StudioPro, which offers more powerful features, including a 3D sculpting tool that resembles the one found in Sketch! (see below).

Strata sells a large number of texture maps as well as simple model libraries. Figure 9.27 shows a scene being manipulated in StrataVision.

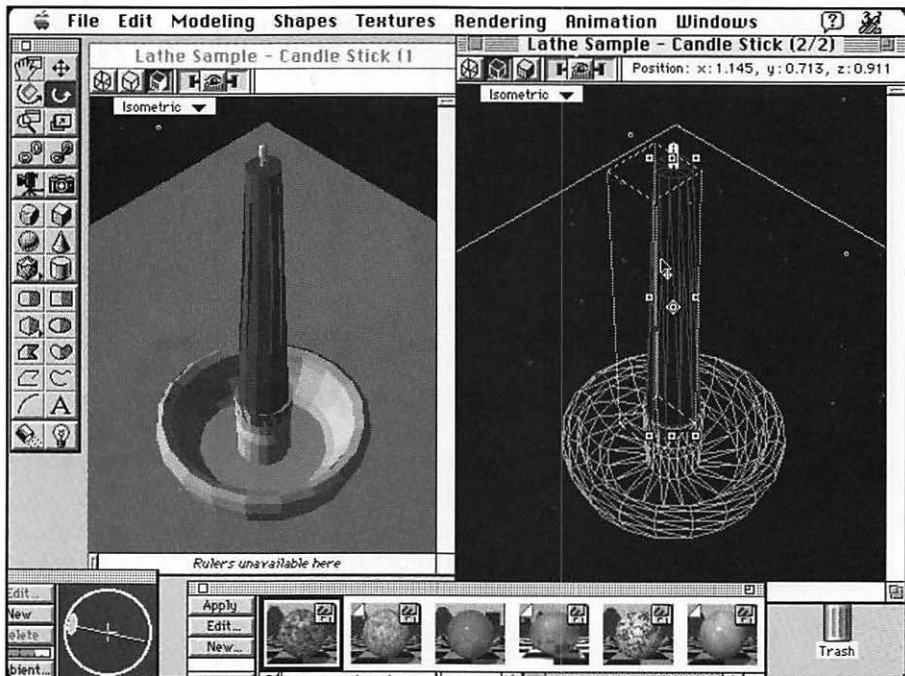


Figure 9.27. Manipulating a scene in StrataVision.

On the downside, Macromedia Three-D requires a powerful computer to provide good response time when manipulating scenes. It can take several seconds to respond to a user's action. If you can afford a Quadra, however, or are willing to put up with the lag in response time, then you may prefer the interface used for creating animation in Macromedia Three-D.

Alias Sketch!

Alias Sketch! is a modeling and rendering engine that lacks animation features. Sketch! more than compensates for this limitation, however, with its powerful tools for creating three-dimensional shapes. The capability to draw a freeform shape, extrude the shape, and then simply grab a point and pull it to deform the shape became available to Macintosh 3D users only with the release of Sketch!. Figure 9.29 shows a figure being modeled in Sketch!.

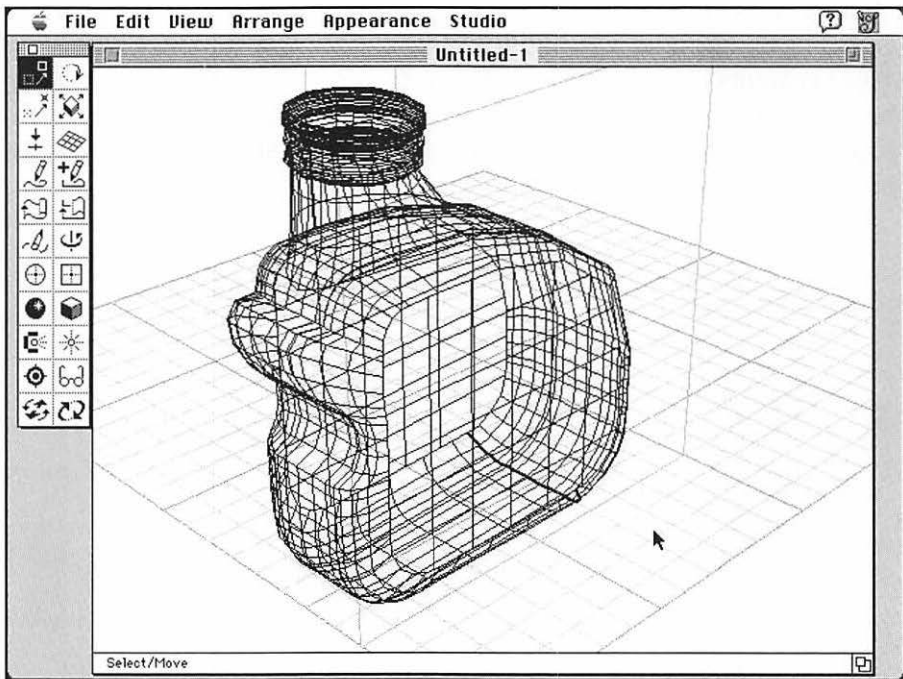


Figure 9.29. Modeling in Alias Sketch!

Sketch! also has a great feature for matting a 3D object into an image—for example, a photograph. You can open a PICT file, and then manipulate the object so that it fits within the scene. Sketch! provides tools for defining in which plane the object is oriented. After you position and light the object, Sketch! renders the object to create a final scene that contains the background picture and the added object.

Be aware that you must have lots of memory and a fast processor to use Sketch!.

All the rest?

No user can expect to learn—let alone buy—every 3D modeling program that is currently available for the Macintosh. The applications mentioned above are those that the author has the most experience with; you may find that one of the many other available products better suits your needs.

Some other programs to consider include MacroModel (which provides freeform modeling similar to that in Sketch!), Electric Image (a popular application for television and film production due to its rendering engine speed), and ShowPlace (an easy-to-use program for arranging objects to render a scene using Pixar's MacRenderMan).

A couple of unusual tools

Two programs, covered in the graphics chapter, use 3D modeling techniques to create graphics. KPT Bryce enables you to create landscapes, while TextureScape enables you to create textured backgrounds.

Hardware Requirements

Three-D modeling applications are some of the most demanding applications available both in memory and processor requirements. You need at least 8 MB of memory, and some programs require nearly 20 MB when using complicated models. Even with these amounts of memory, rendering and manipulating complex models takes a long time on even the fastest Macintosh.

Most of these programs when running on a 68K machine require an FPU (Floating Point Unit). The FPU is not available on all Macintosh models, but may be an option. If you think you may want to work with 3D, make sure that you buy a computer that has an FPU. Power Macintosh versions will run much faster on the Power Macintosh than a 68K version will run on the fastest Quadra (68K machine). Unfortunately, most 3D applications require an FPU, and because the Power Macintosh does not include one, you may not be able to run a 68K application on the Power Macintosh.

If you want to render a large number of images for an animation, you may need a hardware accelerator (discussed earlier in this chapter). Although the 12 hours required to render a single image may be acceptable on a once-a-week basis, it won't work when you need to render 30 frames a week. Of course, you have this option only if your software supports hardware or network acceleration.

For more information on the topic of Macintosh hardware, see Chapter 5, "Hardware."

The Future

Among the makers of the many 3D applications currently available, no single developer has a stranglehold on the market—unlike other market segments such as photo retouching and illustration. The benefit of this wide-open market is that with no clear industry leader, each company has to work hard to improve its software.

On the downside, the 3D market is much smaller than the other markets, and probably will remain so. Whether such a small market can support all these companies remains to be seen. At one time, AT&T Graphics marketed MacTopas, a sophisticated 3D modeling and rendering program. AT&T has since let the product revert to the original developers (who now sell the product). AT&T may have decided to abandon the 3D Mac market to pursue other interests, or they may have found it unprofitable. (They were selling the product for \$7,000!)

Three-D applications continue to be important tools for multimedia developers. As the processing power of the Macintosh improves, you can expect the performance of these programs to improve, as well.

The next chapter discusses QuickTime, a topic of particular interest to those creating 3D animations.

QuickTime

10

chapter

In reality QuickTime doesn't synchronize the video with the sound track. Each track is synchronized with a time track. Because the sound and video tracks are synchronized to the time track, they appear to be synchronized to each other.

What is QuickTime? The simple answer to that question is that QuickTime is software, developed by Apple, that supports time-based media on the Macintosh. An example of time-based media is video—a sequence of images that are displayed onscreen to create the illusion of motion. The images are time-based because in order to retain synchronization with a sound track, the images must be displayed at the correct time in relation to the sound track.

But there's much more to QuickTime than just maintaining synchronization between sound and video tracks. And QuickTime isn't just video—QuickTime can also be used for animation or even text. A full discussion of QuickTime could fill one or two other books. The discussion in this chapter introduces you to QuickTime software and the ways it can help you in preparing your multimedia presentations.

This chapter begins with an explanation of how QuickTime works and why it's important. If you are already familiar with QuickTime, you may want to meet the rest of us later in the chapter. The section “Installing QuickTime” tells you how to install the QuickTime extension on your Macintosh and play QuickTime movies. The “Recording Video” section explains the basics of how to record video using QuickTime on a Macintosh. The remaining sections introduce some video digitizing boards, accelerators, and software products you may consider using in your work with QuickTime.

Understanding the Importance of QuickTime

QuickTime adds support for time-based media to the basic Macintosh system architecture. The most common media that require this kind of support are video, animation, and sound. Sound has been supported on the Macintosh since its introduction, but it's important that QuickTime supports sound because QuickTime can synchronize sound tracks with video tracks. Similarly, there have been applications that supported animation since the early days of the Macintosh, but while these applications usually let the animator adjust the rate at which animations are

played, and even add sound to animations, none of the programs closely synchronized these elements. The most common example of this problem is that animations would play at different speeds when played on different machines.

QuickTime provides this synchronization. Most of this chapter centers on QuickTime's application to video, but remember that much of this discussion also applies to animation.

QuickTime stores video digitally—that is, it converts the frames of video to digital images (computer representations of the image) that are stored on the computer's hard disk. When the video is “played,” these images are read from the disk and displayed on the computer screen. The advantage of storing the video this way is that the computer can very quickly and easily access any part of the video sequence, just as it can quickly and easily access other information on a hard disk. Another advantage is that after the frames are stored on the hard disk, they can be manipulated using a number of different editing applications (which are discussed at the end of this chapter).

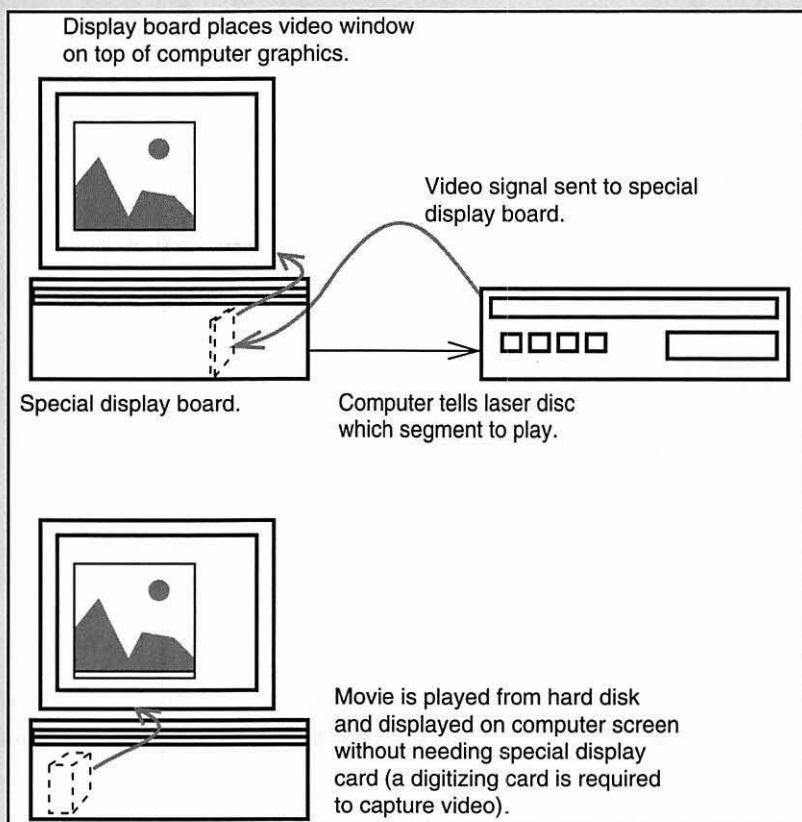
The disadvantage of storing video this way is that the computer has to do a lot of work to display the video sequences. So much work is required that most Macintosh models can only display comparatively small movies—small both in size, about 240 by 180 pixels, and frame rate, around 15 frames per second (fps), although the fastest Macintosh models can manage 320 by 240 at around 15 fps. This is much smaller than video seen on television sets, which is 640 by 480 pixels and 30 fps. It is possible to improve this performance by adding hardware acceleration (see the section later in this chapter).

Don't confuse the discussion of digital video in this chapter (which is essentially digital video on a Macintosh) with digital video in other mediums. For example, Digital Video recorders already exist for recording HDTV and NTSC format video. These recorders are tape-based systems that record the video signal digitally on the magnetic tape.

Note

You don't have to store video sequences on your hard disk to play video on a computer screen. It is possible to take an analog video signal (most video signals you will encounter—from VCRs to cable television—are analog) and display it on the screen. To do this you will need a source (such as a laser disc player) and a video display board to display the video images on the computer monitor.

The signals go into the display board, which adds them to the graphics being output from the computer, and displays the results on the screen. The computer itself is only peripherally (forgive the pun) involved in the video; although the video image is displayed by the video card, the computer processor does nothing except tell the display board where on the screen to add the video to the graphics.



Even without digital video, being able to play video on a computer monitor is quite an achievement. If you have a laser disc player you can control the player from the computer and create a sophisticated presentation. But this achievement is not as exciting or as useful as digital video (storing the video on the computer). Interacting with analog video from an external source is much more difficult. This chapter devotes its discussion to the topic of true digital video.

What's so neat about digital video?

It's easy to understand the importance of synchronizing time-based events. You probably can understand why people want to synchronize animation and sound on the Macintosh, but what's so neat—or useful—about these tiny windows playing video on the Macintosh screen? Well, nothing—and everything—is neat about digital video. Digital video today is often unimpressive because using it on the computer is very difficult (as you see later in this chapter). Digital video files are huge in comparison to the average graphics and text files, and the image quality and the frame rate of the video are usually very low.

Digital video, however, has several benefits for the multimedia producer or user. First, it is new, and computer people get very excited when they can do new things with their computers. Second, after the digital video is in your computer, you can edit, manipulate, and distribute the material to others, with relative ease. Digital video also is very easy to add to multimedia presentations (the process is much easier than hooking up your VCR and trying to cue things from there). Third, the computer can randomly access the movie—quickly jump to any part of the movie. It can even play it backwards! Finally, as you will see later in this chapter, the addition of special hardware makes it possible to play video that almost rivals what you see on your television set.

But, to be honest, we all look forward to the day when we easily can record and play back television-quality video on a computer—in other words, the day when we can watch *Saturday Night Live* on our computer (now there would be a giant step forward in multimedia!). When that happens, why will you need your TV? And how will you tell the difference between your TV and computer? These questions present some interesting philosophical issues, but we will leave them to the big thinkers and get back to digital video and QuickTime.

What's the difference between animation and video?

Wait a minute, you're thinking, what's the difference between animation and digital video? In truth, the two aren't much different. The Saturday morning cartoons on television are good examples of how animation can be video and how video can be animated.

Subtle differences between animation and video emerge when you work with your computer. Animation programs have been available for the Macintosh since its inception. Digital video, on the other hand, has only been around for a few years. The big difference between animation and digital video is the amount of data each requires the computer to deal with. Animation is much easier to store and play on a computer than video because animation files tend to be simpler and smaller than digital video files. Even if your animation uses very low frame rates, viewers probably won't tell you, "That doesn't look very lifelike." The background may be one static graphic, and the animated character a sequence of seven or eight small images that repeat as the character moves across the screen. This arrangement saves memory—and also means that not much changes onscreen from one frame to another.

Video's makeup is very different. The frame rates must be higher or movement looks unrealistic—we expect more of "real-life" images than we do of animated characters! More importantly, even if one video frame appears to be very similar to another, they may be completely different—for instance, a slight move in the camera changes the colors of all the pixels in a frame. Figure 10.1 illustrates the difference between sequences of animated frames and video frames. These differences are important when choosing a means to compress the QuickTime movies.

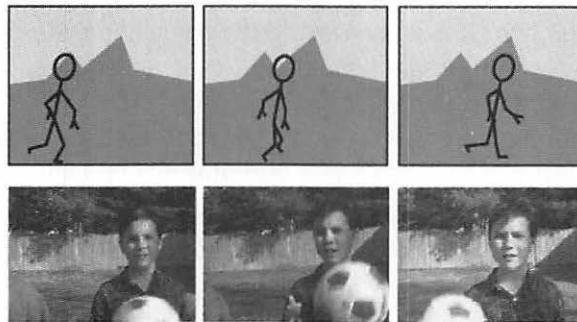


Figure 10.1. The difference between a sequence of animation frames and a sequence of video frames. In video a large amount of the scene changes even though the "scene" remains the same.

How does QuickTime work?

To achieve digital video the computer needs two things. First, the computer must be capable of handling video's large amounts of data. Normally, a computer application will load what it needs to work with into memory before it performs another operation. Director, for example, likes to load all the cast members into memory before displaying the animation. While the best performance is achieved by displaying from memory (time to go from memory to the screen is minimal compared to the time it takes to read from disk), it is not possible to load multi-megabyte movies when you only have 8 MB of memory available. To solve this problem, QuickTime uses **streaming** (essentially the computer displays one frame as it reads the following frame from the disk). Second, the computer needs some kind of timing mechanism that keeps video and sound synchronized. Timing is particularly important when you display people speaking because without synchronization their lips won't move in synch with the sound.

QuickTime fulfills both needs. QuickTime has many features, but the two most important are its compressors and its timing mechanism. The compressors enable the program to reduce the frames to a manageable size, so the computer can stream the information from the disk. The timing mechanism ensures that multiple tracks maintain synchronization.

Without compression, the large amount of data would prevent even the fastest hard disk from playing a movie—by the time the computer had read the first frame from the disk, the time for displaying the frame would have passed. Even with compression, QuickTime may not be capable of playing all the frames in the movie—for example, a Quadra is capable of capturing and playing a much larger movie than a slower machine, such as an LC. In theory, the timing mechanism in QuickTime handles this problem by dropping frames. A 20-frame movie may play at only 10 frames per second on an LC, in which case QuickTime plays only every second frame in the movie. QuickTime always tries to maintain the quality of the sound, since any loss or skip in sound is very noticeable.

The details of the algorithms that provide compression are not as important as the fact that QuickTime provides support for compression. Although QuickTime comes with several different compression algorithms (called compressors), you can add other compressors or even use hardware-based compressors to improve the performance of QuickTime movies.

QuickTime involves much more than the information covered here—this section has breezed over the details to tell you about the important areas related to multimedia. In Chapter 7, “Graphics,” you can read about JPEG—a QuickTime compressor that is useful for working with still images. QuickTime text tracks are described in the Text chapter, while QuickTime support for Audio CDs and MIDI files is covered in the Sound chapter. You can learn more about QuickTime by picking up a copy of either *Cool Mac QuickTime* or the *QuickTime Handbook*, both published by Hayden Books.

Installing QuickTime and Playing Movies

QuickTime is a single file, called QuickTime. You must place this file in the System folder (the Extensions folder, if you are running System 7), and then restart your computer to add the QuickTime functionality to your system (see figure 10.2). Two other files provide additional functionality. The PowerPlug contains customized code for Power Macintosh models, while the QuickTime Musical Instruments file contains the synthesized musical instruments used when playing MIDI files (see Chapter 11, “Sound”).

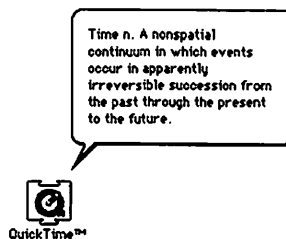


Figure 10.2. The QuickTime extension with its Help balloon.

At this writing, QuickTime only runs on a color-capable Macintosh—for example, the Macintosh II models, the LC models, the SE/30, most PowerBooks, all Centris and Quadra models, and Power Macintosh models. QuickTime does not run on a Macintosh Plus, Classic, SE, or PowerBook 100.

Your computer must be running System 6.0.7 or later. The Sound Manager (part of the operating System) changed with version 6.0.7, and QuickTime requires the new Sound Manager. Further, if you run System 6 and have a pre-Mac IIci computer, you must install Color QuickDraw. Color QuickDraw is on the Printer disk of the System disks that came with your Macintosh, and it must be installed manually. Better yet, install System 7 and don't worry about it because Color QuickDraw is built into System 7.

When you restart the Macintosh, the QuickTime icon appears on-screen during the startup sequence. The icon is the only visible evidence that QuickTime is installed, unless you have an application that requires QuickTime to run, or you have a movie and an application that plays QuickTime movies.

Tip

To play a movie with QuickTime, you must have an application that is capable of playing QuickTime movies. You need something like Movie Player (released with QuickTime and distributed by some user groups and bulletin boards), Popcorn (a shareware program distributed by user groups and bulletin boards), or one of several utility players that have been released with applications.

If you don't have a player application, but must see the movie, you can play it—but you have to restart your computer to do so. Name the movie StartupMovie and place it in your System folder. Then restart your system. The movie should play during the startup. This technique is recommended only for emergencies!

Recording Video

After you have installed QuickTime and played a few movies, you may want to start recording your own movies. To record movies you need items included in the following list (and shown in figure 10.3).

1. A video source: this is the video your Macintosh will digitize. The source can be a VCR, camcorder, or laser disc player.

2. A video digitizing board: usually installed inside the computer, the video digitizing board converts the analog video signal to a digital image. The most common boards are NuBus cards that plug into one of the slots in the computer. Another video digitizing board is a version of SuperMac's VideoSpigot that fits the LC's PDS slot. All AV Macintosh models have video capture circuitry built-in, so you don't need an additional board (unless you need hardware compression, see below).
3. Cables: use these to connect the video digitizing board to the source. Cables are very important! RCA-style or the newer S-Video cables are commonly used.
4. Digitizing application: an application that works with QuickTime and the digitizing board to capture a sequence of images to a QuickTime movie. No digitizing application comes with QuickTime (though the application Movie Recorder is included with the QuickTime Starter Kit sold by Apple). Most of the video digitizing boards now available come with some application for capturing video to QuickTime; AV Macs include a simple recording utility.
5. Sound digitizer: an optional feature. If you want to record sound, you must digitize the sound (see Chapter 11, "Sound"). The video digitizer may include a sound digitizer (such as the Radius VideoVision) or it may not (the SuperMac VideoSpigot does not). If no sound digitizer is included, then you can use the built-in sound recording hardware (if your Macintosh has it) or you can use a MacRecorder (also described in Chapter 11).

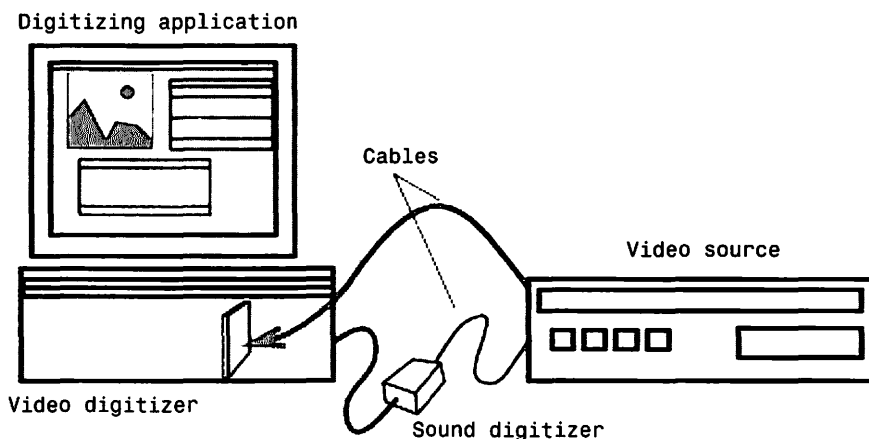


Figure 10.3. Everything you need to record QuickTime movies.

Tip

If you want to record from television, record the material to tape first and then digitize the material from the videotape; you then have more than one chance to get it right. Also, you must use a VCR or a television with a monitor-out option to record from television. Very few digitizing boards have a built-in television tuner, and therefore cannot directly capture a signal coming from, for example, your cable network.

The compressor's role in digitizing

The frame rate of the movies you capture depends primarily upon the speed of the computer you use. The biggest bottleneck is the time it takes to compress an image, although the time it takes for the hard disk to store the information can affect performance. Your choice of compressor also affects the movie's frame rate.

Generally, when we talk about **compressing** a movie, we refer to either digitizing a movie using the compressor, or recompressing a movie that has already been captured. When we talk about **decompressing** a movie, we refer to the process involved in playing the movie back on the computer.

QuickTime comes with several compressors; some are better for still images and others are better for movies. QuickTime's current compressors are None, Video, Photo JPEG, Graphics, Animation, and Cinepak. These compressors are discussed in more detail later in this chapter.

Tip

None? What's a None compressor, you ask? The None compressor does no compression; it just passes the information out to be stored on the hard disk. Because the None compressor doesn't compress, it can sometimes save the frames to disk much faster than any of the other compressors. The resulting movies are very large, but have higher frame rates. You can recompress such a movie with another compressor (usually the Video or Compact Video compressor) to reduce the size of the movies.



Included on the CD-ROM is a movie (Movie Test) and a still image (Image Test) that have been compressed using each of the compressors. Examine these movies and images to see the different sizes and performance that each compressor provides.

The most commonly used compressor for video clips is the Video Compressor—a good general-purpose video compressor. The advantage of this compressor over the Compact Video compressor is that the Video Compressor is almost symmetrical (see the following Note).

Note

A **symmetrical** compressor takes about the same time to compress a movie as it does to decompress the movie. An **asymmetrical** compressor may take a lot longer to compress a movie than to decompress it.

Asymmetrical compressors compress movie frames more efficiently because they can spend more time examining each frame and finding the best way to compress it. Asymmetrical compressors make movies much smaller, but are slower and require an extra step because you must capture the movie first and then recompress it.

You can use a symmetrical compressor to capture a movie in real time and then play the movie back; you can use an asymmetrical compressor to play a movie back, but not to capture a movie. Of course, if a compressor takes 10 minutes to compress a frame and 10 minutes to decompress the same frame, the compressor is classed as symmetrical, but is not useful for playing the movie. As a rule, therefore, you can assume that a symmetrical compressor also is a fast compressor.

The following section describes the compressors included in QuickTime, and offers some recommendations for their use.

QuickTime compressors

As stated earlier, QuickTime comes with several compressors. Each of these compressors has its own strengths. The following information can help you select the compressor best suited to your individual task.

- **Photo JPEG:** This compressor reduces the size of photographic images dramatically, but the images retain exceptional detail (usually, you cannot detect the difference between the JPEG-compressed image and the original). The high compression factors make it useful for reducing the size of photographic images, but the JPEG compressor takes several seconds to decompress large images.
- **Graphics:** This compressor is optimized for still images created using an 8-bit palette. It is not suitable for video.
- **Animation:** This compressor works best with computer-generated animation. It does not work well with video because of the “noise” inherent in video sequences (**noise** describes the way pixels in video images change color subtly from frame to frame, even though the photographed objects remain the same color).
- **Video:** This compressor works best with digital video—sequences of images of photographic nature. It does not work well with computer-generated animation because the algorithm does not handle drastic changes in color from one pixel to another (often seen in cartoon animation where only a few bold colors are used).
- **Cinepak:** This compressor is extremely efficient for compressing video sequences, but is very slow. You can compensate for this slowness by using the None or Video compressor to capture sequences, and then use the Cinepak compressor to recompress the sequences. Cinepak-compressed movies typically are half the size of movies compressed using the Video compressor, but visually are still of high quality.

Some tips for compressing video clips

As a QuickTime user, you are likely to want to compress video clips. You can experiment to determine which compressor best accomplishes this task on your computer.

To begin, try capturing a clip from a videotape or disc, so that you can capture the same clip two times; capture the clip once with the Video compressor and once with the None compressor. When you capture with the None compressor, you must recompress the clip afterward (okay, you don't have to, but uncompressed clips are very large and disk space is still expensive).

See which compressor provides the best performance during the capturing phase. The None compressor may provide the highest frame rate. If you have a fast computer, however, you may prefer using the Video compressor directly. The advantage of recording with the Video compressor is that you don't have to mess around with recompression. If you are going to give the clips to others, press them on a CD, or have serious space problems, you need to recompress the clips using the Cinepak compressor.

Several applications that capture video (for example, Avid VideoShop, Apple's Movie Recorder on the QuickTime Starter Kit, and Adobe Premiere) provide an option that automatically records using the None compressor and then immediately post-compresses using the compressor of your choice. This option saves you the hassle of manual recompression, but still requires a lot of time.

Selecting and Using Digitizing Hardware

In order to record movies you must install a video digitizing board in your computer. The board converts the analog video signal to a digital image. The following sections describe some of the video digitizing boards currently available for Macintosh users, and explore the issue of hardware acceleration.

Video digitizing boards

You can use the following overview as an information resource when you select your video digitizing board. The overview covers only some of the digitizing boards currently on the market, but can show you what features to look for in selecting the right board for your system.

SuperMac's VideoSpigot

This low-cost board became tremendously popular soon after its release. VideoSpigot's low cost and reasonably high quality of captured sequences makes the board popular with both beginners and seasoned professionals. The board's initial popularity was boosted by the fact that the first VideoSpigot boards included a copy of Adobe Premier at no extra cost!

Included with the VideoSpigot is a special application called ScreenPlay, which uses SuperMac's own compressor to capture movies (see figure 10.4). This application provides much better performance than any of the built-in compressors, and captures at much higher frame rates. A IIci, for example, captures at about 12 frames per second (**fps**) with a RasterOps 24STV, and over 20 fps with a VideoSpigot.

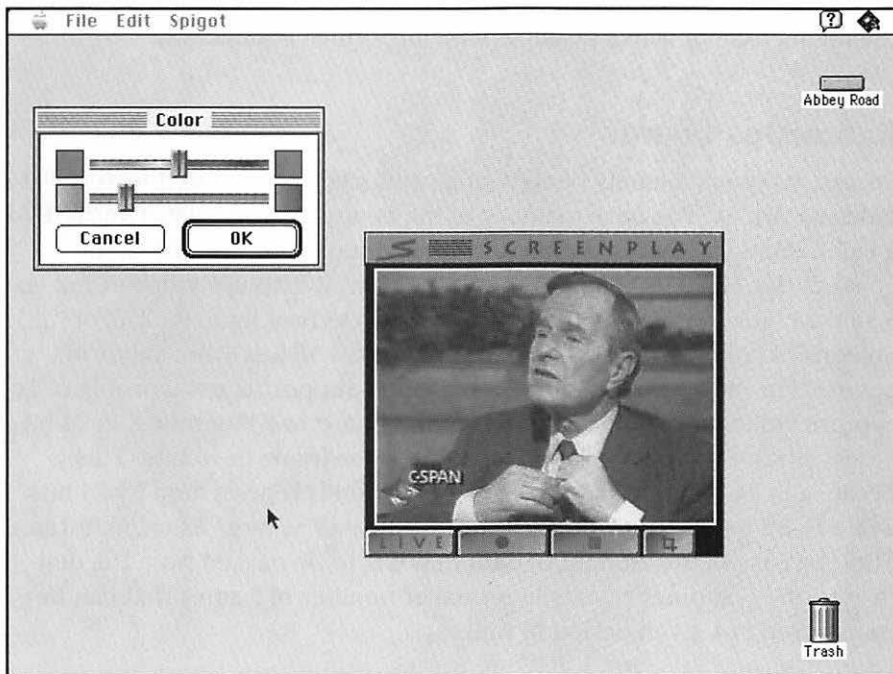


Figure 10.4. The SuperMac VideoSpigot ScreenPlay Application.

The VideoSpigot board comes in two versions: a NuBus board for regular Mac II-class machines, and an LC version that plugs into the PDS slot of the LC. Those users with one foot in the Windows world can purchase a PC version of the VideoSpigot.

If you are considering buying a VideoSpigot, be aware that although the board is very good for small frame capture (anything less than 1/4 screen, 320 pixels by 240 pixels), it does not perform as well with larger frames. This limitation becomes an issue only if you want to capture large single frames from videotape—otherwise, you are unlikely to notice a difference in your movies. Another limitation of the VideoSpigot board is that it has limited controls for brightness and contrast. These controls are particularly important when your source material is poorly or unevenly lit. Further, the VideoSpigot sometimes puts black pixels in extremely bright (saturated) areas when recording from tape. You may or may not encounter this problem—the source material and source device seem to be the determining factor.

Finally, the VideoSpigot only has an RCA video plug-in, so if you have a VCR with S-Video out (either Hi-8 or S-VHS) you cannot take advantage of the higher quality signal possible with an S-Video connection.

RasterOps boards

RasterOps offers a family of digitizing products with different features at different prices. The base member of the family is the 24STV. The 24STV is a video digitizing board combined with a video display board. If you connect the 24STV to your computer monitor, it displays video in 1, 2, 4, 8, and 24 bits; that means the board displays in two, four, 16, 256, or millions of colors. The 24STV does not support 16 bits (thousands of colors). For most applications, lack of 16-bit support is not a problem. To capture video, however, you must set the board to 24-bit mode. In 24-bit mode, a frame of video is larger than the same frame in 16 bits. This is because in 24-bit mode, the color of each pixel is represented by 24 bits while 16-bit uses 16 bits. So a 24-bit frame is half as large as a 16-bit frame. This increase in the amount of data that has to be passed from the digitizer to the computer results in a smaller number of frames that can be transferred in a given period of time.

A Ilci can capture at the rate of about 12 fps with a frame size of 180 by 160 using the 24STV board. Although the 24STV frame-rate performance isn't equal to that of the VideoSpigot, the quality of its captures is much higher. The improved quality is particularly noticeable if you are capturing large frames. Also, the 24STV brightness and contrast adjustments are much wider than VideoSpigot's.

The 24STV performs best with either a very fast Macintosh, or in non-real time capturing (see the following Note). The 24STV can display a window with video in it. When you want to display video from a laser disc during a presentation, this board is ideal.

Included with the 24STV is the MediaGrabber application, which you can use to capture still images and QuickTime (see figure 10.5).

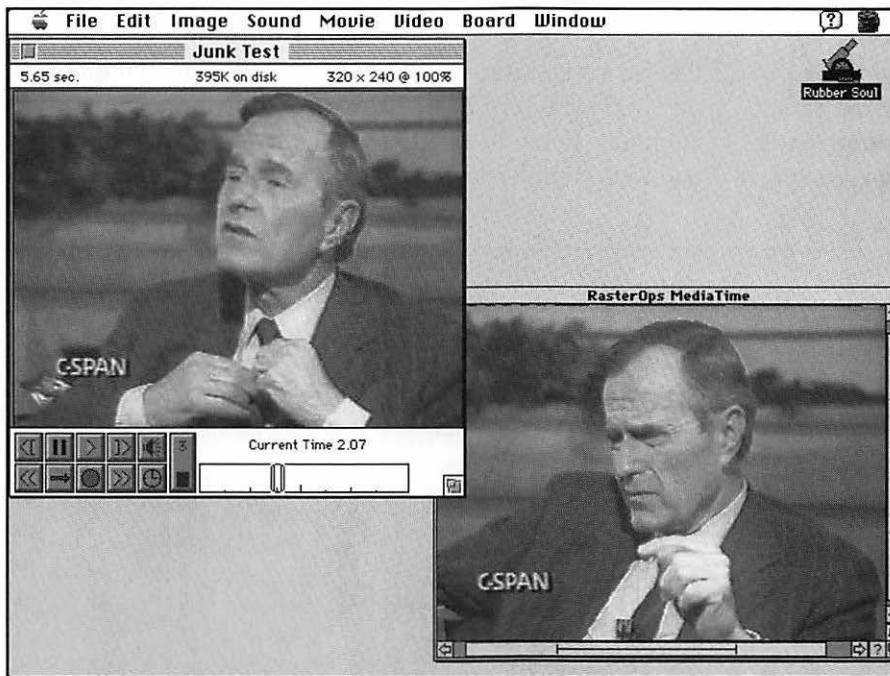


Figure 10.5. MediaGrabber, the digitizing application included with the RasterOps digitizer.

The 24STV is a good general-purpose board, but you also may consider the RasterOps MediaTime board (see figure 10.6). The MediaTime board has the same video circuitry as the 24STV, but adds the audio circuitry from an AudioMedia audio board and provides support for CD-quality audio. For a little extra cash, you can get the 24MXTV. Its functionality is similar to that of the 24STV, but the 24MXTV also has graphics acceleration and its display circuitry works in 16 bits as well as 24 bits. Capturing in 16 bits accelerates the movie capture because the amount of data being captured is smaller.

All these boards can be accelerated by the addition of a plug-in daughter board called MoviePak. This board has hardware compression which can compress the video image much faster than the software compressors can, making it possible to do full-frame video.

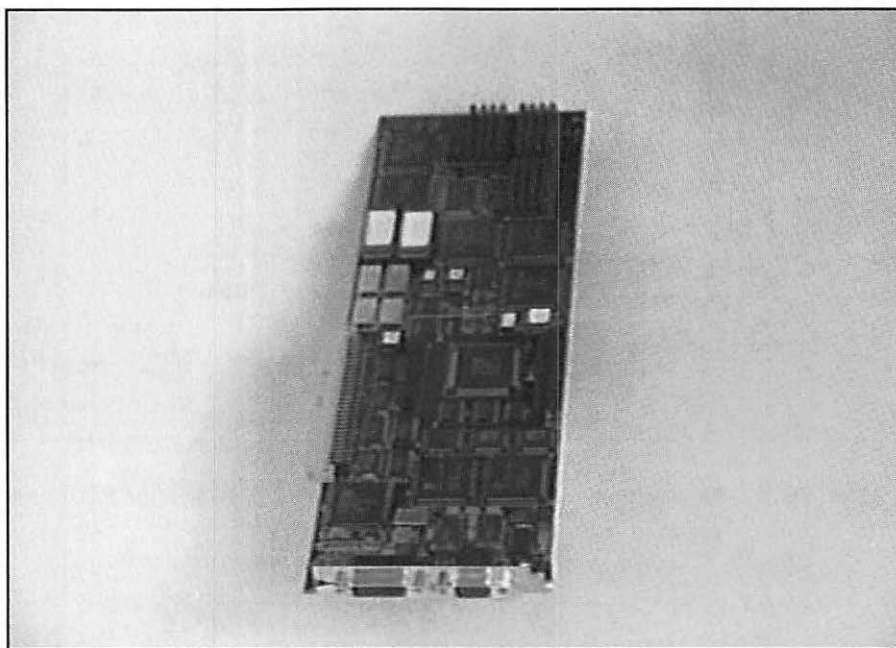


Figure 10.6. The RasterOps MediaTime board.

Note

When a compressor is unable to compress and save a frame before the next frame in the video must be compressed, you can use one of two methods to capture in non-real time. Both methods require special software.

Multi-pass capturing plays the tape at the usual speed and captures as many frames as possible in a single pass, and then performs additional passes until all frames are captured. As an example, if the compressor can only capture one in six frames per pass, the compressor captures frames 1, 7, 13, and so on in the first pass; the tape rewinds, the compressor captures frames 2, 8, 14, and so on. The process repeats until all frames are captured. The advantage of multi-pass capture is that it doesn't place any extra stress on the videotape. The disadvantage of this method is that it requires a professional VCR with time-code support in order for QuickTime to know which frames have been captured.

Step-and-grab capturing places the VCR into pause mode, grabs the frame, then steps the VCR forward to the next frame and grabs that frame. This method places more stress on the videotape and requires a VCR with a clean pause. The method does not require time-code support, however, so you can use a less-expensive VCR.

Radius VideoVision

The Radius VideoVision costs twice as much as the RasterOps 24STV, but offers more features. The VideoVision has a built-in 8-bits sound digitizer (see Chapter 11, "Sound"). The Radius VideoVision enables sound mixing, so you can mix sound from the Mac and from an external source. The board has a built-in video encoder to enable you to record to videotape, and it has a function called **convolution**—a special filter that improves the quality of Macintosh graphics when recorded on videotape.

As a video display and video digitizing card, the Radius VideoVision rates about the same as the 24STV, but has slightly better quality. Capture rates are about the same in both boards. If you need the video output capability, consider the VideoVision; otherwise, you may want to go with a cheaper alternative. (But read the following section on hardware acceleration before making your final decision.)

The VideoVision Studio board is a plug-in board for the VideoVision which provides hardware compression, making it possible to compress full-frame video.

Hardware acceleration

As you can see from the preceding discussion, most existing digitizing boards, coupled with QuickTime, are capable of only very low frame rates at small frame sizes. How are you going to see *Saturday Night Live* at full-screen size and at high frame rates?

The current answer to this problem is the use of hardware acceleration of the compression algorithms. This involves adding a plug-in board that is specially designed to run compression algorithms and can increase the capture rate and frame rate.

RasterOps offers a plug-in board called MoviePak that plugs into several of the RasterOps boards, including the 24STV, MediaTime, and 24MXTV. Radius offers VideoVision Studio, a daughter board for their VideoVision product.

The benefit of these systems is that they make it possible to capture larger images, but these systems are not perfect—yet.

To capture full-frame (640 by 480) at 30 fps with these boards you need a very powerful Macintosh—they require a lot of horsepower and fast hard disks to work properly. The movie files are very large, and a second of video can be between 20 and 60 MB! Currently, the quality of movies produced with these boards barely rivals the quality of VHS video, so the boards aren't suitable for broadcast video applications. Another disadvantage is that others can't play these movies without the hardware boards, and the boards are expensive.

As a rule, you need hardware compression only if you want an off-line editing system, or if you want to capture large frame-rate and frame-size movies, reduce the size and frame rate, and compress them with the Compact Video compressor.

Other compression algorithms

While QuickTime comes with a collection of standard compression algorithms, other algorithms can be added. This is done by installing an extension (a small file which contains the algorithm) into the System folder. In the future other compression routines will probably be added to QuickTime.

You may encounter the following algorithms in the near future:

- **INDEO:** This algorithm was developed by Intel, and is available for the PC and Macintosh. Since it is not bundled with QuickTime it has not yet found wide acceptance. Performance and usage of this compressor is similar to that of Cinepak.
- **MPEG:** This is an “industry standard” algorithm. The algorithm was developed and has been adopted by a large number of companies. While MPEG was originally conceived to be simply the motion picture equivalent of JPEG, interest in Interactive Television and media has resulted in MPEG becoming the de facto standard for these systems.

While the performance of MPEG is very good, to play back at full-frame rates still requires special hardware. Creating MPEG sequences also requires special hardware. (Currently most sequences are sent to service bureaus and compressed on large computers because of the time it takes to compress these sequences.)

QuickTime 2.0 offers support for MPEG, but, in fact, without a special plug-in board (Apple is supposed to be selling one soon at less than \$500), you cannot play an MPEG movie. Apple may offer a software playback compressor for MPEG in the future. Similarly, one developer has already announced that it will be offering an MPEG compression option for the Macintosh which will cost considerably more. For these reasons, MPEG remains only suitable for use in specially dedicated hardware situations.

Audio

QuickTime’s support for audio is important. The synching of audio to video is vital for realistic playback. But there are advantages to using QuickTime for audio tracks—it’s possible to have a QuickTime movie that just contains audio. When you open such a movie, all you will see is the

standard QuickTime controller. You can play the audio track, and use the controller to jump around within the audio (something you can't do when sounds are stored as Resources—see Chapter 11, “Sound”).

QuickTime also provides support for importing Audio CD tracks and MIDI files and converting them into QuickTime movies (see Chapter 11, “Sound”).

QuickTime VR

Apple has recently released QuickTime VR (Virtual Reality) to software developers, but not to the general public. QuickTime VR adds a level of interactivity to QuickTime movies. The user can explore what seems to be a three-dimensional world by clicking on the QuickTime frame. This capability comes at the expense of movie size—every possible angle must be stored as a frame in the movie. (QuickTime doesn't generate the frame from a 3D model the way a product such as Virtus does.)

There's an advantage and disadvantage to this method of providing a virtual reality experience. The quality of the image can be very high; playback performance is also high because the frames are already pre-computed, but the movies are also large.

Using QuickTime Movies

There's more to QuickTime than just capturing video to a QuickTime movie. Once a movie has been created, it can be played in a number of applications. There also are several tools available for editing QuickTime movies.

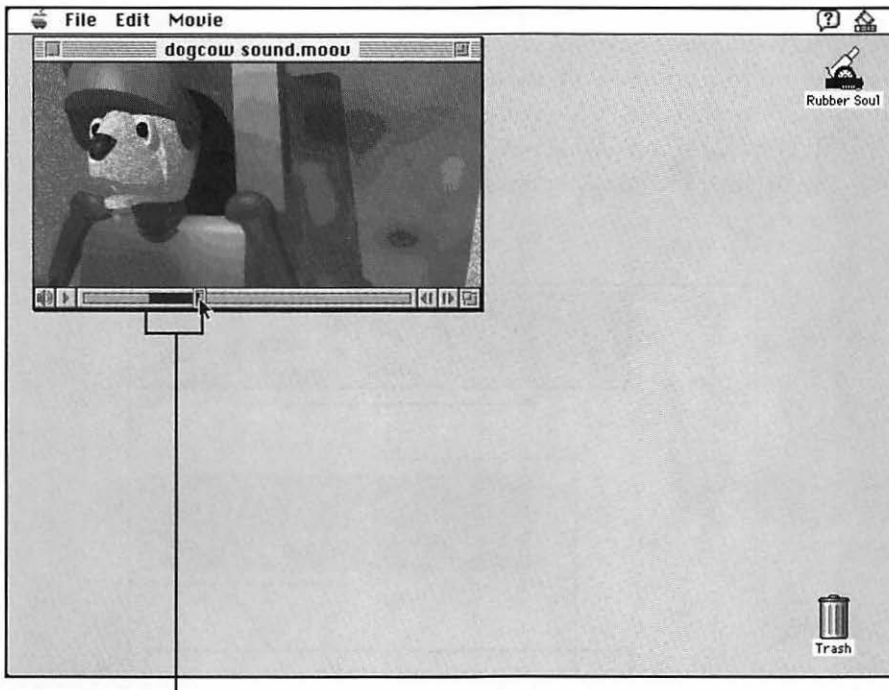
Play them

There are a number of applications that play QuickTime movies, and most developers are adding support for QuickTime to their applications where it is appropriate. These programs include Director (see Chapter 15, “Authoring Environments,” for more information on Director), Action! (an interactive presentation program from Macromedia, described in Chapter 14, “Interactive Presentations”), Special Delivery (an interactive presentation program from ITC, also described in Chapter 14), and Persuasion

from Aldus, to name just a few. Okay, let's not forget MovieWorks, Cinemation, and the flashiest of all programs, Microsoft Word! Also, Apple offers Movie Player, an application for playing movies that is on the Apple QuickTime Starter Kit.

Edit them

Editing is the fun part! Several applications enable you to edit QuickTime movies. Some of the programs mentioned above supply some editing capabilities. For example, MoviePlayer—the player application included on Apple's QuickTime Starter Kit—enables you to select a range of frames in a movie, cut or copy the range, and then paste them into another movie or somewhere else in the same movie (see figure 10.7).



Range of frames selected in the movie.

Figure 10.7. Selecting a range of frames in MoviePlayer.

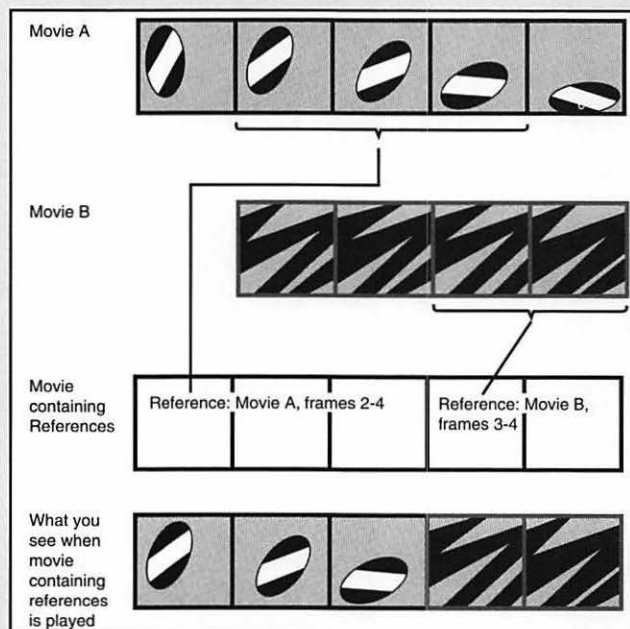
If you want to do more sophisticated things, such as adding special effects, transitions, or titles, you need to investigate one of the QuickTime editing applications.

Note

Before proceeding further, we need to talk briefly about references. QuickTime movies can be large. Even with compression, a typical movie clip can be many megabytes; copying and pasting files of that size can become painful, difficult, or even impossible. Apple came up with a solution. When you select part of a movie and copy that movie clip to the Clipboard, the system copies only the first frame in the clip sequence and some pointers to the selected data in the original file. The pointers act like an index to the clip.

When you paste the clip into another file, the system only inserts the pointers to the original file. If you paste into a program that doesn't support QuickTime, you get only the first frame of the sequence.

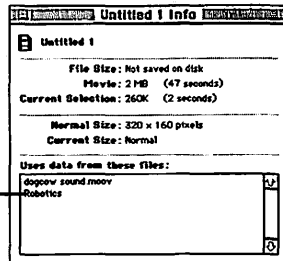
As an example, when you make changes to a movie in MoviePlayer and then select Save As, a Save dialog box with extra options at the bottom appears. These options enable you to save the movie as a reference file (which still contains the pointers to the original file) or as a stand-alone movie (which copies the information from the original file into the new file). This figure shows how references work.



Using references saves disk space because it prevents multiple repetitions of information. If you give the new file to a friend or coworker, however, you also must give him a copy of any referenced file or he cannot play the movie!

Some programs use references, and some don't; you must be familiar with how a program works. And not all programs display the references used in a movie the way Movie Player does. When you create a new movie using editing programs such as VideoShop and Premiere, the programs usually give you the options of using references or creating a stand-alone movie. Others give you no options—you must experiment with the program or read the manual to determine if the program enables references.

The untitled movie contains sections from these movies.



Using Programs That Support QuickTime

This section presents a brief overview of some of the hundreds of programs that support QuickTime. The overview doesn't cover all the programs that can play QuickTime movies (like Microsoft Word, WordPerfect, or Persuasion) because that information is too lengthy for this chapter, and the individual details of these programs are not very interesting.

Utilities

There are two screen capture utilities that record activities on the screen as QuickTime movies and then enable you to play them back. The programs, Spectator from Baseline Publishing, and CameraMan from Motion Works, offer similar functionality. Both can capture either the whole screen or part of the screen to a movie, and both enable you to add sound afterward by recording a narration as the movie plays. Figure 10.8 shows the CameraMan Settings screen and its options.



Figure 10.8. Options for recording a sequence in CameraMan.

CameraMan and Spectator are useful for creating training movies, but have many other uses, as well. One of the more imaginative applications of these programs used PageMaker to manipulate a scanned image and then recorded the process in CameraMan.

When using these programs, be aware that attempting to capture a full-sized screen in color results in very low frame rates—perhaps only 4 fps or less. Reducing the area to be captured improves the performance. Alternatively, working in 1-bit color rather than 8-bit (if you can) makes much smaller movies. If you use 1-bit color, you may be able to fit a one-minute movie on a single floppy disk.

Both programs support sound. They will add sound effects representing mouse-clicks, as well as enable you to record a narration to go with the movie. Figure 10.9 shows sound in CameraMan.



Included on the CD-ROM is an application called MoviePlay, which is supplied with CameraMan. MoviePlay is a basic QuickTime movie-playing application. Several of the movies on the disc demonstrating other products were created using CameraMan.

Motion Works is releasing CameraMan as part of a collection of multimedia utilities called—cleverly enough—Multimedia Utilities.

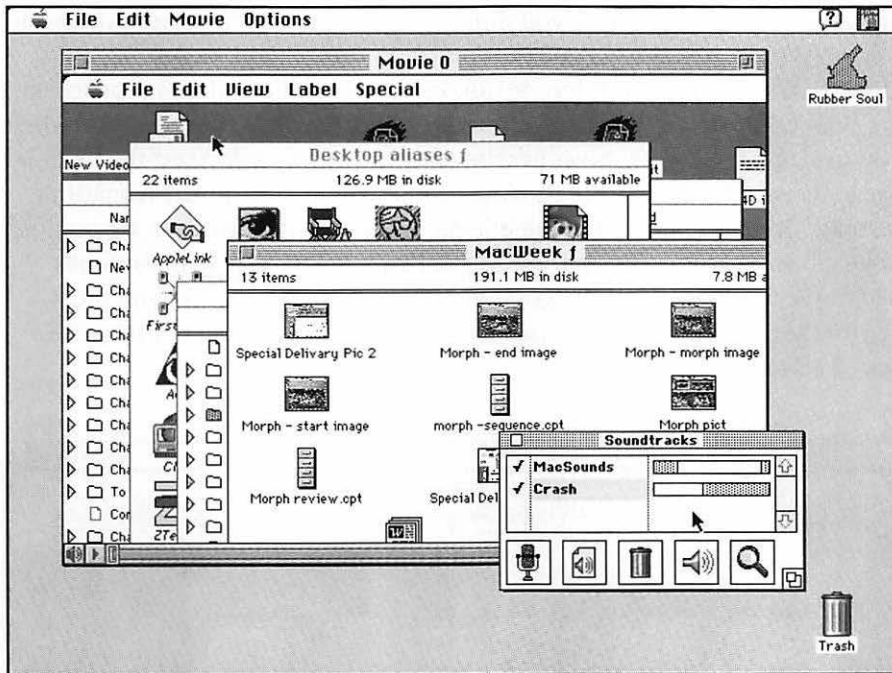


Figure 10.9. Adding sound to a CameraMan movie.

Editing

If you are seriously interested in creating and working with QuickTime movies, you may benefit from purchasing a QuickTime editing application. Two such editing programs are currently available: Adobe's Premiere and Avid's VideoShop (see figures 10.10 and 10.11). With either of these programs you can edit QuickTime movies, apply effects, add transitions, and add titles.

The two programs target slightly different markets. Premiere aims at the high-end editing market, and VideoShop at the general market. Premiere supports time code, for example, and VideoShop requires a plug-in external module to enable time-code support.

These editing applications have other differences you should consider. Premiere simulates (in a Preview window) the playback of the movie you are editing; VideoShop generates the effect as you add it to a movie, so that you can see the effect immediately. What does this mean? When you

add an effect in VideoShop, you must wait a few seconds or minutes while VideoShop creates the frames that make up the transition. Adding an effect to the score in Premiere is quicker because it requires no processing time to generate the effect. Premiere can't play the effect for you when you preview the movie, however, because it hasn't performed the calculations necessary to generate the final frames that make up the transition. Instead, Premiere simulates the effects in the Preview window. If you don't have a fast machine or you want to see exactly what a transition looks like, the Preview may not be good enough. If that is the case, you must create that portion of the movie anyway, and thereby negate Premiere's speed advantage.

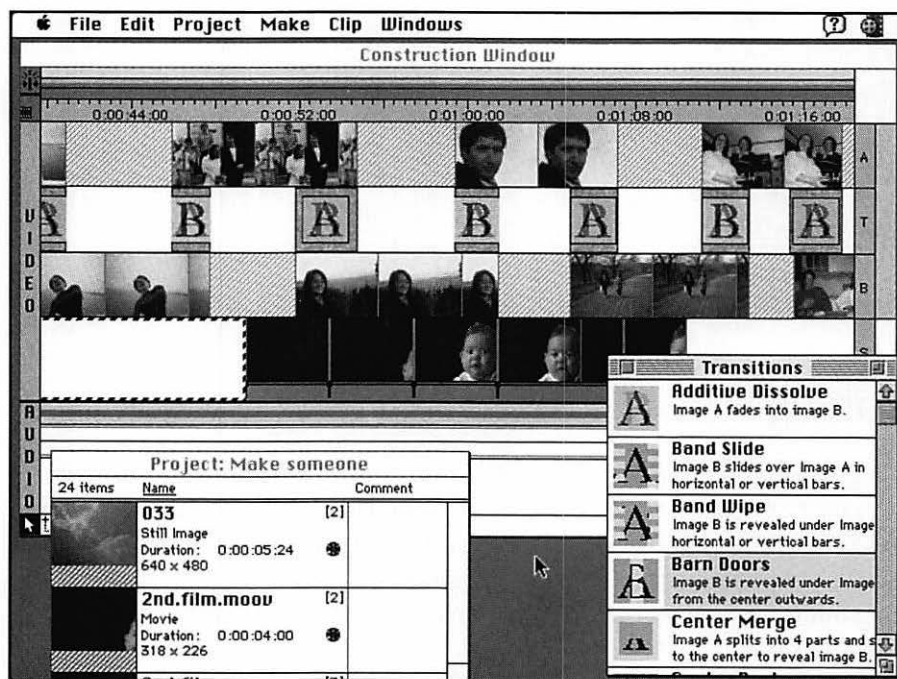


Figure 10.10. The Premiere editing application.

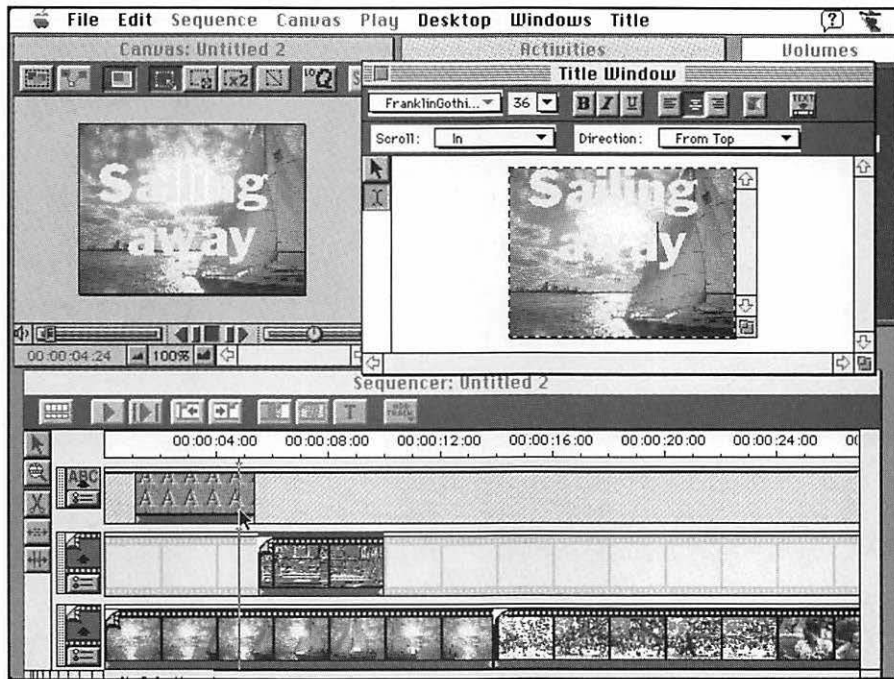


Figure 10.11. The VideoShop editing application.

Post-processing tools

Post-processing tools are advanced tools that really are intended for adding special effects to clips. VideoFusion (see figure 10.12) and After Image are two such programs that work with QuickTime. They both offer a large collection of effects, many of which aren't currently available in VideoShop or Premiere. Both Premiere and VideoShop offer some effects, however, and most people need to get an editing program before they get something for processing clips. Therefore, check out the basic programs and see what you really need before making a selection.

VideoFusion provides a spreadsheet-like interface into which clips are pasted and then selected when applying effects.

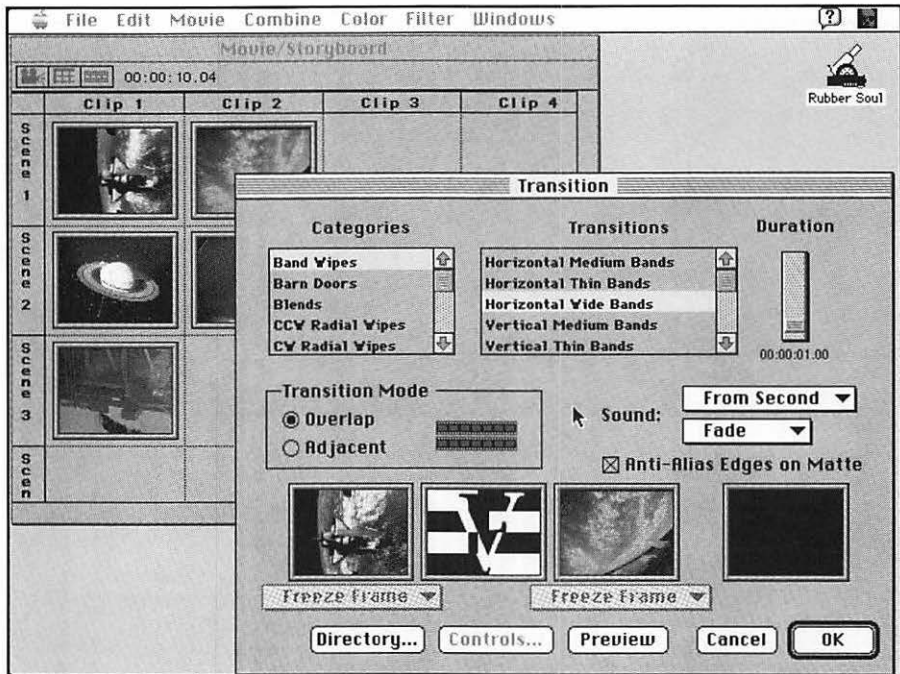


Figure 10.12. VideoFusion.

After Image, from Aldus, is particularly suited to creating animation of images and video sequences. Aldus also offers After Effects, which is essentially the same program, but which can support much larger image sizes than After Image.

Video Graffiti, from Neil Media, actually lets you paint onto a QuickTime movie. You can do this frame by frame, or while the movie is running “live.” While Video Graffiti is an impressive program, you need a powerful computer to really make it work.

It is possible to export clips from Premiere in a Filmstrip format, which can be read by Photoshop. You can then paint on the individual frames. Unfortunately, the Filmstrip file format is not compressed, so the size of the files becomes very large, making it awkward to work on long sequences.

Sound Edit 16 enables you to open QuickTime movies and then edit the movie sound tracks.

Cross-platform Applications

This chapter cannot end before mentioning that Apple has released QuickTime for the PC. QuickTime for Windows will play QuickTime movies (created on the Macintosh) on a PC running Windows. Although QuickTime for Windows currently doesn't enable you to do all the Macintosh editing functions, future versions of the program may offer more editing capabilities.

Microsoft has released a product called Video for Windows that provides many of the features of QuickTime. You can use Video for Windows to create and edit video (on the PC); the product even includes a converter that converts QuickTime movies to the Video for Windows format.

It makes you wonder: If it weren't for Apple, where would PC developers get ideas for new products?

The Future

Apple is enthusiastic about QuickTime and continues to improve the product. You can expect to see new compressors and hardware support (both from Apple and third parties). You also can look for the addition of support for interactivity within movies—enabling you to jump to another part of a movie by clicking within the movie, for example. New computers with built-in support for QuickTime acceleration are likely to appear. And, you can expect the frame size and frame rate of movies you can record and play to improve until they rival television. With these advances, the difference between a computer and a TV/VCR will diminish—and we may get less and less work done, as well.

But that's all in the future, and until then, it's time to get on to the next chapter and worry about what we hear when we watch video. *Saturday Night Live* may be fun to watch, but you need sound to hear the punch line (see figure 10.13).

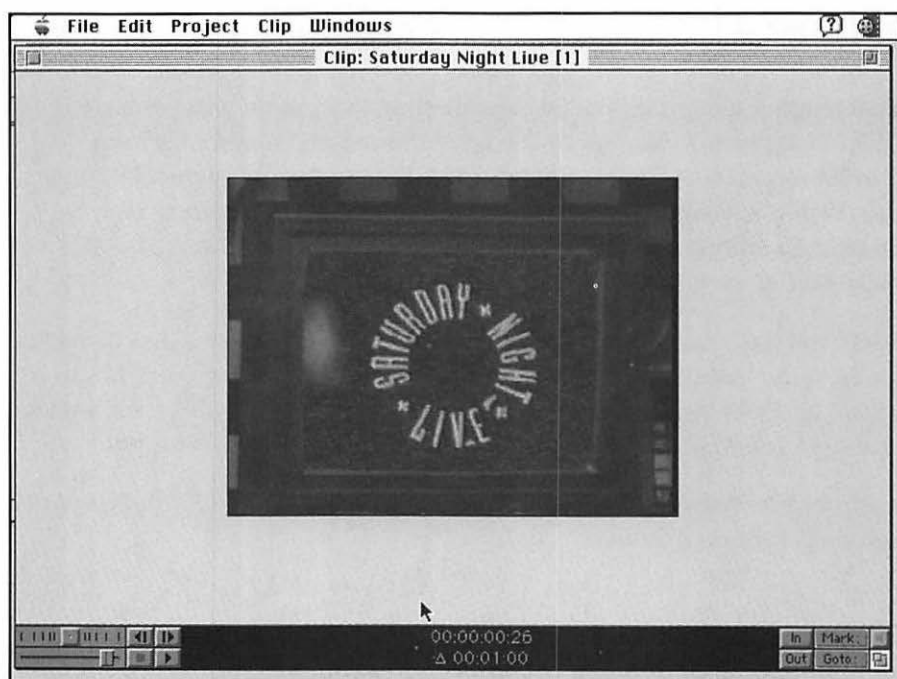


Figure 10.13. *Saturday Night Live* on my computer.

Sound

11

chapter

Perhaps the most overlooked aspect of multimedia is sound—audio. It's so easy to get wrapped up in the visuals that you completely overlook the quality of what you hear.

Not sure if your speaker is the problem? It's very rare for the Macintosh speaker to fail—try restarting the computer first to see if that fixes the problem. If the speaker still doesn't work (and you have checked the volume in the Sound control panel), then try plugging headphones into the sound jack in the back of the Macintosh.

And yet the Macintosh has a tremendous advantage over the millions of PCs out there. Every Macintosh comes with a small speaker and the System software to play sound. It means that you can have sound in any multimedia presentation or game without having to worry whether the user will be able to hear it. Short of a malfunctioning speaker (which has been known to happen), the sound will play.

Storing Sound

It's important to know a little about how sound is stored and used on the Macintosh—for example, its limitations and how it can be improved.

Macintosh built-in sound hardware

The hardware built into all Macintosh models supports at least 8-bit sound sampled at 22 kilohertz (kHz). Many new Macintosh models (including all AV and Power Macintosh models) support 16-bit sound sampled at 44.1 kHz. These two numbers—the number of bits used per sample, and the frequency—have a direct relationship to the quality of the sound that a Macintosh can play. Remember that computers work with numbers. They're digital. Sound is analog—a continuous sound wave. To “record” a sound, the computer measures the level of the sound and converts it to a number. This process is repeated many times a second, and these numbers are then used to recreate the sound during playback.

The kilohertz represents the frequency (how often) a sample is taken. Hertz means “per second,” so 5 hertz would be five times a second, and 22 kilohertz is 22,000 samples per second (which seems like a lot, but you will soon see that it's not as good as it could be). The bit is the accuracy with which the sample is stored. With 8 bits, a computer can store 256 different values, so from the lowest to the highest point in the signal there are only 256 steps. With 16-bit sound there are 65,536 values from the lowest to the highest signal. CDs store sound at 16-bits, 44.1 kHz. Figure 11.1 shows how the analog wave form is sampled and converted to digital information.

One final issue is whether the sound is mono or stereo. Obviously, a stereo sound is double the size of a mono sound. New Macintosh models support stereo recording and playback, but some older models, while they will play a stereo sound, actually output it as a mono sound.

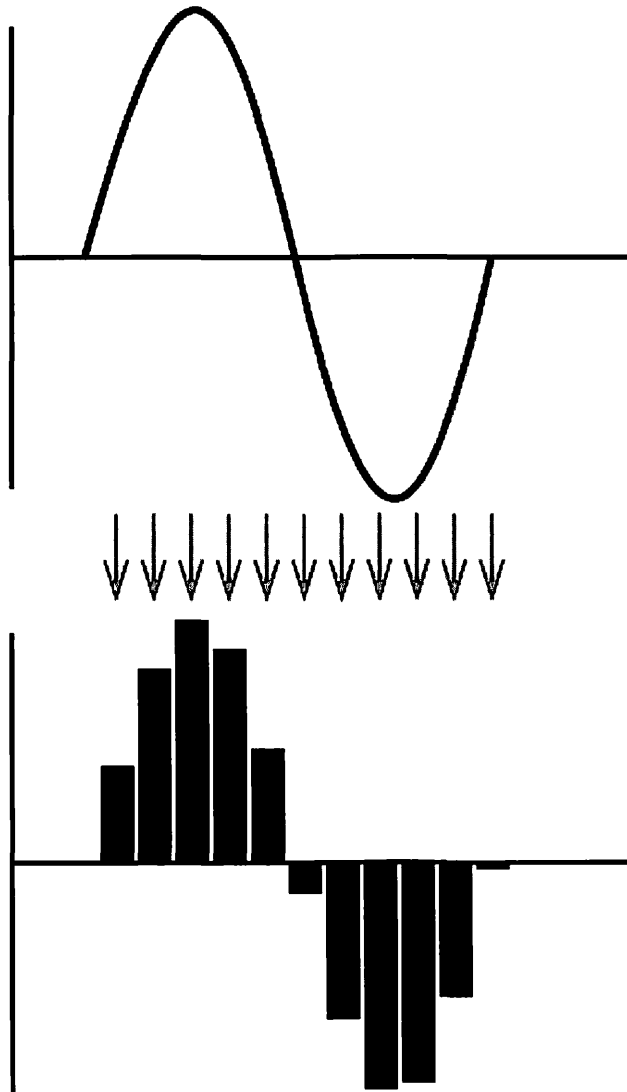


Figure 11.1. The Macintosh represents analog sound waves in a digital form.

Macintosh sound quality

A 16-bit sound at 44.1 kHz is of CD quality, so obviously, if your equipment supports it, that is the ideal way to store a sound. However, while many new Macs (including Power Macintosh models—the 6100, 7100, and 8100) support 16-bit sound, the vast majority of current Macintosh models do not. Also, the size of the files is an issue. A CD-ROM, which can hold about 600 MB, can only hold a little more than an hour of 16-bit, stereo audio (double that if you record in mono). For these reasons, it still pays to use sound at lower bit depths.

An 8-bit sound at 22 kHz doesn't sound too bad when played through the internal speaker of your Macintosh, but sampling noise will be evident if you listen using headphones, or if you plug external speakers into the headphone jack at the back of the Macintosh. This manifests itself as low-level noise heard in the background, and it is most noticeable in quiet passages. If you are going to play the sound loudly through large speakers, definitely consider using 16 bits if you can.

The Macintosh internal hardware is not capable of playing sounds at rates higher than 22 kHz at 8 bits, or 44.1 at 16 bits. You can record at lower sampling rates (5, 7, and 11 kHz being the other choices). These are all 8 bits, but the lower sampling frequency means that the number of samples is smaller and the size of the sound file becomes smaller. Of course, the quality of the sound decreases, too.

Lower sample rates

The biggest reason for sampling at lower rates is that the size of the sound file is reduced. A ten-second sound at 22 kHz may take up 224K, but the same sound at 11 kHz is 116K, and at 7 kHz it is 77K. This is particularly important if you want to use a lot of sound and distribute it on a floppy disk. But you also must consider the quality of the sound that the listener will hear. As the sampling rate goes down, the quality of the sound also deteriorates.

For most productions I use either 22 kHz or 11 kHz. I frequently use 11 kHz because the difference between 22 kHz and 11 kHz will not be noticed by most listeners when they are using the Macintosh internal speaker. They will notice the difference if listening through headphones or playing

external loudspeakers. If the sounds are soft with a lot of quiet passages, then 22 kHz will sound much better than 11 kHz. The other two sample rates (7 and 5 kHz) introduce a lot of noise and should be used only in desperation or for a particular effect. Some people think 5 kHz sampling sounds like a telephone conversation. I think it sounds a lot worse, but that's for you to decide.



The CD-ROM contains a demo program called Sound Quality that contains the same sound sampled at different qualities. You can play the sounds and decide which sound quality you prefer.

Sound compression

You can use a form of sound compression called MACE (Macintosh Audio Compression and Expansion) to make your 8-bit sounds smaller. The samples are made at 22 kHz, but the values are compressed when they are stored in the file, and then decompressed during playback. The compression scheme is not “lossless;” that is, the quality of the sound is not completely preserved. Also, some applications do not support compressed sound format.

Compressed sounds are smaller than 22 kHz sounds, but then so are sounds sampled at lower rates (11 or 7 kHz.) You should try compression options to see whether you prefer them when compared to sounds sampled at lower sampling rates. I prefer 11 kHz sound to the minimum compression factor (which results in sound files just a little smaller than the sound file at 11 kHz).

Recording Sound

Until the Mac LC and IIsi were released, the Macintosh did not include sound digitizing hardware. You could play sound with your Mac, but if you wanted to record sound, you had to buy an external piece of hardware. The Mac LC and IIsi changed that. They included hardware for digitizing sound at 8 bits, and they also came with a microphone. Since then, all the new Macs have come with this functionality built in, so you can start recording sound with a Mac straight out of the box, though you may have to buy a microphone. Figure 11.2 shows the microphone included with the original PowerBooks.

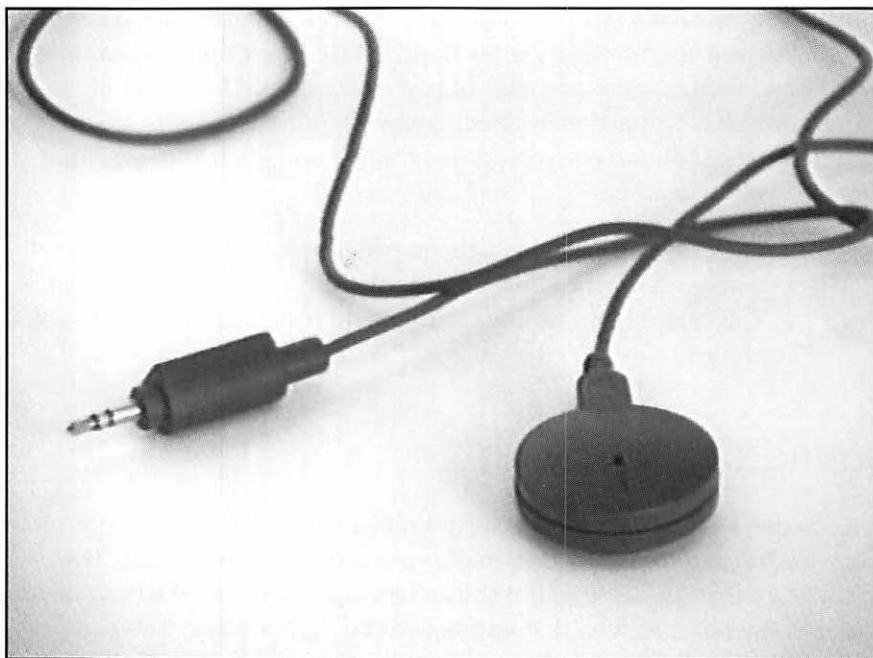


Figure 11.2. The microphone included with some Macs that have built-in sound digitizing hardware.

If you have an older Macintosh, you can still buy an external piece of hardware to enable you to record sound. You just need to buy a MacRecorder (made by Macromedia, and sold for around \$240 mail order). The MacRecorder is a box that attaches to the modem or printer port of your computer and adds the hardware necessary to record sounds. Figure 11.3 shows a MacRecorder.

Upgrade to System 7.0

If you aren't currently running System 7.0 then you really, really, really should consider upgrading. Yes, you will need to have at least 4 MB of memory, but memory is fairly cheap, and System 7.0 provides a lot of neat features (not the least of which are that you can change the icons of folders and applications easily, and you can play sounds by double-clicking them)!

Perhaps a more important reason is that more and more applications will run only in System 7.0, or at best, certain functions do not work unless System 7.0 is running.

If you absolutely, positively, can't run System 7.0 just yet, then at least make sure you are running System 6.0.7 or 6.0.8. Apple updated the Sound Manager (part of the operating system) when it introduced System 6.0.7, and several things, including QuickTime, do not work under earlier versions of the system.

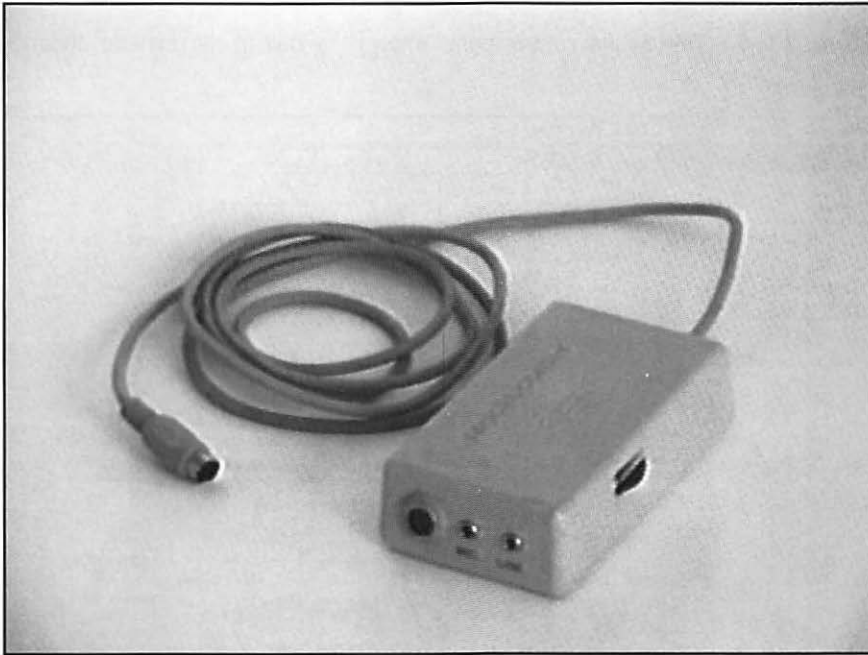


Figure 11.3. MacRecorder box.

If you only want to record new beep sounds for your computer to play, and you have a microphone or MacRecorder, then you can record the beep sounds in the Sound control panel. Along the bottom of the Sound control panel you see a list of the sound recording hardware devices attached to your computer. If your Macintosh has a built-in microphone, you will see an icon representing the microphone. If you have a MacRecorder, then you see the MacRecorder icon. Figure 11.4 shows the

Sound control panel with the built-in microphone option. Figure 11.5 shows the Sound control panel with the MacRecorder icon.



Figure 11.4. The Sound control panel shows the built-in microphone option.

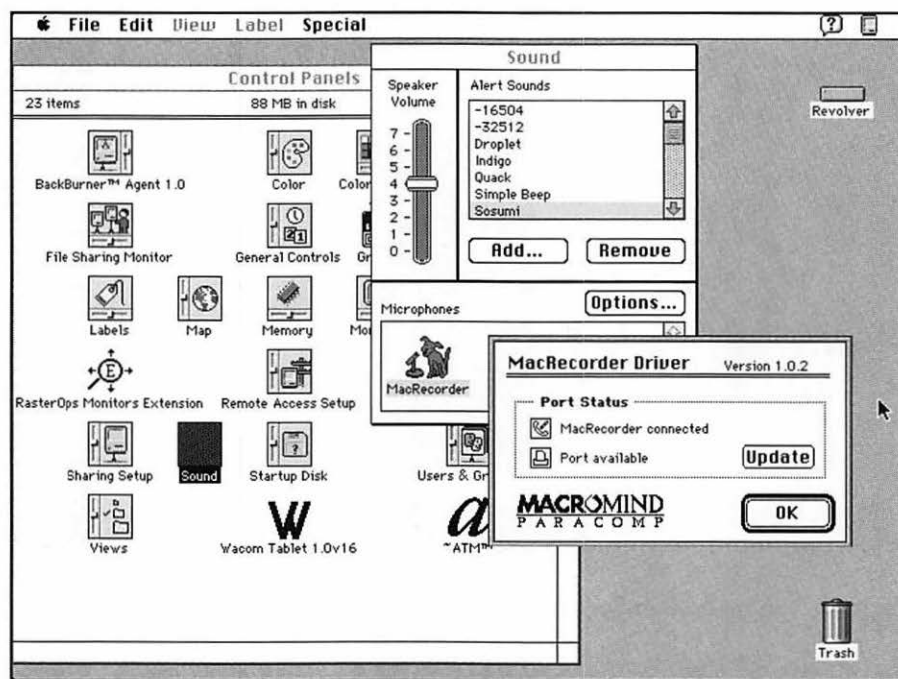


Figure 11.5. The Sound control panel enables you to choose an input sound device (in this case the MacRecorder). Clicking the Options button shows which serial port the MacRecorder is connected to.

Select the icon (if it is not already selected) and then click the Add button. A window appears for recording sounds. To record a sound, click the button labeled Record and then click the Stop button to end the recording. Click the button labeled Play to review the sound. Figure 11.6 shows the Record Sound dialog box. If you like the sound, save it by clicking the Save button; otherwise record another sound or click Cancel to close this window.

When you save a sound, the sound is stored in the System file, and you can use it as a new beep sound by selecting it in the Alert Sounds field of the Sounds control panel.

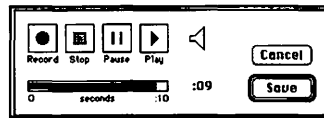


Figure 11.6. The Record Sound dialog box.

You may need to obtain the MacRecorder driver in order to use the MacRecorder with the Sound control panel. The MacRecorder driver is a small piece of software that enables the System to communicate with the MacRecorder hardware. You don't need the driver if you are using Sound Edit, but it is necessary if you want to record sound in the Sound control panel, use the Audio palette in HyperCard, or record sound in QuickTime. Most add-on sound hardware will require the installation of some kind of software driver—which is placed in the System folder—before the Macintosh will be able to work with it. The VideoVision video board, which includes sound digitizing hardware, has its own software driver.

Note

When recording sound with a computer, audio people like to refer to the process as “sampling” (recording samples of the audio). But the process is also a conversion from analog to digital, so computer people often use the term “digitizing.” Is it sound sampling or sound digitizing, and does it really matter? As long as you know what you mean, and understand when someone uses the other term, I don't think it matters which term you use.

Editing Sound

Recording sounds using the Sound control panel is interesting, but you may soon wish that you could edit the sounds. If that's the case, and you don't want to spend a lot of money, then you may consider using the Audio Control panel included with HyperCard. This is a set of XCMDs (see the chapter on HyperCard if you don't know what an XCMD is) that enables you to record sounds, perform simple editing, and save sounds in HyperCard stacks. It's useful, too, if you actually want to use the sounds in HyperCard. Figure 11.7 shows the HyperCard audio palette, and figure 11.8 shows the HyperCard Sound Editing window, available by clicking the Edit button on the sound palette.

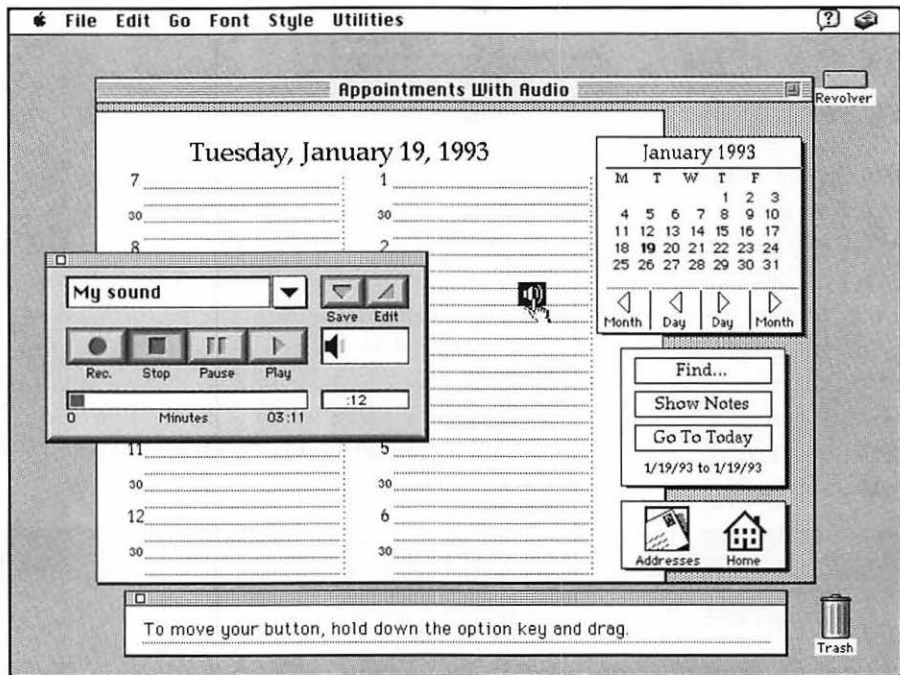


Figure 11.7. The HyperCard audio palette enables you to record and edit sounds.

The more adventurous should definitely get a copy of either Sound Edit 16 from Macromedia or AudioShop from Opcode. Even if you have a Macintosh with a built-in microphone, you should consider getting one of these programs because they enable you to edit sounds and apply effects such as echo and reverb.

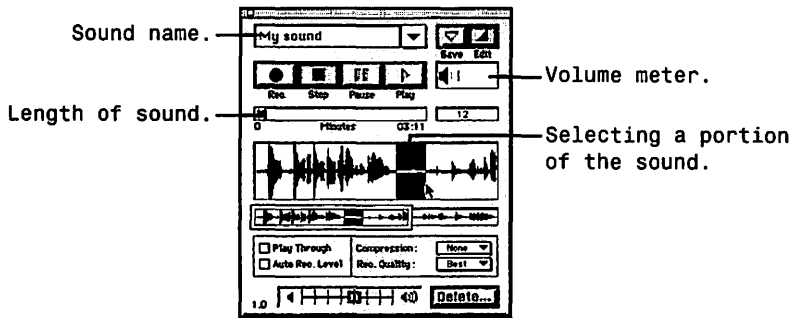


Figure 11.8. HyperCard's simple Sound Editing window.

Saving Sound and Using Sound in Other Applications

One of the most confusing things about sound is the large number of file formats in which sound can be saved. The information stored in the file formats is essentially the same (that is, the sound data), but you really need to know about these formats. When you get a sound file in one format, you may not be able to use it in a particular application.

The most common sound format on the Macintosh is a sound resource, which has a file type of 'snd.' In the Macintosh file system, a file can contain either data or resources. Resources are chunks of information stored inside a data file that are accessed separately from the data stored in the file. The resources are accessed using the Resource Manager, which is part of the operating system. It's kind of like a minidatabase (don't tell Apple I said that—one of the things Apple tells programmers not to do is use the Resource Manager as a database).

Sound editing applications like Sound Edit 16 enable you to save files as resources within files. HyperCard works with sounds only in this format. You can store several sounds as separate resources inside a HyperCard stack, and the stack will play the sounds. To play a sound, simply create a button with the script `Play 'sound name'.`

Another file format that has become a standard is the Sound Edit file. This is the native file format of the Sound Edit application, and because Sound Edit was practically the only application for recording sounds for years on

the Macintosh, this format has become fairly well known. Many programs can read files in the Sound Edit format.

AIFF is a relatively new sound format that has become popular. AIFF stands for Audio Interchange File Format, and there are two important features of this file format. First, you can transfer sounds in this format to computers other than the Macintosh. More important to multimedia developers and users is that, unlike sound resources, an AIFF file can be played from your hard disk. (Sound resources have to be loaded entirely into memory before they can be played. This not only requires time to load the sound, but it also means that you are limited by the amount of memory you have available. If you don't have enough memory, you can't play the sound). On the downside, playing sound from a disk does mean that the disk is being accessed, and this can cause problems if you are trying to read something else from the hard disk at the same time!

There are some shareware utility programs available that you might find useful for working with sound files.

Using Advanced Sound Techniques

There's more to recording good sound than connecting a microphone and clicking the record button. The following tips will help you do a better job.

Get the best sound at 11 kHz

If you decide to use sounds recorded at 11 kHz, and you are using a program like Sound Edit, then you should record the sound first at 22 kHz. Then copy the sound across to an empty 11 kHz file. Sound Edit will convert the sound to 11 kHz, and the sound is generally a little better in quality than what you would get if the sound had been sampled at 11 kHz directly. This is because the program takes two samples and averages them in the conversion process. This produces a sample that is more accurate than the single sample made when digitizing at 11 kHz.

Use a tape recorder

For best sound quality, record a sound with a tape recorder first and then digitize the sound from the recording. Do this for two reasons. First, the quality of the microphone included with the Mac and in the MacRecorder, is not that good. Second, tape is much cheaper and more plentiful than computer memory, so you can compare several takes and then record just the right one. Third, you can get the level right before recording the sound.

To record from a tape recorder to the Macintosh built-in hardware, you will need a cable and an adapter. The Radio Shack part numbers are #42-2461 attenuating cable, and #275-375 adapter.

Watch out for clipping

When recording sound, you must ensure that the sound does not clip—that the sound level is not too high. Clipping results in a very distorted sound and is usually not pleasant. If clipping occurs, adjust the level and re-record.

Avoid recording too low

Just as important, don't record with a sound level too low—to hear the sound you'll have to amplify it, and this will accentuate noise if you are recording in 8 bits. The quality of the sound is reduced because you are effectively lowering the range of the sample used to represent the sound.

So how do you get the level right? Some of the recording applications provide a visual read-out with a peak meter just like a traditional tape recorder. If the application provides some kind of meter, you need to adjust the level so that the loudest noise doesn't reach the maximum level of the meter. But don't believe the meter! Often these software meters aren't very accurate, so you should always record some sound to see if there is any clipping, or if it's too low—don't just go by the meter. If there is no meter, then you will have to experiment to see what sounds the best.

Get a good original for better sound

Although you can achieve a lot of effects using programs such as Sound Edit 16 and AudioShop, you are still limited by the quality of the original sound. Although post-processing can turn a good sound into a great sound, it's generally not possible to turn a poor sound into even an average sound. You must concentrate on getting the best quality original rather than planning on fixing it in the mix later on.

Additional Hardware

Don't have an AV Macintosh or a Power Macintosh, but still want to record high-quality sound? You can purchase a NuBus card that offers much better sound quality. DigiDesign makes the AudioMedia II card for around \$1,000. This board is capable of recording at 16 bit, 44.1 kHz. The AudioMedia board provides great quality sound as well as the tools to work with the sound files. You can even buy an application called DECK from DigiDesign that has an interface resembling a four-track tape recorder; it enables you to record and mix sounds.

You will need a lot of disk space to record sounds, and a reasonably fast hard disk as well. And if you give someone a copy of your presentation, they must have the AudioMedia card to play back the sound at high quality.

Using CD

If you have a CD player attached to your computer, you are probably wondering how to integrate the sound from an audio CD into a presentation. Until recently, you couldn't! You could put an audio CD into the CD-ROM drive, and most drives would recognize the disc and even let you "play" the disc (through an external amplifier). But you couldn't copy the files to your hard disk and then open, edit, or play them. This is because the audio CD uses a very different format for storing sound than that used by any Macintosh application.

QuickTime has changed that. Provided you have one of the new generation CD drives (the Apple CD300 or 300i), you can open CD audio tracks

and convert them into QuickTime movies. You can then use these files in any application that supports QuickTime format movies.

To do this, insert an Audio CD into the CD-ROM drive. It should appear on the desktop. Using any application that can open QuickTime movies (such as Premiere), simply choose Open and go to the Audio Drive. The tracks of the disk should be listed. Choose a track. A dialog box will appear asking you to save the file on another disk drive—you can't save on a CD-ROM (see figure 11.9).

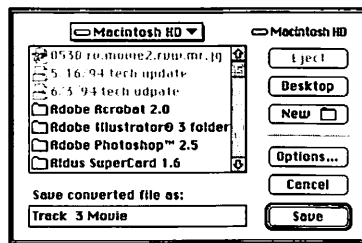


Figure 11.9. The Save dialog box.

Note that one of the buttons is labeled Options. Click that button and the Audio CD Import Options dialog box appears (see figure 11.10). In this dialog box, you can choose the sound quality, as well as the part of the track that you want to convert.

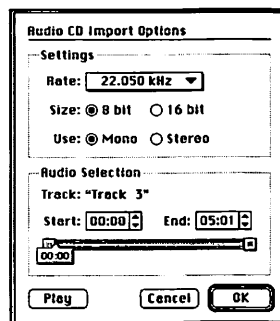


Figure 11.10. The Audio CD Import Options dialog box.

To convert just a part of the track, use the slider at the bottom of the dialog box and simply click and drag the box at each end to the desired location in the track. You can preview the track by clicking the Play button (see figure 11.11).

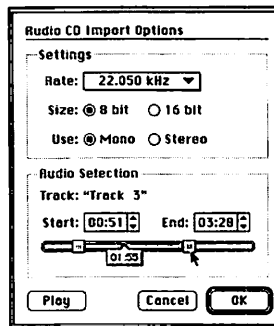


Figure 11.11. Adjusting the part of the track to be converted.

Click OK when you are happy with the selection, and then click the Save button to save the file. A thermometer window will appear indicating that the movie is being imported. The audio track is converted into a QuickTime movie which contains just the audio track. You can play the audio track in any application that supports QuickTime.

While you can digitize CDs for your own enjoyment, remember that for any other purpose you must get permission from the record company to do this (see Chapter 16, "Clip Media and Copyright").

You can get some XCMDs that will "play" a track from an audio CD when you tell it to. But the sound won't come out of the Macintosh internal speaker—you must have headphones or an external speaker connected to the CD player.

Using MIDI

This chapter is almost over, and I haven't said a word about **MIDI** (Musical Instrument Digital Interface). MIDI is a method of communicating with electronic musical instruments (for example, electronic keyboards). MIDI tells an instrument which notes to play and for how long, and also other information such as how loudly to play the notes. It's a very compact way of sending sound information from one instrument to another, or from the computer to an instrument and back again.

There are many applications available from companies such as Opcode and Passport that enable you to communicate with a MIDI device using the Macintosh. These are often called **MIDI sequencers**, and with them

you can play a composition on an instrument and record the MIDI information on the computer. You can edit the file and then play back the new sound by sending it back to the instrument. For the computer to be able to communicate with a MIDI instrument, you need a device that plugs into the Macintosh serial port and converts the signals coming from the port into a MIDI signal. How the computer communicates with the instrument is shown in figure 11.12.

Because of the compressed nature of MIDI files, and the fact that all the computer has to do is send the information out through the serial port, the computer doesn't have to do a lot of processing. It can be doing other interesting things, such as playing an animation. Several applications support MIDI playback, including Passport Producer Pro. Producer provides no editing or creation capabilities, so you will need a recording and editing application in addition to this program.

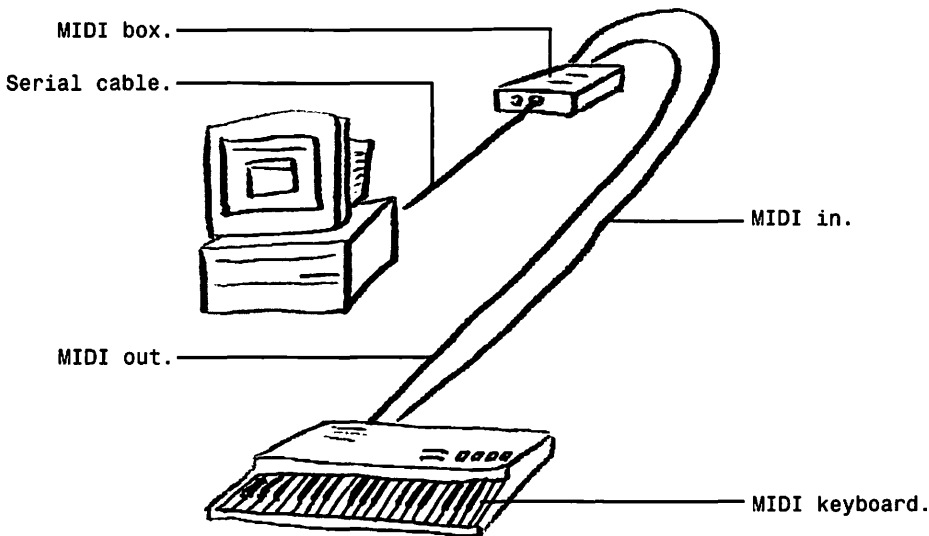


Figure 11.12. How the Macintosh communicates with an electronic instrument.

Finally, though MIDI has several advantages, it has in the past required additional hardware (the instrument and MIDI box) to play the sound.

That has changed with the addition of MIDI support in QuickTime. QuickTime can now take a MIDI file and play it back, creating the sound using synthesized instrument sounds which Apple has licensed from

Roland. QuickTime 2.0 comes with a file containing a small library of different instrument sounds.

Creating a MIDI file requires a MIDI editor. These programs are beyond the scope of this book. But importing a MIDI sound is easy, providing you have a MIDI file (and the file type must be MIDI or QuickTime won't recognize the file as a MIDI file). You open and convert the file much the same way as a CD Audio track is converted.

Using an application that can open QuickTime files, open the MIDI file. The button on the dialog box should be labeled Convert. When you click Convert, a second dialog box appears asking you to name the new file. Click the Options buttons. A new dialog box appears listing the current instruments in the file (see figure 11.13).

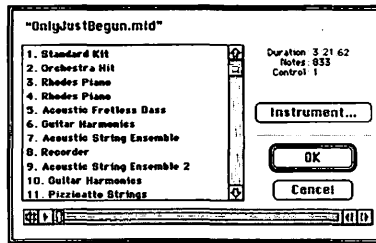


Figure 11.13. The MIDI conversion dialog box.

A MIDI sound will consist of at least one instrument track, although a complicated song will contain many tracks. Each instrument has its own track, and they are listed in the dialog box with an instrument attached to each. You can change the instrument by clicking on the track/instrument and clicking the Instrument button. A window appears that enables you to choose an instrument by category and type, and even hear the instrument by clicking on a keyboard graphic (see figure 11.14).

When you choose a new instrument, you can preview the sound by clicking the Play button and dragging the slider to the part of the song you want to hear. Once you are happy with the song, click Save and QuickTime will convert the MIDI file into a QuickTime movie, which can be used in any QuickTime compatible application.

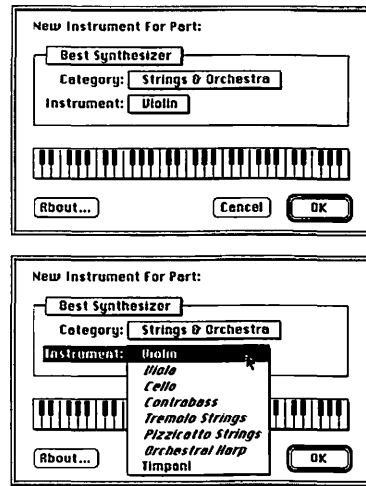


Figure 11.14. Choosing a new instrument.

Making the Macintosh Speak

With computer-synthesized speech, the computer constructs what sounds like a human voice based on English text provided to the synthesizer. It has been available on many computers for several years now. The quality of even the best talking programs does not rival the quality of the human voice, and for many applications, digitized narration sounds much better. But for some applications, a synthesized voice makes more sense. Some examples might include reading mail or reading text files that come from a database that changes continually.

For some time there had been only one application available for making the Macintosh speak. This program was called MacinTalk, and it was written for Apple back when the Macintosh was first introduced.

Apple recently updated MacinTalk with PowerTalk. Unfortunately, while it is a small improvement in audio quality, it is now a very large collection of files—and Apple is not widely distributing the product. It's something to consider for the right application, but you cannot assume that most users will have, or will want to install this software.

Looking Ahead

As more machines supporting 16-bit sound become available, users will begin to expect better quality sounding productions. Unfortunately, that means bigger files. However, the area of sound compression has been barely touched. Only a year ago Sony released the MiniDisc, which used special audio compression algorithms to drastically reduce the size of 16-bit audio files. (What used to fit on a 600 MB CD can now be squeezed onto a magneto-optical that holds only about 150 MB of data.) The compression is lossy. The sound quality isn't as good as a regular CD, but only the most critical listeners will notice under ideal listening situations. Hopefully some form of compression will soon be available so that we can better deal with these 16-bit sounds!

This chapter completes the section on the basic data types of multimedia. The next chapter looks at the programs used to bring these different data types together to form the final production.

Clip Media and Copyright

12

chapter

Whether you are a graphic designer, electronic artist, or multimedia producer, you may on occasion lack the time or skill to create all the material necessary for a production. Perhaps you lack the time to fly out to San Francisco and photograph the San Francisco-Oakland Bay Bridge, or you don't have the equipment to scan in images of different textures. For these and many other occasions, the best answer may be to obtain production material from clip media collections or stock houses.

Clip media collections originally began with the advent of desktop publishing. Several companies started selling collections of images—called clip art—that could be used for desktop publishing. Now companies are selling clip media collections—collections of sound, images, animations, and QuickTime movies.

A more traditional source for designers has been stock houses. Used by advertising agencies and corporate graphics departments, stock houses assemble collections of photographs and movie and video clips that they license for use.

This chapter discusses some production material sources. It also addresses issues of copyright and rights of ownership that apply to multimedia—in particular the necessity for getting releases from models.

Stock Houses

Stock houses and film libraries are the traditional sources for photographs and movie clips. These repositories of visual media have served the advertising and television industries for some time. Many of the generic images seen in magazines (a mountain, a running stream, and so on) are available from stock houses. Film libraries provide access to a large number of historic film clips such as the Empire State Building, as well as generic video like a busy street. Rarely are images such as clips from movies or news images available from these sources—you must go to the holders of the copyright for that kind of material.

As a multimedia producer, you can obtain production material from stock houses and film libraries. However, many stock houses have little or no experience with multimedia and may be reluctant to license their materials for such use. Further, their rates may be high—often charging per second of video—since most photo stock houses are accustomed to

dealing with advertising agencies and charge accordingly. Also, they will search and find a clip that matches your needs, rather than sell you a collection of material that may or may not be useful to you. The search and individual licensing costs more. Check the classified advertising sections of magazines like *New Media* and *Macworld* for possible sources.

Video Tape Library Ltd.
Video Tape Stock Footage
1509 North Crescent Heights Blvd., Suite 2
Los Angeles, CA 90046
(213) 656-4330

Jasmine Stock Video/Music
(310) 277-7523

Clip Media

Several multimedia clip collections are currently available. Most clip collections are distributed on CD-ROMs because they can store a large quantity of material and are inexpensive to produce. These collections often include photographs, graphics, sound, animation, and QuickTime clips—all on a single compact disc. Collections sell from less than \$100 up to \$399, depending upon the source and the type of material.

There also are several single media collections, such as background images and music discs. The arrival of Photo-CD (see Chapter 7, “Graphics”) has resulted in several discs of photographs only. These vary widely in subject matter and price.

Corel publishes a series of CD-ROMs containing Photo CD images. The price per disc is very low (under \$50!), and each disc holds 100 images. Included on each disc is a program called CoralMOSAIC which you can use to view the images. Although they are stored in Photo CD format you can open the images with almost any graphics application.

Corel Corporation
Phone: (613) 728-3733 Fax: (613) 761-9176

Jasmine Multimedia Publishing also sells a series of CD-ROMs containing photographs. They don't use Photo CD to compress the images. These discs contain a larger selection of material, but they cost more, too.

Jasmine Multimedia Publishing
6746 Valjean Ave., Suite 100
Van Nuys, CA 91406
Phone: (800) 798-7535 (818) 780-3344 Fax: (818) 780-8705

The following paragraphs discuss some of the clip media collections currently on the market.

- *ClipMedia* includes sounds, animation, and background images. The image shown in the following figure is from a catalog created for the ClipMedia disc using Aldus Fetch (a catalog program sold separately and designed for cataloging multimedia clip collections). Figure 12.1 shows some of the material contained on the ClipMedia disc. The index shown in the figure was generated using Aldus Fetch, a program that is useful for indexing a large collection of media (see the section on Media cataloging utilities). The ClipMedia collection is available in both Mac and PC versions from:

Macromedia
600 Townsend Street
San Francisco, CA 94103
(415) 252-2000

- *Multiple Media Tour* is a collection of various multimedia materials from Audio Visual Group. The Multiple Media Tour CD-ROM includes its own catalog of the source material in the collection (see figure 12.2). The catalog is available in six languages, including Japanese. Multiple Media Tour is available from:

Audio Visual Group
7 Mystic St., Suite 203
Arlington, MA 02174
(617) 646-9050

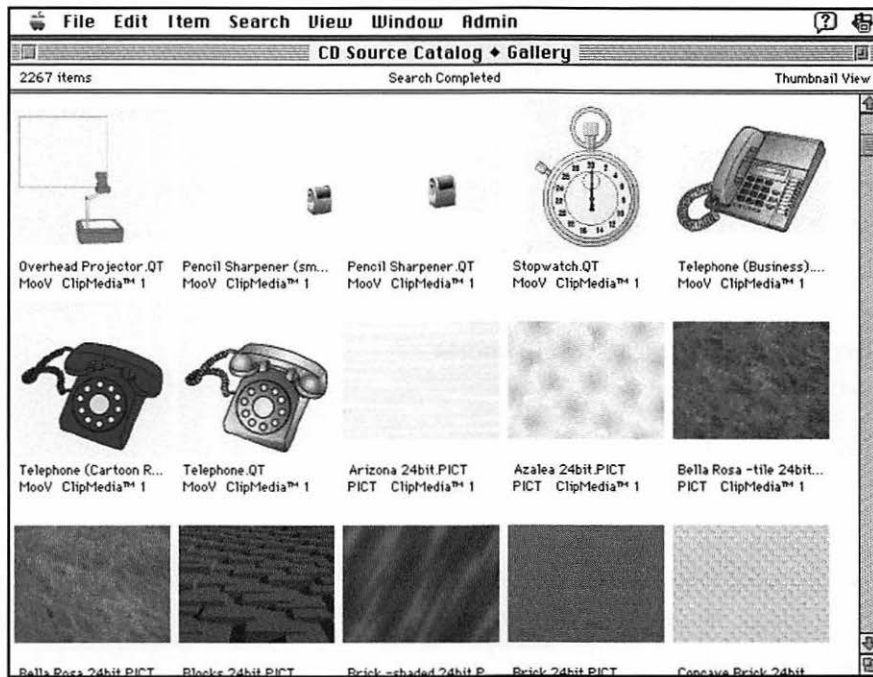


Figure 12.1. Some of the material in the ClipMedia collection.

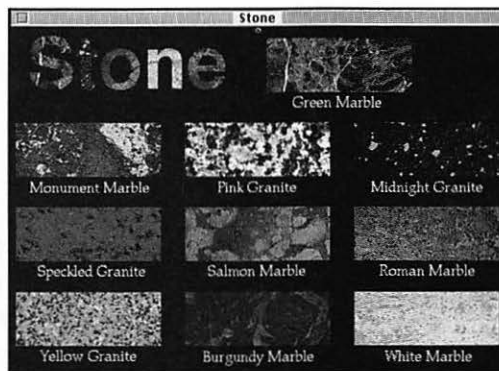


Figure 12.2. A portion of the Multiple Media Tour catalog of source material.

Other clip collections include:

Clip Time

Images for presentations.

Alpha Technologies Group, Inc.

6921 Cable Drive, Suite 100

Marriottsville, MD 21104

(301) 781-4200

Digital Video Library

Video clips of cities, places, and sports.

Educorp

7434 Trade Street

San Diego, CA 92121-2410

(800) 843-9497 (US and Canada)

(619) 536-9999 (international)

WraptureReels One

Video footage, animation, and audio tracks for presentations.

Form & Function

San Francisco, CA

(415) 664-4010

Multi Media Music

A diverse collection of music created for computerized multimedia productions.

Filler Tracks

6534 Sunset Boulevard

Hollywood, CA 90028

(800) 877-0078

Although some of these collections offer hundreds of different clips and may be very good values, to determine whether or not a collection saves you money you must consider how many of its clips you are likely to use. In some cases, you may need to use only a few clips to recoup the cost of the collection.

Licensing

An important issue before you buy a clip disc is determining what the licensing arrangements are for the materials. Some vendors offer

materials with unlimited rights. (You can essentially use the material in any production, commercial or otherwise, as long as you don't simply repackage the material as is.) Others limit the commercial uses of material. Often you can't use the material for broadcast purposes, but you can use it for any other commercial use, and some vendors sell discs which are essentially directories of their stock media collection. If you want to use the material for any purpose, you have to pay an additional licensing fee.

For this reason, you should check the licensing agreement before you purchase any clip media to make sure the license is compatible with your intended use.

Media Cataloging Utilities

Whether you buy clip media discs or just start collecting your own material, you will soon accumulate a large collection of files, and finding the clip you need will become more difficult. A number of companies are now selling utilities for cataloging the many different files and media types. Nearly all of these programs produce a catalog of thumbnails (small graphic representations of the files) that you can view, as well as some kind of search function that will search by file name, file type, or another characteristic. Some of these programs are multi-user and work over a network, so they are particularly appropriate for businesses with large collections of material spread over several machines.

Two of the programs available are:

Aldus Fetch (illustrated in figure 12.1)
Aldus Corporation
411 First Avenue South
Seattle, WA 98104-2871
(206) 622-5500

Kodak Shoebox Image Manager
Eastman Kodak Company
343 State Street
Rochester NY 14650
(800) CD-KODAK (235-6325)

Creating Your Own Clip Media

Two new programs enable you to create your own textured backgrounds and images. KPT's Bryce and Specular's TextureScape provide a method for creating an almost unlimited number of custom images. These two programs are described in more depth in Chapter 7, "Graphics." They may prove a useful alternative to clip media sources.

Copyright Laws

If you are creating a production for commercial distribution, or if you are using works created by others, you need to be aware of copyright laws and their application to multimedia.

Copyright and your work

Copyright laws extend to multimedia productions. You are wise to obtain a copyright for any multimedia production you create, even if you have no intention of selling your work to others. A copyright, to some extent, protects your work (or a derivative of your work) from being copied and sold by others. A copyright protects work for the life of the work's creator plus 50 years.

Obtaining Copyright for your work

Gaining copyright is very simple. Under current copyright law, a work is protected by copyright as soon as the work is created. You must indicate, however, that the work is protected. To indicate the copyright, place the following notation on the work in an obvious location:

© Year Name of author

For example,

© 1994 Michael D. Murie

Some minor variations within the notation are possible, and a sound work should have the ® symbol instead of the © symbol. This symbol indicates that the work is a sound recording that cannot be perceived visually.

Although you are not required to register your work with the Copyright Office, you may benefit from doing so. If someone copies your work, you must register before you can file a suit against them. Further, you receive a reduced amount of damages if your work is unregistered at the time the infringement takes place. If you anticipate any possibility of future infringement on your work, register the work with the Copyright Office.

Employee and Work for Hire

Although copyright law assumes that a work's creator holds the copyright for that work, this assumption does not hold true if you are an employee or in the case of *Work for Hire*. Work created by an employee can be copyrighted by the employer. But if you work as a freelancer, then the person who hires you does not automatically gain copyright. Instead, you retain the copyright.

If a work is created as a Work for Hire, the Work is the property of—and can be copyrighted by—the person who hired the creator. If you hire me to create an animation for you under a Work for Hire agreement, for example, you own the work and its copyright. Both artists and those who hire them need to be clear on who will own the copyright to works produced under employment agreements.

A Work for Hire must be designated as such in a contractual agreement. If the agreement does not state that the work is a Work for Hire, the work is the property of the creator. The creator can assign the copyright to someone else, but this assignment requires extra paper work.

The employer can claim to be the creative force behind the creator's work, and therefore the rightful owner of the work's copyright. Say, for example, that you hire me to take a photograph—you tell me where to stand and where to point the camera, and you arrange the subjects of the photograph. Because you directed me in almost every aspect of this photograph's creation, you have a strong claim to the ownership of the photograph's copyright. Although you may win the copyright to this work, you must do so in court. The safest arrangement, therefore, is to specify who is to be the owner of a work before the project begins.

Copyright and the work of others

At the same time, as a multimedia producer you must be aware of the rights of others. You cannot take another's work and use it to create a derivative work unless you obtain the permission of the artist who created the original. Copyright law expressly grants to the owner of the original work the right to make copies and derivatives. Some famous cases have set the precedent for this law, such as that of the sculptor who made a sculpture that was based on a photographer's published photograph. The court judged the sculpture to be a derivative work.

Some exceptions exist to the copyright limits on the use of the works of others. These exceptions fall under the *Fair Use* section of copyright law. Fair Use grants to others the use of protected works under certain conditions, such as for criticism, research, and satire. Fair Use also permits the reproduction of an original work as an element of another work, such as in the creation of a montage. The law stipulates, however, that the reproduction(s) cannot be a major portion of the new creation.

One factor in determining the fair use of another's work is whether the use may detract from the original creator's attempts to sell the work now or in the future.

You should be very careful about using material under the Fair Use clause for any commercial or public use. For anything other than a prototype, you should either gain permission to use protected material, buy material from a clip media collection or stock house, or create your own new material.

Releases

When your productions include photographs or other likenesses of individuals, you must get a release from the individuals depicted in those likenesses. A *release* is a written agreement that states that an individual grants to you (the multimedia producer) the right to use that individual's likeness in any way you please. Remember that the release must specify that you can use the likeness in any way; otherwise, your agreement is open to interpretation.

Obtaining a release is particularly important when your work is to be distributed commercially.

The Future

Ted Nelson, who coined the term “Hypertext,” has spent the last decade promoting Xanadu, a world-wide electronic publishing medium. The idea behind Xanadu is that it serves as a single system for publishing materials. Users can access the material, though they may have to pay a charge. The user can copy and excerpt material, use the material to create new works, and then publish these new works on the network. When another user accesses the new work, the system calculates the royalties for the use of the new work. A percentage of the royalties goes to the creator of the original work, and a percentage goes to the creator of the new work.

Sound complicated? Sure does, and maybe that’s why no such system currently exists.

Electronic networks for clip art, sounds, and even movies, are likely to appear in the future. There are at least two services for distributing commercial photographs, though these are primarily intended for large graphic houses and media organizations.

In the future, end-users will benefit greatly from electronic networks of publishable materials. Not only will the networks bring down the price of multimedia clip art, but they will speed up the work of multimedia producers. At last, you’ll be able to quickly find and use a copy of that perfect background for your slide presentation!

The electronic publishing explosion poses a challenge for the creative industry because digital media is so easy to copy and distribute. Much work remains to be done in educating end-users about what is permissible when working with the creations of others.

IV

PART

Multimedia Environments

Choosing the Right Development Environment

13

chapter

A development environment is the glue that combines different media elements—images, sounds, text, and animation—into a coherent project that the user can explore. Although programs such as Photoshop and Infini-D enable you to create bitmapped graphics or three-dimensional illustrations, they don't enable you to add buttons or data fields. Similarly, though Premiere is a sophisticated tool for editing QuickTime movies, QuickTime does not enable the user to do anything except play the movie from beginning to end and stop and start it manually. This really limits the productions you create, and that's why you will generally need to have access to some kind of development environment such as HyperCard, SuperCard, Director, Producer Pro, or Special Delivery. These programs are described in more depth in Chapter 15, "Authoring Environments," and Chapter 14, "Interactive Presentations."

QuickTime VR, an addition to QuickTime, provides some level of interactivity. Don't be surprised if future versions of QuickTime add even more interactive support.

The Different Environments

Even though this book devotes considerable attention to Director and HyperCard, they are not the only multimedia development environments available. They are two of the most flexible development environments, but with this flexibility comes added complexity. Other tools may be better suited to your application. The tools available for multimedia development can be divided into three categories:

1. Slide presentation environments
2. Interactive presentation environments
3. Authoring environments

Figure 13.1 shows how these environments relate to one another, to QuickTime, and to programming an application from scratch.

Slide presentations

Slide presentation programs such as Persuasion from Aldus and PowerPoint from Microsoft are based on the traditional slide presentation metaphor. You create a series of screens that are linked together in a sequence, and you show them in that order. Although it's possible to jump from one slide to any other slide, it's usually expected that the presentation will proceed in a certain sequence.

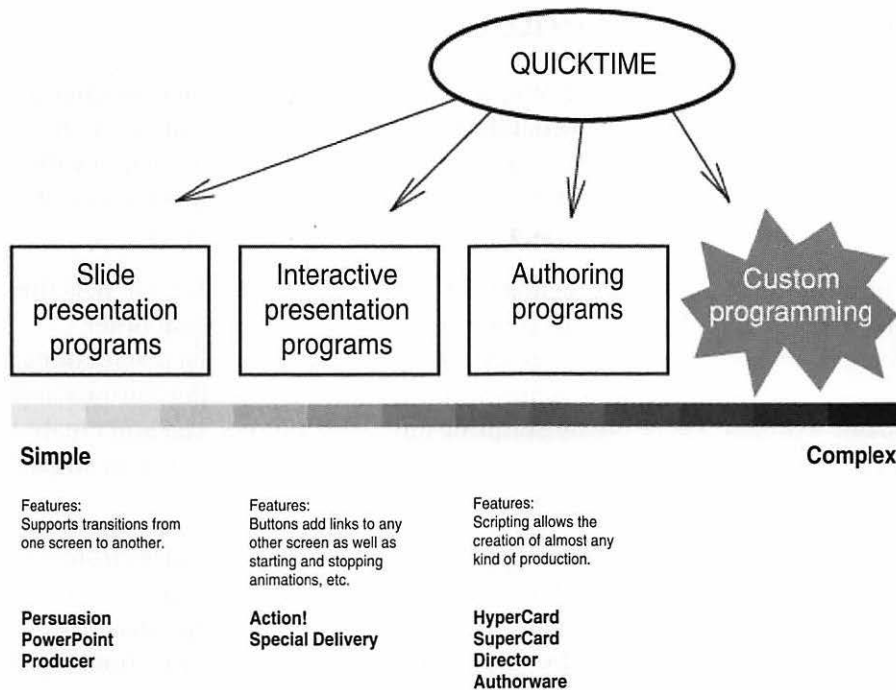


Figure 13.1. The relationship of the different multimedia development environments.

Many of the principles used in creating a sequential presentation are the same as those used in creating an interactive presentation, but there are some specific things to remember when creating a presentation that will be projected or used with a speech. These things include avoiding type that is too small to read when projected and limiting the number of points per slide. These hints are just the “tip of the iceberg” on this subject; there are numerous books devoted to creating presentations.

PowerPoint or Persuasion may be suitable for your application. They don’t, however, offer the functionality and flexibility of the other programs discussed here, so they will not be covered in any greater depth. If you do use PowerPoint or Persuasion, you will still use many of the other tools described in Part III, “Multimedia Tools.”

Interactive presentations

The interactive presentation programs enable the multimedia designer to add buttons that, when clicked, take the user to another location in the presentation. Most of these programs have some things in common with the slide presentation programs. They often use the concept of slides or screens, and you can add transitions from one screen to another.

The addition of buttons that cause different things to happen, such as the playing of a QuickTime movie, or a transition to one of several other screens, makes it possible to create much richer multimedia productions. Action! from Macromedia and Special Delivery from ITM offer buttons and linking features. There are no scripting functions, but you can still create quite complicated presentations. These programs are described in some depth in Chapter 14, “Interactive Presentations.”

Also covered in Chapter 14 are Apple’s Media Tool, Authorware (from Macromedia), and Producer Pro (from Passport). These programs provide some additional options for programming, but these options are not as easy or flexible as the authoring tools listed below, so in this book they are classified as Interactive Presentation tools.

Authoring programs

Scripting really means programming; you write “scripts” using a programming language that the development environment understands. The scripts, or programs, perform various functions such as importing a text file or sorting a list of words. If you haven’t programmed before, don’t be put off by the term *programming*. It’s surprisingly easy to write scripts.

Programs such as HyperCard, Director, and SuperCard add another element to the multimedia developer’s arsenal—scripting. With scripting it is possible to create very complex interactions between the user and the presentation. For example, by clicking a single button, the user might inquire about a subject and then find all the information available on that subject in the presentation.

These authoring environments can be used to create slide presentations or simple interactive presentations, but what sets them apart from the other classes of programs is the capability to write scripts and create complex interfaces.

Just because HyperCard, SuperCard, and Director are in the same category, they don’t have the same strengths and weaknesses, nor should they even necessarily be used to create the same kinds of multimedia presentations. Their differences are explained in more depth in Chapter 15, “Authoring Environments.”

Custom programming

It is possible that none of the commercial programs available will be able to perform the task you want to accomplish. In that case, you may have to resort to custom programming. Though custom programming makes it possible to do almost anything imaginable, it is much more expensive and time consuming than any of the commercial applications already mentioned.

It is beyond the scope of this book to discuss this avenue of development in more depth. If you are considering this route, you should find someone with experience in Macintosh software design and development to provide advice about the feasibility and cost of your desired project.

QuickTime

You are probably wondering where QuickTime fits into the world of multimedia development depicted in figure 13.1. As you have probably noticed, QuickTime is off on its own with a circle pointing to each of the other areas. That's because QuickTime is not really a development environment like the others. It more closely resembles a data type, just as PICT is a data type for graphics, and AIFF is a data type for sound.

Though QuickTime can be used to create movies and may even replace the slide presentation programs for some jobs, it is far more likely that QuickTime will be used to provide animation or video from within one of the development environments.

QuickTime VR does provide some interactive support, but you will still find that for most applications you will need to use an authoring or presentation tool to make a completely interactive production.

The Decision

Many factors go into choosing a development environment, not the least of which are cost, ease of use, and features. Another factor of increasing importance is cross-platform capability; this most often means that you can create a production that runs on Windows as well as Macintosh. Each of the programs has different strengths and weaknesses, making it even more difficult to choose among them.

Here is one recommendation, however. If you are interested in multimedia development, then you should consider beginning with either HyperCard or SuperCard. They are both inexpensive and can be used to create a wide range of presentations. Either would be a good introduction to the world of multimedia development. Director is great if you are interested in animation, but if all you need are buttons and data fields, then Director is too much—in terms of both cost and complexity. Chapter 15 is devoted to an in-depth description of the authoring programs HyperCard, SuperCard, and Director.

Action!, Special Delivery, Apple Media Tool, Authorware, and Producer Pro are worth considering if you don't have any complex scripting needs. Reading Chapter 14, "Interactive Presentations," may help you to decide among the programs.

Interactive Presentations

14

chapter

The programs covered in this chapter, Action!, Astound, Producer Pro, Authorware, Apple Media Tool, and Special Delivery, represent a midpoint between the more traditional slide-based presentation programs PowerPoint and Persuasion, and the authoring environments Director, HyperCard, and SuperCard. These programs are very different from one another, and yet they are similar—they add buttons and links to the basic model of a sequential slide presentation, enabling you to create interactive presentations. These programs, however, don't provide all the features and the scripting capabilities of the authoring environments. While this may limit the capabilities of these programs, it also makes them easier to learn and use. This can be particularly important from a maintenance point of view. Although not everyone would want to create productions using these tools, most anyone with some knowledge of Macintosh programs would be able to do simple maintenance to a production—change a background image, or a piece of text. This may not be the case for Director and SuperCard.

The information in this chapter tells you what interactive presentation programs are, what they do, and how you can use them in your work. The chapter discusses both the capabilities and the limitations of working with these programs.

Understanding Presentation Programs

Although you cannot use presentation programs to create the sophisticated interfaces or complex programs possible with authoring environments, they are perfectly acceptable for creating interactive sales brochures, most kiosk applications, and demos. You also can use presentation programs to create training and help systems.

Most of these programs cannot be extended the way HyperCard and Director can be. Only Authorware supports XCMDs and XFCNS. Apple Media Tool can be programmed (more on that later); support for AppleScript makes Producer Pro extendable also, but not with the same ease and flexibility as HyperCard and Director.

MovieWorks and Cinemation also can be used to create interactive presentations because they offer buttons and interactivity, but these programs come from an animation (rather than presentation) background and, therefore, don't resemble traditional presentation programs. Chapter 8, "Animation," describes the MovieWorks and Cinemation programs.

Using Interactive Presentations

Presentation programs are marketed primarily to users who create sequential presentations and want to add the interest of interactivity to their presentations. As mentioned earlier, presentation programs are good tools for creating such things as kiosk applications, interactive brochures, training systems, and demos.

Adding interactivity, however, also adds a level of complexity that you may not want in your presentation. Do you really want to be trying to remember which button to press while nervously making a presentation to the president of the company? And when making a presentation, how often do you really need to branch off in different directions for different audiences? You may want to start and stop a QuickTime movie or an animation at an appropriate time, but that may be the extent of interaction you need in the presentation.

Another problem in creating the animations and graphical elements is that doing so requires a skill many presenters lack. Action! goes a long way toward solving this problem by providing a large collection of document templates that users can adapt for their own presentations.

In spite of these problems, presentation programs do have application in the areas previously mentioned, and as a multimedia producer you may want to have one of these programs available for simple presentations. Action! and Astound! both can create standalone documents that you can distribute free of charge.

Using Action! and Special Delivery

The remainder of this chapter is devoted to a brief description of the applications.

Action!

Action!, from Macromedia, is available in Macintosh and Windows versions. Although the interface and functions are quite similar in both

versions, the current programs do not share a common file format. As a result, you cannot move a presentation created in Action! on the Macintosh over to the Windows platform (Macromedia is working on offering that functionality). Figure 14.1 shows a presentation being edited in Action!.

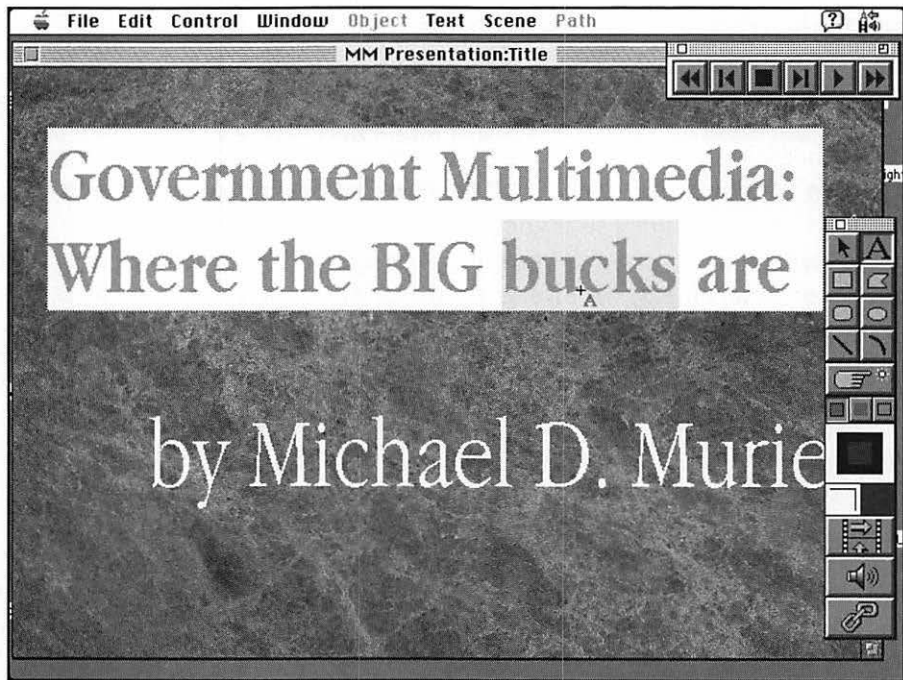


Figure 14.1. Editing a presentation in Action!.

You organize presentations in Action! into **scenes**. Scenes can contain a background image, buttons, and text fields, as well as transitions and animation effects. Scenes are viewed in the Scene Sorter window, which provides thumbnails of the individual scenes (see figure 14.2).

Each scene in an Action presentation begins at a specific time within the presentation; each scene has a specific duration (for example, two minutes). You also must define the time at which an object appears in the scene and how long the object stays onscreen. You make these definitions with the Time window (at the bottom of the screen shown in figure 14.2) or in the Edit Object dialog box (shown in figure 14.3).

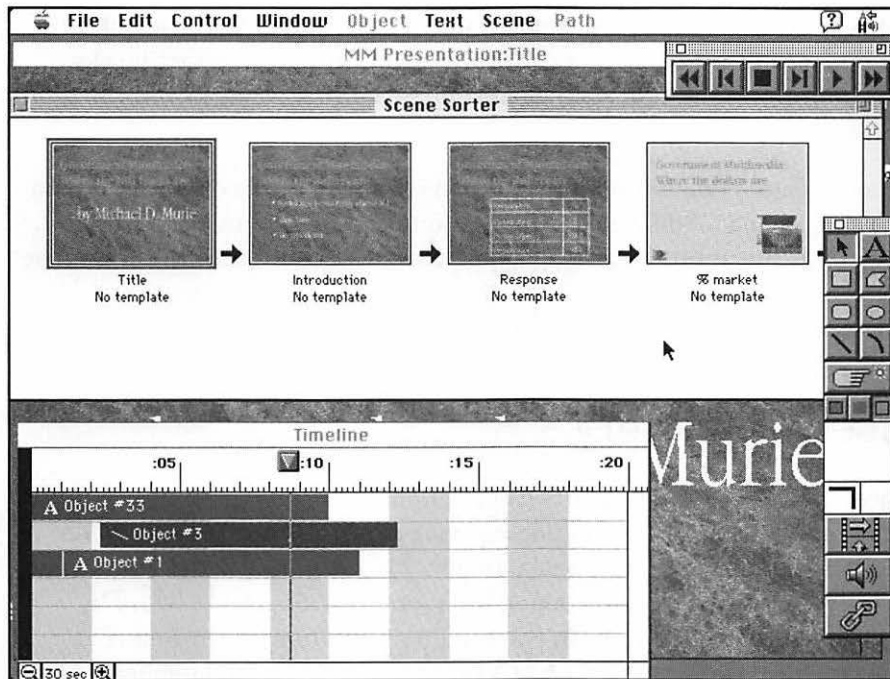


Figure 14.2. The Scene Sorter and Time window.

You can interrupt the presentation by adding a pause at any time within the Time window. The Time window displays the start time and duration of all objects in a scene, and can be used to alter these parameters by clicking and dragging.

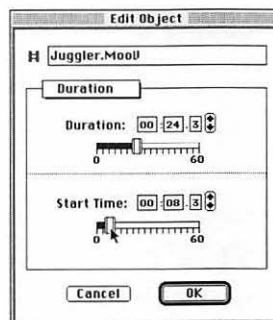


Figure 14.3. The Edit Object dialog box.

You can turn any onscreen element into a button by defining a Link parameter in the Edit Object dialog box. **Linking** adds interactive elements to a presentation such as moving to another scene or returning to the last scene.

While creating time-based presentations is much more complicated than creating regular slide-based presentations, Action! makes it easier for those starting out by providing a collection of templates that you can use to create a new presentation quickly. You also can create your own templates.

Special Delivery

Special Delivery differs greatly from Action! (and most other presentation programs) in the way you define interactivity within the program. The basic building blocks of a Special Delivery presentation are slides, similar to those in Action!, but you define interactivity in Special Delivery by drawing links from buttons to the screen or other onscreen elements. Figure 14.4 shows the creation of a presentation in Special Delivery. At the bottom of the figure, you can see the Map section, which provides thumbnail views of individual screens from the presentation.

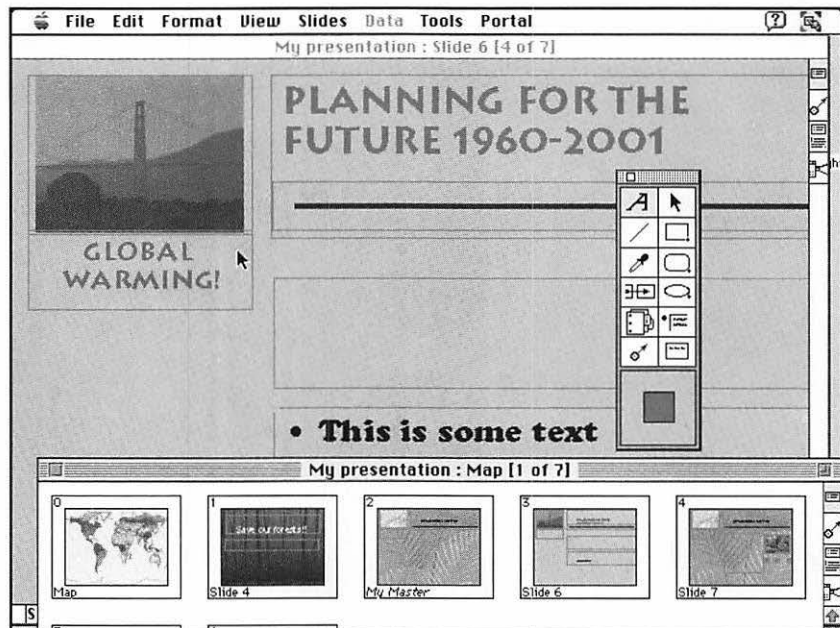


Figure 14.4. A presentation in Special Delivery.

To place a graphic, a piece of text, or a QuickTime movie on the screen, you first draw a **portal**. A portal is a rectangle that “holds” the item. You can use portals to crop and resize graphics, as well as to add interactivity. Special Delivery has three kinds of portals: rectangle, rounded rectangle, and oval.

A separate layer in the slide contains the interactive links. You create these links by selecting the button tool and dragging from one portal to another portal, or to the screen. A predefined collection of commands is used to perform different functions, such as playing QuickTime movies or moving to other screens. Special Delivery supports several basic transitions for moving from one scene to another. Figure 14.5 shows a link being defined in Special Delivery. When the button being defined in this figure is pressed during a presentation, a QuickTime movie plays. Figure 14.6 shows how to create a link from a button to an object—for example, between a button that starts a movie playing and the movie itself.

Portal #2 contains the object acted upon (such as a QuickTime movie).

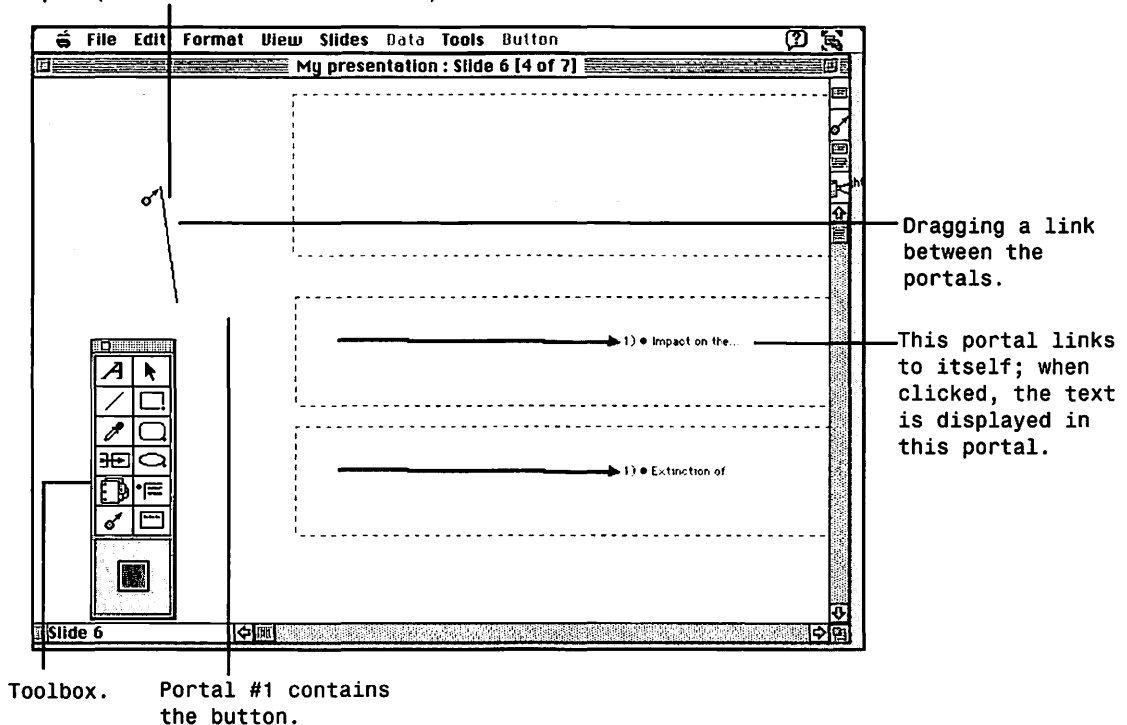


Figure 14.5. Defining a Link in Special Delivery by dragging from one portal to another.

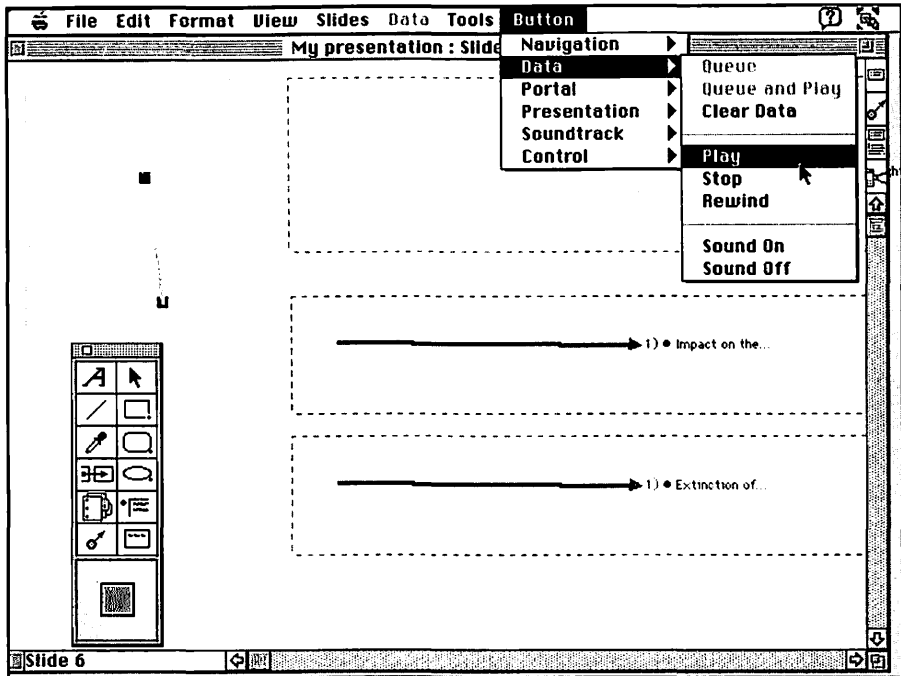


Figure 14.6. Defining the command to be sent when the button is clicked.

Like Persuasion and PowerPoint, Special Delivery supports speaker notes—a printed document for the speaker or attendees that shows a reduced image of the screen and some text notes. Special Delivery also can display these notes on one monitor and the presentation on a second monitor, if you use a computer that is connected to two monitors.

Producer Pro

Passport Producer Pro is a hybrid application for creating multimedia productions. The company that created it, Passport, has a background in audio software, and the program reflects this—it has support for MIDI; elements are arranged in tracks; and the program operates almost like a sequencer program.

A presentation in Producer Pro is made up of several tracks. Each track can hold either a sound, a QuickTime movie, a graphic, text, MIDI file, or a button that causes an action when the user clicks on it. You can synchronize several things by arranging them side by side in separate tracks

at the same point in time. Other features include a standalone player that can be used to distribute productions created with Producer. A Windows player makes it possible to take presentations cross-platform. Figure 14.7 shows a presentation arranged in Producer Pro.

Producer Pro also supports AppleScript, Apple's language for interapplication control. Using AppleScript you can write scripts which cause scriptable applications to do things. For example, you can have a Producer Pro presentation add a record in a FileMaker database by telling FileMaker what to do. To do this you need a copy of AppleScript (a system extension available from Apple), and the application you want to control must have been updated to support AppleScript. While support for AppleScript does make it possible to do things not possible with Producer alone, you cannot write scripts that control elements in the presentation itself. So you cannot create script-controlled animations like you can in Director.

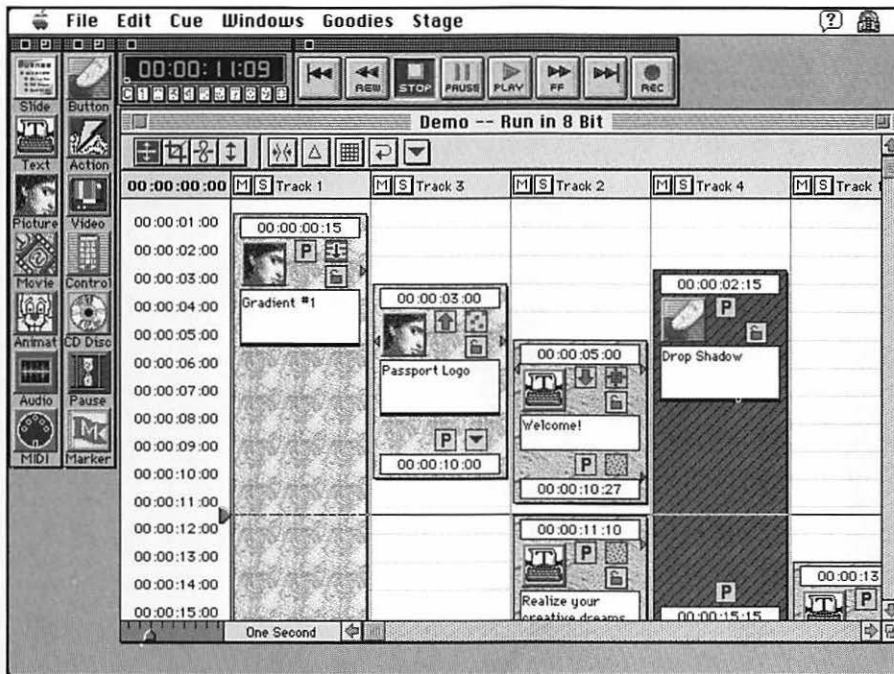


Figure 14.7. Passport Producer Pro.

Authorware

Authorware was originally designed for producing training programs and is very good for sequential presentations and question-and-answer-format presentations. You can play Director animations directly within Authorware presentations. Authorware is also cross-platform—it runs on Macintosh and Windows—which is an important consideration for some projects. The more formal structure of Authorware (you create the production's structure in a linked list that resembles the logic diagram for a computer program) is an easy way to enter interactive training programs. Another advantage is that it is easy to copy parts of a presentation from one Authorware file to another. It's not so easy to do this in HyperCard, SuperCard, or Director.

Authorware does not provide the flexibility of Director, but the fact that you can play Director animations from within Authorware does make it possible to create almost anything (though of course, that means that you need both programs)!

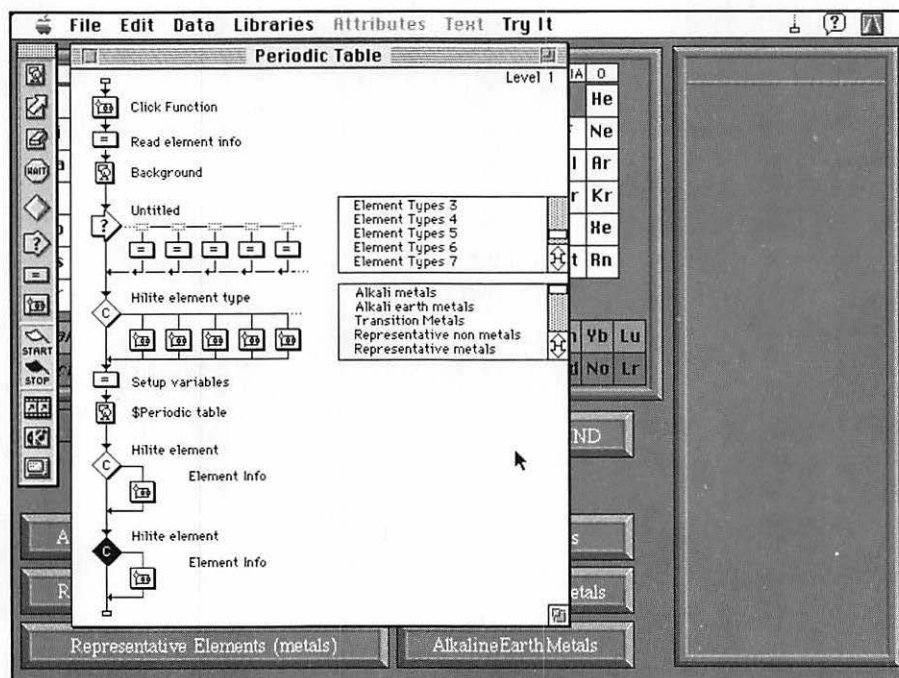


Figure 14.8. An Authorware production.

Apple Media Tool

The Apple Media Tool is actually two products. The Media Tool is an easy-to-use program for creating interactive productions. It provides buttons, fields, and graphic support that enable you to create productions without scripting. Second, there is a Windows converter that turns Apple Media Tool productions into movies that run under Windows.

If you need more flexibility, the Apple Media Tool Programming Language provides a scripting language called Key. It's a very powerful and flexible environment (in fact, the Apple Media Tool was written using the Media Tool Programming Language). In this respect, the Media Tool Programming Language is more capable than the authoring tools Director, SuperCard, and HyperCard. However, programming in this language is much more complicated than the scripting in those programs, so don't even consider the Media Tool Programming Language if the idea of HyperCard scripting frightens you.

The Future

There has been tremendous development in this category of software. These programs don't offer scripting, but they offer flexible options for customizing interactive presentations. For many projects, that is all you need. All of these programs offer different interfaces. While ease-of-use is a serious issue (particularly if you are going to be working with these programs every day), the most likely influence on which tool will be right for your project is features, and of course, cost.

The next chapter looks at some of the interactive presentation packages currently on the market.

Authoring Environments

15

chapter

The authoring environments HyperCard, SuperCard, and Director provide the multimedia producer with a great deal of freedom. You can use these programs to create just about any type of production. They provide a great deal of power and flexibility, and yet are much easier to learn than a traditional programming environment or language such as C or Pascal. All three of these environments feature some kind of programming language. HyperCard uses a language called HyperTalk; SuperCard uses a superset of HyperTalk called SuperTalk; while Director uses a language called Lingo. These languages provide the power to create complex multimedia presentations and interfaces, featuring buttons, fields, and other interface components that are familiar to the Macintosh user.

Although you can use these authoring tools for nearly every production task, they aren't necessarily the best tools for every job. You may find that one of the interactive presentation tools (described in the previous chapter) can do exactly what you want and requires less effort than using the tools described in this chapter. For that reason, study both chapters and then decide which tool is right for you. You may need to know how to use several tools in order to create your productions.

HyperCard—What Is It?

Hypertext is a term that describes documents that have links between words to other references for the word. For example, if you are reading an electronic version of this book and see a reference to hypertext somewhere else in this book, clicking on the word takes you to this description.

Although HyperCard was introduced by Apple back in 1987, it remains unfamiliar to many Macintosh users. Is Hypercard an address book, a database, a hypertext environment, or something else? In some respects, HyperCard is all and none of these things. The best description of HyperCard I have heard comes from the program's creator, Bill Atkinson, who describes it as a "do-it-yourself construction set"—a simple environment for creating just about anything you want.

HyperCard has been used to create address books, children's games, electronic books, demos, and help systems. The following sections describe HyperCard and should give you a better idea of how it works and what it does.

How does it work?

The HyperCard authoring environment consists of an application (HyperCard) and the data files that HyperCard works with, called **stacks**. In many ways a stack is to HyperCard what a text file is to Microsoft

Word—it's the document that HyperCard works with to do things. HyperCard's capabilities are very different, however, than those of a program such as Microsoft Word.

When you start HyperCard and open a new stack, you see an empty window (see figure 15.1). This window is your view of the stack's contents. The Tools palette, also shown in figure 15.1, provides a number of tools that you can use to manipulate the stack. With the exception of the top three (you learn about those later in the chapter), all of the tools are paint tools, similar to those you might see in a program such as MacPaint or Canvas. You can select a tool and start drawing on the stack. You can draw a picture, for example.

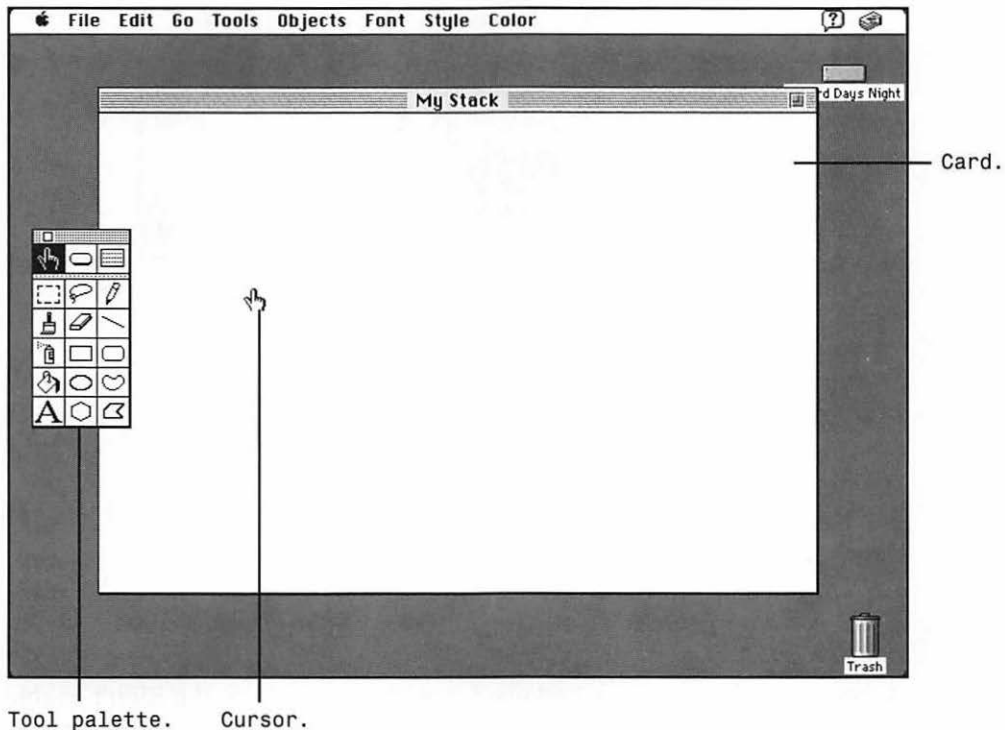


Figure 15.1. An empty HyperCard stack and the Tools palette.

HyperCard doesn't limit you to one page in a stack—you don't have to open another stack to create another illustration. A stack contains **cards**, which are screenfuls of information. To create a new card, you simply choose New Card from the Edit menu. When you choose New Card, the

stack screen becomes blank. Don't worry, what you drew on the first screen—the contents of the first card—still exists in the stack. To go back to the original card, simply choose Prev from the Go menu. The Next command in the Go menu returns you to the new card that you created.

Figure 15.2 shows the relationship of cards to the stack. Cards have a very specific order; card two always follows card one, unless you delete one of the cards or insert a new card between two cards. A stack can contain an almost unlimited number of cards.

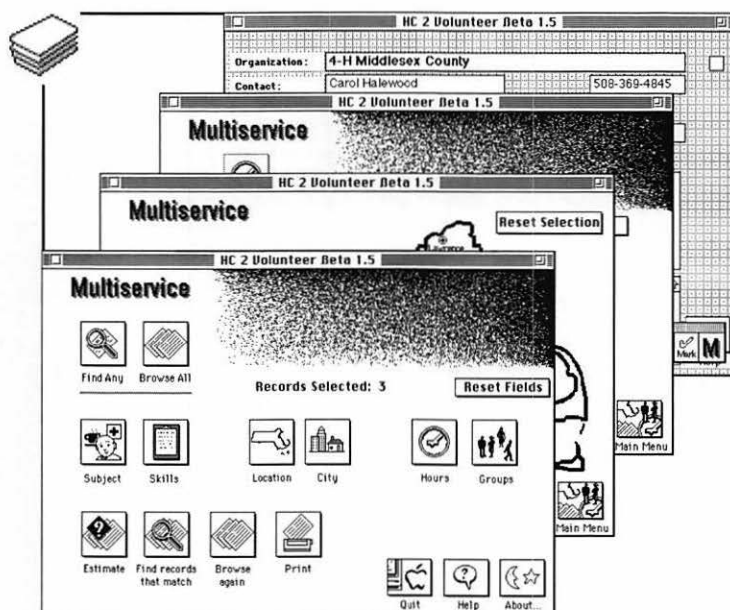


Figure 15.2. The relationship of HyperCard's cards to a stack.

Fields

HyperCard would have limited use for many applications if it didn't enable you to store information—**text**—in the cards. The Text tool, shown at the bottom of the Tool palette illustrated in figure 15.4, enables you to create text, but it only creates bitmapped text; after you write the text it is converted into a graphic made up of a pattern of dots on the screen—just like something painted on the screen in a paint program. You can erase part of the text using the eraser tool (see figure 15.3) because the computer does not recognize the text as information. For this reason, you

cannot search or change the text. To store text in a format that you can change and work easily with, you must first create a **field** to contain the text. A field is a container for text information.

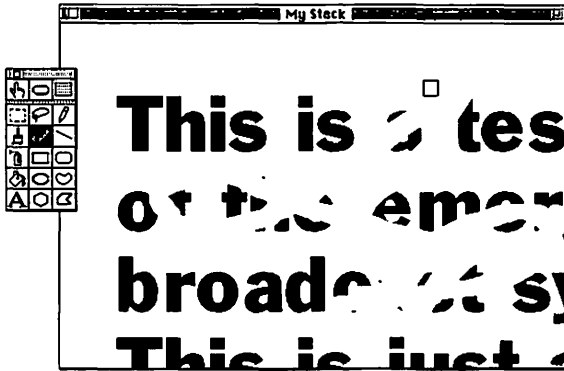


Figure 15.3. Bitmapped text that has been partly erased with the Eraser tool.

To create a field, select the Field tool shown in figure 15.4. Then place the tool over the card and hold down the Command key; the cursor changes to a crosshair shape. Continue to hold down the Command key while you click the mouse button and drag out a rectangle; then release the mouse button. HyperCard creates a text field (see figure 15.5).

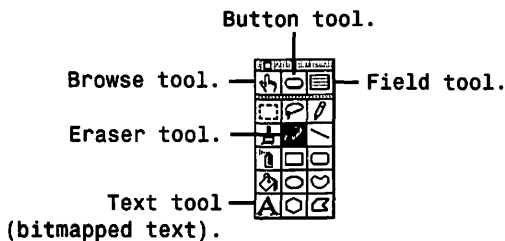


Figure 15.4. The Browse, Button, and Field tools.

After you create a field, you can enter information into it. You must first return to the browse mode to enter information (you can move and resize the text field while the Field tool is selected, but you can't enter information). To return to the browse mode, click on the Browse tool.

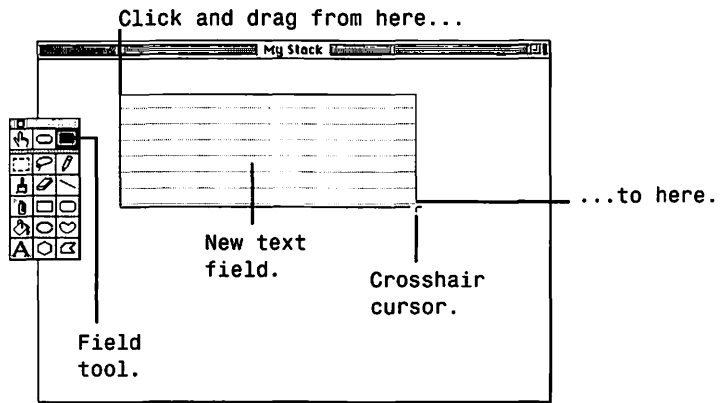


Figure 15.5. Drawing out a field using the Field tool.

The field seems to disappear! Don't worry, it's still there. Move the cursor to the place where you drew the text field; the cursor changes from the hand shape to an I-beam. You can click and type some text into the field (see figure 15.6).

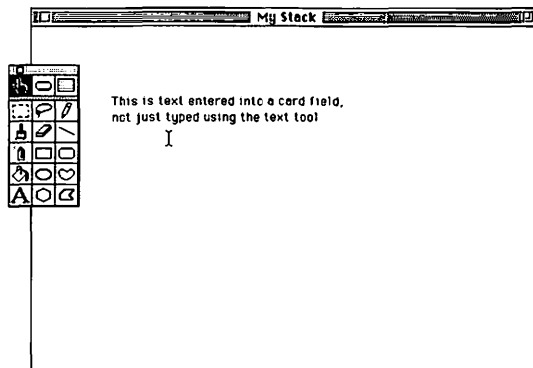


Figure 15.6. Entering text into a field.

If you want to change the font or change the field to a more visible position on the page, you can do so by editing the parameters for the field. To begin this procedure, return to the Field tool and double-click on the field. A dialog box appears that provides access to a number of parameters for the field (see figure 15.7).

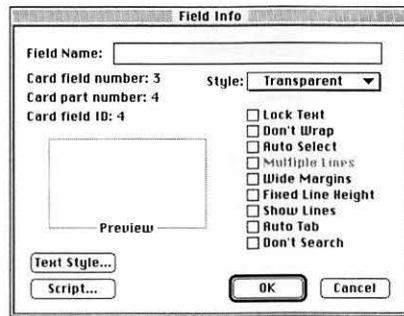


Figure 15.7. Field parameters.

The Style options pop-up menu controls the field's appearance. Transparent is the default style option. Any background or card graphics will be visible underneath a transparent field. When you use the Opaque option and move the field over other fields or bitmapped graphics, they are obscured. The Rectangle option draws a box around the field, and Shadow adds a drop shadow to the field. The Scrolling option turns the field into a scrolling field into which you can easily put a large amount of information. Figure 15.8 illustrates the effects produced with HyperCard's field options.

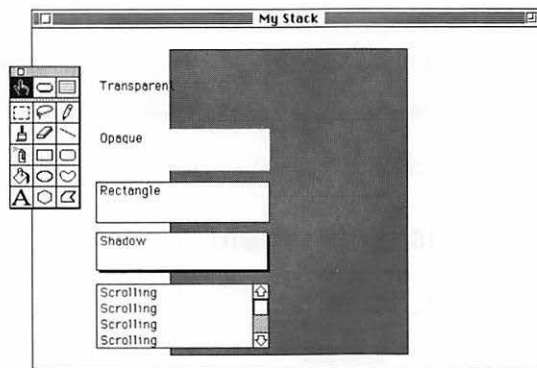


Figure 15.8. The different types of fields in HyperCard.

The other options (Lock Text, Show Lines, and so on) all affect how you interact with the field and the format of the field's contents.

Clicking Font brings up a font dialog box that enables you to change the default font for the field (see figure 15.9). Clicking Script brings up a script window in which you can enter a script for the field. The next section of this chapter describes scripts in more detail.

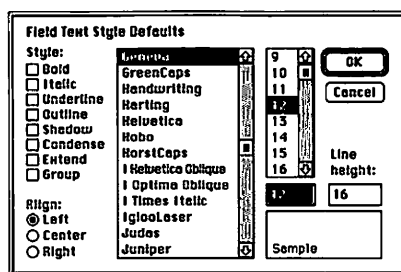


Figure 15.9. The font dialog box for the field.

Finally, notice that the top of the field dialog box contains a Field Name box (which is blank in the figure) and lists a Card field number and Card field ID (refer to figure 15.7). These elements are important in scripting because they identify these objects in the scripts.

Set the field style to Rectangle and close the dialog box. Choose the Browse tool and you will see that the field now has a rectangle around it. If you go to the second card in the stack (by choosing Next Card from the Go To menu), you see that the field does not appear on that card. The field you created is called a **card field** and exists only on the card where you created it (see figure 15.10).

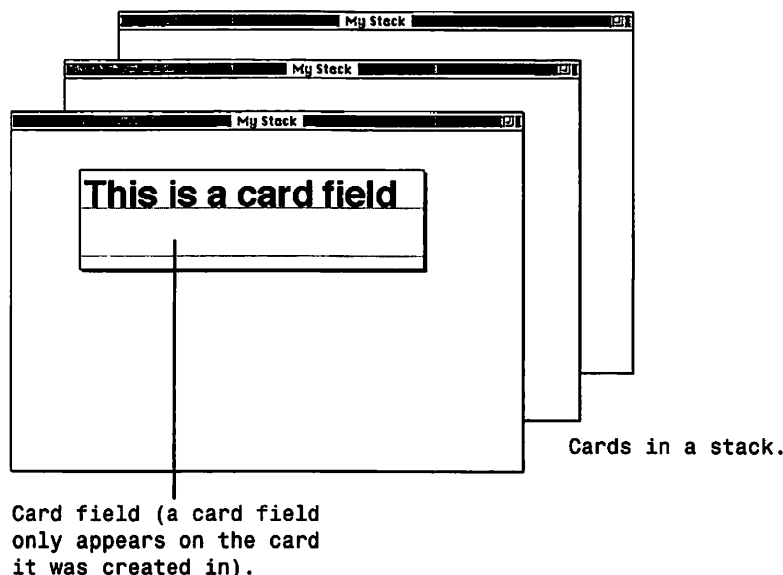


Figure 15.10. The relationship of a card field to the cards in a stack.

While a card field is useful for some applications, it is impractical for others. If you want to create an address book, for example, you will find it tiresome to recreate the fields in every card in a stack. To avoid such repetitive tasks, you can use HyperCard's backgrounds to duplicate common elements throughout several cards.

Backgrounds

You may not realize it, but the two cards that you created in the stack share a common parent, called a **background**. Anything you draw in the background of a card appears in all the cards attached to that background. To add something to the background of the cards in your stack, choose Background from the Go To menu. Notice when you do this that everything on the card disappears, and that the menu bar has a striped outline. Draw something in the background, or type the word "Background" using the text tool and then choose Background again to switch back to the card; you now see both the contents of the card and the contents of the background. When you go to the next card, you see that it has the same background contents (see figure 15.11).

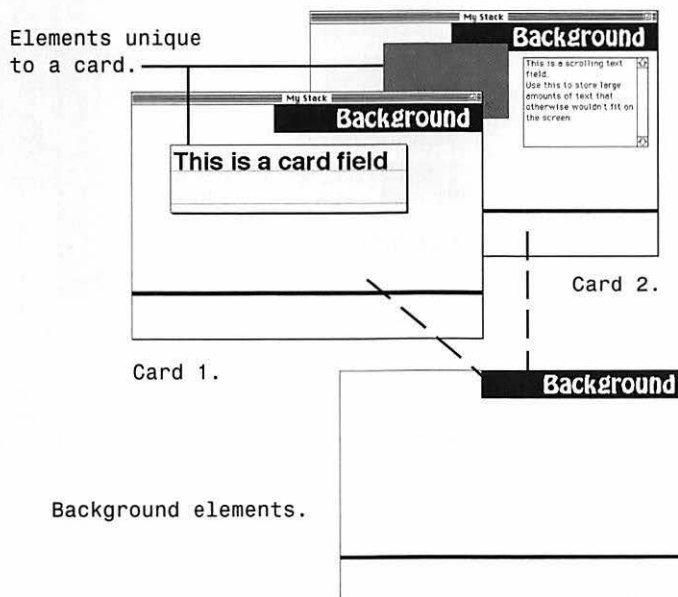


Figure 15.11. Three cards that share a common background.

If you add a field to a background, that field exists in all the cards attached to that background. But the contents of the field in each card will be unique. If you choose New Card from the Edit menu to add a new card, the new card is attached to the current background and will contain all the items defined for that background.

What if you don't want the background to appear in a card? With HyperCard, you can have multiple backgrounds in a stack. To create a new card with a different background, choose New Background from the Edit menu. A new blank card, attached to a new background, appears. (HyperCard automatically creates a new card when you choose New Background because every background must be attached to a card.) Figure 15.12 shows how different backgrounds can be attached to different cards in a single stack.

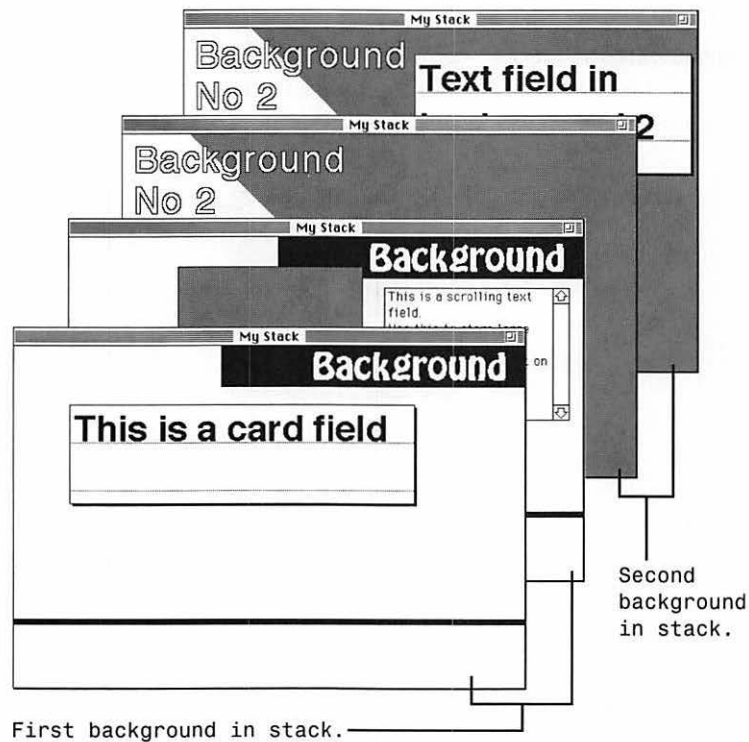


Figure 15.12. Four cards attached to two backgrounds in a single stack.

How do you know which background is attached to a card? Choosing the Bkgnd Info command from the Objects menu displays the identification information of the background that is attached to the current card (see figure 15.13).

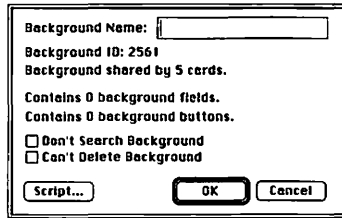


Figure 15.13. Identification information for the background attached to a card.

HyperCard attaches the background to a card when you create the card; you cannot change the background attached to an existing card (but you can delete the card). HyperCard attaches the background that is current when you choose New Card from the Edit menu. If you are in Background 1 when you choose New Card, HyperCard attaches the New Card to Background 1; if Background 2 is current when you choose New Card, HyperCard attaches Background 2 to the new card.

An important factor to remember about backgrounds is that a field created in a background exists in all the cards of that background. This arrangement is particularly useful if you are working with data-base-type applications; for example, if you are storing names and addresses in a stack.

You add a field to a background much the same way that you add a field to a card, except you add the background field while in the edit background mode (choose Background from the Edit menu).

Buttons and scripts

A **button** is an object that the user can click on to make something happen. A sound may play, or the presentation may go to another card. Buttons are important, too, because they enable you to build an interface.

Creating a button that takes you to another card is surprisingly easy. You begin by adding a button, using a process similar to that for adding a field to a card. Choose the Button tool, the middle tool at the top of the tools palette, and draw a rectangle on a card (you can add a button to a background, but for this example, add the button to a card).

Double-click the mouse button and the button parameter dialog box appears, as shown in figure 15.14. Choose Rectangle from the Style column to outline the button. Type a name for the button in the button name field; for this example, type the name Go to next card. Next, click on the Show Name and Auto Hilite buttons. The Show Name option displays the name in the button; the Auto Hilite option highlights (displays in reverse tones) the button when the user clicks on it.

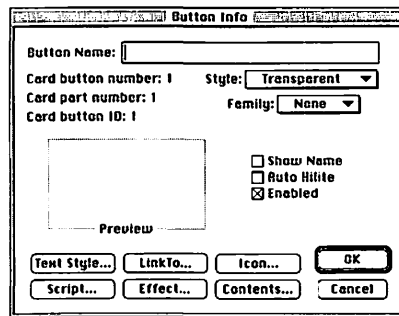


Figure 15.14. The button parameters dialog box.

You can choose to have HyperCard automatically link your newly created button to another card, so that when you click on the button HyperCard goes to the selected card. To create this link, click on the LinkTo button. A palette, shown in figure 15.15, appears with three options: This Card, This Stack, and Cancel. If the palette is in the way, you can move it to the side of the screen, but do not close or cancel the palette. Instead, use the Go To commands in the Go To menu to go to another card in the stack. When you are on another card, click on the This Card button; HyperCard creates a link to the current card from the button in the original card and then returns you to the card that contains the button.

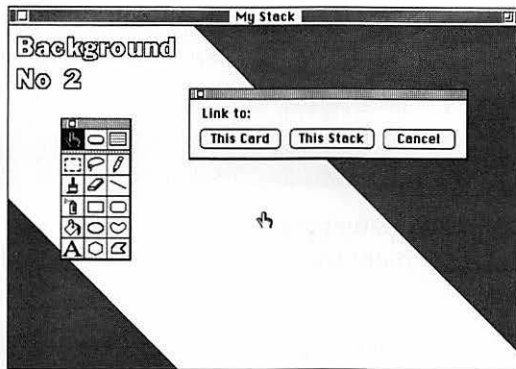


Figure 15.15. The Link to palette.

Let's try the button out. Choose the Browse tool and click the Go To Next Card button. The button highlights and takes you to the linked card.

You may be wondering how HyperCard performs this action. Go back to the first card, either by pressing the Escape key on your keyboard or by choosing the Previous command from the Go To menu. Select the Button tool and double-click the button on the card. When the button preferences dialog box opens, click on the Script button. The Script window opens (see figure 15.16).

The Script window contains a series of instructions that make up the button's **script**. The information in the script tells HyperCard what to do when the button is activated. The script shown in figure 15.16 reads as follows:

```
on mouseUp
  go to card id 4873
end mouseUp
```

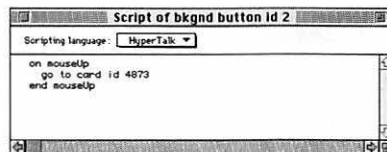


Figure 15.16. The script window.

The ID number in your script window will not match that shown in figure 15.17. Each time you create a new card in a stack, HyperCard assigns a unique ID number to the card and uses that number to identify that card.

Note

Let's talk about browsing versus editing. When either the Button, Field, or any of the paint tools is selected, HyperCard is in editing mode.

Only when the Browse tool (the hand) is chosen do buttons actually work. The Browse tool must be chosen before HyperCard will allow you to type text in fields. When the Button tool is chosen you can do nothing to Text fields; when the Field tool is chosen you can only move and resize fields.

HyperTalk

A HyperCard script is written in the programming language HyperTalk. HyperTalk is similar to the Pascal programming language, but is an English-like language.

Don't be afraid of the term **programming language**. You don't have to be a programmer to use HyperCard, you need only have a willingness to experiment. If you want to experiment a little with HyperTalk, try adding the line:

```
visual effect iris open slow
```

just before the line, `go to card id 1234`. After you add the line, the script looks like this:

```
on mouseUp
    visual effect iris open slow
    go to card id 1234
end mouseUp
```

Choose the Browse tool and click the button again; you will see a visual effect that happens as HyperCard transfers to the other card. You can select from a large collection of visual effects including dissolve, wipe, and checkerboard.

HyperTalk provides a wide range of other commands. This introduction barely scratches the surface. HyperTalk supports strings, variables, and control structures, which enable you to do nearly all the things that sophisticated programming languages do.

Before we leave HyperCard programming, it is important to briefly talk about how HyperCard knows to execute a script.

The script example we looked at began with the words `on mouseUp` and ended with the words `end mouseUp`. These script lines perform two very important functions. The words `on` and `end` indicate the beginning and the end of a script. You may want to store (in the same button) several scripts that perform different functions—for example, one script may play the sound “Click Me” when the mouse is over the button, while another script handles the action when the button is clicked. You therefore must use the `on` and `end` indicators to tell HyperCard where each script begins and ends. The term `mouseUp` is the name of the routine. You must spell the name the same way at both the beginning and the end of the script. If the name is not identical in both places, the formatting of the script is incorrect, and HyperCard will generate an error message when executing the script.

HyperCard spends a lot of time generating messages as the user does things like open a card, click on a button, or type in a field. HyperCard sends these messages to the scripts associated with the buttons, fields, and cards. If a script exists with the same name as a message, then that script is executed.

For example, if you click on a button, when you release the mouse, the message `mouseUp` is sent to the button. If there is a script labeled `mouseUp` attached to the button, then that script will be executed. If there is no `mouseUp` script in the button, HyperCard sends the message to the card script, then the background script, then the stack script, and finally the Home stack script (see figure 15.17).

As a result of HyperCard’s message sending capabilities, you could put a single script in the stack, and have that script do something whenever the user clicks on any button. (Although this arrangement is possible, it’s not one you are likely to want.)

It’s beyond the scope of this book to teach you all you need to know about HyperTalk programming. Check the next section “Where to go from here?” to find out where to learn more.

Part IV

HyperCard sends many other messages, including `mouseDown`, `mouseWithin` and `mouseLeave`. Your button could contain scripts for all these actions:

```
on mouseWithin  
on mouseDown  
on mouseUp  
on mouseLeave
```

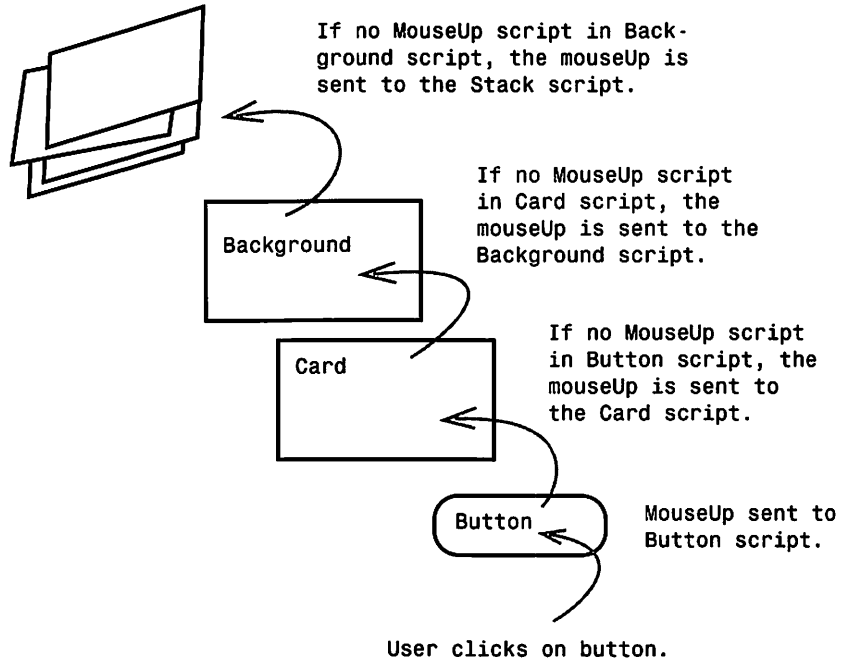


Figure 15.17. Message sending in HyperCard.

AppleScript

The latest version of HyperCard (2.2) adds support for AppleScript. AppleScript is a language that resembles HyperTalk, but was developed to support Apple Events, system software that enables programs to communicate and control one another. If you have an application that supports Apple Events, then by using AppleScript you can write a routine which tells that program to do something—such as tell a database to add a record.

AppleScript comes with an editor for writing AppleScripts, but with HyperCard you could create an easy-to-use interface for doing complex tasks—different buttons in a stack would do different things.

An AppleScript can be placed anywhere that a HyperTalk script is placed—simply choose the scripting language from the language pop-up dialog box in the Script window (see figure 15.18).

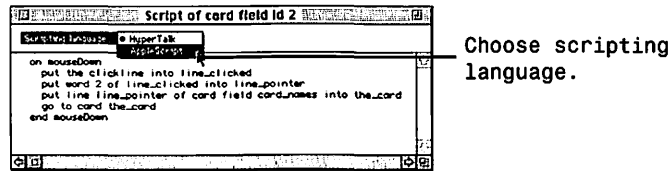


Figure 15.18. Choosing the scripting language.

While AppleScript is useful for talking to other applications, and in fact, you can control HyperCard itself using AppleScript, I would recommend using HyperTalk within HyperCard itself. I would also recommend using AppleScript only if you need to control other programs.

The intricacies of AppleScript are beyond the scope of this book. Therefore, you might want to check out *The Tao of AppleScript, Second Edition*.

Color

HyperCard does not currently support color within the application, although it is possible to add color to your stacks. This may seem like a subtle difference, but it is an important one.

Prior to version 2.2 an XCMD called ColorizeHC was available that enabled you to, under script control, display color graphics within cards. XCMDs are small plug-in programs that add functionality to HyperCard. When using ColorizeHC buttons, fields appear on top of the graphics, but are not colorized themselves. While it is possible to fairly seamlessly create a color stack, you have to create the graphics in other programs, and it takes some scripting knowledge and tweaking to get the graphics to display.

With HyperCard 2.2, Apple added new support for color—specifically a Color menu which, when chosen only offers two options, one of which is

Open Coloring Tools. When you do this, a color palette opens and two new menus are added—Items and Effects. Using the color palette, you can choose and colorize fields and buttons (applying a single color to them), as well as import color graphics and drag them around on the screen to create your desired presentation. The color graphics files (which must be in PICT format) can be stored externally, or imported into the stack as resources.

You can also add transition effects, such as wipes, which work in color (the scripted visual effects do not work in color). These color transitions can be attached to the stack, a background, or a card. The Select Effect window enables you to choose the effect and the length of time the effect takes.

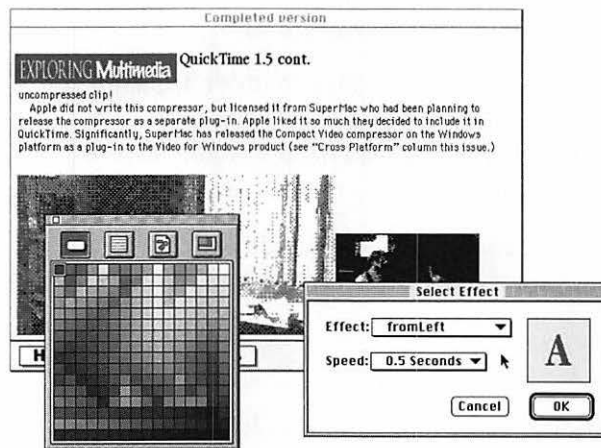


Figure 15.19. The Select Effect window.

While this makes it possible to create quite convincing color productions, rather than update HyperCard to support color, what Apple has created is a more complex XCMD that does a better job of shielding the user from interacting with the external routines. No longer do you have to use scripts to tell the routines where and when to display the graphics. But, because it's an XCMD, it takes several seconds to load the color tools, and the color transitions are applied in a completely different way to the standard HyperCard transitions. Buttons can't have color graphics attached to them, only a single color, and you can't display color graphics under script control.

Still, it is now possible to create simple color presentations fairly easily in HyperCard (and this is shown in more depth in Chapter 16.)

What is it good for?

HyperCard can do so many things—from small databases to kids' games—that describing its benefits can be difficult. An easier approach may be to describe HyperCard's limitations. HyperCard does not replace a real programming environment. HyperCard has many limitations that make it impossible to create an application of any kind: the window size and number of windows is limited, the button types are limited, and the implementation of menus is difficult. HyperCard's support for color is somewhat limited, but if you aren't using color, it runs faster than SuperCard—which supports color, but is a little slower as a result. With HyperCard 2.2 it is possible to turn a stack into a standalone application. Essentially, this means that the HyperCard program code is added to the stack to create a standalone application. Although HyperCard supports text quite well, it is not appropriate for use with large database applications.

A major strength of HyperCard is that it is expandable. It is possible to add additional functions to HyperCard by writing XFCNs. XCMDs and XFCNs are small programs that perform a specific function. You can copy and paste XCMDs and XFCNs into HyperCard stacks to add new functions, using either Apple's ResEdit or some other resource moving utility. QuickTime support and the display of color images on top of cards are just two of the functions that you can add with XCMDs.

Although you can write your own XCMDs, doing so requires some experience with Macintosh programming and is not for the faint-hearted. One exception to this rule exists: Heizer software sells a product called *CompileIt!* which takes a HyperTalk script that you have written and tested in HyperCard, and compiles the script into an XCMD. The advantage of this arrangement is that, in many cases, the compiled XCMD executes much faster than the HyperTalk script executed by HyperCard.

Where to go from here?

This section has only scratched the surface of how to use HyperCard. A number of very good books on HyperCard are currently available that explain the details of HyperTalk programming in greater depth. You may want to check these out:

***HyperCard Stack Design Guideline*, Apple Computer. New York: Addison-Wesley, 1987.** A bit outdated (it doesn't cover the newer version 2.0), but does provide a lot of tips and suggestions for beginners.

***Cool Mac Stacks*, David Drucker. Indianapolis, IN: Hayden Books, 1992.** A short introduction to HyperCard with a lot of examples.

***The Complete HyperCard 2.2 Handbook*, Danny Goodman. New York: Random House 1993.** A very complete HyperCard reference.

HyperCard also comes with a very good electronic reference stack that explains nearly all of the HyperTalk commands, complete with examples (see figure 15.20). You can browse through that stack and use it as a reference.

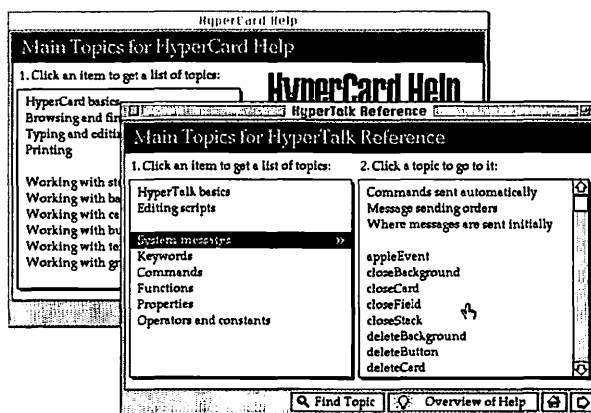


Figure 15.20. The HyperCard online reference.

Finally, stacks created by other users are a great reference. Many stacks are available from online sources and computer-user groups. You can get some of these stacks, take them apart, and examine the scripts to find out how to create your own projects.

The future

The future of HyperCard looks much brighter now that Apple has taken it back from Claris and released a new version. Apple originally released HyperCard and shipped it with every Macintosh, and then later gave the program to Claris, where it languished for a couple of years. Apple's primary reason for reclaiming HyperCard was to use it as a front-end to AppleScript. Apple's renewed interest in HyperCard should ensure its existence for some time to come (we hope).

The following section discusses another authoring environment, SuperCard, that offers a similar but different range of tools for the multimedia producer.

SuperCard—What Is It?

In its functions, SuperCard is very similar to HyperCard. SuperCard uses a language that is similar in syntax and shares many (but not all) of the commands used in HyperTalk. In fact, you can convert HyperCard stacks into SuperCard.

What is the difference between the two environments? The two most important differences are that SuperCard supports multiple windows of all kinds (plain windows, dialog boxes, palettes), and it also supports color—color graphics can be turned into buttons and there are even color paint tools. If you are working on a project that requires either of these functions, SuperCard is a natural choice over HyperCard.

On the downside, SuperCard is slightly more complicated to work with than HyperCard, and is a little slower. Most users who need color, however, are willing to sacrifice some speed to get it.

How does it work?

Like HyperCard, SuperCard has documents (called **projects**) that are run by the SuperCard application. Whereas HyperCard has built-in editing functions, however, SuperCard has a separate application for document editing, called SuperEdit. You create and edit your project using SuperEdit, and you run the project using SuperCard. Some people find this

separate editing application inconvenient, particularly during the debug-
 ging—making it work—stage of development.

SuperCard enables you to build a **standalone** application. In a standalone application, the code required to run the project is copied into the file, creating an application that can run on machines that don't have SuperCard installed. HyperCard 2.2 recently added a similar capability.

When SuperEdit opens a project, it displays a list of the project's win-
 dows. A SuperEdit project **window** is like a stack in HyperCard—
 a stack provides a window in which cards and backgrounds exist. The
 relationship of cards and backgrounds to the window in SuperCard is the
 same as their relationship to a stack in HyperCard, but in SuperCard, you
 can have multiple windows in the same project. Figure 15.21 shows
 SuperEdit being used to edit a simple project.

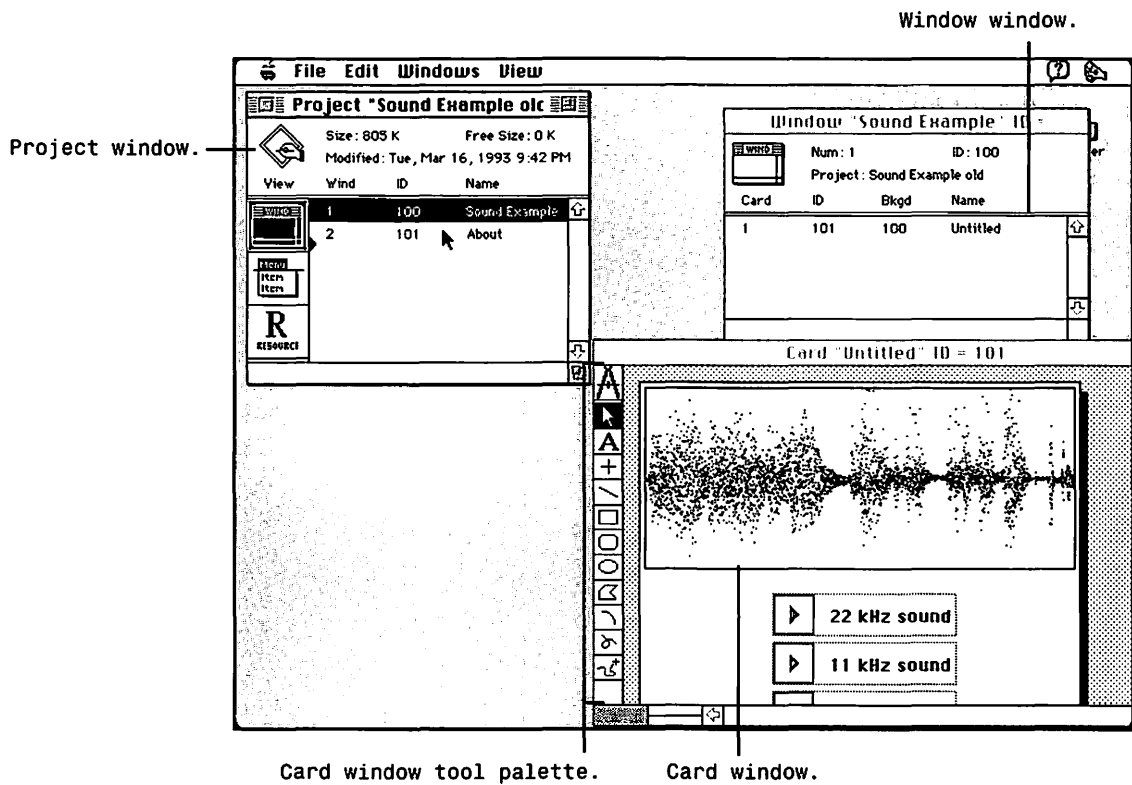


Figure 15.21. The SuperEdit application.

SuperEdit attaches scripts to cards or buttons in the same way that HyperCard does. Note in figure 15.21 that, unlike the tear-off palette in HyperCard, SuperCard tools are arranged on the left side of the edit window in SuperEdit.

When you use SuperCard, you have the option of locating scripts in the project. You can call such a script from any of the windows. This arrangement is similar to that of the Home stack in HyperCard. The scripts in SuperCard are very similar to those in HyperCard, as you can see in the example in figure 15.22.

Setting up menus is easier in SuperCard than in HyperCard because SuperEdit provides a separate list of menus. You call up the list by clicking on the Menus button in the Project window. You can open and edit each menu item from a list and attach a script to the menu item that will be executed when the item is selected. Figure 15.23 shows where this is done.

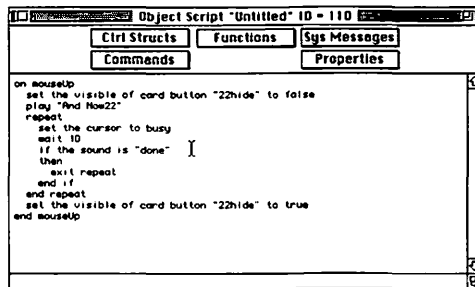


Figure 15.22. The Script dialog box.

What is it good for?

SuperCard's strengths lie in its multiple windows and color support. SuperCard is the tool to use if you require these capabilities. In nearly all other respects, you can do exactly the same things with either HyperCard or SuperCard. Further, many of the add-on applications for HyperCard, such as CompileIt!, also work with SuperCard. Remember, however, that you sacrifice a bit of speed with SuperCard due to the added overhead required with color support. (The lag in performance is slight; most users won't even notice it!)

Part IV

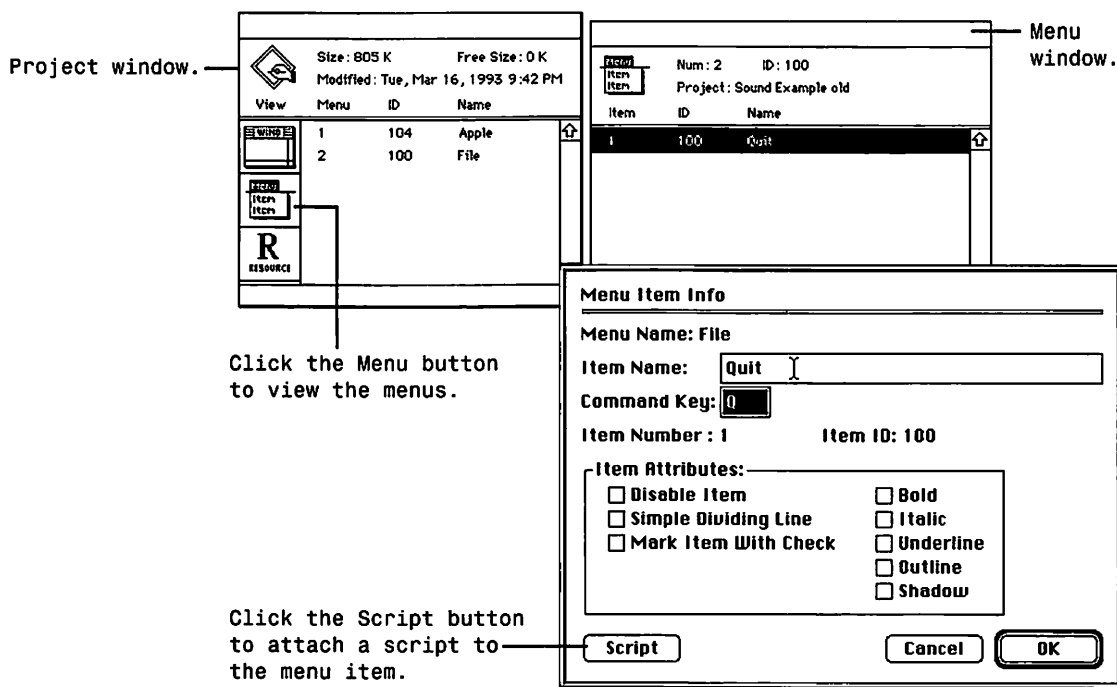


Figure 15.23. Creating a menu in SuperCard.

Where to go from here?

Fewer support books and reference materials are available for SuperCard than for HyperCard. Due to the similarity of the two programs, however, you may find that HyperCard references and example stacks help you in scripting and using SuperCard. Even if you know HyperCard, however, you should refer to the manuals that come with SuperCard to familiarize yourself with the differences in syntax between the two programs. For more information on SuperCard check out:

***Inside SuperCard*, Andrew Himes and Craig Ragland. Redmond WA: Microsoft Press, 1990.** An introduction to SuperCard with emphasis on SuperTalk programming.

The future

SuperCard, like HyperCard, has had a difficult few years. Originally developed by Silicon Beach Software, SuperCard suffered when Silicon Beach was purchased by Aldus. No update was released for two years—until recently SuperCard still did not support QuickTime!

In 1994 SuperCard was sold to a new company called Allegient Technology. Allegient has already released a maintenance update (version 1.7) which added support for QuickTime, and is working on a new version, as well as a Windows Player.

SuperCard deserves to survive, and hopefully Allegient will be able to continue development.

While SuperCard resembles HyperCard, Director is a very different kind of authoring tool.

Director—What Is It?

Director began life as an animation program for the Macintosh called VideoWorks. For several years it was the only animation program available, and for that reason it became very well known and has been used for all sorts of applications.

About the time HyperCard was released, a scripting language (called Lingo) was added to VideoWorks. Lingo is very similar to HyperTalk in syntax (but it's not the same, which can cause confusion when working regularly in both environments).

VideoWorks has gone through a name change, and the publisher—MacroMind—has changed its name to Macromedia. But the product is still very rooted in the world of animation.

Director is somewhat difficult to learn and use, particularly learning scripting. Director remains very popular, however, principally because no other program does exactly what Director does. Macromedia has released frequent updates to keep the program current.

In 1994 Macromedia released Director 4.0. This new version was more powerful than previous versions, featured a new file format optimized for reading from CD-ROM, and several other new features, but the most important thing about Director 4.0 is that it is now available for Windows. The file format is common—an animation created on the Macintosh can be copied across and played on Windows, providing you have a copy of Director on both machines.

You can turn a Director animation into a standalone application that can be distributed to others. This can be done with either the Macintosh or Windows versions.

How does it work?

The concept behind Director is very simple. Director has three major parts: the **Stage** (the window in which an animation plays); the **Cast** (a database of graphic images that are displayed on the Stage); and the **Score** (a spreadsheet that records when and where a cast member is visible on the Stage). These parts are shown in figure 15.24.

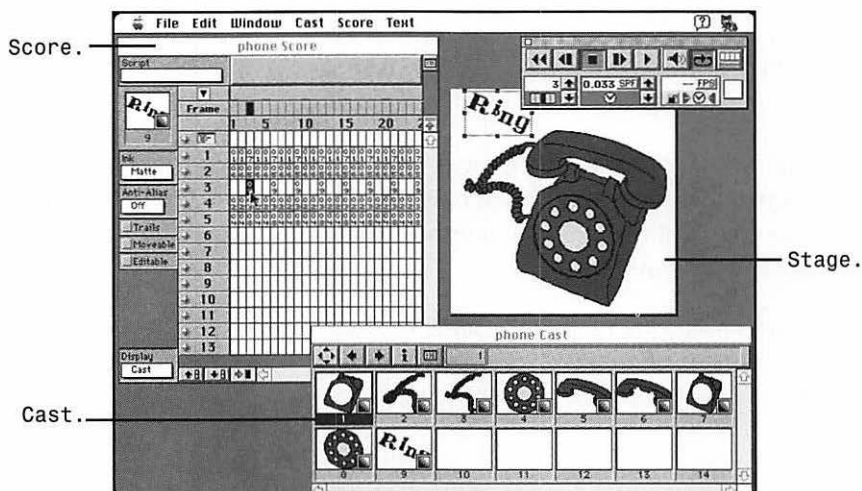


Figure 15.24. The Stage, Cast, and Score in Director.

Director includes other elements (such as a Paint window for creating and editing the cast, and a Text window for entering text), but they only support the three major functions.

To create an animation, you must first create the objects that you want to animate. You can draw these objects in the Paint window, or copy them from another paint program and paste them directly into the Cast window. Figure 15.25 shows a cast member being edited in the Paint window.

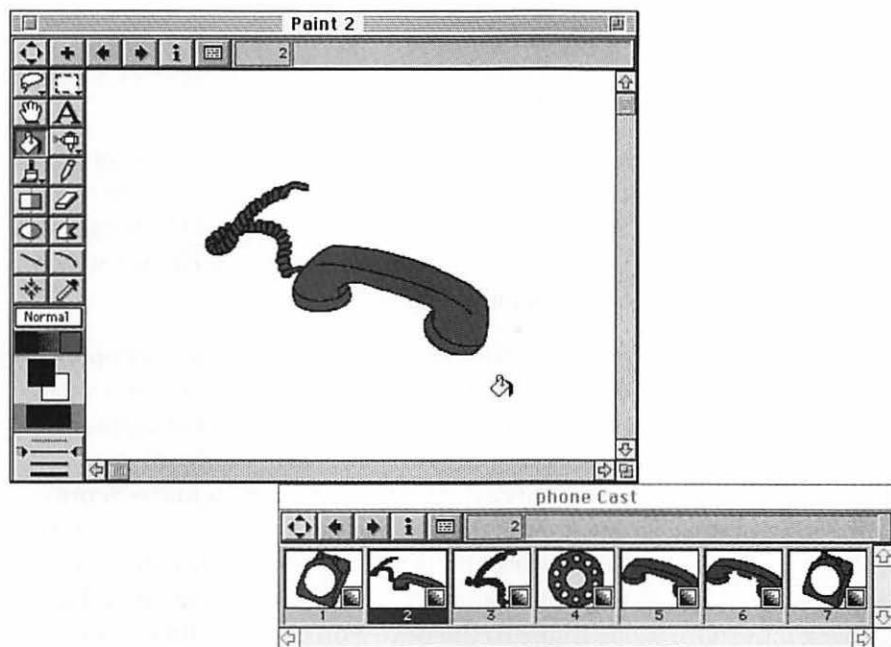


Figure 15.25. The Cast window and Paint window.

After you have created the cast objects, it's time to arrange them on the Stage. To accomplish this task, simply click and drag individual cast members from the Cast window onto the Stage. If you look in the Score window, you see that the Score now contains a mark that indicates the cast member in the current frame. Figure 15.26 shows the cast member on the Stage and in the Score window.

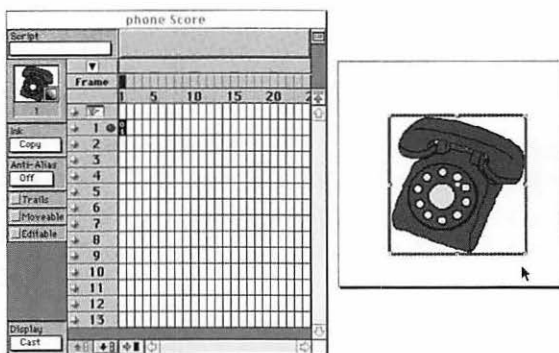


Figure 15.26. A cast member on the Stage and in the Score.

In the Score window, the vertical columns indicate a single frame in the animation. The location of a single cast member is stored in one of the squares in the column. A cast member can be in any one of the horizontal rows (called a **channel**), but the order of its appearance indicates which cast member is drawn on top of another.

If you click in the next vertical column in the Score, the Stage becomes blank. The Stage is blank because you added the cast member to a single frame only. You can either drag the cast member to the Stage again, or select and copy the first block in the Score, and paste it into the next frame. If you copy and paste the cast member information in the Score, the location of the cast member is the same in both frames. To alter the location in the second frame, click and drag the cast member slightly away from the original location. Figure 15.27 shows a cast member that has been moved from one frame to the next. You continue this process to create the animation. You play the animation by clicking on the Play button at the top of the main Director screen.

Director stores Lingo scripts in the Score. Figure 15.28 shows an example of scripts in Director. You can attach the scripts to either a cast member or a frame. You can add commands to a frame; for example, you may want to add a command to go to a specific frame (to create a loop), or a command to pause the animation, or to cause some other time-based event to occur. Director executes scripts attached to cast members only when you click on the cast member. Attaching scripts to objects enables you to implement buttons and other user interface functions in Director.

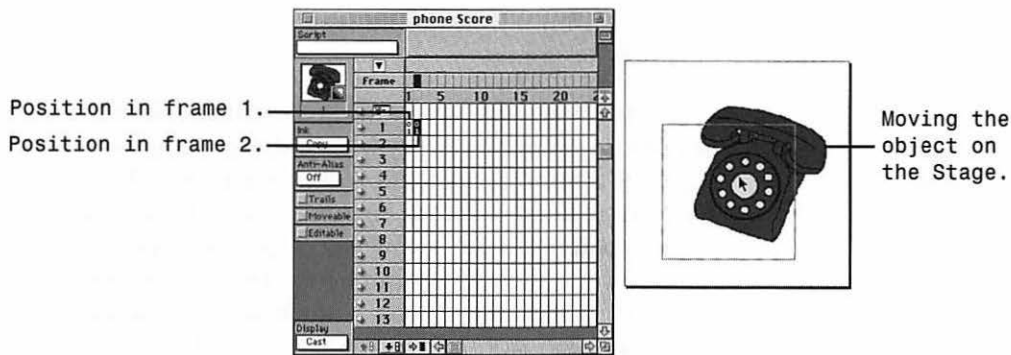


Figure 15.27. Moving a cast member from one location to another.

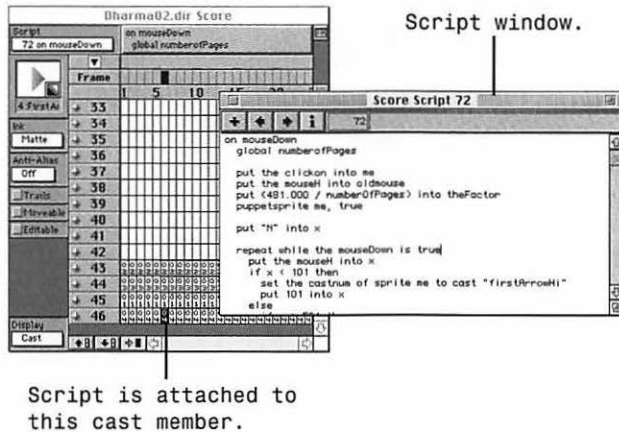


Figure 15.28. Scripts in Director.

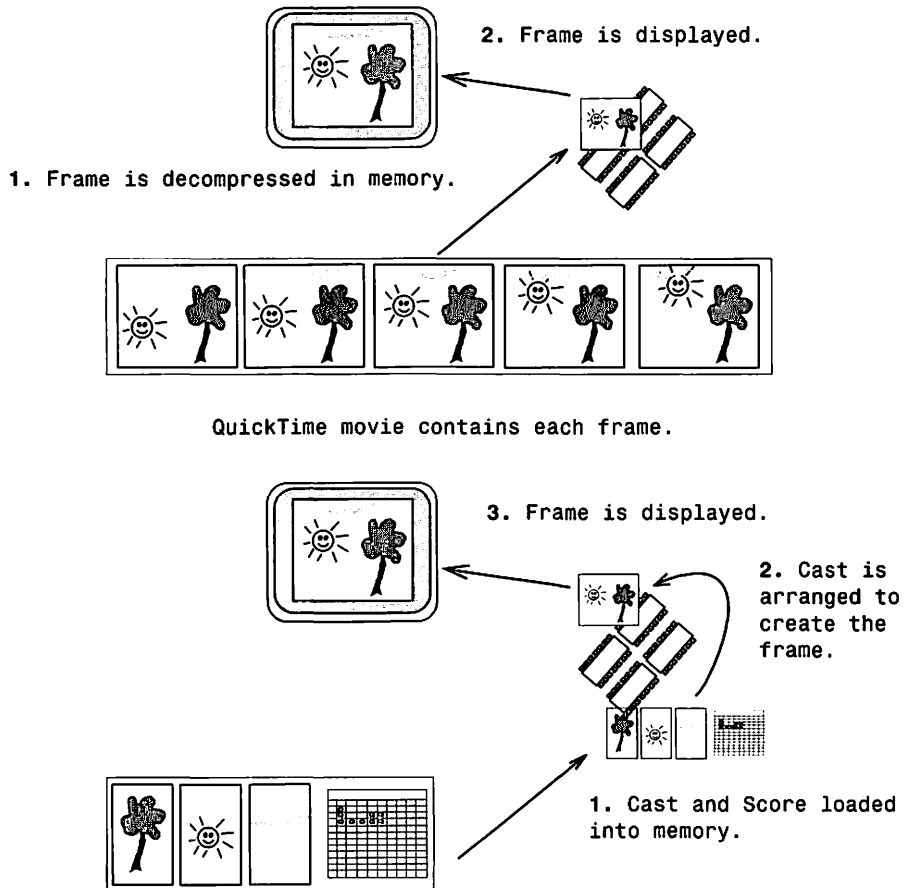
When Director plays an animation, it calculates which cast members are in the frame and where, and draws each frame of the movie from the Stage and Cast information. The frames of the movie don't exist until you run it. This method of storing and playing animation differs from that used by QuickTime. QuickTime stores all the frames of the movie in sequence on the disk. When you play a movie back in QuickTime the program reads the frames from the disk and displays them sequentially on the screen.

Director, on the other hand, loads all the Cast window members into memory and then starts displaying them based on the information in the Stage window. The advantages of this method of storing and displaying animation is that the movies are much easier to edit, and the movies are generally much smaller on the disk than QuickTime movies (because

Part IV

Director essentially is compressing the movies by only storing one copy of any particular cast member or graphic).

The disadvantage of this animation method is that Director has to load everything into memory before it can start playing and, therefore, consumes much more memory. Director needs to store all the cast members in memory, whereas QuickTime needs to store only a couple of frames in memory. Further, if you use mattes (which enable one object to appear through gaps in another object), generating a frame requires much more processing. This increase in processing results in a much slower playback than that of a movie that has been pre-computed.



Director file contains individual elements (Cast) and how they are arranged for each frame (Score).

Figure 15.29. Illustrates the different methods QuickTime and Director use to store animation.

The disadvantages of Director are most apparent in the way it handles memory. For best performance it must load all cast members into memory before playing an animation. But this is not always possible. You can have Director load cast members as it needs them. This works best with interactive presentations, and worse with animations (an animation may start playing and pause in the middle as Director loads the rest of the animation graphics). You can use script commands to tell Director when to load graphics, but this can be difficult to fine tune—particularly for several different memory situations; how the movie performs on a 4 MB machine will be different than an 8 MB machine.

If you are creating a large project, you may be able to split the production into several separate files and then link them together. This can make production easier, but the pause during the loading of the next movie can be excruciatingly long, particularly if the movie is playing from a CD-ROM.

That's not to say that Director doesn't have its advantages. QuickTime does not currently support interactivity (buttons that can be clicked, scripts, and so forth), and you can't create animation in QuickTime—you need another tool. Further, Director can export animations to QuickTime.

What is it good for?

Director can be used for just about anything. The animation capabilities make it particularly attractive for projects that require a lot of visual action. Director is not particularly strong with text, and if you are developing what is primarily a button-clicking interface with little or no animation, then you may be much better off using HyperCard, SuperCard, or another available tool.

Although QuickTime enables applications such as HyperCard to play animation, you must use another tool to create the animation. Director is a good tool for that job.

Director supports a feature similar to XCMDs, called **XObjects**. XObjects provide a capability in Director similar to that provided by XCMDs in HyperCard, but with a slightly different format. In some, but not all cases, Director can work with some XCMDs, by using a special XObject that knows how to communicate with XCMDs. Director also has a sophisticated scripting function, called **factories**, that enables you to create very complex routines and functions. Beginners should avoid using this function, however, until they are completely comfortable with Director's Lingo language.

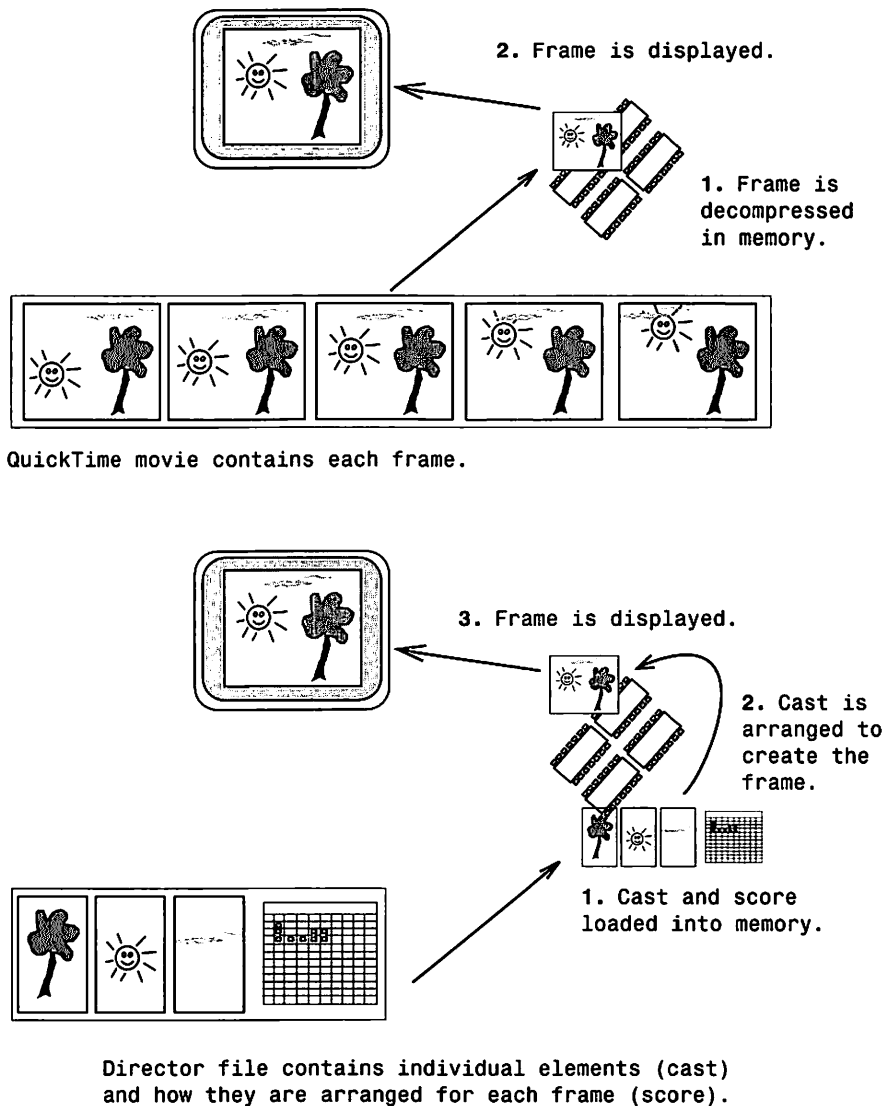


Figure 15.30. How Director stores movies versus QuickTime.

An important advantage Director has over HyperCard and SuperCard is the cross-platform capability. Not only is a version available for Windows, but it can play the files created with the Macintosh version, making it possible to create a single presentation that is distributed to users with different machines. Projectors (Director's name for standalone movies) can be created for Macintosh and Windows that link to the same file.

Where to go from here?

Macromedia and certified trainers around the country offer courses and classes on using Director. If you are a beginning Director user, and need to get something done quickly, you may benefit from attending one of these courses. Director seems to be one of those programs that lots of people have—or think they should have—but that few people know how to use. It can take a while to grasp the concepts of how Director works, and that's why a hands-on class could make the difference between a successful project and a product box gathering dust on the shelf.

You might want to try *Macromedia Director Design Guide*, **Cathy Clarke and Lee Swearingen**. Indianapolis, IN: Hayden Books, 1994. It illustrates newly invented multimedia production processes within Director in an attempt to provide inspiration for experienced and aspiring multimedia developers.

Other Programs to Consider

There are still other programs that you may want to consider using for the development of your multimedia projects. The most obvious of these is another Macromedia program, Authorware. This program is described in more depth in Chapter 14, "Interactive Presentations". Also, the Apple Media Tool (also described in Chapter 14) has a programming environment which is very powerful, yet more difficult to learn.

You also can choose from among some iconic programming languages for the Macintosh. Two such environments are Serious and Prograph. Although you still have to learn a programming language to use them, these tools provide some basic building blocks for creating your own programs, as well as a different interface for designing the logic of a program using icons and links between these icons. Although some people consider iconic programming languages to be easier to use than traditional programming environments, they are more complicated than programs such as Director or HyperCard. The iconic programming languages, however, also provide much more power.

In choosing an environment, you must consider a number of questions: Which tool will best suit your needs? How long will it take you to learn the tool? And how long will it take to produce the final production?

Part IV

Sometimes you must trade some functionality in the final production for a reduction of time spent in learning and using more sophisticated tools. Only your individual needs can determine your decision.

Finally, while all of the programs described here have their strengths and weaknesses, and one may be better suited to a particular task than another, it is also true that in many cases you could create an acceptable production with any of the programs. If you already know one program, it may make more sense to use that rather than try to learn every program.

This chapter has reviewed the most sophisticated multimedia authoring tools available. The next chapter goes step-by-step through the process of creating a prototype using HyperCard. For those just starting out, this chapter will serve as an introduction and tutorial for creating multimedia projects.



Multimedia Workshops

Putting It to Work

16

chapter

You've looked at the design examples, and read about all the tools and authoring environments. Now fasten your seat belts and get ready to put it all to use! This chapter shows you how to create a project using HyperCard. Starting at the beginning, we will assemble an electronic interactive publication—the project that was specified in Chapter 3, “Developing a Multimedia Project.” The following chapter takes you through creating an electronic advertisement using Director.

What is the Project?

Exploring Multimedia is a monthly Macintosh multimedia newsletter whose primary audience is Macintosh multimedia producers and users. There's only one problem—it appears on paper, and while a paper version is the easiest to produce and distribute, the editor would like to have an electronic version that can be distributed through bulletin boards and on disk as a promotional item.

Beta is computer-
ese for “Still not
finished.”

Using material from the first issue of the newsletter (the text and pictures), the electronic version will be assembled in HyperCard. This is a reasonably easy exercise—all that's required is a main card that contains the table of contents (the articles in the issue) and then a series of cards for each article. For this beta version only one article will be implemented.

The complete publication will look something like figure 16.1, but for this exercise we will implement only the main card and one article.

Getting Started



Included on the disc, in the folder labeled Workshop files (within the Chapter 16 folder), is a collection of data files that make up the first article. The images are in 8-bit color—though you can use them on a grayscale monitor. You'll find the text for the article, as well as two HyperCard stacks. One stack (Table stack) contains a table used on one of the pages. The other stack (Colorizing HyperCard 1.0) contains XCMDs (small plug-in routines) that can be used to display color images in HyperCard versions prior to 2.2. However, this chapter assumes that you are using version 2.2; you really should upgrade to that version if you are serious about using HyperCard.

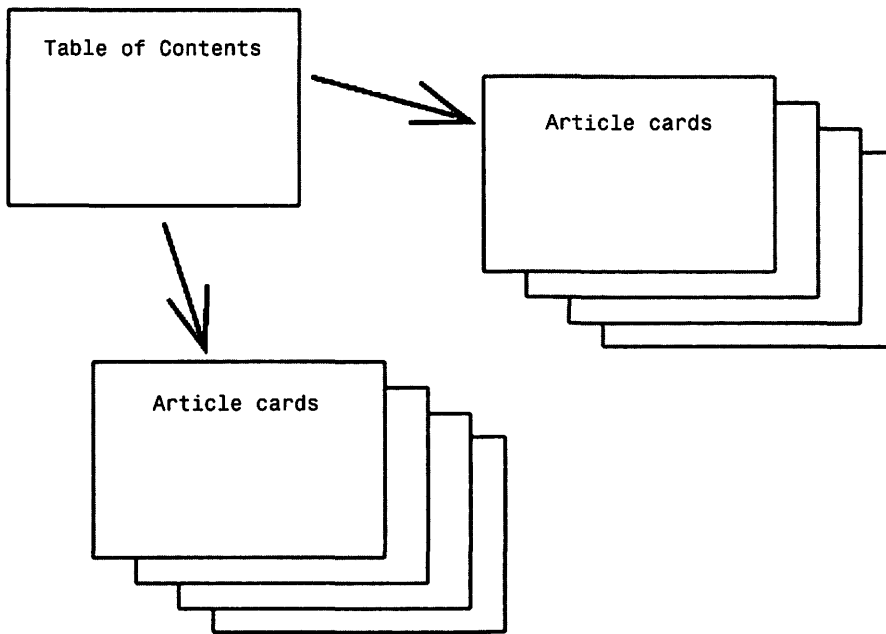


Figure 16.1. The operational diagram for the electronic version of *Exploring Multimedia*.

You should copy the folder labeled Workshop files from the CD-ROM to your hard disk before you begin work. It will make it easier if the graphics are in the same folder as the stack that you create. If they aren't, the routine that displays them won't be able to find the graphics.

If you are running HyperCard under MultiFinder on System 6, or you are using System 7, you should increase the amount of memory allocated to the HyperCard application to at least 2 MB. You must do this because the Color routines use a large amount of memory. If you don't increase the memory available, the routines may not work correctly and the color graphics will not be displayed.

To increase the amount of RAM allocated by the Finder to HyperCard while HyperCard is *not* running, select the HyperCard application icon (by clicking it once) and then type the Get Info command (⌘-I) to display the Info dialog box. Change the preferred Size in the Memory requirements section of the Info dialog box to at least 2,000K, or 3,000K if you have at least 5 MB of RAM in your computer.

Step 1: Creating a New Stack

Launch HyperCard by either double-clicking the application or the Home Stack. In either case the Home stack opens. Choose New Stack from the File menu. A dialog box appears asking you to name the new stack and define the size of the stack window (see figure 16.2).

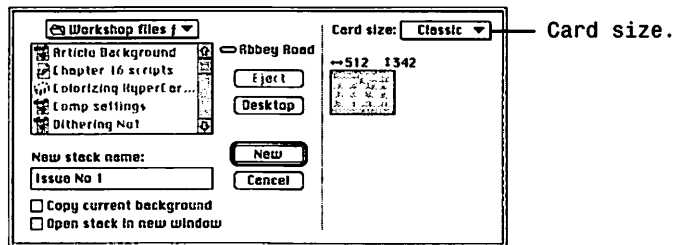


Figure 16.2. The New Stack dialog box.

Type a name for the stack—Exploring Multimedia might be a good title. Make sure that the card size is 512 by 342 (Classic) in the card size on the right-hand side of the dialog box. All the graphics have been created for this size window. Do not select the checkboxes for Copy current background or Open stack in new window. Click the New button.

HyperCard will open a new stack which contains one empty card (see figure 16.3).

Displaying color graphics

The next step is adding a color graphic to the first card of the stack. This card will be the title/contents page of the publication; it will have a different image from the rest of the publication.

To add a color graphic you must switch to HyperCard's color tools. A dialog box (see figure 16.4) appears informing you that the Color Tools must add resources and scripts to the stack. This is because the Color Tools are actually a collection of XCMDs that must be copied to any stack that makes use of them. Click OK.

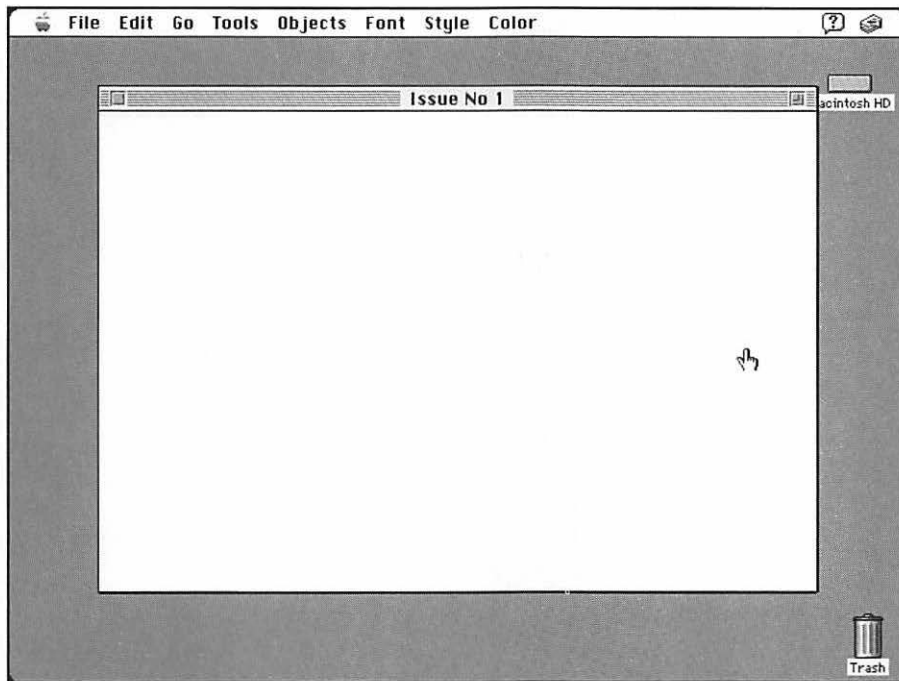


Figure 16.3. The new stack.

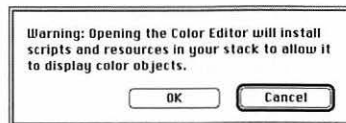


Figure 16.4. Color Editor warning message.

A color palette appears, and two new menus—Items and Effects—are added to the menu bar (see figure 16.5). The color palette is divided into two parts: a row of buttons across the top and a color palette. The color palette is used to apply colors to buttons, fields, and graphic elements. The button tool must be chosen to select buttons to apply color to them, and the field tool must be used to select fields. The PICT button (labeled with the PICT file type icon) is used to edit PICT images on the card. When this button is double-clicked, a dialog box prompts you to display on the current card. By clicking this button only once, PICT images on the current card can be moved or deleted.

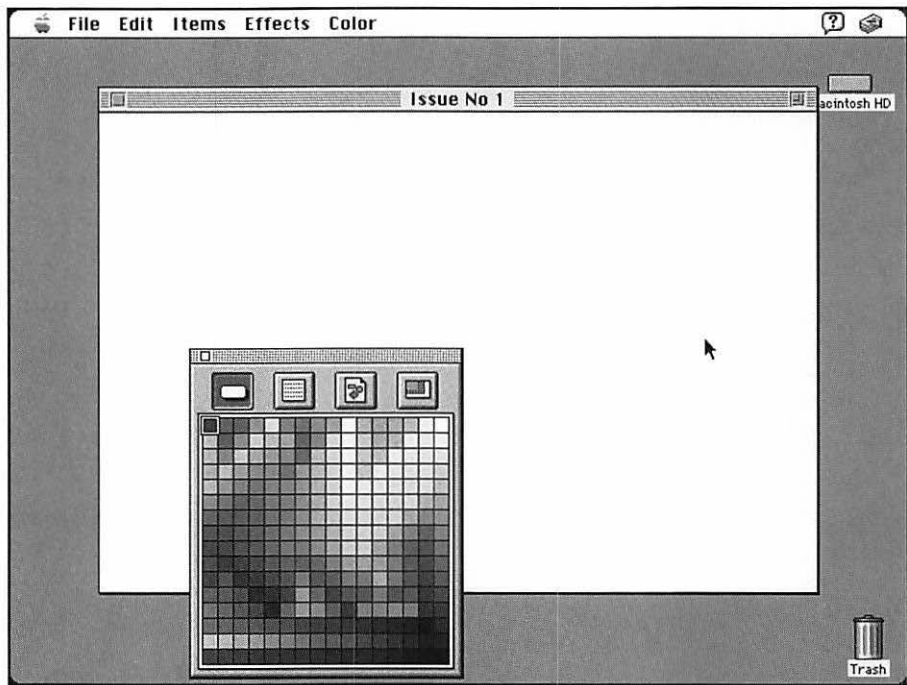


Figure 16.5. The Color Tools palette and menu items.

Before adding the color graphic you must decide how you want to store the graphic. HyperCard's Color Tools offers two choices. One is to store links to PICT files stored on the disk. When the images have to be displayed, they are read from the original files. The alternative is to add the PICT files as resources to the HyperCard stack. This makes the stack larger, but means that you only have to distribute one file to users. For this stack, you will import the graphics.

You can either double-click the PICT button or choose New Picture from the Items menu. In either case, the Select a Picture to Place window appears (see figure 16.6). This window provides a list of the images that have already been imported into the stack. You can import new images and add or delete them to the current card.

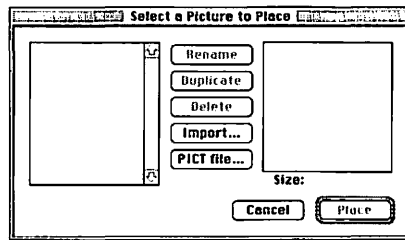


Figure 16.6. The Select a Picture to Place window.

Click Import and choose the file “Master Bkgnd” from the Workshop folder. The graphic will appear in the Select a Picture to Place window. Note that the image has only been imported into the stack. You must now add it to the current card (if you want to). Make sure the graphic is selected and choose Place.

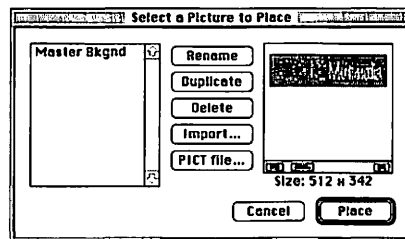


Figure 16.7. An image imported into the stack.

The graphic will now appear in the stack, though it probably isn’t where you want it to appear (see figure 16.8). Click and drag the graphic so that it covers the entire card—the “crawling ants,” or selection marquee (the dotted lines that surround the selected graphic), should be visible on all four sides of the image.

While you’re here, you may want to look through the menus available in the Color Tools. Commands in the Items menu enable you to change the stacking order of objects on the card. You can add several objects to the same card, and then change which one appears on top of the other. The Effects menu enables you to define the color transitions between cards, backgrounds, or stacks. If you want to add a transition, feel free to do so.

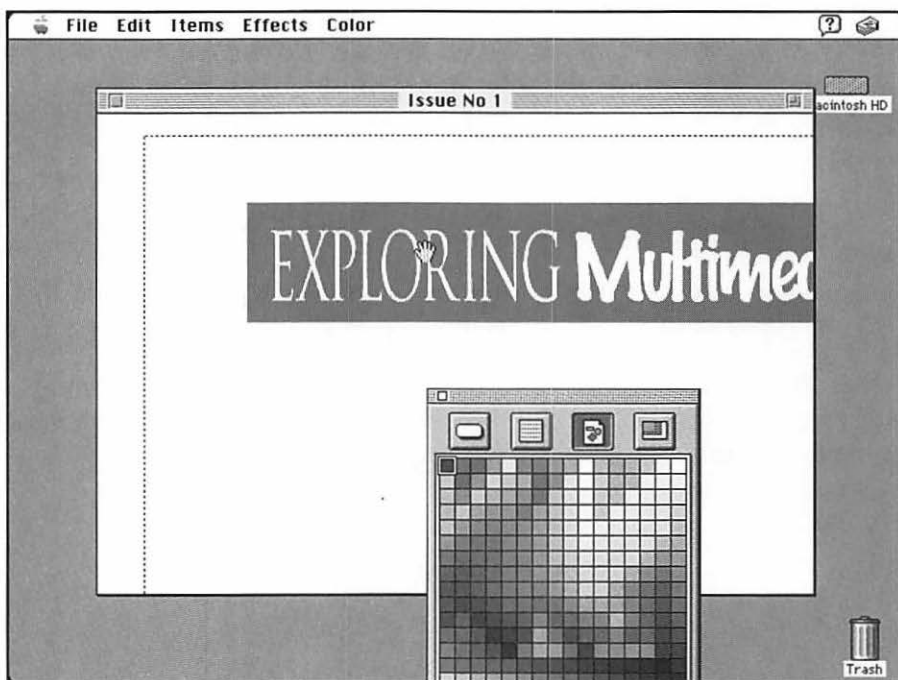


Figure 16.8. The image placed on the current card.

Exit Color Tools by choosing Close Coloring Tools from the Color menu.

Notice that the background has buttons drawn on it; these buttons don't work yet.

Adding a Quit button

Choose the \mathcal{H} key and the cursor will change to a cross-hair. Drag out a button that covers the Quit button (see figure 16.9).

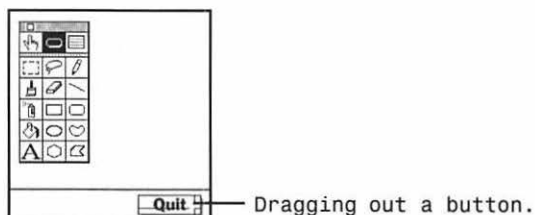


Figure 16.9. Dragging out a button.

With the button tool still selected, double-click the button. This will open up the Button Info dialog box. (Double-clicking is a shortcut for clicking once on the button to select it and then going to the Objects menu and choosing Button Info.) Check the Auto Hilite checkbox (see figure 16.10).

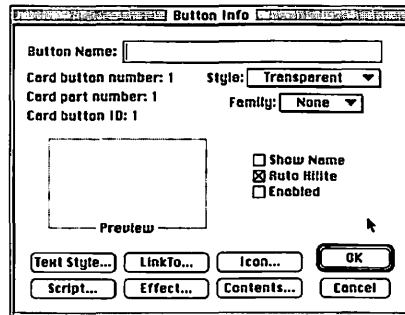


Figure 16.10. The Auto Hilite checkbox.

Click Script and add the following script to the button:

```
on mouseUp
  domenu "Quit HyperCard"
end mouseUp
```

This script executes a command that is in one of HyperCard's menus—the Quit HyperCard command. You can use the `domenu` instruction in your script to execute any command that is available in HyperCard's menus.

Close the Script window and choose the Browse tool (the hand in the Tools palette). Clicking on the Quit button should now quit HyperCard.

Step 2: Adding Another Background

The next step is to add a new background in the stack. This background will be used to display the content cards—the cards that contain the text and pictures that make up an article.

Each article will consist of a sequence of cards that share this common background. The background will contain a field for storing the text, buttons for navigation, standard routines for displaying the color background, and any picture elements.

Add a second background

Open the stack (if it isn't already open). You should be on the first card (with the color graphic).

From the Objects menu, choose New Background. The screen will change—the menu bar will have a dashed pattern surrounding it. This indicates that you are editing the background. A new card also has been added to the stack. The window will be blank (white).

You are going to use a single graphic as the background for all the cards in this article. Instead of adding this graphic to each card individually, you will add it to the background. To do this you must switch to the Color Tools while still editing the background (which is indicated by the dash marks around the menu bar).

Because you just added the background to the stack, HyperCard should have switched into the edit background mode, and the dash marks will be visible (see the menu bar in figure 16.11). If they aren't visible, choose Background from the Edit menu.

Then add the graphic “Article Background” by repeating the steps used to add the previous graphic—remember to first choose Open Coloring Tools from the Color menu. After you're done, the stack should look something like that in figure 16.11.

Add a background field and buttons

Now you will add a background field and a background button, and start placing the text onto the card. Close the Color Tools by choosing Close Coloring Tools from the Color menu.

If you aren't currently editing the background, choose Background from the Edit menu (or press ⌘-B) to edit the contents of the background. Choose the Field editing tool from the Tools palette. Hold down the ⌘ key and drag out a text field to fill the area between the *Exploring Multimedia* logo and the buttons at the bottom of the screen (see figure 16.12).

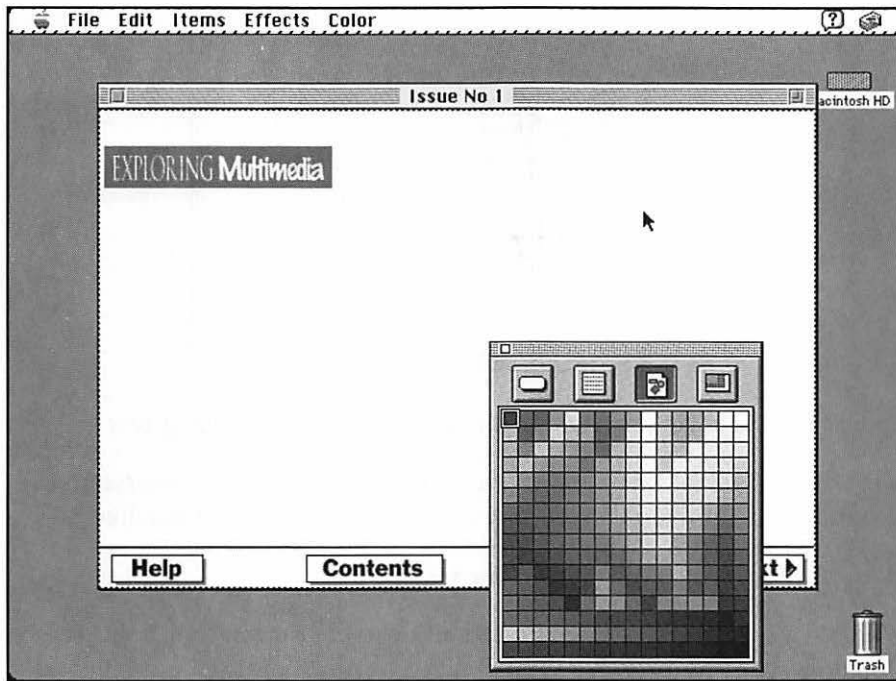


Figure 16.11. The color background for the article cards.

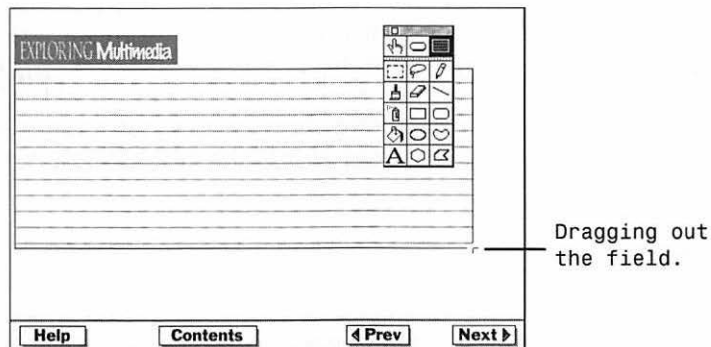


Figure 16.12. Adding a background field to contain text.

Then change the default font used to display the text. To do this, double-click the text field (with the field tool still selected). When the Field Info dialog box appears, click the Text Style button, and then choose the font you want to use in the field (see figure 16.13).

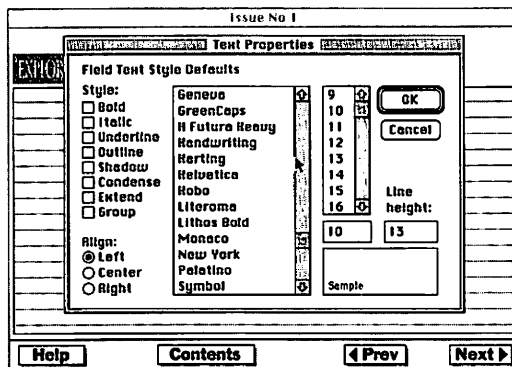


Figure 16.13. Choosing the field's default font in the font dialog box.

In this case, I am using 10-point Geneva because I know that everyone will have that font, but you can use any other font and size that you like.

Close the Font dialog box by clicking OK.

Now add a button to the background. Choose the Button tool, and then hold down the ⌘ key and drag to create a button over the Contents area on the background.

Double-click the button (to bring up the Button Info dialog box) and click the Link To button. The Link To palette appears. Go to the previous card by pressing ⌘-2 and then click on This Card button. HyperCard will now jump back to the original card.

Open the script for the button (a shortcut is to hold down the Option and ⌘ keys and click on the button). You will see a script that resembles this one:

```
on mouseUp
    go to card id 2854
end mouseUp
```

Don't be surprised if the ID number is different—HyperCard randomly assigns ID numbers to objects.

Don't change the card ID number—leave it as the number that you already have.

Now add the Previous and Next buttons in the same way—except don't use the Link To command; instead, just add the following scripts:

PREV button:

```
on mouseUp
    go to previous card
end mouseUp
```

NEXT button:

```
on mouseUp
    go to next card
end mouseUp
```

These two scripts move from one card to the Next card, or to the Previous card.

Now it's time to start adding some content.

Step 3: Adding Content

The text of the story is contained in the document “QuickTime 1.5 text.” The illustrations for the article are in the following PICT files:

- Movie size
- Dithering No1
- Dithering No2
- Text track
- Comp settings

The text and images will be displayed on a series of the article cards.

You should be on the first content card of the stack. If you are currently editing the Background (the menu has the dashed lines around it), then choose Background from the Edit menu to switch back to the card.

Add a title

First, you're going to add a title. You are going to create the title in a large display typeface. Since the user might not have this typeface, it will be created as a bitmap (painting) on the card.

Choose the Text tool from the paint tools in the Tool Palette. Click next to the logo and add the text "QuickTime 1.5" (see figure 16.14). To change the size and font, press ⌘-T ; the font dialog box appears. Choose any font and size you like, but choose something reasonably large—say 30- or 40-point.

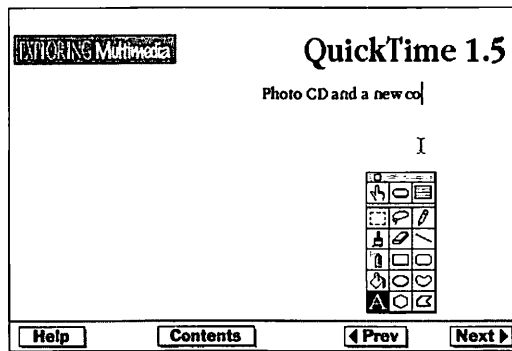


Figure 16.14. Adding the title text.

Click somewhere else on the card to “set” the text—do this instead of pressing the return key to add another line. (If you simply press return to add a second line, the new line will have the same font size as the first.) Click underneath the heading and add a second line that reads “Photo CD and a new compressor” in a smaller size of the same font.

If you don't like the placement of the text, you can move it (see figure 16.15). Use the marquee or lasso tool to select the piece of text and drag it to another location on the screen.

Copying text

Open the “QuickTime 1.5 text” file. Copying the text from your word processor to the HyperCard stack is a slightly tedious process. You have to copy some text, paste it into the field, and then add more text until the field is full.

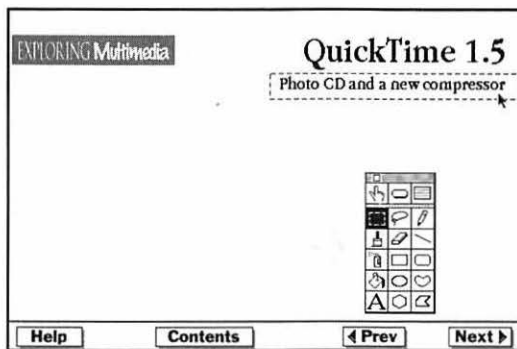


Figure 16.15. Moving a bitmapped title.

It is possible to write a script that will import a standard text file. It is then possible to place the text and work out where the text is located in the field (by knowing how many lines are visible on the screen, finding the location of a character, and then automatically flowing the text into the field).

Because you don't want to copy too much text, set up the font and size of the text in your word processor to be the same as the card. This will give you a very good idea of how much text to copy (see figure 16.16). (The process will be much easier if you can run HyperCard and your word processor at the same time.)

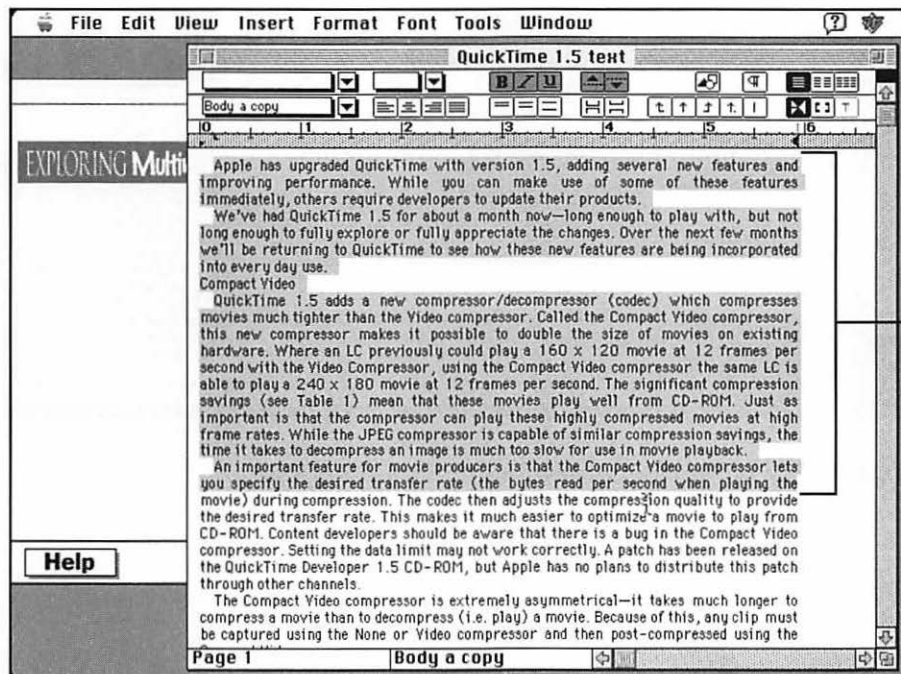


Figure 16.16. Selecting text to copy into the HyperCard stack.

Go to the card with the browse tool selected and click in the text field. (Remember, we created the text field in the background of the article card.) Paste the text into the field. You should now have a card with text, much like figure 16.14.

If you didn't paste enough text, or pasted too much, you will have to either select and copy more text, or remove some text (being careful to save the text for the next card).

In figure 16.17, too much text was selected, so the last line was removed (see figure 16.18).

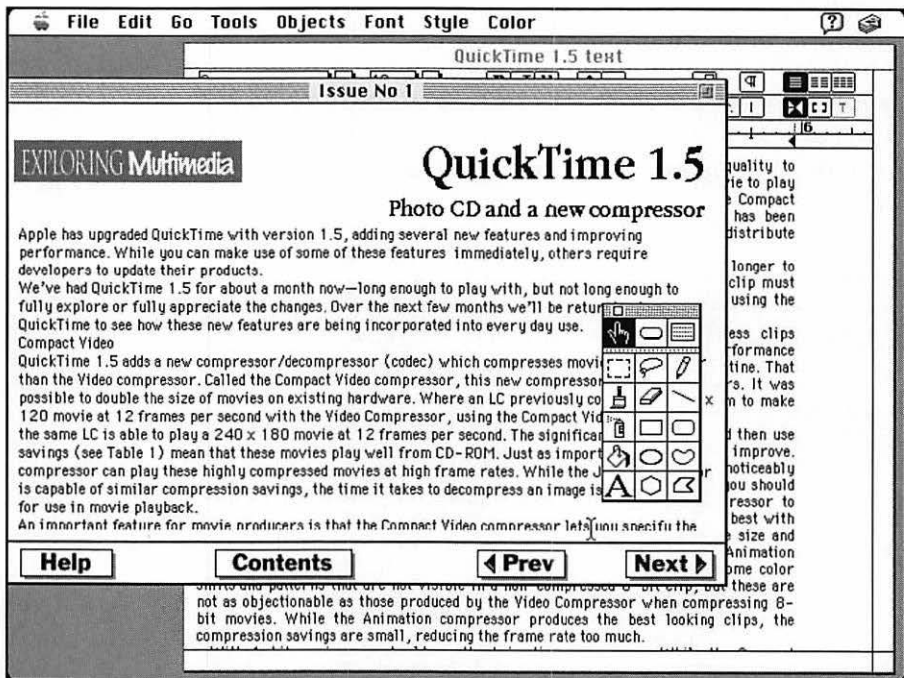


Figure 16.17. Pasting text into the HyperCard article card.

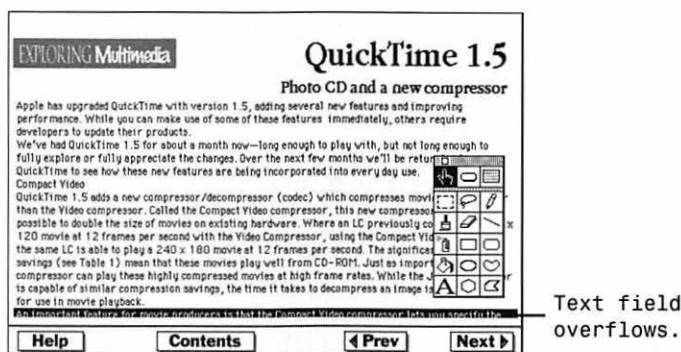


Figure 16.18. Selecting excess lines and deleting them.

Formatting text

Now that the text is in place, it would be nice to add some formatting. HyperCard does not support tabs, and no text attributes will come across (bold, italic, and so forth). You can imitate tabs with spaces, and HyperCard does support styles and other typefaces within a text field.

To imitate tabs at the beginning of each paragraph, simply go to the beginning of a paragraph and add some spaces (three or four will do—just make sure you are consistent).

To add formatting to the subtopic “Compact Video,” select the text and press **⌘-T**. The text dialog box will appear (see figure 16.19). Choose bold and close the dialog box.

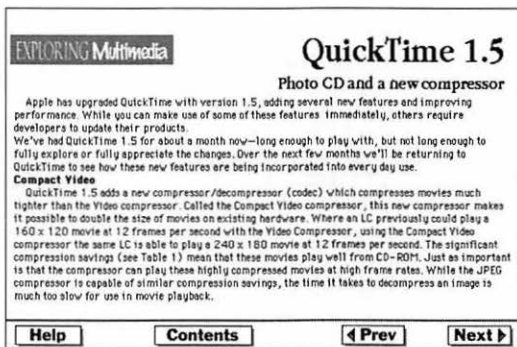


Figure 16.19. Formatted text.

Step 4: Adding a Picture to a Card

The next card has a single picture on it with a caption. The appearance of the final card is displayed in figure 16.20.

Add a new card by choosing New Card from the Edit menu. Use the Color Tools to add the graphic “Movie Size” to the card. Follow figure 16.20 as a guide. Add the caption by typing the text directly into the field below the graphic. You can do this simply by clicking at the bottom of the card (in the background text field that’s on the card) and adding the text:

The Compact Video Compressor doubles the size of movies playable on CD-ROM from 180 by 160 to 240 by 180.

Change the text style to bold by selecting the text, pressing ⌘-T, and choosing Bold for the text style.



Figure 16.20. Article card with pictures displayed on it.

Step 5: Adding the Other Cards

The third card has only text on it (see figure 16.21). You can, of course, change the layout if you like, but that’s the way the original prototype was set up. Create a new card (⌘-N), add Article Background to the PICT Info field (so that the background is used), and add the text. Remember to format it with spaces for tabs.

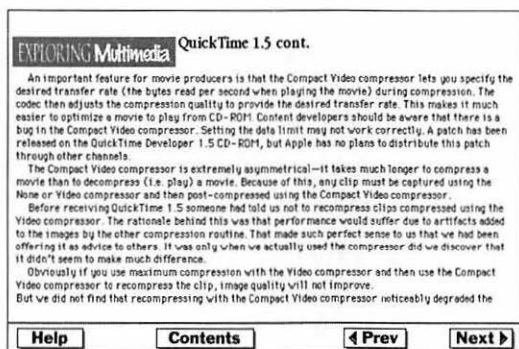


Figure 16.21. The third article card.

Remember that as you add new cards you must add the text `Article Background` to the field `PICT Info` so that the color background is displayed.

Manually wrapping text

The fourth article card (see figure 16.22) has a table. This was created as bitmapped text on the prototype. When creating tables it is sometimes easier to work with bitmapped text than to try to rearrange and align fields.

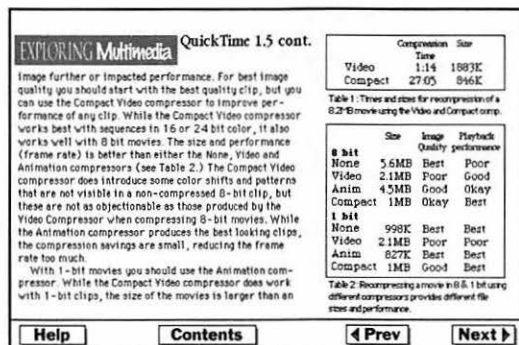


Figure 16.22. The fourth card has a table, and the text has to be wrapped.

You can recreate the table, or you can copy it—there's a copy in the stack `Table Stack`.

One problem with placing text into this card is that the text will overlap the table. There are two solutions to this problem. One is to create a unique card field which is sized correctly. This is probably the easiest thing to do—and you are welcome to do that if you want. Alternatively, you can manually rewrap the lines by placing a return at the end of each line so that the lines aren't too long (see figure 16.23).

I used the latter method because I wanted to create an automatic index of the stack; this was easier to do by keeping everything in one field. It makes searching easier if you only have to look in one field—though you could probably use a method similar to the one that we used with the graphics (by having a background field that contains the name of a card field used to place text on the screen).

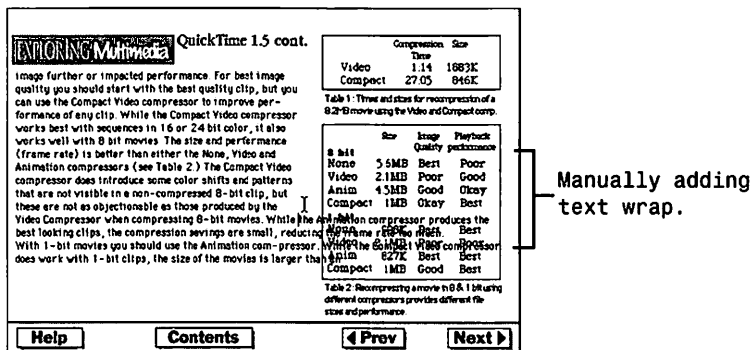


Figure 16.23. Manually wrapping the lines.

Cards 5 through 10

Finish the article by adding the rest of the text and the other graphics as necessary (as in figures 16.24 through 16.29).

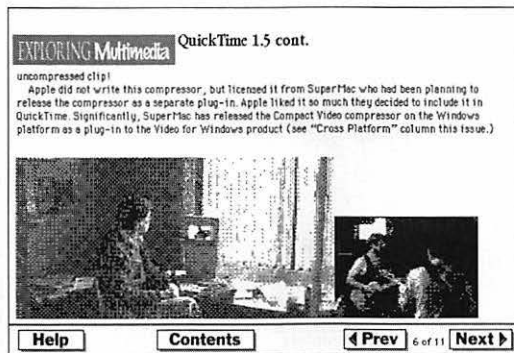


Figure 16.24. The fifth card contains graphic Dithering No1.

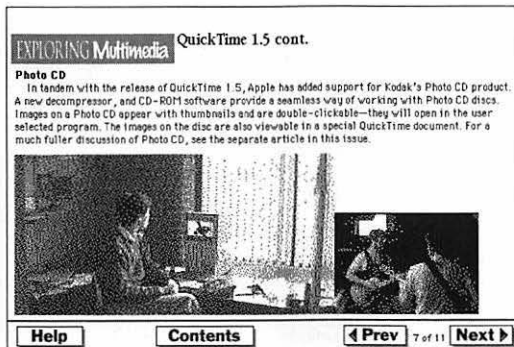


Figure 16.25. The sixth card contains graphic Dithering No2.

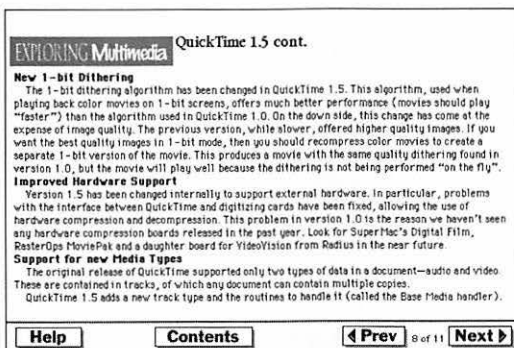


Figure 16.26. The seventh card contains text only.

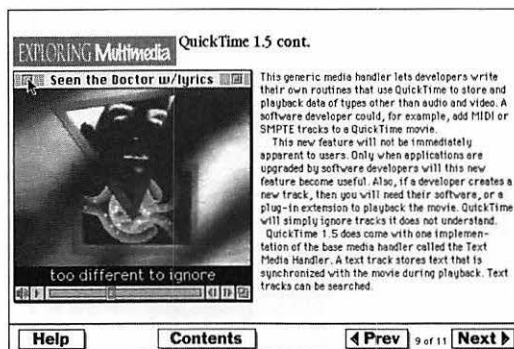


Figure 16.27. The eighth card contains graphic Text Track and text.

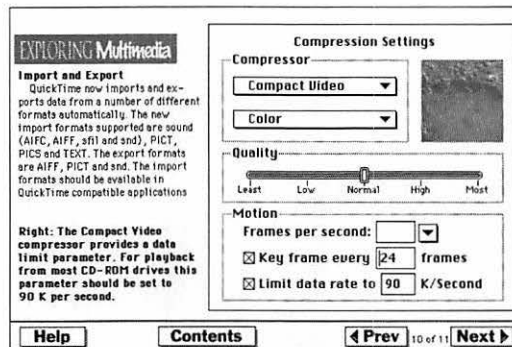


Figure 16.28. The ninth card contains graphic Comp Settings and text.

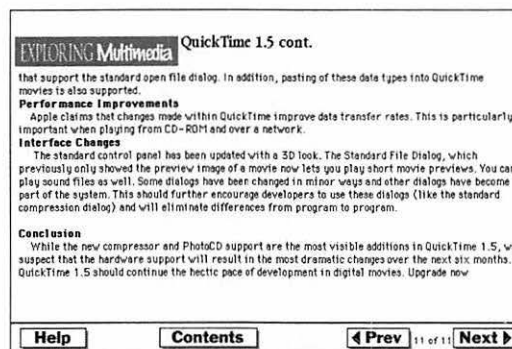


Figure 16.29. The tenth card is text only.

Step 6: Some Finishing Touches

You may have noticed that in the “final” version there is a small subheading on each card listing the title for the article. This was added as a paint text element. Also, between the Next and Previous button there is a field that contains the number of the card, and the total number of cards (2 of 11, 3 of 11, and so on).

Add the page number field as a background field. (You should center the text to make it look good.) Name the field “page.” A script is used to put the value into the field. Choose Bkgnd Info from the Objects menu. Choose Script and add the following script:

```
on opencard
    put the number of cards in this stack into number_cards
    put the number of this card into card_number
    put card_number & " of " & number_cards into background field page
    pass openCard
end opencard
```

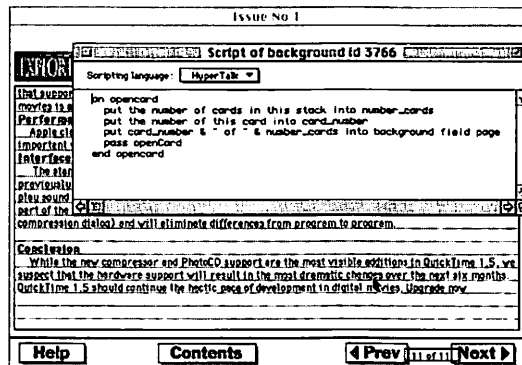


Figure 16.30. Background script which displays the card number.

Part V

This script takes the number of cards in the stack and the number of the current card, and puts these values into the background field added to the stack. The two values are separated by a string (the “ of ” part of the script).

The labels `number_cards` and `card_number` are variables. They are containers into which we can place numbers or text for use at a later point in time. In this case, the variables are used in the third line of the script.

The `pass openCard` command makes sure that the message that a card has been opened is passed along to other parts of the stack (specifically, if you had a stack script which executed when a card was opened). If the `pass openCard` instruction wasn't included, then a stack script would not receive the `openCard` message (see the HyperCard message hierarchy diagram in Chapter 15). In this example, you don't have such a script, but it's good programming practice to include this command in case you decide to add such a command.

Note that this script says there are 11 cards—which is the total number of cards including the Content card.

You should lock the text fields so that the script will execute when you click on the field (go into the Field Info dialog box for each field and click the Lock Text checkbox, as shown in figure 16.31).

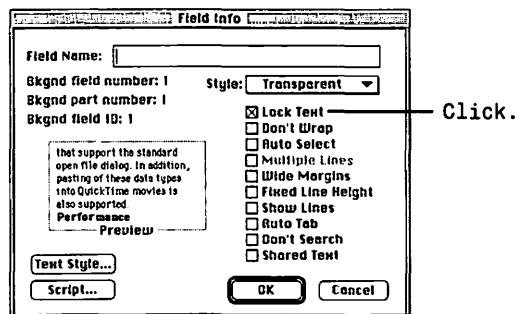


Figure 16.31. Locking the text field.

Adding a content list

You are now going to add a content list to the first card of the stack. Go to the first card and add two fields. One will be reasonably large, and will contain the titles of the articles. The second field can be smaller; it will be used to point to the cards that contain the article.

Add the two fields. They should look something like figure 16.32. Name the second smaller field “card_names.” Do this by choosing the field tool and double-clicking the field. In the Field Info dialog box type the name in the Field Name field.



Figure 16.32. Two fields added for a content list.

Go to the first card of the QuickTime article, and choose Card Info from the Objects menu. Type the name “QuickTime” into the Card Name (see figure 16.33).

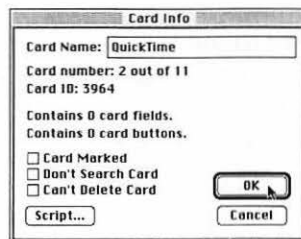


Figure 16.33. Adding a card name.

By naming the card, you can write a script which goes to that card name, rather than having to use the card ID number. You must store this name somewhere, and that's what the second field (the one named `card_names`) will be used for. Type "QuickTime" into the `card_names` field. Type the title "QuickTime article" into the first line of the larger field.

Now add a script to the first field which, when clicked, will show which line you clicked on (the "QuickTime article" title). Then the script will check the second field for the name of the card that contains the article, in order to jump to that card.

Why use the second field? You could name the card "QuickTime article" and then simply jump to the card by using the title stored in the field that we clicked on. But there's a limit to card names; you don't want to name a card "Anatomy of a multimedia disaster." By using the two fields, one containing the title, and the other the location of the article, you provide the most flexibility.

Add the following script to the first field—the one containing the title (see figure 16.34). Note: you must lock this field, or the script will not be executed when you click on the field:

```
on mouseDown
    put the clickline into line_clicked
    put word 2 of line_clicked into line_pointer
    put line line_pointer of card field card_names into the_card
    go to card the_card
end mouseDown
```

The clickline tells HyperCard to put the line that the user clicked into the variable `line_clicked`. HyperCard actually puts the message "line x of field y" where x and y are the values for the line and field clicked. You only want to know what line the user clicked; that's why the next line takes word 2 (the second word in the string "line x of field y," which will be the value of x) into `line_pointer`. The value in `line_pointer` is then used to see what is in the same line of the second field `card_names` (in this case, it will be the value "QuickTime"), and the last line goes to the card with that label.

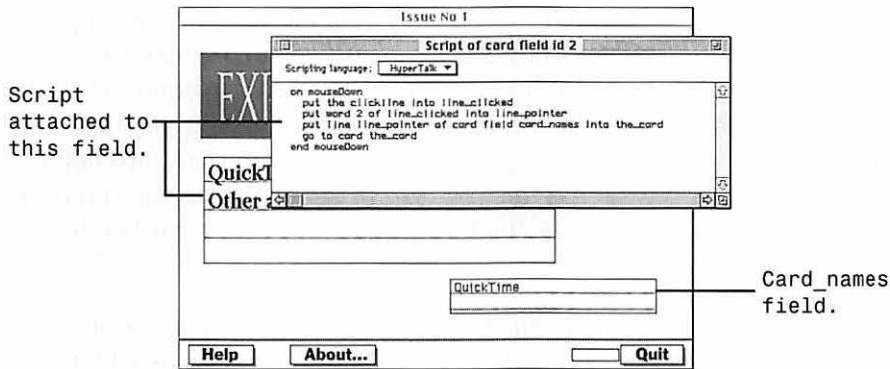


Figure 16.34. Adding the script to the field.

Now that you have this script working, you can add other entries simply by adding a new line to the first field, adding a card and naming it, and then placing that name into the second field.

Finally, you will want to hide the card field `card_names`. To do this, open the message window by entering \mathbb{H} -M. Enter the instruction:

```
hide card field card_names
```

Press the return key. The field `card_names` will be hidden.

Things That Haven't Been Done

This project could be expanded by adding sound and QuickTime movies. HyperCard supports sound (you have to place sound resources in the stack and then use the command `Play (sound name)` to play a sound). There are XCMDs for playing QuickTime movies. You can even add animation using the XCMDs for the programs Director, ADDmotion, and Animation Works. Be warned that as you add these extra functions, the stacks will become larger and more complicated, and you may have to increase the memory allocation for HyperCard again!

As a finishing touch, we have yet to add an About screen (which explains what this stack is), or even a Help screen(s). If you want to add these, you should probably create another background with a different background picture, and then add buttons to the first screen that links to these cards.

You could also add another article after the first one. One interesting problem is the Prev-Next buttons. Should they take you to the next article, or should the Prev button not be available on the first card, and the Next button not available on the last card of an article? That's for you to decide. In this prototype, clicking the Next button on the last card takes you to the next card (which is the first card in the stack), so you end up at the table of contents. Similarly, the Prev button on the first card in the article also takes you to the table of contents card.

If you do decide to change the operation of these buttons so that they only take you to the next or previous card in the article (you would use the table of contents button to exit from the article), you will have to hide the next and previous buttons in the background on the first and last card of the article. That would mean either creating two new backgrounds (without the appropriate buttons), or you could take a small area of plain color as a graphic and display that on top of the background. Even further, you could take the buttons off the background and display the buttons as separate elements. You will also have to alter the scripts of the buttons.

Performance-wise, the first option would probably be the best and easiest to do, while the second option is the most efficient—you'd only need one graphic and only have to display it on the first and last cards.

Other functionality that you might consider adding is an index or dictionary of words used in the articles. When you click on a word, the different locations are shown, and clicking on one of these displays the context of the word. Clicking on the context description takes you to that article.

Moving On

Using what you have learned here, you can now start creating your own HyperCard stacks to hold other information. You should consider getting a HyperCard book to help you learn scripting if you want to create more complex projects.

There are a lot of projects that have been created in HyperCard and other tools; you should try and see as many as possible to get ideas for implementing your own projects.

Finally, don't be afraid to dive right in and do something. As you have seen in this chapter, it is very easy to put together a simple project using a tool such as HyperCard. As long as you start with a clear design, work in chunks, and experiment as you go, you should be able to create your own projects without too much trouble. Good luck!

Putting Director to Work

17
chapter

This chapter is an extension of the previous one—it will walk you through the creation of a multimedia project using Macromedia's Director in the same way that the previous chapter took you through the creation of a project using HyperCard. But you don't have to be familiar with the previous chapter to be successful with this one. Director and HyperCard are very different programs—understanding one may give you a slight advantage, but it's not required.

However, if you plan to learn both programs, I recommend that you start with HyperCard. It's an easier program to learn, and you'll get comfortable with the scripting language (which is similar to Director's) before plunging into the depths of Director!

The Project

The sample project you are going to create is an interactive kiosk application—a tour guide for the city of London. Why London? Well, I happen to have the materials at hand (photographs, a map, etc.), and so it seemed the obvious choice. Feel free to change this example to the city, town, or household of your choice.

For this project, we'll keep it simple. There will be an opening screen, and then a Menu screen with choices that lead the user to three submodules: A Map, Calendar, and Places of Interest (see figure 17.1).

The Map screen will display a map of the city of London. Pressing a button will overlay a map of the London Underground (train system). Other buttons on the map will display information about places of interest.

The Calendar will display information about events happening in the current week. A slider will enable you to see the events for each day.

Finally, Places of Interest will present a list of places of interest—an alternative method of accessing the information available in the Map.

All the elements for this project are on the CD-ROM in the Workshop folder (within the Chapter 17 folder). There also is a finished working version in both Director and Projector formats. You can run the Projector version even if you don't have a copy of Director.

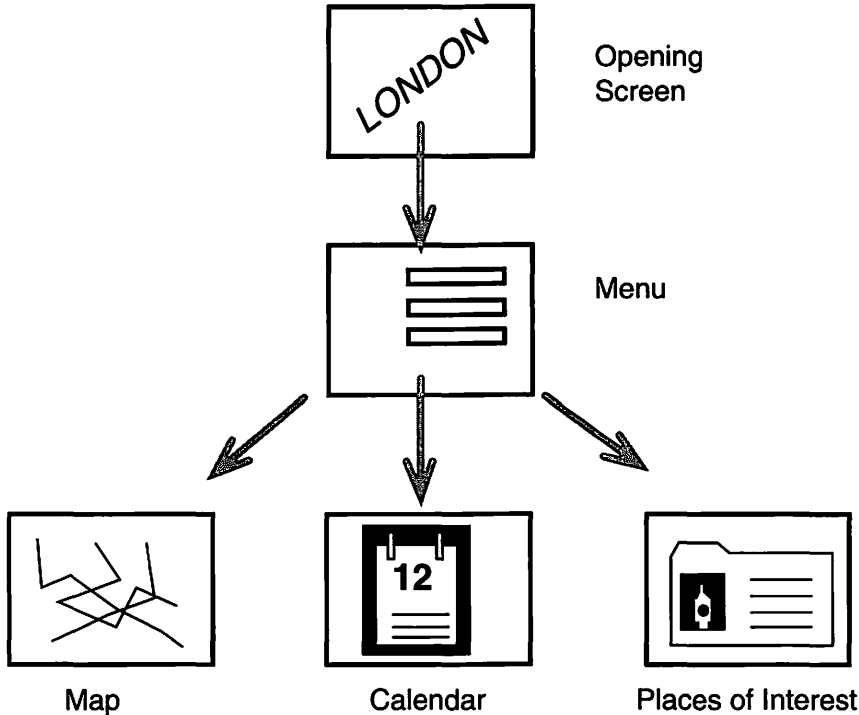


Figure 17.1. The screens that make up the project.

While this project takes you through the development of a multimedia project in Director, it should not be considered a lesson in animation using Director. Creating animation is a very different task, and it is not necessary for the project that you are developing here.

Getting Started

Where to start? The first thing you need to do is to create a sketch of the project (see figure 17.1) and a list of features. You also need to gather your materials—images, sounds, QuickTime movies, etc. Once you’ve done that, it’s time to start creating.

It’s important to remember that in Director the Cast is where things are stored, and the Score is where things are arranged chronologically. I strongly encourage you to read the description of Director in Chapter 15 before proceeding.

Open a new document in Director (or simply launch the program and it will open a new document for you). Before you import any graphics, make sure that the size of the Stage (the place where graphics appear onscreen) is the correct size. It will default either to the size of the monitor, or to the size of the last movie that you had open. Since all the graphics for this project have been created for a specific size—512 by 342—you will change the Stage to that size.

Choose Preferences from the File Menu. In the upper left-hand corner of the dialog box, choose 9-inch monitor as the Stage size (see figure 17.2). Notice that the 9-inch monitor size equals 512 by 342 pixels. You should also center the Stage by choosing Centered (top right in the dialog box).

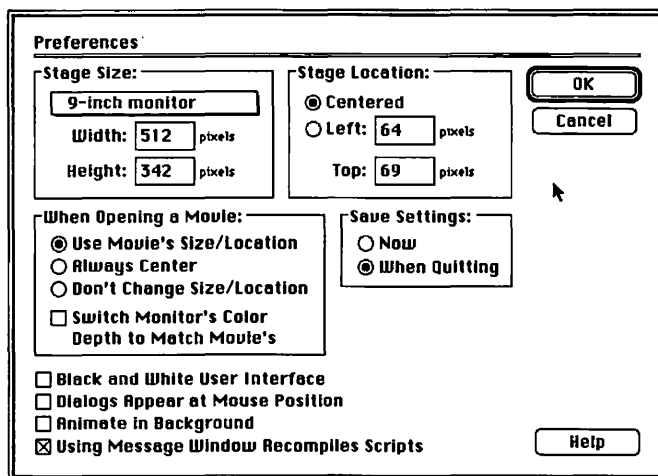


Figure 17.2. Choosing Stage size.

Importing graphics

Now that you have chosen the Stage size, you'll import the graphics that make up the opening screens. The opening will consist of a sequence of screens (with dissolves between each one) that lead to the menu screen.

Choose Import from the File menu. A dialog box appears asking you to choose a file or multiple files to import (see figure 17.3). Notice the pull-down menu for file types; it defaults to PICT files. If you click on the menu

you will see all the file types that Director can import. While Director can import many different file types, you must choose the file type you are importing before you can choose the file. Notice that you can import all the files in the current folder (of the same type) by choosing Import All.

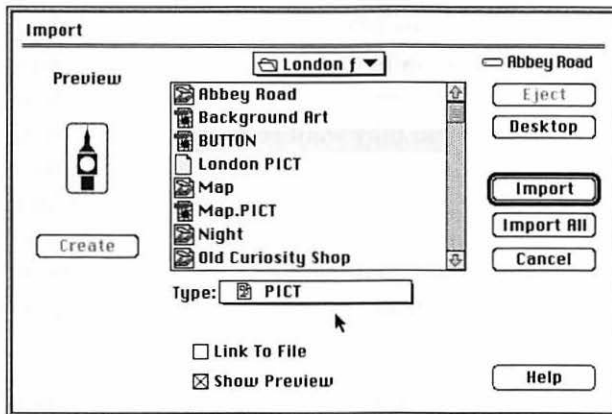


Figure 17.3. The Import File dialog box.

You can import the file Background Art by choosing it and clicking the Import button. Director will import the file and place it in the Cast window. Director opens the Cast window automatically to show you the file it has imported (see figure 17.4).

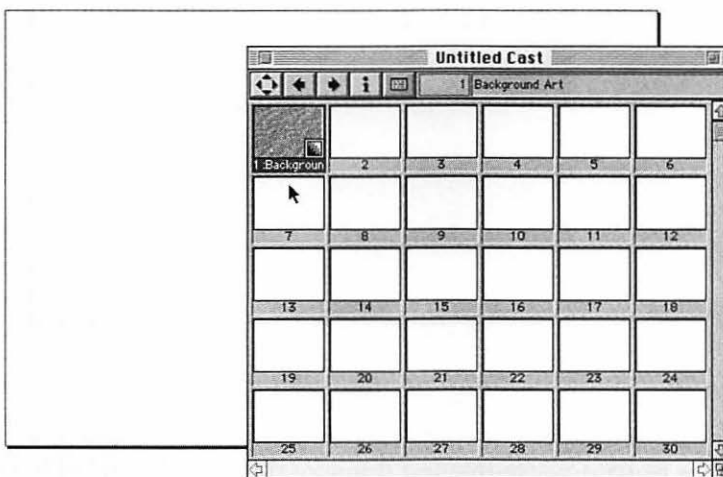


Figure 17.4. The Background Art imported into the Cast.

Placing a graphic on the Stage

To place this graphic on the Stage, simply select it in the Cast window (the cursor should change to a hand) and drag the cursor onto the Stage window. A dotted line should appear representing the graphic. Release the mouse button and an image of the cast member should appear on the Stage. The image of a cast member on the stage is referred to as a sprite. To center the sprite on the Stage, you can click and drag the selected sprite until it is centered (you may want to close the Cast window while you are doing this), or use the arrow keys to tweak the location, or choose Sprite Info from the Score menu to set the location numerically. As an alternative to centering a sprite after it has been placed on the Stage, select the cast member in the Cast window and choose Cast to Time from the Cast menu. This automatically places and centers the sprite on the stage.

Now open the Score by choosing Score from the Window menu. Notice that there is a character in the first frame of the first channel of the Score. This represents the cast member that you just placed on the Stage.

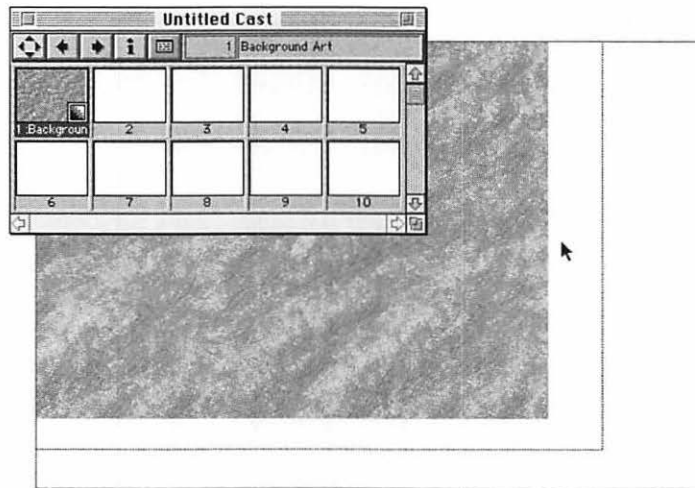


Figure 17.5. Placing the graphic on the Stage.

Now open the Score by choosing Score from the Window menu. Notice that there is a character in the first frame of the first channel of the Score. This represents the cast member that you just placed on the Stage.

But this graphic is only on the Stage for one frame, and you want to use the graphic in several frames as a background for your opening. What you want to do is to extend the graphic over several frames—there are several ways you can do this.

1. You could place the graphic manually into each frame. To do this you would go to frame 2, drag the graphic into position, go to frame 3, and so on. This would become rather tedious.

Note

How do you go to the next frame in a Director movie? Like many other aspects of Director, there are several ways to go from frame to frame in a Director movie. One way is by simply clicking in a channel of the Score on the frame that you are interested in. You also can click in the Frame channel at the top of the Score to move the playback head. (The playback head is a black bar that travels in the Score as a movie plays indicating the Frame being displayed.) You don't have to move in the playback head; clicking in any cell in the Score will have the same effect (though you may select something in the Score when you do this).

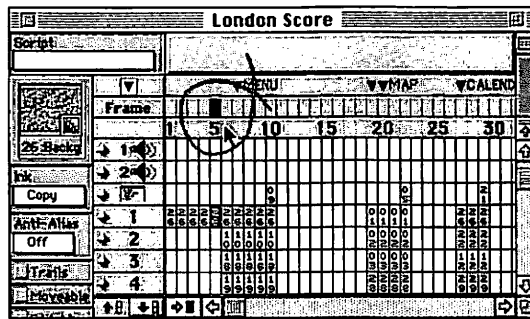


Figure 17.6. The Playback head.

2. You can cut and paste within the Score. You can copy either a single cell or several cells in the Score and paste them at another location. To do this, simply click on the cell(s) you want to copy, choose copy from the Edit menu, select the cell you want to copy to, and choose Paste. You can do this repeatedly, but it would get tedious.

3. You can use Director's In-Between feature to repeat the same graphic across multiple cells. In-Between is designed to help in the creation of animations. Imagine that you want to have a ball move across the screen. You know where you want the ball to start and stop. Generally, you have to create all the in-between frames by manually dragging the object to a new location for each frame.

The In-Between feature does this for you. You place an object in frame one, copy it to frame twelve, and reposition for the final position of the object. The object must be in the same channel or this won't work.

After clicking and dragging in the Score to select the range of frames, choose In-Between Linear from the Score menu. Director will automatically create the in-between locations (by dividing the distance traveled into equal segments). In-Between Special provides added features, enabling you to add acceleration or deceleration effects to the movement.

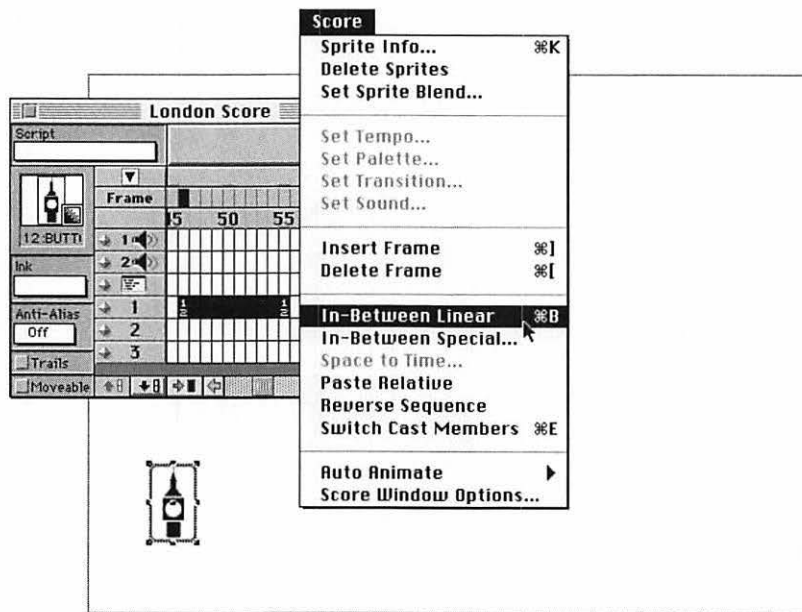


Figure 17.7. Creating In-Between animation.

One more tip—If the object stays in one place, you don't need to create the last frame location. Simply choose the first frame in the Score and the

frames you want to place the object in. Choose In-Between Linear (⌘-B). Director will copy the object, in the same location, across the selected frames.

Use In-Between Linear to copy the background across the first five frames of the movie.

Then import the following graphics: Old Curiosity Shop, Westminster Abbey, Night, and London PICT.

Arranging more objects

Now that you have several objects in the Cast, you want to create a series of frames that build into the final screen—that is, one image after another is added to the screen until the final effect is seen.

The easiest way to do this is to build the final screen, and then create the sequence that leads up to that screen. To do this, click somewhere in the fifth frame of the movie so that the playback head moves to that frame in the Score. Drag the other graphic elements onto the Stage and arrange them as you like.

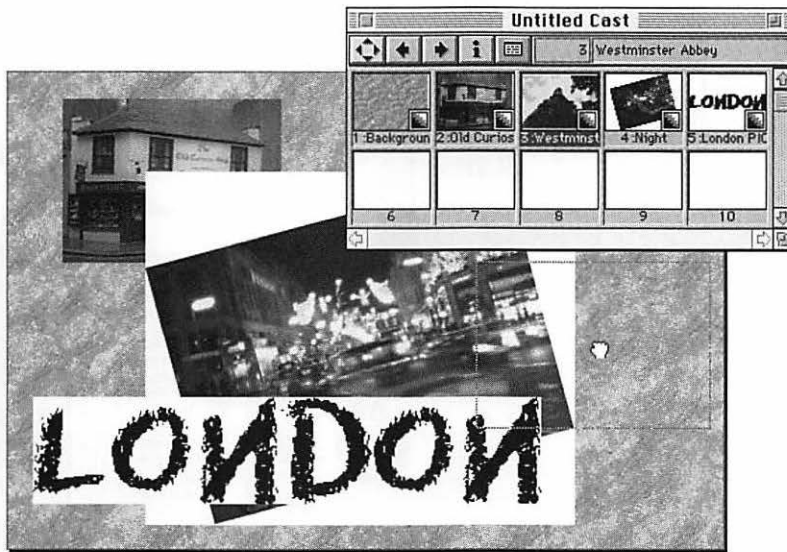


Figure 17.8. Arranging the objects.

You may have noticed that while the Westminster and Curiosity shop graphics look acceptable, the Night image is surrounded by a white matte. This is a result of the ink applied to the sprite. Ink effects define the appearance of the sprite on the Stage, and in this case, the Copy ink effect is chosen by default. Choosing Matte ink will remove the white border.

There are two ways to change the ink effect: either open the Score, select the object in the Score, and then choose the ink effect from the Ink pop-up menu on the left-hand side of the Score window (see figure 17.9). Or, you can use a shortcut—click the sprite on the Stage. The sprite should be selected (a dotted line appears around it). Release the mouse button; then holding the ⌘ key down, click on the sprite again. A pop-up menu appears, enabling you to choose the ink for the sprite.

Notice that by choosing an ink effect in the Score you can change the ink for multiple sprites in multiple frames (simply select them all first); changing it on the Stage only changes the selected sprite in the current frame.

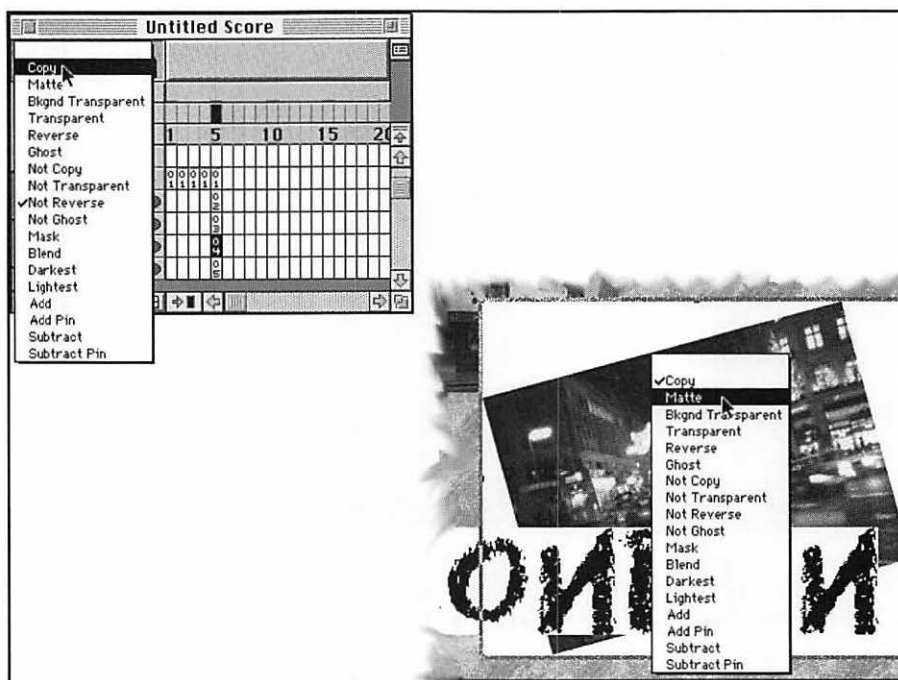


Figure 17.9. Choosing an ink effect.

Choose the Ghost ink effect for the London title. This should change the title to white. You should end up with something that resembles figure 17.10 (it doesn't have to be exact). A Director movie has been saved on the CD-ROM called Step 1.

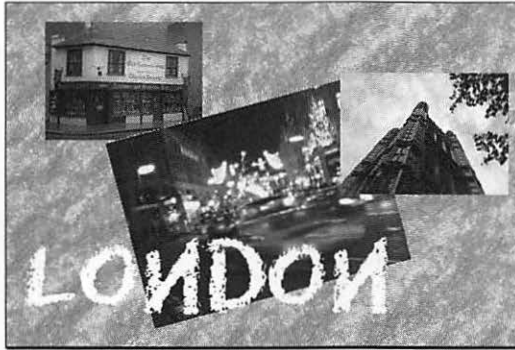


Figure 17.10. Choosing the Ghost ink effect.

Note

Having trouble selecting objects on the Stage? Director has this annoying habit of not letting you select an object that is on top of a currently selected object. For example, say you click and select the Background graphic on the Stage. You will not be able to select another graphic by clicking it. To deselect the object, you must open the Score window and click elsewhere in the Score (to deselect the object).

Remember this if you ever find yourself trying to select an object and nothing seems to happen.

Creating the opening

The opening will start with the background in frame one. The second frame will add one graphic to the background; the third will add a second; the fourth adds a third graphic; and then the title will appear at frame five. You already created frame five, so now it is reasonably simple to place the graphics in the other frames and achieve the desired results.

Open the Score. You should see the first few frames of the movie. Click on one of the graphics in frame five—other than the background—step 1 in Figure 17.11. Choose Copy Cells from the Edit menu (⌘-C). Select the same row (channel) in the Score at frame 2 (step 2 in Figure 17.11) and choose Paste Cells (⌘-V). The object has been pasted at frame 2 (step 3 in figure 17.11). Then click and drag to select the range of frames for the graphic (step 4 in figure 17.11). Choose In-Between Linear from the Score menu (step 5 in figure 17.11). The object now appears in frames 2 through 5.

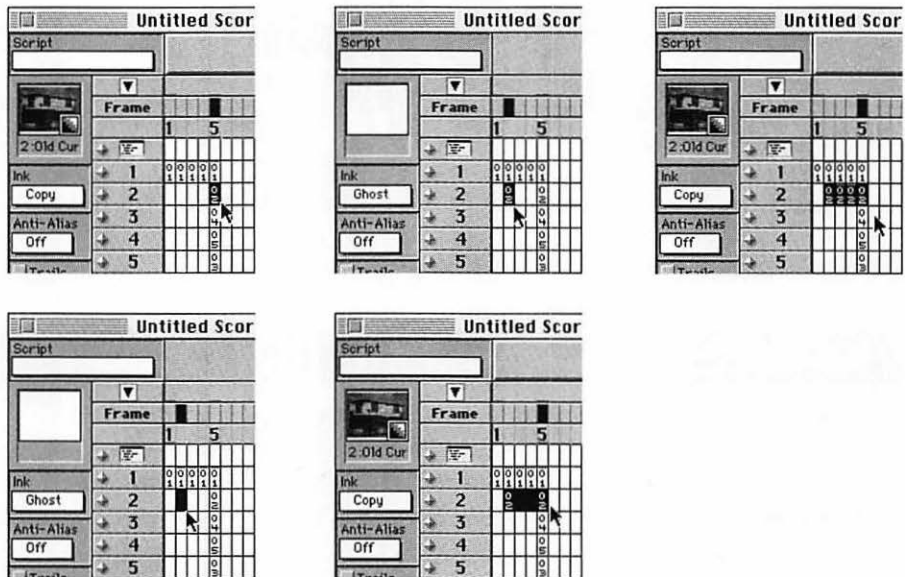


Figure 17.11. Placing the object into multiple frames.

Repeat the process for the other objects—except go to frame 3 for one, and frame 4 for the other—obviously you won't need to use In-Between for the last two objects. Your movie should now look like figure 17.12 (saved as the Step 2 movie on the CD-ROM).

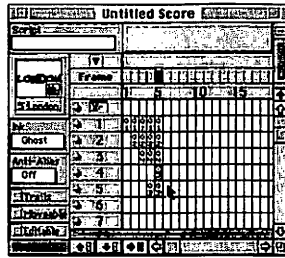


Figure 17.12. All objects arranged for the opening.

Adding effects and setting the Tempo

Try running the animation. Choose Rewind (⌘-R is the shortcut) then choose Play (⌘-P). You may want to hide the Score and any other windows while doing this. A shortcut to hide windows is ⌘-1. Pressing ⌘-1 a second time will display them again.

What happens? The animation probably runs extremely fast—much faster than you want, and you'd like to add a transition effect to give the opening more impact.

Note

If the movie is still running, then you have the loop option in the Control Panel turned on. Press ⌘-. (period) to stop the movie.

Open the Score window and scroll up so that you can see the six channels used specifically for (top to bottom): tempo, color palette, transitions, sound, and scripts (see figure 17.13).

Tempo is used to set the frame rate (frames per second), a delay (wait for X seconds), or even to wait for a specific action (the user clicks, or a sound finishes playing). The color palette is used if you have graphics that use adaptive palettes. This project doesn't use any, so don't worry about it at the moment. The transition channel is used to indicate the transition that occurs when the selected frame is displayed. The script channel stores scripts that are executed while a frame is displayed.

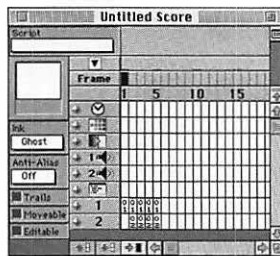


Figure 17.13. The channels at the top of the Score window.

First, you want to extend the background into the sixth frame (which is where the menu will be located). Copy the background into frame 6.

Set the Tempo. Double-click in the Tempo channel of the fifth frame of the movie. The Set Tempo dialog box appears. Adjust the Wait slider to three seconds (see figure 17.14). Run the movie. It should play just as fast as before, but will pause on the fifth frame before proceeding to the sixth (which only contains the background).

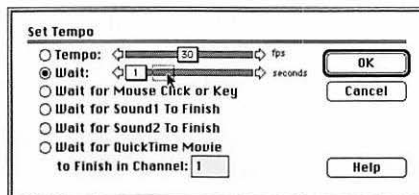


Figure 17.14 The Set Tempo dialog box.

Then apply a transition. Double-click in the transition channel for the second frame of the movie. When the Set Transition dialog box appears (see figure 17.15), choose a transition. Now play the movie.

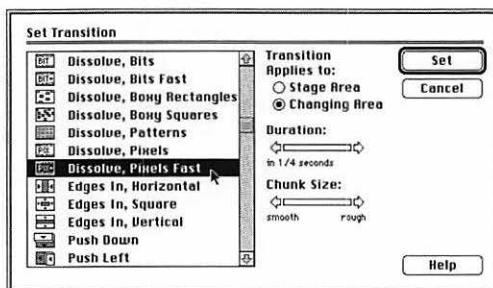


Figure 17.15. The Set Transition dialog box.

What happens? The movie will play as before, but there should be a transition between frame 1 and frame 2. Make sure that there is a difference between frame 1 and frame 2 of your movie—there should be an additional graphic in frame 2—if the frames are the same, the transition will happen, but you won't see it.

Apply transitions to frames 2 through 6. Play the movie. Experiment with the transition effects. Your movie Score should resemble figure 17.16 (movie step 3).

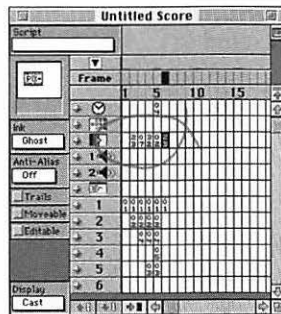


Figure 17.16. Transitions added to the movie.

Adding Scripts

Now comes the fun part. You are going to create the Menu screen. Open the Paint window (⌘-5). The window will open showing an existing graphic in the Cast. You want to create a new graphic. Click the Add button with the plus sign at the top left of the Paint window. A new blank cast member will be displayed in the Paint window.

Use the paint tools to create a button (see figure 17.17). Use the text tool to type the title. Then use the rectangle tool to create the button shape (or use whatever shape you want). Create a button for Map. When you are finished, click the Add button again and create another button for Calendar, and repeat the process for a button for Places of Interest.

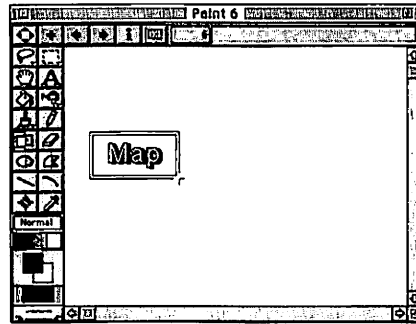


Figure 17.17. Creating a button.

If you open the Cast window, you should see the new objects there. You want the objects to appear in frame 6, so drag them to that frame and arrange them on the Stage. Then extend both the background and these buttons into frame 7.

Frame 7 is where you are going to ask for user input (i.e., the user clicks a button to make something happen). In this case, the user clicks the graphic buttons you have just created. You will make the Director movie loop on frame 7 while it is waiting for user input. Why not just redisplay frame 6? Well, frame 6 contains a transition, and transitions take time. If you looped in frame 6, then the transition would happen again and again. While the transition is happening, any user input is ignored—so the user would click and nothing would happen.

Now that you have extended the Score, let's add a frame script that loops. At frame 7, click in the Script channel. A Script window appears (see figure 17.18). The script already has a handler name (on exitframe) and the end of the handler (end); all you have to do is add the script between those two lines. Add the following line: go to the frame. The script should look like this:

```
on exitFrame
    go to the frame
end
```

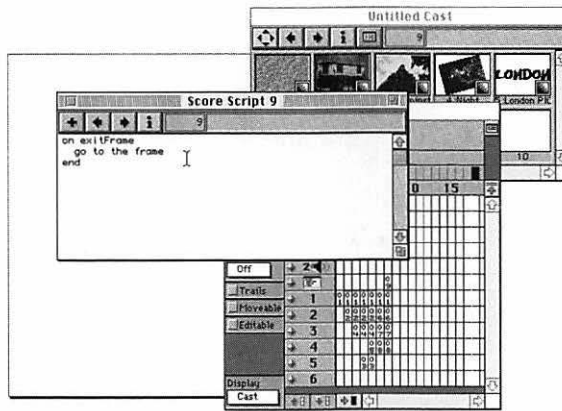


Figure 17.18. Adding a script to the Score.

Click in the Close box of the Script window. Play the movie. What happens? It should stop at frame 7. What does the script do? The instruction “go to” tells Director to go somewhere. The second part of the instruction tells Director to go to “the frame,” which is a variable that contains the current frame number. As a result, Director keeps going back to the current frame. This movie is labeled step 4 on the CD-ROM.

Navigating within a Movie

You’re now going to go somewhere interesting. Let’s create a new frame at frame 10. (Leave the in-between frames blank—you aren’t going to use them.)

Import the graphic Map PICT and place the cast member in frame 10. Now add a label to the frame (see figure 17.19). To do this, go to the label well (it’s the downward pointing arrow above the word “frame” in the Score window). Click and drag the label well to the right in the Score—an arrow (or label) will follow the cursor. Place the arrow in frame 10, release the mouse, and type Map. The label named Map should appear next to the label.



Figure 17.19. Adding a label to a frame.

The nice thing about a label is that you can add a script that says “go to frame *labelname*.” You also can say “go to frame 10.” This would have the same net effect, but, if you were to add or delete frames in the Score, then your instruction would be incorrect (frame 10 in the Score would no longer contain the frame you were interested in). Labels solve this problem because they move with the frame if other frames are added or deleted.

Now click the Map button in frame 7 to select it. Click in the pop-up menu labeled Script at the top left in the Score window. Choose New and a score script window will appear. Add the instruction “go to frame Map.” The script window should look like this:

```
on mouseUp
    go to frame "Map"
end
```

Then close the Script window and play the movie. What happens? When you click the Map button, the movie should jump to the Map frame. You want to pause there, so add the “go to the frame” statement to frame 10.

But you need to get back! Let’s add a button to go back to the Menu. Import the graphic London Button and place it in frame 10. Add a score script for this button that goes back to frame 7. Or, you could add a label “Menu” to frame 7 (movie step 5) and use the *labelname* instead of the frame number.

Adding the underground

You have a map of the underground that you are going to overlay on the map. Note that the map doesn't closely match the existing street map. (This is a demo after all!)

Import the graphics Tube Map and Tube Button. Import the Tube Button twice (two copies) and add the text “Show Tube” to one and “Hide Tube” to the other. Do this in the space below the Tube logo in the Paint window.

Place the Tube Map in frame 10. Make sure the ink is set to Transparent. You will have to move the London button to a lower channel number if the Tube map is displayed over the top of the button. Add the Hide Tube button to the screen (see figure 17.20).

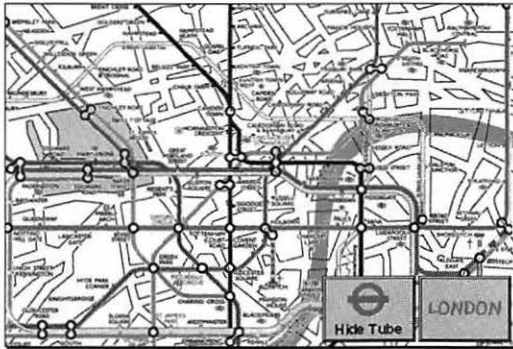


Figure 17.20. Adding the Hide Tube button to the screen.

Now you are going to add a script to a cast member. So far, you have added scripts to sprites in the Score. A script attached to a cast member will work anywhere the cast member appears in the Score.

To add a script to a cast member, select the cast member in the Cast window and then choose Cast Member Info (⌘-I) from the Cast menu. The Cast Member Info dialog box will appear (see figure 17.21). Click the script button to add a script.

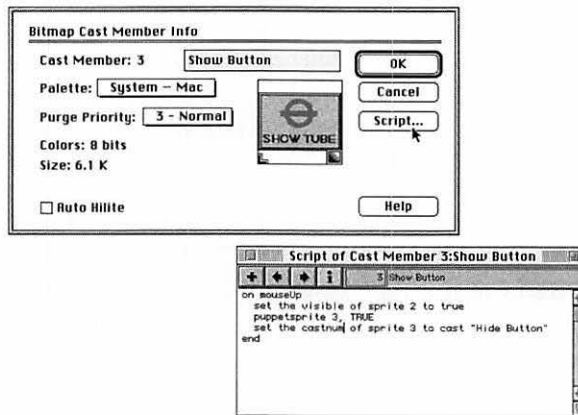


Figure 17.21. Adding a script to a Cast Member.

Notice that in the Cast Member Info dialog box you can also name cast members. Name the Show Tube button “Show Button” and add the following script to the Show Tube cast member:

```
on mouseUp
    put the clickon into me
    set the visible of sprite 3 to true
    puppetSprite me, TRUE
    set the castnum of sprite me to cast "Hide Button"
end
```

Add the following cast script to the Hide Tube button (and name it “Hide Button”):

```
on mouseUp
    put the clickon into me
    set the visible of sprite 3 to false
    puppetSprite me, TRUE
    set the castnum of sprite me to cast "Show Button"
end
```

What is this script doing? Simply put, you are hiding or displaying the channel that contains the UnderGround Map graphic, and switching which button is being displayed.

Take the Hide Button script as an example. The first line (put the clickon into me) finds out which channel the button is located in. This is used later on. The next line (set the visible of sprite 3 to false) turns off the visibility of channel 3. This assumes that the UnderGround Map is in channel 3 of the Score. If it isn't, either move it, or change the numeral 3 in the script to the channel containing the Map.

The next line (puppetSprite me, TRUE) is turning on the puppetSprite property of the channel "me." (Remember "me" contains the channel you just clicked in, therefore, it is the channel that this button is in.) The puppetSprite property must be turned on if you want to use scripts to move objects in Director.

The last line switches the cast member in the channel to the cast member "Show Button." The Show Button script performs the reverse effect.

Now play the movie. What happens? You should see the tube map. When you click the Hide Tube button, the map should disappear, and the button should switch (movie step 6).

Problems!

But wait! This isn't working out exactly right. When you return to the menu you may have noticed that the Tube button remains visible. Also, if you had hidden the tube map, then one of the buttons in the menu screen is probably not visible. Why? The puppetSprite property essentially takes away control from the Score and gives it to scripts you write. Therefore, if you turn on the puppetSprite property, then nothing the Score does will affect that channel until you turn off the property. Similarly, the Hide Tube button hides one of the channels. If you don't turn it back on, then all sorts of undesired effects can happen.

To fix these problems, add the following sprite script to the London Button (do this in the Score, not to the cast member):

```
on mouseUp
    set the visible of sprite 3 to true
```

```

puppetSprite 5, FALSE

go to frame "Menu"

end

```

Note

The channel numbers may not be the same for your movie. Step 6a contains the fixed movie.

Adding an information page

You're now going to add some information screens to the map. Import the graphics Button, Abbey Road Info, and Westminster Info. Place the Button graphic onto the Map in two locations. Then drag one of the Info cast members onto the stage (see figure 17.22).

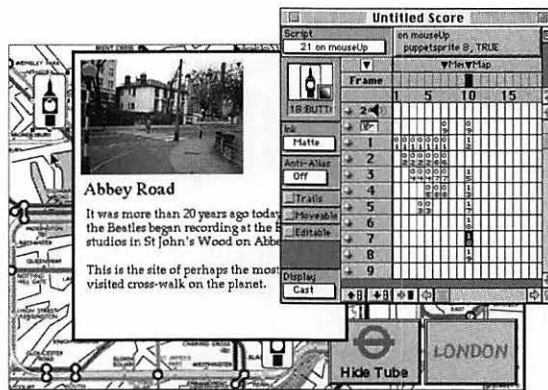


Figure 17.22. Adding an Info graphic.

This feature will work much like the Map feature—except that you will show and hide a channel that contains the Info graphic, and you'll switch the channel to the graphic you want. You'll also set the location of the graphic.

Make note of which channel the Info graphic is located in. In the script for the Map button, turn off the channel (set the visible to false) so that when you go to the Map frame the Info graphic won't be visible.

In the Score, attach the following sprite script to one of the button graphics:

```
on mouseUp
    puppetSprite 8, TRUE
    set the castnum of sprite 8 to cast "Westminster Info"
    set the locH of sprite 8 to 250
    set the locV of sprite 8 to 150
    set the visible of sprite 8 to true
    updatestage
end
```

Attach this script to the other one:

```
on mouseUp
    puppetSprite 8, TRUE
    set the castnum of sprite 8 to cast "Abbey Road Info"
    set the locH of sprite 8 to 200
    set the locV of sprite 8 to 150
    set the visible of sprite 8 to true
    updatestage
end
```

Note that you are turning on the puppetSprite property, switching the graphic, setting the locH and locV (the horizontal and vertical location of the graphic relative to the upper left corner of the stage), and making the graphic visible. The updatestage command simply tells Director to redraw the screen.

To the London button, add a command to turn off the puppetSprite property, and to the Info graphic in the Score, add the following:

```
on mouseUp
    set the visible of sprite 8 to false
end
```

This hides the graphic if the user clicks on it (movie step 7).

Advanced Scripting

The Calendar uses many of the ideas you have seen already, but it has two additions—it enables the user to drag a button across the screen, and it updates a graphic based on the location of the button (see figure 17.23).

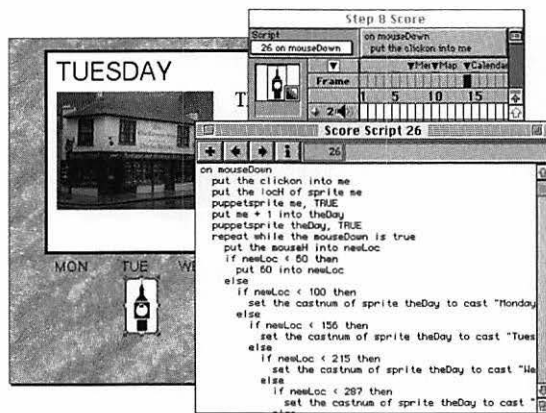


Figure 17.23. Implementing the Calendar.

This is actually easier than it seems. You've already set the location of an object using the Info graphic. You can update a graphic based on a button location by getting the mouseH (the current location of the cursor) and setting the locH of the graphic to a corresponding location. This limits the button to only horizontal movement. If you wanted it to move in all directions, you would have to use the mouseV to set the locV as well.

Look in movie 7 on the CD-ROM at the script for the button in the Calendar. You'll see that a repeat loop is used:

```
repeat while the mouseDown is true
    ...
end repeat
```

The statements inside a repeat loop are repeated as long as the conditions specified are true. In this case, the repeat loop is executed until the mouse is released (or conversely, it is executed while the mouse is down). Inside this loop you set the new location for the button and update the graphic.

Updating the Calendar is simple. There are seven graphics labeled Monday through Sunday, and the routine selects one to display depending upon the horizontal location of the button. This is where a number of “if” statements are used to find the location of the button.

The numerical values used were generated by first creating the sliding button. Then an instruction (put the mouseH) was placed inside the repeat loop, and the Message window was opened. Running the routine, Director displays the location of the cursor in the Message window as you drag the cursor around. You can use this to find the approximate location for each test.

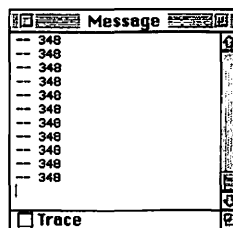


Figure 17.24. Displaying the MouseH in the Message window.

Here is the complete script:

```
on mouseDown
    put the clickon into me
    put the locH of sprite me
    puppetSprite me, TRUE
```

continues

Continued

```
put me + 1 into theDay
puppetSprite theDay, TRUE
repeat while the mouseDown is true
    put the mouseH into newLoc
    if newLoc < 60 then
        put 60 into newLoc
    else
        if newLoc < 100 then
            set the castnum of sprite theDay to cast "Monday"
        else
            if newLoc < 156 then
                set the castnum of sprite theDay to cast "Tuesday"
            else
                if newLoc < 215 then
                    set the castnum of sprite theDay to cast "Wednesday"
                else
                    if newLoc < 287 then
                        set the castnum of sprite theDay to cast "Thursday"
                    else
                        if newLoc < 346 then
                            set the castnum of sprite theDay to cast "Friday"
                        else
                            if newLoc < 407 then
                                set the castnum of sprite theDay to cast "Saturday"
                            else
                                set the castnum of sprite theDay to cast "Sunday"
                            if newLoc > 443 then
                                put 443 into newLoc
```

```

        end if
    end if
end if
end if
end if
end if
end if
end if
set the locH of sprite me to newLoc
updatestage
end repeat
end repeat

end

```

Note two things: This script is a MouseDown handler, not a MouseUp handler. If you used a MouseUp handler, then the script would never work (because the repeat loop is dependent upon the mouse still being depressed)! Also, the updatestage is required because while the mouse is down, Director won't redraw the screen unless you tell it to by using the updatestage command.

Where to from Here?

Add the Places of Interest screen. This could be an alphabetical list of places that you click, or it could be a directory of images that you click. (It's up to you!)

Using the basics you have learned here you should be able to create quite sophisticated productions. Even if you only use the “go to frame” command, you will be able to create interesting interactive presentations.

What's on the Disc



The CD-ROM included with this book consists of a wide range of things—from demos and sample files to images and QuickTime movies.

Different applications and files on the disk have different requirements; I recommend a **color Macintosh** with **System 7.0** and **4 MB of memory** (and a **CD-ROM drive!**) as a minimum. Some of the applications require more memory, and some must be copied to your hard disk. The QuickTime movies must be viewed with an application that can play QuickTime movies (such as CameraMan's MoviePlay and Adobe Premiere). You also must install the QuickTime extension, which is included on the disk. Some demos and the "Putting it to work" files require HyperCard (not included).

Some of the applications on this disc are shareware. These applications are provided to you on a trial basis. If you like them, you should send the appropriate shareware fee to the author of the application. (Read the Read Me files with each application to find out more.)

The purchase of the *Multimedia Starter Kit for Macintosh* does not include purchase of these programs. Please support shareware authors.

The following is a short description of the contents of the disc.

Tour the Disc

This is a QuickTime movie that is a guide to the contents of the disc. The movie was captured using CameraMan (see Chapter 10, "QuickTime") and the narration was added using the CameraMan editor. Note that MoviePlay (a utility for playing QuickTime movies that is distributed with CameraMan) is included on the disc and can be used to play this and other QuickTime movies.

QuickTime

This is version 2.0 of the QuickTime extension, and also includes the QuickTime PowerPlug (for Power Macintosh computers) and the musical instruments file (for MIDI support). Drag these to your System Folder to

install them (you'll have to restart your computer, too). You don't need the PowerPlug unless you are running a Power Macintosh. You must be running a machine that supports QuickTime (see Chapter 10, "QuickTime," for more information).

Cover Art

This is a PICT file of the cover of the book. Why is it here? Well, we had to put it somewhere!

Software Developers

This folder contains demo versions or sample files that were supplied to us by various multimedia software developers. The inclusion of a product in this folder does not constitute an endorsement from us. Rather, these are the companies that made material available to us within our schedule.

Vividus

Samples and a self-running demo of Cinemation, the animation and presentation program. Copy the folder to your hard disk to run the demos.

Specular International

Demo version of Infini-D 2.6, and LogoMotion, as well as sample images, QuickTime movies, and an interactive presentation on Infini-D.

Playmation

Sample images and movies generated using Playmation, a 3D modeling tool. (Movies must be played with a QuickTime movie player.)

No Hands Software

Copy of Common Ground and demonstration documents. Common Ground is an electronic publishing tool.

Virtus

WalkThrough is a 3D modeling application. This folder contains sample images and movies created using WalkThrough. Also enclosed are three Voyager documents. These are 3D models with the playback software from WalkThrough included. They enable you to explore the 3D world, but you can't manipulate it.

Heizer Software

Demo versions of MasterScript, WindowScript, and CompileIt! These tools provide added functionality to HyperCard stacks.

Fractal Design

Demo version of Painter 2.0. Copy the folder to your hard disk to run the demos.

Multiple Media Tour

Catalog for the Multiple Media Tour, a multimedia clip art collection from Audio Visual Group.

Strata

Sample images generated using Strata 3D.

Adobe

Demo versions of Premiere 4.0, Photoshop 2.0, and Illustrator 5.0. Also included are some sample movies created using Premiere. Copy the folders to your hard disk to run the demos.

Morph

Demo version of Morph. Use this program to create magical transitions from one image to another.

PROmotion

Demo version of this animation package.

VideoFusion

Sample movies and an interactive presentation about VideoFusion. (Movies must be viewed with a QuickTime movie player.)

VideoToolKit

A demo movie created using VideoToolKit. (Movies must be viewed with a QuickTime movie player.)

CameraMan

MoviePlay, a utility for playing QuickTime movies. The “Tour the Disc” movie was created using CameraMan.

Kodak

Demonstration versions of Kodak Create-It Photo CD Presentation software and Kodak Arrange-It Photo CD Portfolio Layout Software.

Michael's

This folder contains some multimedia productions that I have created.

QuickTime Handbook movie

This QuickTime movie was created as an advertisement for the book *QuickTime Handbook*, published by Hayden, written by David Drucker and Michael D. Murie. The movie was put together in a couple of days and was shown at an awards show.

The movie was created in Director and Infini-D (which was used to create the flying television).

Now & Zen

Created almost four years ago as a parody of Apple's Knowledge Navigator videotapes. This was created in Director.

Farmers Market

There are three presentations in this folder. The movies feature a selection of hand-colored photographs by Susan Murie. The first presentation was created in Director, and has full-screen graphics. The other two are quarter-screen and are QuickTime movies. One movie is compressed using the Apple Video compressor, and the other using the Compact Video Compressor.

Dogcow

This movie was created using Director. There are two versions—one compressed using the Apple Video Compressor, and the other compressed with the Compact Video Compressor.

Interactive Skye

An interactive diversion created in Director.

Exploring Multimedia BBS

The software for calling the Exploring Multimedia BBS, a BBS dedicated to Multimedia issues.

EM BBS Demo Movie

A movie created in Director that shows how the Exploring Multimedia BBS software works.

Exploring Multimedia BBS

Contains the software for calling the BBS.

Getting Started with TF

A text document and a Microsoft Word document that explain how to use the software.

TeleFinder/User

This folder contains the software. Drag it to your hard disk. Note that you don't need the software to call the BBS—it's only necessary if you want to use the GUI interface. You can call using regular telecom software. The number of the bulletin board is (617) 666-9447.

TF Prefs 1 bit

This file contains a 1-bit version of the splash screen. Good for use on portables or other machines with only 1-bit screens. Simply drag the TF Prefs file from this folder into the TeleFinder/User folder.

Library

This folder contains images, QuickTime movies, and sounds that you can use in your own productions. While you can use these materials for your own purposes, they are not in the public domain and cannot be given or sold to others. They cannot be distributed as part of another collection or library of material.

Images

This folder contains three folders. Each folder contains the same set of 20 images. One folder contains 24-bit versions of the files, while the other folders contain versions that use the 8-bit system palette and 8-bit adaptive palettes, respectively.

QuickTime clips

This folder contains three folders of clips. The 180 by 135 and 240 by 180 folders contain the same movies in different sizes. All are compressed using the Apple Video Compressor and are 15 frames per second.

The folder 320 by 240 contains the same file, compressed once at normal quality and once at low quality using the Apple Video Compressor.

Sounds

Contains two folders that have the same sounds in Sound Edit and AIFF format (both at 22 kHz).

Book chapters

This folder contains elements relating to specific chapters in the book.

Graphics

Contains files related to Chapter 7, "Graphics."

Pictures

Sample images referenced in the chapter.

Photo CD stuff

Files from a Photo CD.

PCD1642 Slide Show

A QuickTime movie that shows the images contained on a PhotoCD.

Photos

This folder resembles the folder you see when viewing a Photo CD using QuickTime 1.5 or later. A Photo CD contains the same images in five different formats. This folder contains the five folders, and in each folder there are three sample images. You must be running QuickTime 1.5 (or later) to be able to view these.

Animation

Contains an animation created using the program Life Forms (see Chapter 8, “Animation”).

3D

Contains a test image rendered with different qualities (see Chapter 9, “3D”).

QuickTime

Contains files related to Chapter 10, “QuickTime.”

Image Test

This folder contains the same image, compressed using the different QuickTime compressors.

Appendix

Movie Test

This folder contains the same movie compressed using the different QuickTime compressors.

Sound

Contains files related to Chapter 11, “Sound.”

Sound Quality

A demo program that provides an example of a sound sampled at different rates.

Workshop

Contains the files necessary for creating the interactive newsletter project in Chapter 16 and the interactive kiosk in Chapter 17.

Also contains a working version of each project.

Multimedia Starter Kit for the Mac

Index

Symbols

128 MB optical disks, 85
16-bit sound, 222
3D animation, 170-171
 event-driven animation, 171
 key frame animation, 170
 PICS animation, 172-186
 PICT animation, 172-186
 QuickTime animation, 172-186
3D applications, 172-176, 185
 Alias Sketch!, 184-185
 Architectural modeling 3D
 applications, 172-173
 DynaPerspective, 172
 general purpose 3D applications,
 175
 hardware requirements, 185-186
 importing, 160-161
 Infini-D, 180-181
 Macromedia Three-D, 183-184
 ModelShop, 172
 Power Macintosh, 170
 Ray Dream Designer, 177-180
 simple extrusion 3D applications,
 173-174
 StrataVision 3D, 182
 user selection, 176-177
 virtual reality 3D applications,
 175-177
3D model rendering application,
 161-170
3D modeling application, 155-161
3D objects
 advanced freeform modeling,
 159-160
 algorithms, 162-166
 arranging with SceneBuilder, 179
 creating with Light Forge, 178
 DXF format, 160-161

extrusion, 156-157
extrusion modeling, 174
freeform modeling, 158-159
lathing, 157
lighting, 167-168
rendering, 161-170
RenderMan RIB format, 168-169
shading algorithms, 162-164
surface modeling, 165-166
sweeping, 158
8-bit sound, 222

A

accelerated renderers, 169-170
accelerators
 graphics, 82
 hardware, 206
access times
 CD-ROM drives, 85
 hard drives, 83
accounts (Internet), 104
Acrobat utility, 104
Action! (QuickTime movies
 application), 208, 261-264
active-matrix screens, 76
actors (ADD/PROMotion
 applications), 146
adding
 background text buttons
 (HyperCard), 316-319
 background text field
 (HyperCard), 316-319
 backgrounds (HyperCard), 315-319
 color graphics (HyperCard), 310
 Director, 351-358
 HyperCard
 card content lists, 331-333
 cards, 324-328

- text, 319-323
- pictures (HyperCard), 324
- ADDmotion application
 - (animation), 146-148
- Adobe
 - fonts, 93
 - Illustrator, 112
 - Photoshop, 115-116
 - Premiere (QuickTime editing application), 213
- advanced freeform modeling (3D objects), 159-160
- After Image (QuickTime post-processing tool), 215-218
- AIFF (Audio Interchange File Format) sound format, 230
- Aldus
 - Fetch (Clip Media application), 245
 - Freehand, 112
- algorithms
 - ray tracing, 164
 - shading, 162-164
 - surface modeling, 164-166
- Alias Sketch! 3D application, 184-185
- Allegient Technology, 295
- ambient lights (3D object lighting), 167
- amplifying sounds, 231
- animation
 - 3D animation, 139, 170-171
 - actors (ADD/PROMotion applications), 146
 - applications, 180-184
 - cartoon animation, 139
 - cels, 146
 - compressing, 143
 - Director, 297
 - opening, 347-349
 - playing, 299-300
 - dope sheets, 151
 - exporting to applications, 148
 - file formats, 143-144
 - frames, 146
 - In-between animation (Director), 344
 - key frame animation, 145
 - loops, 149
 - Macintosh, 139-144
 - memory management, 142
 - movement exaggerations, 140
 - paths, 146
 - synchronizing sound, 141-143
 - vs. digital video, 191-192
- Animation compressor (QuickTime), 199
- Animation Stand application, 151
- Animation Works (animation application), 148-149
- Animators Workbook, The*, 140
- anti-aliasing images, 133
- anticipation (animation movement exaggerations), 140
- APDA (Apple Programmers and Developers Association), 100
- Apple CD-ROM Titles Sampler CD-ROM, 35, 38-41
 - navigating, 40-41
 - structure, 40-41
- Apple Media Tool application, 269
- Apple, *see* Macintosh
- AppleScript programming languages, 286-287
- applications
 - 3D applications, 172-176, 185
 - Alias Sketch!, 184-185
 - hardware requirements, 185-186
 - Infini-D, 180-181
 - Macromedia Three-D, 183-184
 - Ray Dream Designer, 177-180

Index

StrataVision 3D, 182
 user selection, 176
3D model rendering
 application, 161-170
3D modeling application, 155-161
Action! application, 261-264
ADDmotion (animation), 146-148
animation applications, 180-184
Animation Stand, 151
Animation Works, 148-149
Apple Media Tool application, 269
Authorware application, 268
 chart creating applications, 117
Cinematic, 149-150
ClipMedia application, 242
cross-platform applications
 (QuickTime), 217
digitizing applications, 196
Director
 (animation application), 139-144
Fractal Designs Painter, 116
graph creating applications, 117
graphics, 108, 131-134
interactive presentation
 applications, 261
Life Forms (animation), 144-146
ModelShop, 160
MovieWorks (animation), 150-151
Multimedia applications, 9
Multiple Media Tour
 (Clip Media), 242
object-oriented graphics
 applications, 116
Passport Producer Pro
 application, 266-267
players, font installation, 99
PostScript graphics
 applications, 117
presentation applications, 260
PRomotion (animation), 146-148

QuickDraw, 112
QuickTime, 188-194
QuickTime movie players, 208
QuickTime movies, 208
ResEdit, 100
ScreenPlay (VideoSpigot), 201
 scripting applications, 256
Sketch! 3D modeling, 160
Special Delivery application,
 264-266
 standalone applications
 (SuperCard), 292
Architectural modeling 3D
 applications, 172-173
asymmetrical compressors, 198
audio (Quicktime), 207-208
Audio CD Import Options
 dialog box, 233
Audio palette, 228
Audio Visual Group, 242
authoring environments, 27-28, 256,
 303-304
 Authorware, 303
 Director, 295-303
 Hypercard, 272-291
 Prograph, 303
 selecting, 46
 Serious, 303
 SuperCard, 291-295
Authorware, 303
Authorware application, 268
AV Macintosh, 77-78

B

Background command (Edit menu),
 316
Background command (Go To menu),
 279

backgrounds
 HyperCard, 279-281, 315-319
 Bernoulli disks, 84
 Bezier curves (image resolution), 112
 bit-depth (screens), 22-23
 bitmap-based graphics, 108, 111
 bitmapped fonts, 90, 98-105
 Bkgnd Info command (Objects menu), 281
 BMP format, 119
 boldface type, 6
 books
 Animators Workbook, 140
 Complete HyperCard 2.2 Handbook, 290
 Exploring Multimedia, 308
 HyperCard Stack Design Guideline, 290
 Inside SuperCard, 294
 Internet Starter Kit, 2nd Edition, 102
 Macintosh Multimedia Workshop, 4
 Macromedia Director Design Guide, 303
 QuickTime Handbook, 114
 ResEdit Reference, 100
 Tao of AppleScript, Second Edition, 287
 bounding boxes (shading algorithms), 162
 bump maps (3D objects), 166
 Button Information command (Object menu), 315
 button parameters dialog box (Hypercard), 282
 buttons (HyperCard), 281-284

C

CameraMan (QuickTime movies), 212
 cameras, 124-127
 capturing video images, 205
 card fields (HyperCard), 278
 cards
 third-party cards, 82
 video digitizing cards, 124
 cards (HyperCard stacks), 273
 cards (HyperCard)
 adding, 324-328, 331-333
 customizing, 329-333
 naming, 332
 navigating, 273-274, 332-333
 Cast (Director), 296
 Cast Member Info (Cast menu), 355
 Cast menu commands, 355
 Cast objects (Director)
 adding, 355-358
 arranging, 345-347
 Cast window (Director), 297
 CCD (charged couple device), 124
 CD audio tracks, 232-234
 CD-ROM, 5
 Apple CD-ROM Titles Sampler, 35, 38-41
 drives, 85-86, 130
 MarketPlace, 56-66
 Multi Media Tour, 41-43
 Type On Call (font program), 93
 cels, 146
 Centris (Macintosh), 74
 channels (Director), 298
 displaying, 356
 hiding, 356
 Score windows, 349
 charged couple device, *see* CCD
 chart creating applications, 117
 chips (RISC), 79

Index

- Cinematic (animation application), 149-150
- Cinepak compressor (QuickTime), 199
- clip art graphics, 131
- Clip Media, 241-246
 - Aldus Fetch application, 245
 - Audio Visual Group, 242
 - cataloging utilities, 245
 - Clip TimelImages for presentations, 244
 - copyrighted, 244-246
 - Corel Corporation, 241
 - Digital Video Library, 244
 - Jasmine Multimedia Publishing, 242
 - Kodak Shoebox Image Manager (Clip Media application), 245
 - licensing, 244-245
 - Multi Media Music, 244
 - WraptureReels One, 244
- Clip TimelImages for presentations, 244
- ClipMedia application, 242
- clipping sound, 231
- Close Coloring Tools command (Color menu), 314
- Collage (graphics application), 132
- color
 - hardware, 21
 - graphics, 310-312
 - HyperCard, 287-289
 - multimedia requirements, 81-82
 - processors, 74
- Color menu (HyperCard), 287
- Color menu commands, 314
- Color Tools palette (HyperCard), 312
- commands
 - Color menu, 314
 - Compression menu, JPEG, 120
- Edit menu
 - Background, 316
 - New Background, 280
 - New Card, 273, 324
- File menu
 - Import, 340
 - New Stack, 310
 - Preferences, 340
- Go menu
 - Next, 274
 - Prev, 274
- Go To menu
 - Background, 279
 - Previous, 283
- Macintosh OnLine Reference, 33-34
- Object menu, 315
- Objects menu
 - Bkgnd Info, 281
 - New Background, 316
- pass openCard command, 330
- Quit HyperCard command, 315
- Score menu, 344
- Windows menu, 342
- Common Ground utility, 104
- Complete HyperCard 2.2 Handbook*, 290
- Composer application (MovieWorks), 150
- compressing
 - algorithms, *see* compressors
 - animation, 143
 - digital video, 193
 - files, 120
 - images, 199
 - sound, 223
 - video images, 197-200
- Compression command (Compression menu, JPEG), 120
- compression factor (JPEG), 114, 120

Compression menu, JPEG
 commands, 120
 compressors
 adding to QuickTime, 207
 Animation compressor
 (QuickTime), 199
 asymmetrical compressors, 198
 Cinepak compressor
 (QuickTime), 199
 Graphics compressor
 (QuickTime), 199
 INDEO compressor, 207
 MPEG compressor, 207
 Photo JPEG compressor
 (QuickTime), 199
 Quicktime compressors, 193,
 197-198
 Quiktime compressors, 199
 symmetrical compressors, 198
 Video compressor
 (QuickTime), 199
 converting
 image formats, 134
 video images to digital images, 189
 convolution filter (Radius
 VideoVision), 205
 Cool Mac Stacks, 290
 copying text, 320-323
 copyright laws, 246-247
 Clip Media, 244-246
 Fair Use, 248
 fonts, 99
 release, 248-249
 Corel Corporation, 241
 cosmetic designs, 28-29
 cost (projects), 18
 cross-platform utilities, 104
 custom programming, 257
 customizing
 HyperCard, 329-333
 icons, 100
 text characters, 100

D

DeBabelizer (graphics application),
 134
 decompressing video images, 197-198
 deselecting Stage objects, 347
 design
 interfaces, 46-49
 Macintosh OnLine Reference,
 33-34
 MarketPlace, 63-65
 projects, 45-53
 QuickTime Intro News, 37-38
 dialog boxes
 Audio CD Import Options, 233
 button parameters, 282
 Cast member Info, 356
 Edit Object (Action! presentation),
 263
 Field (HyperCard), 277
 Field Info, 317
 font, 318
 font (HyperCard), 278
 Import File, 341
 MIDI (Musical Instrument Digital
 Interface), 236
 New Stack, 310
 Record Sound dialog box, 227
 Save, 233
 Script (SuperCard), 293
 Set tempo, 350
 Set Transition, 350
 digital
 cameras, 126-127
 images, converting video
 images, 189
 video, 191-197
 Digital Video Library, 244
 digitizers, 196

Index

- digitizing
 - CDs, 232-234
 - boards (sound), 232-238
- Director (animation application), 139-144, 256, 295-303
 - animation, 297
 - opening, 347-349
 - playing, 299-300
 - buttons, 352
 - Cast, 296
 - Cast objects, 345-347, 355-358
 - channels, 298
 - documents, 340-353
 - graphics, 340-341
 - In-between animation, 344
 - information screens, 358-360
 - Ink Effect menu, 346
 - interactive kiosk application, 338-339
 - limitations, 301
 - Lingo, 295
 - Menu screen, 351-354
 - movies
 - navigating, 353-357
 - frames, 343, 354
 - troubleshooting, 357-360
 - Score, 296, 352
 - Script window, 353
 - scripts, 298
 - adding, 351-354
 - advanced scripts, 360-363
 - Stage, 296, 347
 - storing QuickTime movies, 302
 - windows, hiding, 349
 - XObjects, 301
- disks
 - 128 MB optical, 85
 - Bernoulli, 84
 - multi-session disks (Photo CDs), 130-135

- single-session disks (Photo CDs), 130-135
- SyQuest, 84
 - utilities, 104-105
- display boards (graphics), 82
- documents
 - Director, 340-353
 - HTML (Hypertext Markup Language), 102
 - SuperCard projects, 291
- dope sheets (animation), 151
- dpi (dots per inch), 91
- drivers (MacRecorder), 227
- drives, 130
- Duos (Macintosh), 75-77
- DXF format (3D objects), 160-161
- DynaPerspective (3D applications), 172

E

- Edit menu commands
 - Background, 316
 - New Background, 280
 - New Card, 273, 324
- Edit Object dialog box (Action! presentation), 263
- editing
 - Action! presentations, 262
 - applications (QuickTime), 213-215
 - backgrounds (HyperCard), 316
 - QuickTime movies, 209-211
 - Score window, 343-344
 - sounds, 228-229
- Effects menu (HyperCard), 311-313
- electronic newsletters, 26-45, 308
 - Macintosh OnLine Reference (starting hints), 31
 - starting, 29-53

- emulators (PowerPC Macintosh), 79
- entering HyperCard text into
 - fields, 276
- environments, 254-257
- EPS (Encapsulated PostScript) file
 - formats, 113, 119
- event-driven animation (3D animation), 171
- expanding HyperCard, 289
- Exploring Multimedia*, 308
- exporting animation, 148
- extending Director
 - graphics, 343
 - Score, 352
- extrusion (3D objects), 156-157, 174

F

- Fair Use (copyright law), 248
- Field dialog box (HyperCard), 277
- Field Info dialog box, 317
- fields
 - adding to backgrounds, 280
 - card fields (HyperCard), 278
 - HyperCard, 274-279
- File menu commands
 - Import, 340
 - New Stack, 310
 - Preferences, 340
- files
 - compressing, 120
 - EPS (Encapsulated PostScript)
 - files, 113
 - PostScript files, 113
 - sound files, 222
- film clips (Stock houses), 240-241
- Filmstrip Window (Cinematic), 149
- flat shading (shading algorithms), 162-163

- flatbed scanners, 127-128
- font dialog box (HyperCard), 278, 318
- fonts, 100
 - Adobe fonts, 93
 - bitmapped fonts, 90, 98-105
 - copyrighted fonts, 99
 - distributing, 99
 - graphics, 101
 - installing in player applications, 99
 - point size, 90
 - PostScript, 91-93, 98-105, 112
 - styles, 98
 - TrueType, 98-105
- formats
 - BMP format, 119
 - converting, 134
 - EPS (Encapsulated PostScript), 113, 119
 - GIF (Graphics Interchange Format), 119
 - graphic file formats, 117-122
 - JPEG (Joint Photographic Experts Group) file format, 114, 120-122
 - lossy compression formats, 114
 - MacPaint format, 118
 - PICS format (animation), 143-144
 - PICT (picture), 111, 118
 - QuickTime animation compressor, 143-144
 - RenderMan RIB format (3D objects), 168-169
 - RIFF format, 118
 - sound formats, 229-230
 - Targa format, 119
 - TIFF (Tagged Image File Format), 112, 118
- formatting HyperCard text, 323
- Fractal Designs Painter, 116
- frames, 146

Index

freeform modeling (3D objects),
158-159
Freehand (Aldus), 112

G

general purpose 3D applications, 175
GIF (Graphics Interchange
Format), 119
glossaries (Macintosh OnLine
Reference), 32
Go menu command (HyperCard), 97
Next, 274
Prev, 274
Go To menu commands
Background, 279
Previous, 283
Goodman, Danny, 290
Gouraud shading (shading
algorithms), 163
graph creating applications, 117
graphics
accelerators, 82
applications, 131-135
bitmap-based graphics, 108, 111
clip art graphics, 131
Director, 340-343
display boards, 82
file formats, 117-122
fonts, 101
images 114-135
MacDraw, 111
Macintosh, 108-114
object-oriented graphics, 108, 111,
114-122
SuperPaint, 111
see also images
Graphics compressor (QuickTime),
199
Groups program (HyperCard), 95

H

hard drives, 83-85
hardware
3D application requirements,
185-186
accelerators, 206
color, 21
multimedia applications, 81-82
multimedia requirements, 82-88
projects, 21-30
QuickTime, 21
sound (Macintosh), 220-221
sound digitizers, 232
system versions, 21-22
YARC board, 170
Help screens, 333
hertz (sound), 220
HFS (Hierarchical File System), 85
hiding
channels, 356
Director, 349
Hierarchical File System, *see* HFS
Home Stack (HyperCard), 310
HTML (Hypertext Markup Language),
101-102
HyperCard, 256, 272-291
ADDmotion application
(animation), 146-148
AppleScript, 286-287
Audio palette, 228
backgrounds, 279-281, 315-319
buttons, 281-284, 334
card fields, 278
cards, 273, 324-333
color, 287-289
Color menu, 287
Color Tools palette, 312
Effects menu, 311, 313
expanding, 289

fields, 274-279
 Go menu command, 97
 Groups program, 95
 Help screens, 333
 Home Stack, 310
 HyperTalk, 284-286
 Hypertext, 94-98
 Items menu, 311-313
 limitations, 289
 links, 282
 Macintosh OnLine Reference, 31
 Macromedia Director, 61
 MarketPlace, 56
 memory allocation, 309
 online reference, 290
 pictures, 324
 PROMotion application
 (animation), 146-148
 QuickTime movies, 333
 Quit button, 314-315
 Recent window, 97
 scripting languages, 287
 scripts, 283-284
 Select a Picture to Place window,
 313
 stacking order, 313
 stacks, 27, 310
 storing color graphics, 312
 Style options pop-up menu, 277
 text, 319-323
 entering into fields, 276
 styles, 324
 wrapping, 325-326
 titles, 320
 VideoDisc Toolkit, 88
HyperCard Stack Design Guideline,
 290
 HyperTalk programming language,
 284-286

HyperText, 94-98
 internet, 101-104
 navigating, 96-105
 system links, 96
 Hypertext Markup Language, *see*
 HTML
 hyphenated keys, 6

I

icons, 6, 100
 Illustrator (Adobe), 112
 images
 anti-aliasing, 133
 collages, 132
 compressing, 199
 importing, 313
 Kodak Photo CD images, 129-131
 scanning, 127-128
 still video images, 124-125
 texturing, 131
 video images, 123-124, 189-190,
 197-198
 see also graphics
 ImageWriter printer, 109
 Import command (File menu), 340
 Import File dialog box, 341
 importing
 3D applications, 160-161
 Director, graphics, 340-341
 images, 313
 In-between animation (Director), 344
 In-Between Linear command (Score
 menu), 344
 In-Between Special command (Score
 menu), 344
 INDEO compressor, 207
 index (Macintosh OnLine
 Reference), 32

Index

Infini-D, 180-181
 flat shading, 163
 freeform modeling screen, 159
 Gouraud shading, 163
 Phong shading, 164
 ray tracing, 164
 tools, 155
 wireframes, 162
information screens (Director),
 358-360
Ink Effect menu (Director), 346
Inside SuperCard, 294
installing QuickTime, 194-195
interactive kiosk applications
 (Director), 338-339
interactive presentations, 256, 261
interfaces
 cosmetic designs, 28-29
 designs, 46-49
 electronic newsletters, 28-29
 MIDI (Musical Instrument Digital
 Interface), 234-237
 structural designs, 28-29
Internet, 101-104
Internet Starter Kit, 2nd Edition, 102
isometric views (objects), 159
Items menu (HyperCard), 311, 313

J-K

Jag II (graphics application), 133
Jasmine Multimedia Publishing,
 241-242
JPEG (Joint Photographic Experts
 Group) file format, 114, 120-122
key frame animation, 145, 170
Key scripting language, 269
keys, 6
Kodak DCC cameras, 127

Kodak Photo CD images, 129-131
Kodak Shoebox Image Manager (Clip
 Media application), 245
KPT Bryce (graphics
 application), 134

L

labeling movie frames, 354
laser discs, 86-88
laser printers, 109
LaserWriter printers, 112
lathing (3D objects), 157
licensing, Clip Media, 244-245
Life Forms application (animation),
 144-146
Light Forge (Ray Dream
 Designer), 178
lighting (3D objects), 167-168
Lingo (Director), 295
linking text, 94-97
Linking (Action! presentation), 264
links (Hypertext), 96
lofting (3D objects), 158
logos (3D), 156
loops (animation), 149
lossy compression formats, 114

M

MacDraw, 111-112
Macintosh
 animation, 139-144
 AV Macintosh, 77-78
 Centris, 74
 color processors, 74
 computers, purchasing, 80-82
 Duos, 75-77

- graphics, 108-114
- MacDraw, 111-112
- MacPaint, 109-110
- MacRecorder, 224-227
- multimedia-friendly computers, 81
- Performas, 77
- Portable, 75-77
- Powerbook, 75-77
- PowerPC, 78-80
- Quadra 630, 75
- synthesized speech, 237
- technological development, 72-77
- text, 90-93
- Macintosh Human Interface
 - Guidelines, 43-53
- Macintosh Multimedia Workshop*, 4
- Macintosh OnLine Reference, 31-34
- MacPaint, 109-110
 - format, 118
- MacRecorder, 66, 224-227
- Macromedia, 242
- Macromedia Director Design Guide*, 303
- Macromedia Three-D (applications), 183-184
- Macromedia Director, 61
- MacTCP (HTML documents), 102
- MacWorld magazine*, 241
- marketing projects, 19-24
- MarketPlace
 - CD-ROM, 56-66
 - design, 63-65
 - HyperCard, 56
 - prototypes, 57-61
- matting objects, 183
- media options, 84-85
- MediaTracks, 66
- memory
 - HyperCard allocation, 309
 - management in animation, 142

- Menu screen (Director), 351-354
- microphones (sound), 223
- MIDI (Musical Instrument Digital Interface), 234-237
- modeling (3D objects), 159-160
- ModelShop (3D applications), 160, 172
- monitors, 22-23
- movie frames (Director)
 - labeling, 354
 - navigating, 343
- Movie Window (Cinematic), 149
- MoviePlay (QuickTime movies application), 212
- movies (Director)
 - navigating, 353-357
 - playing on QuickTime, 194-195
 - troubleshooting, 357-360
 - see also* video images
- MovieWorks (animation application), 150-151
- MPEG compressor, 207
- Multi Media Music, 244
- Multi Media Tour, 41-43
- multi-pass (RasterOps digitizer), 205
- multi-session disks (Photo CDs), 130-135
- Multimedia application icons, 6
- Multiple Media Tour, 42, 242
- Musical Instrument Digital Interface, *see* MIDI

N

- naming HyperCard, 332
- navigating
 - Apple CD-ROM Titles Sampler, 40-41
 - cards (HyperCard stacks), 274

Index

- Director, 343, 353-357
- HyperCard, 332-333
- Hypertext, 96-105
- Macintosh Human Interface Guidelines Companion, 44-53
- Macintosh OnLine Reference, 31-33
- Multiple Media Tour, 42
- QuickTime Intro News, 36-37
- network renderers, 169-170
- New Background command (Edit menu), 280
- New Background command (Objects menu), 316
- New Card command (Edit menu), 273, 324
- New Media* magazine, 241
- New Stack command (File menu), 310
- New Stack dialog box, 310
- Next command (Go menu), 274
- None compressor (QuickTime), 197

O

- Object menu commands, 315
- object-oriented graphics, 108, 111, 114-122
- objects
 - 3D objects, 156-160
 - creating graphics, 110
 - isometric views, 159
 - matting, 183
 - planes, 159
- Objects menu commands
 - Bkgnd Info, 281
 - New Background, 316
- Ofoto scanner driver, 127

- online references
 - HyperCard, 290
 - Macintosh Human Interface Guidelines Companion, 43-53
 - Macintosh OnLine Reference, 31-34
- opening MIDI files, 236

P

- Paint Editor (MovieWorks), 150
- Paint window (Director), 297
- palettes (HyperCard), 312
- parameters (HyperCard fields), 276
- pass openCard command, 330
- passive-matrix screens, 76
- Passport Producer Pro application, 266-267
- paths, animation, 146
- Performas (Macintosh), 77
- Persuasion
 - QuickTime movies, 208
 - slide presentations, 254
- Phong shading (shading algorithms), 163-164
- Photo CD images, 129-131
- Photo JPEG compressor (QuickTime), 199
- photography (Stock houses), 240-241
- Photoshop (Adobe), 115-116
- PICS format (animation), 143-144, 172-186
- PICT (picture) formats, 111, 118, 172-186
- Picture Compressor utilities, 120
- pictures (HyperCard), 324
- pixel depth, 23
- pixels (picture elements), 108
- pixels per inch, 91

- planes (objects), 159
- playback speed (projects), 22
- playing
 - Director animation, 299-300
 - MIDI files, 235
 - QuickTime movies, 208-209
 - sounds, 229
 - video image sequences on
 - computer screens, 189-190
- plug-in cards, 74
- point lights (3D object lighting), 167
- point size (fonts), 90
- Portables (Macintosh), 75-77
- portals (Special Delivery application), 265
- post-processing tools
 - (QuickTime), 215
- PostScript files, 113
- PostScript fonts, 91-93, 98-105, 112
- PostScript graphics applications, 117
- Power Macintosh (3D applications), 170
- Powerbooks (Macintosh), 75-77
- PowerPC Macintosh, 78-80
- PowerPoint (slide presentation environments), 254
- ppi (pixels per inch), 91
- Preferences command (File menu), 340
- Premiere (QuickTime editing application), 213
- presentation applications, 260
- pressure-sensitive drawing tablets
 - (graphics application), 135
- Prev command (Go menu), 274
- Previous command (Go To menu), 283
- print-to-disk utilities, 104
- printers
 - LaserWriter, 112
 - resolution, 91
- procedural shaders (surface modeling), 165
- program cards, 74
- programming, 27-28, 257
- programming languages, 284-287
- Prograph, 303
- projects
 - constraints, 17-27
 - cost, 18
 - creating, 26-45
 - designs, 45-53
 - developing, 16-17, 65-67
 - development tools, 61-62
 - equipment, 18-19
 - hardware, 21-30
 - marketing, 19-24
 - playback speed, 22
 - prototypes, 49-51
 - RAM (Random Access Memory)
 - requirements, 22
 - SuperCard, 291
 - text, 94
 - time management, 18
 - transmitting, 23-24
 - troubleshooting, 52-53
 - user access, 20-30
- PRomotion application (animation), 146-148
- prototypes
 - MarketPlace, 57-61
 - standardized terminology, 64
 - testing, 64-65
- purchasing Macintosh computers, 80-82

Q

Quadra 630 (Macintosh), 75
QuickDraw, 112
QuickTake 100 (digital camera), 126
QuickTime, 21, 142, 188-194, 257
 After Image (post-processing tool), 215-218
 animation, 172-186
 compressor, 143-144
 audio, 207-208
 compressors, 193, 197-198, 207
 cross-platform applications, 217
 editing applications, 213-215
 installation, 194-195
 playing movies, 194-195
 post-processing tools, 215
 Premiere (editing application), 213
 timing mechanism, 193
 utilities, 212-213
 video images, 193-200
 VideoFusion (post-processing tool), 215-218
 VideoShop (editing application), 213
QuickTime Handbook, 114
QuickTime Intro News, 35-38
QuickTime movies, 333
 CameraMan application, 212
 editing, 209-211
 MoviePlay application, 212
 playing, 208-209
 recording CD audio tracks, 233
 saving as reference files, 210
 Spectator application, 212
 storing, 302
QuickTime Starter Kit, 120
QuickTime VR (Virtual Reality), 208
Quiktime, 199
Quit button (HyperCard), 314-315
Quit HyperCard command, 315

R

Radius VideoVision (video digitizers), 123, 205-218
RAM (Random Access Memory), 22
RasterOps (video digitizing board), 123, 202-205
Ray Dream Designer (3D applications), 177-180
ray tracing algorithms (3D objects), 164
Recent window (HyperCard), 97
Record Sound dialog box, 227
recording
 sound, 223-227, 230-232
 sound files (MacRecorder), 66
 video images, 195-200
reflection maps, 166
registering copyrighted work, 247
release (copyright law), 248-249
removable media options, 84-85
rendering 3D objects, 161-170
RenderMan RIB format (3D objects), 168-169
Replica utility, 104
ResEdit application, 100
ResEdit Reference, 100
resolution
 Bezier curves, 112
 ImageWriter printer, 109
 laser printers, 109
 MacDraw objects, 111
 printers, 91
 screens, 109
resources (sound), 229
RIFF format, 118
RISC chips, 79
run-length encoding (animation compression), 143

S

- sample rates (sound), 222-223
- sampling, *see* recording
- Save dialog box, 233
- saving
 - QuickTime movies as reference files, 210
 - sounds, 229-230
- scanners, 127-128
- SceneBuilder (Ray Dream Designer), 178
- Score (Director), 296, 352
- Score command (Windows menu), 342
- Score menu commands, 344
- Score window, 297, 342
 - channels, 349-350
 - editing, 343-344
- ScreenPlay application (VideoSpigot), 201
- screens
 - active-matrix, 76
 - Apple CD-ROM Titles Sampler, 40-41
 - bit-depth, 22-23
 - dimensions, 22-23
 - Macintosh Human Interface Guidelines Companion, 44-53
 - Macintosh Online References, 31-33
 - multimedia requirements, 81-82
 - Multiple Media Tour, 42
 - passive-matrix, 76
 - pixel depth, 23
 - QuickTime Intro News, 36-37
 - resolution, 109
- Script dialog box (SuperCard), 293
- Script window
 - Director, 353
 - Hypercard, 283
- scripting
 - applications, 256
 - languages (Key), 269
 - see also*, programming languages
- scripts
 - Director, 298
 - adding, 351-354
 - advanced scripts, 360-363
 - HyperCard, 283-284
- Select a Picture to Place window (HyperCard), 313
- Select Effect window (HyperCard), 288
- sequencers, 234
- Serious (authoring environment), 303
- Set tempo dialog box, 350
- Set Transition dialog box, 350
- shading algorithms (3D objects), 162-164
- Silicon Beach Software, 295
- simple extrusion 3D applications, 173-174
- single-session disks (Photo CDs), 130-135
- Sketch! 3D modeling application, 160
- slide presentation environments, 254-255
- slide scanners, 127-128
- sound
 - 16-bit sound, 222
 - 8-bit sound, 222
 - advanced sound techniques, 230-232
 - amplifying, 231
 - applications, 79
 - audio CD recording, 232-234
 - bits, 220
 - clipping, 231
 - compressing, 223
 - digitizers, 196

Index

- digitizing boards, 232-238
- editing, 228-229
- files, 222
- formats, 229-230
- frequency, 220
- hardware (Macintosh), 220-221
- hertz, 220
- levels, 231
- MacRecorder, 66
- microphones, 223
- MIDI sounds, 236
- playing, 229
- recording, 223-227, 230-232
- resources, 229
- sample rates, 222-223
- saving, 229-230
- stereo sound, 221
- storing, 220-223
- synchronizing in animation, 141-143
- Sound control panel, 225
- Sound Edit (sound format), 230
- Sound Editor (MovieWorks), 150
- Special Delivery (QuickTime movies), 208, 264-266
- special effects, video images, 215-218
- Spectator (QuickTime movies application), 212
- spotlights (3D object lighting), 167
- squash and stretch (animation movement exaggerations), 140
- stacks (HyperCard data file), 272-290, 310-313
- Stage (Director), 296, 347
- standalone applications (SuperCard), 292
- standardized terminology (project prototypes), 64
- step-and-grab (RasterOps digitizer), 205
- stereo sound, 221
- still images, 177
- still video cameras, 124-125
- Stock houses, 240-241
- storage devices, 83-88
- storing
 - HyperCard color graphics, 312
 - HyperCard text, 274
 - sound, 220-223
 - video images, 189
- storyboards, 45-46
- StrataVision 3D (3D applications), 182
- streaming video images (QuickTime), 193
- structural designs (interfaces), 28-29
- Style options pop-up menu (HyperCard), 277
- styles
 - fonts, 98
 - text (HyperCard), 324
- SuperCard, 256, 291-295
- SuperEdit application (SuperCard), 292
- SuperPaint (graphics program), 111
- surface modeling (3D objects), 164-166
- sweeping (3D objects), 158
- switching projects, 61-62
- symmetrical compressors, 198
- synchronizing sound in animation, 141
- synthesized speech (Macintosh), 237
- SyQuest disks, 84
- system memory, 81-82
- system versions (hardware), 21-22

T

Tagged Image File Format, *see* TIFF
Tao of AppleScript, Second Edition, 287
 tape recorders, 231
 Targa format, 119
 TCP/IP (Transmission Control Protocol/Internet Protocol), 102
 Tempo (Score window channels), 349
 testing
 project prototypes, 50-51
 prototypes, 64-65
 text
 characters, 100
 HyperCard, 319-323
 Hypertext, 94-98
 linking, 96-97
 Macintosh, 90-93
 multimedia projects, 94
 styles, 324
 threading, 94
 wrapping, 325-326
 Text Editor (MovieWorks), 150
 texture mapping (3D objects), 165-166
 TextureScape (graphics application), 131
 texturing images, 131
 third party applications
 cards, 82
 scanners, 127
 stacks, 290
 threading text, 94
 TIFF (Tagged Image File Format), 112, 118
 time management (projects), 18
 Time window (Action! presentation), 263
 timing mechanism (QuickTime), 193

titles (HyperCard), 320
 tools (Infini-D), 155
 topics (Macintosh OnLine Reference), 31-33
 transfer rates, 83-85
 transition (Score window channels), 350
 Transmission Control Protocol/Internet Protocol, 102
 transmitting projects, 23-24
 transparency maps (texture mapping), 166
 troubleshooting
 Director, 357-360
 projects, 52-53
 TrueType fonts, 98-105
 type (boldface), 6
 type family, *see* fonts
 Type On Call (CD-ROM font program), 93
 type sets, *see* fonts

U

user access, projects, 20-30
 utilities
 Acrobat utility, 104
 Clip Media cataloging utilities, 245
 Common Ground utility, 104
 cross-platform utilities, 104
 disks, 104-105
 Picture Compressor, 120
 QuickTime, 212-213
 Replica utility, 104

Index

V

Video compressor (QuickTime),
198-199
video digitizers, 123-124, 196
video digitizing boards
 filters (Radius VideoVision), 205
 Radius VideoVision, 205-218
 RasterOps, 202-205
 VideoSpigot, 201-202
video images, 123-124
 capturing, 205
 compressing, 197-200
 converting to digital video, 193-197
 decompressing, 197-198
 digital video, 191
 playing sequences on computer
 screens, 189-190
 recording, 195-200
 special effects, 215-218
 storing, 189
 streaming (QuickTime), 193
Video Tape Library Ltd., 241
VideoDisc Toolkit (HyperCard), 88
VideoFusion (QuickTime post-
 processing tool), 215-218
VideoShop (QuickTime editing
 application), 213
VideoSpigot (video digitizer), 123,
 201-202
view (objects), 159
virtual reality 3D applications,
 175-177, 208
Virtus VR (virtual reality 3D
 application), 175

W

windows
 Cast window (Director), 297
 Director, 349
 Filmstrip Window
 (Cinematic), 149
 Movie Window (Cinematic), 149
 Paint window (Director), 297
 Score window, 297, 342-344, 349
 Script window (Hypercard),
 283, 353
 Select a Picture to Place window
 (HyperCard), 313
 Select Effect window
 (HyperCard), 288
 Stage window (Director), 296
 SuperEdit project window, 292
Windows menu commands, 342
wireframe (shading algorithms), 162
World Wide Web, 101-104
wrapping text, 325-326
WraptureReels One, 244

X-Y-Z

Xanadu, 249
XCMDs (HyperCard), 289
XFCNs (HyperCard), 289
XObjects (Director), 301
YARC board (hardware
 accelerator), 170

About the CD-ROM



The CD-ROM included with this book is in HFS (Macintosh) format, and includes a wide range of things—from demos and sample files to images and QuickTime movies. (Note that some of the software is shareware, and is provided to you on a trial basis.) For more information about the disc contents, see “What’s on the Disc.”

Different applications and files on the disk have different requirements; we recommend a **color Macintosh with System 7** or later and **4 MB of memory** (and a **CD-ROM drive!**) as a minimum. Some of the applications require more memory, and some must be copied to your hard disk to work. The QuickTime movies must be viewed with an application that can play QuickTime movies (such as CameraMan’s MoviePlay or Adobe Premiere). You must also install the QuickTime extension, which is included on the disk. Some demos (and the “Putting it to work” files) require HyperCard.

The disc includes:

Tour the disc (a QuickTime movie)

QuickTime 2.0

Cover art

Software demos and samples

Includes demos of the following programs: Vividus’ **Cinematic**; Specular International’s **Infini-D 2.6**; and LogoMotion (with sample images, QuickTime movies, and an interactive presentation); sample images and movies generated using Playmation; No Hands Software’s **Common Ground** (and demo documents); Virtus **WalkThrough** (with sample images and movies, and three Voyager documents that enable you to explore a 3D world); Heizer Software’s **MasterScript**, **WindowScript**, and **CompileIt!**; Fractal Design’s **Painter 2.0**; a catalog for the **Multiple Media Tour** from Audio Visual Group; Adobe’s **Premiere 4.0**, **Photoshop 2.5**, **Illustrator 5.0**, and sample movies created using Premiere; Gryphon Software’s **Morph**; Motion Works’ **PROmotion**; sample movies and an interactive presentation about VideoFusion; a demo movie created using VideoToolKit; CameraMan’s MoviePlay utility; and Roger Wagner Publishing HyperStudio.

Multimedia productions created by the author

Includes: the **QuickTime Handbook** movie; **Now and Zen**; **Farmers Market** movies; **Dogcow**; and **Interactive Skye**.

Exploring Multimedia BBS

Software for calling the Exploring Multimedia bulletin board.

Library

A folder that contains images, QuickTime movies, and sounds.

Chapters

Folder containing elements relating to specific chapters in the book: graphics (including sample images and Photo CD information); a sample animation; a sample 3D image; QuickTime materials; sound materials; and files for the “Putting it to work” chapters.

Everything you need to create brilliant multimedia!

The *Multimedia Starter Kit* gets you going fast with easy steps that teach the basics—covering all the software you need to take advantage of the best Macintosh® multimedia techniques!

- **Create, develop, and fine-tune multimedia projects in no time**
- **Learn specific tips on integrating text, graphics, animation, 3D, QuickTime, sound, and more**
- **Discover expert techniques for choosing and using the right hardware and development environment**
- **Find out step by step how to get the most from HyperCard and Macromedia Director**
- **Get up-to-date information on the best multimedia products and companies**

You get useful information so that you can get started quickly creating your own multimedia projects. Michael D. Murie, a multimedia producer and president of a multimedia consulting firm, understands the challenges you face and the solutions you need. Valuable notes, tips, and shortcuts guarantee the best multimedia results in the least time.

Here's what people said about the previous edition, *Macintosh Multimedia Workshop*!

"Filled with too's and tips for developing multimedia projects, this well-written text is an ideal book for multimedia beginners."

CompuServe Magazine

"...an excellent value."

Australian MacUser

"...leads the reader step-by-step to creating, designing, and implementing multimedia projects."

Computer Publicity News

Other books just tell you about multimedia.
With this kit, you can do it yourself!



You get all the great software you need to work in multimedia!



- **Working demos for creating stunning multimedia projects including Infini-D, Painter, Movie Works, and Morph**
- **Try-out versions of Adobe Photoshop, Illustrator, and Premiere**
- **QuickTime 2.0**
- **Multimedia productions created by the author**
- **A great library of images, QuickTime movies, and sounds**
- **Valuable utilities for designing multimedia and manipulating files**
- **Examples that go with the book's exercises**
- **And much more!**

System Requirements:

Macintosh or Power Macintosh
System 7.0 or higher
4M of RAM CD-ROM Drive



\$30.00 USA / \$40.95 CAN

ISBN 1-56830-113-8



9 781568 301136

9 00000



Category: Multimedia
User Level: All Users