

LEARNING

QUARTZ COMPOSER

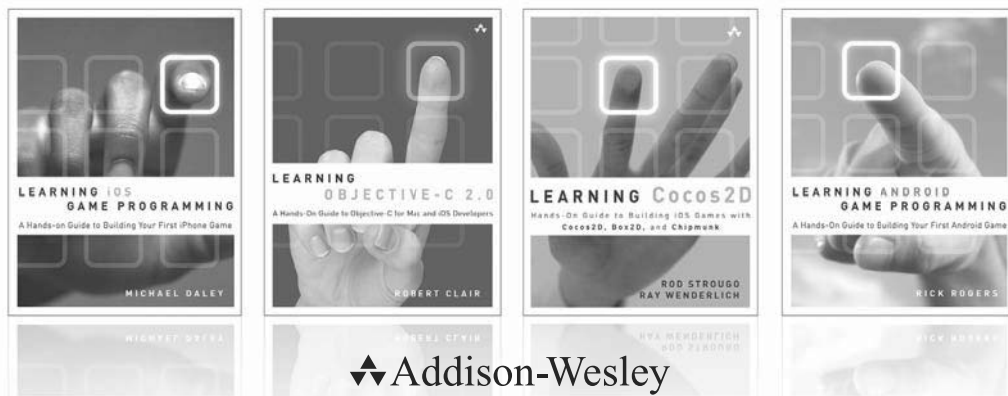
A Hands-on Guide to Creating Motion Graphics with Quartz Composer



GRAHAM ROBINSON
SURYA BUCHWALD

Learning Quartz Composer

Addison-Wesley Learning Series



Visit informit.com/learningseries for a complete list of available publications.

The Addison-Wesley Learning Series is a collection of hands-on programming guides that help you quickly learn a new technology or language so you can apply what you've learned right away.

Each title comes with sample code for the application or applications built in the text. This code is fully annotated and can be reused in your own projects with no strings attached. Many chapters end with a series of exercises to encourage you to reexamine what you have just learned, and to tweak or adjust the code as a way of learning.

Titles in this series take a simple approach: they get you going right away and leave you with the ability to walk off and build your own application and apply the language or technology to whatever you are working on.

◆ Addison-Wesley

informIT.com

Safari[®]
Books Online

Learning Quartz Composer

A Hands-On Guide to Creating
Motion Graphics with
Quartz Composer

Graham Robinson

Surya Buchwald

◆◆Addison-Wesley

Upper Saddle River, NJ • Boston • Indianapolis • San Francisco
New York • Toronto • Montreal • London • Munich • Paris • Madrid
Capetown • Sydney • Tokyo • Singapore • Mexico City

Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this book, and the publisher was aware of a trademark claim, the designations have been printed with initial capital letters or in all capitals.

The authors and publisher have taken care in the preparation of this book, but make no expressed or implied warranty of any kind and assume no responsibility for errors or omissions. No liability is assumed for incidental or consequential damages in connection with or arising out of the use of the information or programs contained herein.

The publisher offers excellent discounts on this book when ordered in quantity for bulk purchases or special sales, which may include electronic versions and/or custom covers and content particular to your business, training goals, marketing focus, and branding interests. For more information, please contact:

U.S. Corporate and Government Sales
(800) 382-3419
corpsales@pearsontechgroup.com

For sales outside the United States, please contact:

International Sales
international@pearson.com

Visit us on the Web: informit.com/aw

Library of Congress Cataloging-in-Publication Data

Robinson, Graham, 1982–

Learning quartz composer : a hands-on guide to creating motion graphics with Quartz composer / Graham Robinson, Surya Buchwald.
p. cm.

Includes index.

ISBN 978-0-321-85758-3 (pbk. : alk. paper)

1. Computer animation. 2. Digital video. 3. Quartz (Electronic resource) I. Buchwald, Surya, 1982- II. Title.

TR897.7.R595 2012

777'.7--dc23

2012015316

Copyright © 2013 Pearson Education, Inc.

All rights reserved. Printed in the United States of America. This publication is protected by copyright, and permission must be obtained from the publisher prior to any prohibited reproduction, storage in a retrieval system, or transmission in any form or by any means, electronic, mechanical, photocopying, recording, or likewise. To obtain permission to use material from this work, please submit a written request to Pearson Education, Inc., Permissions Department, One Lake Street, Upper Saddle River, New Jersey 07458, or you may fax your request to (201) 236-3290.

ISBN-13: 978-0-321-85758-3

ISBN-10: 0-321-85758-5

Text printed in the United States on recycled paper at Courier in Westford, Massachusetts.

First printing, July 2012

Editor-in-Chief

Mark Taub

Acquisitions Editor

Trina MacDonald

Development Editor

Sheri Cain

Managing Editor

John Fuller

Project Editor

Anna Popick

Copy Editor

Jill Hobbs

Indexer

John S. (Jack) Lewis

Proofreader

Diane Freed

Publishing Coordinator

Olivia Basegio

Multimedia Developer

Dan Scherf

Cover Designer

Chuti Prasertsith

Composer

Rob Mauhar



I would like to acknowledge Jesus as my inspiration and the source of my creativity and talent. My wife Natalie, my muse and the most caring, supportive, fun human I have ever met, thank you so much! My father Bruce, without your support and encouragement I never would have been able to become self-employed and start this whole journey.

—Graham

I dedicate this book to my Mom for always telling me to do what I love (it's working!), to my Dad for instilling in me a talent and love for the arts, and to my Grandma Theda for getting us our first computer when I was but a wee lad, giving a jump-start to my tech education. Thanks to my friends in Stargaze and LAVA for encouraging and believing in me when I had the crazy idea to get into making interactive ridiculousness.

—Surya



This page intentionally left blank

Contents at a Glance

Preface **xv**

Acknowledgments **xxi**

About the Authors **xxiii**

I Quartz Beginner 1

- 1 What Is Quartz Composer and Why Should I Learn It? **3**
- 2 The Interface and Playing a Movie **9**
- 3 Adding Visual Effects (Pimping It Out) **21**
- 4 Using LFOs, Interpolation, and Trackballs to Move Stuff **33**
- 5 Debugging (When Things Go Wrong) **43**
- 6 Particles (Little Flying Bits of Bling) **49**
- 7 Mouse Input (Making Your Mouse Do Cool Stuff) **59**
- 8 MIDI Interfacing (Getting Sliders and Knobs Involved) **71**
- 9 Interacting with Audio (Get Stuff Grooving to the Beat) **85**
- 10 Lighting and Timelines (The Dark Side of QC) **91**
- 11 Replication/Iteration (The Bomb) **97**

II Quartz Ninja 107

- 12 Modeling Complex Environments (3D Cities) **109**
 - 13 Create a Cocoa App (Send Quartz to Your Friends) **121**
 - 14 Create a Screensaver **133**
 - 15 Secret Patches, Core Image Filters, and GLSL (Pushing the Boundaries) **137**
- Index **143**
- Patch Index **151**

This page intentionally left blank

Contents

Preface	xv
Acknowledgments	xxi
About the Authors	xxiii

I Quartz Beginner 1

1 What Is Quartz Composer and Why Should I Learn It?	3
Play Video Introduction to Quartz Composer	3
Installing and Setting Up Quartz Composer	5
Outputs	5
Flexibility	7
Experimentation	7
Summary	7
Challenges	8
2 The Interface and Playing a Movie	9
Launching Quartz Composer	9
Editor versus WYSISYG	11
Viewer	13
Patch Library (Creator)	13
Patch Inspector/Patch Variables	16
Playing a Movie Tutorial Instructions	16
Summary	18
Challenges	19
3 Adding Visual Effects (Pimping It Out)	21
Adding a Filter	21
Filter Chains and Layering	23
Filter Tools	24
Image Crop	25
Rendering Destination Dimensions	28
Core Image FX and FPS	30
Summary	32
Challenges	32

4 Using LFOs, Interpolation, and Trackballs to Move Stuff	33
Interpolation Patch: Do Stuff for a Bit	33
Interpolation as an Amazing Calculator	36
LFO	37
Hierarchies with Environment Patches: Trackball and 3D Transformation	39
Trackball	40
3D Transformation	41
Summary	42
Challenges	42
5 Debugging (When Things Go Wrong)	43
Using Image with String	43
Debugging Tips	45
Interactive Placement Mode	46
Debug Mode	46
Debug Mode in Leopard	46
Profile Mode	46
Summary	46
Challenges	47
6 Particles (Little Flying Bits of Bling)	49
Add to Library (Creating a Clip in Leopard)	49
Starting Point Composition	51
The Particle System	51
Real-World Modeling	54
Rain	54
Fire	55
Blend Modes	57
Summary	57
Challenges	57
7 Mouse Input (Making Your Mouse Do Cool Stuff)	59
Particle Systems Control	59
Smoothing Input	61

Drag-and-Drop Interaction	64
Controlling a Kaleidoscope	68
Summary	69
Challenges	69
8 MIDI Interfacing (Getting Sliders and Knobs Involved)	71
MIDI Notes	71
How to Get the Information	71
MIDI Virtual Macros	73
MIDI Notes to Control Sprites	76
Adding in MIDI Controllers	82
Further Control	84
Summary	84
Challenges	84
9 Interacting with Audio (Get Stuff Grooving to the Beat)	85
Working with Audio Input	85
Output	89
Export	89
Summary	90
Challenges	90
10 Lighting and Timelines (The Dark Side of QC)	91
Video Tutorial on Lighting	92
Timelines	94
Summary	96
Challenges	96
11 Replication/Iteration (The Bomb)	97
Demo: Replicate in Space	97
Iteration	101
Summary	105
Challenges	105

II Quartz Ninja	107
12 Modeling Complex Environments (3D Cities)	109
Texturing	110
One City Building	112
Beat Reaction	112
Macro It Up	113
Duplication	114
Video Walls	115
Replicating	116
Master Scale	117
Finishing the Plan	118
Camera	119
Summary	119
Challenges	119
13 Create a Cocoa App (Send Quartz to Your Friends)	121
Xcode	121
Demo: Create an Application with Snow Leopard/Leopard	122
Demo: Create an Application with Lion	124
More Features: Publishing Inputs with Snow Leopard and Leopard	127
More Features: Publishing Inputs with Lion	129
Summary	131
Challenges	131
14 Create a Screensaver	133
Making the Screensaver	133
Adding Options	135
Summary	136
Challenges	136
15 Secret Patches, Core Image Filters, and GLSL (Pushing the Boundaries)	137
Private Patches	137
Plugins	138

JavaScript	139
GLSL	139
Core Image Filters	139
OpenCL	142
Summary	142
Challenges	142
Index	143
Patch Index	151

This page intentionally left blank

Preface

Welcome to *Learning Quartz Composer!* We guarantee this will be the most fun geek book you have read, and by the end your digital world will be a better-looking place. Whether you dream of live visuals, interactive installations, Cocoa apps, dashboard widgets, or extra awesomeness for your film and motion graphics projects, Quartz Composer will enable you to develop beautiful solutions in amazingly short periods of time.

With the introduction of Quartz Composer in Mac OS X Tiger, Apple delivered a very powerful and unique tool, and with each operating system upgrade it becomes better and better. Quartz Composer is like your graphics card's special sauce; hidden away on your Developer Tools disk, it's your Mac's best kept secret.

Creating with Quartz Composer is superfast because it is a live, constantly rendering environment. Thus, if you make a change, you will see the result immediately, rather than having to wait for RAM previews or long renders. In performance environments, a Quartz Composer file can take live inputs from music or cameras, allowing for unique interaction and improvisation. Another massive advantage is that you don't ever have to define your project dimensions, so you can work on them freely and later choose to output a video file to devices ranging from a tiny phone screen to a high-definition video editing program.

So if Quartz Composer is so great, why isn't everyone using it? Well, there is a little bit more to the story. Quartz Composer is a graphical programming environment, which sounds scary enough to make most creative types run for the hills. When you add in an unusual (though highly usable) interface, you can see why it has remained in the dark. Fear not—we will break it all down into plain English and give you the confidence to do anything you want with this handy tool.

This book launches you directly into building and manipulating beautiful compositions. Each concept is introduced as part of a hands-on project, with video tutorial, steadily building your “qc-fu” and demonstrating/encouraging experimentation every step of the way. The projects start out very simple, and the first focus is always on beautiful visual feedback, so you know why you are learning what you are learning and want to explore the systems they are creating.

Audience for This Book

With only the very basics of computer literacy, this book/DVD combination launches the unsophisticated user into creating art projects, visuals for a band or party, wild

screensavers, and RSS-powered trade-show kiosks. For anyone with a programming background, the material quickly opens up a new world of visual potential.

Who Should Read This Book

The target audience for this book consists of Maker types: people who are delighted and excited by projects that enable them to create new things from what they have, but who need a helping hand to get them going. The nature of Quartz Composer means that its appeal spans many genres. Motion graphics designers, filmmakers, VJs, artists, interactive programmers, and Cocoa developers—all can learn something here that will apply to their jobs tomorrow.

Who Shouldn't Read This Book

If you are an advanced Quartz Composer user looking for detailed knowledge about using GLSL and OpenCL in Quartz Composer or creating your own plugins in Objective-C Quartz Composer, this book may be a little too basic. However, even a long-time Quartz Composer user could benefit from some of the tips and tricks we've discovered on our own learning journeys.

We'd Like to Hear from You

This book is about your experimentation, and we expect great things from you, so please drop in and share what you have created. You will also be able to access any updates, download the book's projects, and more at this site: <http://iloveqc.org>.

As a reader of this book, you are our most important critic and commentator. We value your opinion and want to know what we're doing right, what we could do better, which areas you'd like to see us publish in, and any other words of wisdom you're willing to pass our way.

When you write to the publisher, please be sure to include this book's title and the names of the authors, as well as your name, phone, and/or email address. The editor will carefully review your comments and share them with the authors and others who have worked on this book. Please note that due to the volume of email we cannot respond to all inquiries/comments.

Email: trina.macdonald@pearson.com

Mail: Trina MacDonald
Senior Acquisitions Editor, Addison-Wesley
Pearson Education, Inc.
1249 8th Street
Berkeley, CA 94710 USA

For more information about Pearson Education books or conferences, see our website at: <http://iloveqc.org>.

Organization of This Book

There are 14 chapters in this book, each of which builds on the last, transforming you from total beginner to Quartz Composer Ninja. The book is divided into two parts: Part I teaches the basics of how the different tools or patches can be used and Part II builds on what you have learned to make more advanced compositions.

Part I: Quartz Beginner

- **Chapter 1, “What Is Quartz Composer and Why Should I Learn It?”**
This chapter introduces Quartz Composer, explaining what it is and how it can be used. It describes the range of outputs—Quartz file, movie, screensaver, and so on. The emphasis here is on flexibility and encouragement for the reader to experiment at every stage.
- **Chapter 2, “The Interface and Playing a Movie.”** This chapter covers the very basics of Quartz Composer launching it, the layout of the interface, and the concept of the Quartz Composer Editor versus a traditional “what you see is what you get” (WYSIWYG) program.
- **Chapter 3, “Adding Visual Effects (Pimping It Out).”** Quartz Composer comes with a variety of built-in image filters for effects. It’s easy to start routing your graphics and video through these filters, but some quirks and caveats that pop up could easily frustrate you. This chapter introduces the different types of effects available and the tools you’ll need to come to grips with them.
- **Chapter 4, “Using LFOs, Interpolation, and Trackballs to Move Stuff.”** One of the important concepts of the book is teaching you to create beautiful organic motion; Chapter 4 describes the tools you need to do so. The best part of the lesson is that all of these tools can work together and allow you to control many different things, from size to positioning to color. Trackballs and the **3D Transformation** patches help you control which part of your virtual world you are looking at.
- **Chapter 5, “Debugging (When Things Go Wrong).”** With the power of experimentation in Quartz Composer comes the inevitable “What did I do wrong?” moments. There are some helpful patches to get users through these tough times.
- **Chapter 6, “Particles (Little Flying Bits of Bling).”** This chapter explains what particles are, along with Quartz Composer’s **Particle System** patch and how to use it to make cool stuff like rain and fire. You can even use an image or movie as the particle! A brief introduction to blend modes for layering images and video together is provided as well.
- **Chapter 7, “Mouse Input (Making Your Mouse Do Cool Stuff).”** The mouse is an excellent input device for interactive work, and most computers

will have one (or else a trackpad). In this chapter, we teach you how to put the mouse to work inside Quartz Composer.

- **Chapter 8, “MIDI Interfacing (Getting Sliders and Knobs Involved).”** MIDI controllers have historically been used to control audio software and hardware. With Quartz Composer, you can now use keyboards, drum machines, and banks of sliders and knobs to control visual images.
- **Chapter 9, “Interacting with Audio (Get Stuff Grooving to the Beat).”** Now that you have an understanding of the use of LFOs, the mouse, and MIDI, the concept of using audio processing to control values within compositions can be easily introduced. Initially, ways to manipulate the volume peak and microphone input are demonstrated for a quick “Wow” factor, but then we move on to splitting the spectrum, using smooth and math functions to enhance the aesthetics of the application. In addition, we cover how to export your compositions as normal movie files.
- **Chapter 10, “Lighting and Timelines (The Dark Side of QC).”** This chapter introduces the topic of lighting, including all settings that a computer light has but the light in your room doesn’t. Experimentation with light, its positioning and its strength, and getting other controller objects involved are encouraged. Those readers who are used to normal editing and motion graphics packages will be happy to find out how Quartz Composer’s timelines work.
- **Chapter 11, “Replication/Iteration (The Bomb).”** Why have just one interesting interactive object when you can have hundreds? One of the great things about Quartz Composer is that once you have created a single object, you can make hundreds without any of that boring copy-and-paste nonsense!

Part II: Quartz Ninja

- **Chapter 12, “Modeling Complex Environments (3D Cities).”** In this chapter, we take a lot of what you have learned through the earlier chapters and use it to create an awesome audio reactive city scene. Readers learn how to turn cubes into buildings, to create floors, and to create more complex camera moves.
- **Chapter 13, “Create a Cocoa App (Send Quartz to Your Friends).”** Apple makes it easy for amateurs to create native applications that can manipulate Quartz Composer compositions. This chapter covers the basics of using Xcode and guides you all the way through publishing an application.
- **Chapter 14, “Create a Screensaver.”** Building on the earlier discussions of LFO and interpolation, this chapter uses patch time and random output to demonstrate longer-term, more gradually developing patches. It also explains how to wrap up and install patches as screensavers.
- **Chapter 15, “Secret Patches, Core Image Filters, and GLSL (Pushing the Boundaries).”** Quartz provides a rich feature set and many objects for

developing stunning compositions. Beyond its own capabilities, many interesting and exciting possibilities are provided by third-party plugins, access to the shader language GLSL, Core Image filters, and OpenCL. This chapter introduces the plugins included with Quartz Composer, including how to install them, how to access Kineme's work, and how to make your own Core Image filter.

Tutorial Videos and DVD Resources

With the book comes a great DVD, complete with a video tutorial for each chapter. The book and tutorials work together, so you can see exactly how to accomplish the more tricky bits covered in the book. The DVD also includes sample projects for each chapter, as well as some images we refer to in the chapters. Enjoy!

This page intentionally left blank

Acknowledgments

This book has been a lot of fun to write and simply would not have happened without a lot of time, effort, and help from these fantastic people:

- Our first editor at Addison-Wesley, Chuck Toporek, and his superstar assistant, Romny French. Chuck found our online video tutorials and made this all happen. Chuck and Romny were succeeded by editor Trina MacDonald and assistant Olivia Basegio, who helped us make it over the finish line. Without everyone's support, guidance, and encouragement, we would never have been able to become authors.
- Dr. Monica Schraefel, Dr. Mike Poppelton, and Dr. Eric Cooke at Southampton University, who opened up the world of computer science to me (Graham) and allowed me the freedom to explore the more creative sides of it.

This page intentionally left blank

About the Authors

Graham Robinson runs Shakinda Productions in Belfast, Ireland, specializing in innovative projection design and interactive visual systems creation. He believes that technology allows us to create art that can inspire humanity and transform society, and has performed audio-visual sets as VJ Shakinda worldwide.

Surya Buchwald runs MMMLabs in Portland, Oregon, creating interactive experiences for Nike, Intel, Scion, and others. He traverses the globe as the VJ for The Glitch Mob, bringing the Quartz Composer magic to fans all over. He also creates interactive video instruments and performs with them as Momo the Monster.

This page intentionally left blank

Chapter 1

What Is Quartz Composer and Why Should I Learn It?

Welcome to the first chapter! Are you ready to discover a whole new world? This chapter introduces Quartz Composer, explains how it differs from other applications you may have used before, and describes the many different end products you can create with it. We'll explain just how flexible it is and why you should always experiment, experiment, experiment!

Terms

The following terms are used throughout this book:

- **QC:** Quartz Composer.
 - **Patch:** the basic building blocks of QC. Each little box in the editor is a patch.
 - **Composition:** any saved QC file but normally a collection of patches you arrange to do something.
 - **Noodle:** the flexible lines that connect patches, created by clicking and dragging from a patch output or double-clicking on the patch output and then moving the mouse. Officially these lines are called “patchcords”—but that’s a bit boring so we’ll call them noodles.
-

Play Video Introduction to Quartz Composer

Still not really sure what Quartz Composer is? Quartz Composer (QC) is a node-based graphical programming language. If that sounds complex, it simply means that you will be connecting boxes with squiggly lines instead of editing a timeline or using a drawing tool (see Figure 1.1). QC allows you to build up things—for example, you can take a QuickTime video, pass it through a filter, combine it with shapes, and display that on the screen. It takes some getting used to, but between reading this book and viewing its DVD, you’ll be a QC ninja in no time.

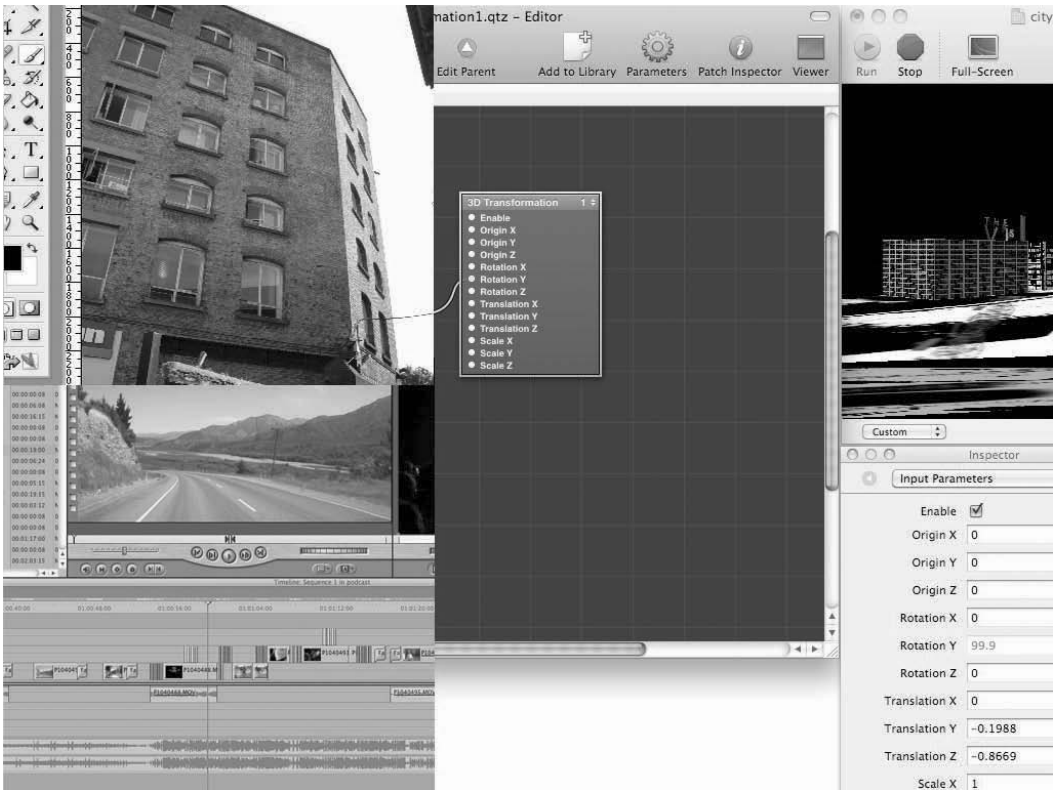


Figure 1.1 Quartz Composer interface versus Photoshop and Final Cut Pro



Play the video titled “Introduction to Quartz Composer.”

If you haven’t done so already, pop in the DVD and check out a few examples to see what’s possible with Quartz Composer.

At this stage, if you haven’t already set up Quartz Composer, you need to get it on your machine. This is a painless process, but it will involve digging out your original Mac install disks, (or rocking over to Mac dev at <http://developer.apple.com>, setting up an account, and downloading the installer).

Installing and Setting Up Quartz Composer



Play the video titled “Installing and Setting Up Quartz Composer.”

Follow these steps to install Quartz Composer:

1. Grab your Snow Leopard or Leopard Install DVD and stick it in your Mac.
2. Browse to Optional Installs > XcodeTools and double-click `XcodeTools.mpkg`.
3. Step through the installer and restart your machine.
4. Quartz Composer is now installed to Macintosh HD > Developer > Applications > Quartz Composer. If you drag the icon to your dock, it will add a launch shortcut.

Alternatively, if you are running Mac OSX Lion, follow these steps:

1. Open the App Store.
2. Search for “Xcode.”
3. Download and step through the installer.
4. Quartz Composer is now installed to Macintosh HD > Developer > Applications > Quartz Composer. If you drag the icon to your dock, it will add a launch shortcut.

Congratulations, you are ready to rock!

Outputs

There are many different ways to output your QC productions (or “compositions,” as we’ll call them). Most simply, you can share them as Quartz composition files (extension `.qtz`), which means other people will be able to see exactly how you made your composition and add their own ideas to it—a great choice for collaboration. If you save a Quartz composition to `~/Library/Screen Savers` or the `/Library/Screen Savers` folder, it’s ready to be a screensaver and will appear in your Screen Saver Preference Panel (see Figure 1.2).

If you want to include your composition in a webpage or dashboard widget (which load via the same WebKit plugin), use the `<embed>` tag. (For more information, go to http://developer.apple.com/documentation/GraphicsImaging/Conceptual/QuartzComposer/qc_webkit/chapter_8_section_1.html#//apple_ref/doc/uid/TP40001357-CH3-SW6.)

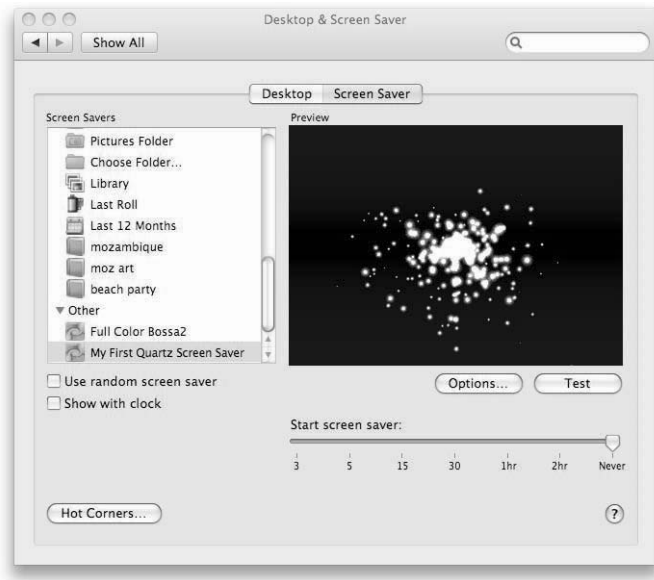


Figure 1.2 Quartz composition as a screensaver

Alternatively, you can export the composition as a QuickTime movie of a specified length and resolution. This method means that anyone with QuickTime installed can watch your composition and, if they have the QuickTime player version 7 or higher, will be able to see any interactive elements of your composition. Once you have the composition formatted as a movie, you can upload it to YouTube or load it into Final Cut, Video Jockey software, or other programs, just like any other movie file.

Not enough for you? Using Cocoa, it's simple to create an application that runs your composition. (See Chapter 13.)

Getting even geekier, you can “publish” certain controls of your compositions so they can be bound with an interface builder to create interfaces for other programs you may have written (http://vidvox.net/wiki/index.php/QuartzComposer_Adding_a_published_input). The publishing process can also be used in a specific Video Jockey setup with VDMX to create some very powerful live performance experiences.

Surely that's enough—but wait, there's more! Using a QCRenderer class, you can run a QC file in any OpenGL context. If you're not sure what an OpenGL context is, don't panic. . . . I'm covering all the bases here just to show you just how many different ways there are to output your creations from QC.

Flexibility

As you can see, there are many, many different ways to put your QC files to work—but that’s only half the story. The range of what you can put in a Quartz composition is also formidable.

QC has an impressive range of filters that come from the core image library. You can have a lot of fun just putting photos, videos, or Photoshop files through a couple of filters. QC can also combine images using 31 different blending modes, some of which Photoshop users will recognize—Add, Alpha, Multiply, and so on—and some they won’t—Luminescence Premultiply, anyone?

As well as being a great compositor of images and movies, QC loves live data streams, including audio being picked up by a microphone, RSS feeds from the Internet, MIDI and OSC data from other programs, JavaScript patches within a composition, and data from the ambient light and motion sensors from a MacBook Pro. All of these data streams can be manipulated and represented in incredible visualizations.

Three-dimensional environments can be created; 3D objects loaded, lit, textured, and animated; and camera moves animated on timelines. Lots of smooth and simple controller patches are also built into QC, which allow complex and organic movements to be created simply and quickly.

The possibilities are endless.

Experimentation

This is where you come in: Experiment, experiment, experiment. It is impossible for us, as authors, to highlight all of the possibilities provided by QC. To get the most from QC, you need to take the initiative and play around. Anywhere you see an input to a patch, think, “What could I connect to that?” The beauty of QC is that it’s live, so once you have that thought, just noodle out from the closest patch and find out what happens. Then try another hole, a different value, or another controller. Keep going until your composition looks nothing like what it was when you started!

We are expecting some awesome results from you, our apprentices, so in preparation, we created a special site for your creations at <http://www.iloveqc.org>. As soon as you have something that looks a bit different, get your experiments online and let everyone else enjoy them. Share your compositions, learn from others’ work, and collaborate. Let the QC explosion begin!

Summary

In this chapter we covered what Quartz Composer is, how to install it, and which kinds of wonderful things you can do with it. We also introduced experimentation as one of the key concepts to help you learn QC: Play around as much as possible and have fun!

Challenges

This chapter described how to install Quartz Composer, so why not start it up and play with some of the templates? If you get stuck, don't worry; just head to Chapter 2, where we will explain how to get started.

Index

Numbers

- 2 Sided Lighting, 94
- 3D (three-dimensional) coordinates, 16, 18
- 3D (three-dimensional) environments, 7. *See also* modeling complex environments
- 3D (three-dimensional) Transformation
 - Audio Input, 87
 - camera, 119
 - city block duplication, 114
 - city block replication, 116
 - hierarchies with Environment patches, 39–42
 - Lighting, 93
 - starting point composition, 51

A

- Add to Library
 - defined, 11
 - OSDebugger, 45
 - particles, 49–50
- adding visual effects. *See* visual effects
- Alpha option, 60–61
- Ambient Light, 93
- Amount, FPS, 30–31
- Amplitude, 39
- anchor point, 41
- app creation. *See* Cocoa app
- Attenuation, 93
- Attraction, 53–54
- Attribute Inspector, 123, 127
- Audio Band In, 114
- Audio Input
 - beat reactive building, 112–113
 - challenges, 90
 - output, 89–90
 - summary, 90
 - working with, 85–89
- Audio Multiplier, 114
- Audio Processor
 - beat reactive building, 112–113

- city block duplication, 114
- defined, 88–89
- modeling complex environments, 110

B

- bass, 85
- beat reaction, 112–113
- Billboard
 - adding visual effects, 32
 - Core Image Filters, 140
 - creating screensaver, 134
 - drag-and-drop interaction, 64
 - Kaleidoscope control, 68–69
 - playing a movie, 16, 18
 - using Image with String, 43–45
- Blend modes
 - particles, 57
 - real-world modeling, 55–56
- Blending
 - Particle System parameters, 54
 - Replicate in Space, 100
- Blendmode Explorer, 57
- blurs
 - Kineme Quartz Crystal, 90
 - Zoom Blur patch, 30–31
- Boxes
 - adding in MIDI controllers, 82–83
 - MIDI Notes to control Sprites, 78, 81
- Build Automation, 124, 127
- building texturing, 110–111

C

- C Boolean, 78
- calculator with Interpolation, 36–37
- camera
 - 3D cities, 119
 - Kaleidoscope control, 68
- City Block macro, 114
- cityscape modeling. *See* modeling complex environments

- Clear
 - Core Image Filters, 140
 - drag-and-drop interaction, 64
 - Image with String, 43
 - Interpolation patch, 34
 - iteration, 101
 - Kaleidoscope control, 68
 - MIDI interfacing, 72
 - Particle System, 59
 - starting point composition, 51
 - Cocoa app
 - challenges, 131
 - creating with Leopard/Snow Leopard, 122–124
 - creating with Lion, 124–127
 - publishing inputs with Leopard/Snow Leopard, 127–129
 - publishing inputs with Lion, 129–131
 - QC outputs, 6
 - summary, 131
 - Xcode, 121
 - Color
 - Iterator, 101
 - Lighting, 93
 - Particle System, 53
 - comet, 61–64
 - compiler, 121
 - complex environment modeling. *See* modeling complex environments
 - compositions, 5–6
 - Compute Specular Light Separately, 94
 - controls
 - adding in MIDI controllers, 82–83
 - further MIDI control, 84
 - Kaleidoscope, 68–69
 - MIDI Notes to control Sprites, 76–81
 - Particle System, 59–61
 - Core Image
 - filters, 139–142
 - FX and FPS, 30–32
 - cos waveforms, 38
 - Create Macro, 11
 - creating Cocoa app. *See* Cocoa app
 - Cube
 - Audio Input, 87
 - Interpolation, 34
 - Iterator, 101–105
 - LFO, 38
 - one city building, 112
 - Replicate in Space, 97–100
 - video walls, 115–116
 - Current Index, 102
 - Current Position, 102–104
- D**
- data streams, 7
 - Debug mode, 46
 - debugging
 - challenges, 47
 - summary, 46–47
 - tips, 45–46
 - using Image with String, 43–45
 - Xcode, 121
 - Decreasing Duration, 62
 - Decreasing Scale, 87
 - downloads
 - Midi Keys, 72
 - Soundflower, 86
 - drag-and-drop interaction, 64–67
 - duplicating 3D cities, 114
 - duration, 62
- E**
- Edit Parent, 11
 - Editor vs. WYSIWYG, 11–12
 - <embed> tag, 5
 - Envelope, 102–103
 - environment modeling. *See* modeling complex environments
 - Environment patches
 - Lighting, 91–94
 - Trackball and 3D Transformation, 39–42
 - experimentation, 7
 - Exponential In-Out, 64
 - exports
 - audio, 89–90
 - QC introduction, 5–6
 - from Viewer, 13
- F**
- faders
 - adding in MIDI controllers, 82
 - MIDI Notes to control Sprites, 78–79

- filter chains, 23–24
 - filters
 - adding, 21–23
 - Core Image, 139–142
 - Kaleidoscope. *See* Kaleidoscope
 - QC flexibility, 7
 - tools, 24–30
 - Final Rotation, 99–100
 - Final Translation, 99
 - fire modeling, 55–56
 - flame image, 55–56
 - flexibility, 7
 - Floor Cube, 118
 - floor level, 115
 - FPS (Frames Per Second)
 - Core Image and, 30–32
 - defined, 13
 - frequency
 - defined, 85
 - Structure Index Member, 89
- G**
- GLSL (OpenGL Shading Language), 139
 - Gravity, 54
- H**
- hierarchies with Environment patches, 39–42
 - HSL Color
 - adding mouse control, 60–61
 - MIDI Notes to control Sprites, 77
 - Hue, 61
 - Hue Adjust, 133
- I**
- IDE (integrated development environment), 121
 - Image
 - Iterator, 101
 - Kaleidoscope control, 68
 - real-world modeling, 55–56
 - texturing, 110
 - Image Crop
 - defined, 25–27
 - Rendering Destination Dimensions, 28–30
 - Image Dimensions, 133–134
 - Image with String
 - debugging, 43–45
 - Iterator, 103
 - importing QC library items, 123–124
 - Increasing Duration, 62, 64
 - Increasing Scale, 87
 - Input Splitter
 - adding options to screensaver, 135–136
 - City Block macro, 113–114
 - defined, 44–45
 - Iterator, 101, 104
 - Master Scale, 117
 - inputs
 - audio. *See* Audio Input
 - City Block macro, 114
 - Master Scale, 117–118
 - MIDI. *See* MIDI interfacing
 - mouse. *See* mouse input
 - publishing with Leopard/Snow Leopard, 127–129
 - publishing with Lion, 129–131
 - installing Quartz Composer, 5
 - integrated development environment (IDE), 121
 - interacting with audio. *See* Audio Input
 - Interaction patches, 64–67
 - interaction port, 64, 66
 - Interactive Placement mode, 46
 - interface
 - challenges, 19
 - Editor vs. WYSIWYG, 11–12
 - launching, 9–10
 - Patch Inspector/patch variables, 16, 17
 - Patch Library (Creator), 13, 15
 - playing a movie, 16, 18
 - summary, 18
 - Viewer, 13, 14
 - Interface Builder, 121, 123, 125–126
 - Interpolation
 - adding options to screensaver, 136
 - as calculator, 36–37
 - defined, 33–36
 - Iterator, 102
 - LFO patch and, 38
 - smoothing input, 64
 - summary, 42

Iterated String Cubes
 publishing inputs with Leopard/Snow
 Leopard, 127–129
 publishing inputs with Lion, 129–131
 Iterations, 102

Iterator
 challenges, 105
 exercise, 101–105
 summary, 105

J

Javascript patch
 defined, 139
 MIDI Virtual Macros and, 73–76

K

Kaleidoscope
 adding filter, 22–23
 control with mouse input, 68–69
 Image Crop, 25–27
 layering, 23–24
 Rendering Destination Dimensions, 28–30
 keyboards, MIDI. *See* MIDI interfacing
 keyframe timeline animation, 94–96
 Kineme Quartz Crystal, 90
 Kineme Safe Audio Input, 90
 knobs in MIDI controllers, 82

L

launching QC interface, 9–10
 layering filters, 23–24
 Leopard
 audio outputs, 90
 creating Cocoa app with, 122–124
 Debug mode, 46
 launching QC, 9
 publishing inputs with, 127–129
 QC installation, 5
 Timelines in, 94
 LFO (low-frequency oscillator)
 creating screensaver, 133–134
 defined, 37–39
 Iterator, 104–105
 OSDebugger, 45

summary, 42
 Virtual Macros, 50

Lifetime

challenges, 69
 Particle System parameters, 53

Lighting

finishing plan, 119
 Iterator, 101
 overview, 91
 Replicate in Space, 98–99
 summary, 96
 video tutorial, 92–94, 95

Lion. *See* Mac OSX Lion

Log patch, 46

looping Interpolation, 35

low-frequency oscillator (LFO). *See* LFO
 (low-frequency oscillator)

M

Mac OSX Lion

audio, 86
 creating Cocoa app with, 124–127
 publishing inputs with, 129–131
 QC installation, 5

macros

3D cities, 113–114
 duplication, 114
 MIDI Notes to control Sprites, 78
 video walls, 115–116

Master Scale, 117–118

Material Shininess, 94

Material Specularity, 94

Math

Audio Input, 87
 creating screensaver, 134
 Javascript patch and, 139
 Kaleidoscope control, 68
 MIDI Notes to control Sprites, 78
 Rendering Destination Dimensions, 28
 Replicate in Space, 97–100

Mathematical Expression

beat reactive building, 113
 City Block macro, 113–114
 Javascript patch and, 139
 Master Scale, 117–118

MIDI Channels, 72

- MIDI interfacing
 - adding in MIDI Controllers, 82–83
 - challenges, 84
 - further control, 84
 - MIDI Notes, 71–73
 - MIDI Notes to control Sprites, 76–81
 - summary, 84
 - Virtual Macros, 73–76
- Midi Keys, 72
- MIDI Notes
 - to control Sprites, 76–81
 - defined, 71–73
- mirror looping, 35
- modeling
 - defined, 49
 - real-world modeling, 54–57
- modeling complex environments
 - beat reaction, 112–113
 - camera, 119
 - challenges, 119
 - duplication, 114
 - finishing plan, 118–119
 - macros, 113–114
 - Master Scale, 117–118
 - one city building, 112
 - overview, 109–110
 - replication, 116
 - summary, 119
 - texturing, 110–112
 - video walls, 115–116
- Momentum Scrolling, 65–66
- mouse input
 - challenges, 69
 - drag-and-drop interaction, 64–67
 - Kaleidoscope control, 68–69
 - Particle System control, 59–61
 - Replicate in Space, 97–100
 - smoothing input, 61–64
 - summary, 69
- movie playing, 16, 18
- Multiply Blend Mode
 - defined, 24
 - Rendering Destination Dimensions, 28–29

N

- Native Core Image Rendering, 32
- Network Send/Receive, 84

- .nib files, 123
- node-based graphical programming
 - language, 3
- noodle, 3
- noteIn, 73–74
- noteOut, 73–74

O

- Observed Octaves, 72
- Offset, 39
- one city building, 112
- Open Sound Control (OSC), 84
- OpenCL (Open Computing Language), 142
- OpenGL context, 6
- OpenGL Shading Language (GLSL), 139
- orbiting animation, 67
- organic motion, 37–39
- Origin, 99
- OSC (Open Sound Control), 84
- OSDebugger
 - MIDI Notes and, 72–73
 - overview, 45
- Output Splitter, 77
- outputs
 - Audio Input, 89–90
 - Audio Processor, 88–89
 - Iterator Variables, 102
 - MIDI Notes and, 73
 - MIDI Virtual Macros, 73–76
 - Quartz Composer, 5–6

P

- parameters
 - adding options to screensaver, 135–136
 - adding to apps with Lion, 129–131
 - adding to apps with Snow Leopard/Leopard, 127–129
 - creating screensaver, 134
 - Particle System, 53–54
 - patch, 16
 - Replicate in Space, 99
 - Smooth, 62, 64
- Particle Count, 53
- Particle System
 - challenges, 131
 - control with mouse input, 59–61

- Particle System (*continued*)
 - Mouse challenges, 69
 - particles, 51–54
 - real-world modeling, 55–56
 - smoothing input, 61–64
 - particles
 - Add to Library, 49–50
 - Blend modes, 57
 - challenges, 57
 - creating app with Leopard/Snow Leopard, 122–124
 - creating Cocoa app with Lion, 124–127
 - Particle System, 51–54
 - real-world modeling, 54–57
 - starting point composition, 51
 - summary, 57
 - Patch Inspector
 - defined, 11
 - OpenCL, 142
 - overview, 16, 17
 - Patch Library (Creator)
 - Editor vs. WYSIWYG, 11
 - overview, 13, 15
 - Pixellate filter, 21–22
 - Virtual Macros, 50
 - visual effects, 32
 - Patch Parameters, 11
 - patch variables, 16, 17
 - patches, 3
 - Period
 - adding options to screensaver, 136
 - LFO patch options, 39
 - Phase, 39
 - Pixellate filter
 - adding, 21–22
 - Image Crop, 25
 - layering, 23–24
 - Rendering Destination Dimensions, 28–30
 - playing a movie, 16, 18
 - plugins, 138
 - Positioning Lighting, 93
 - private patches, 137–138
 - Process the Image
 - adding options to screensaver, 135–136
 - creating screensaver, 133
 - Profile mode, 46
 - properties
 - Javascript patch, 73
 - in Patch Inspector, 16, 17
 - Publish Outputs, 74
 - publishing inputs
 - with Leopard/Snow Leopard, 127–129
 - with Lion, 129–131
 - publishing QC outputs, 6
 - PWM Ratio, 39
- ## Q
- QC (Quartz Composer)
 - challenges, 8
 - experimentation, 7
 - flexibility, 7
 - installation and setup, 5
 - interface. *See* interface
 - MIDI Notes, 71
 - outputs, 5–6
 - summary, 7
 - video introduction, 3–4
 - QCRenderer class, 6
 - QuickTime
 - exporting audio to, 89–90
 - QC outputs, 6
- ## R
- rain modeling, 54–55
 - random waveforms, 38
 - real-world modeling, 54–57
 - Red, Green, Blue, Opacity, Delta, 54
 - render-type patches, 52
 - Rendering Destination Dimensions
 - defined, 28–30
 - MIDI Notes to control Sprites, 78, 81
 - Rendering Mode
 - debugging tips, 45–46
 - defined, 13
 - Rendering Options, 124
 - Replicate in Space
 - city block replication, 116
 - demo, 97–101
 - modeling complex environments, 110

- replication
 - 3D cities, 116
 - iteration and, 101–105
 - Replicate in Space, 97–101
 - summary, 105
- Result debugging, 45
- rotation
 - 3D Transformation, 41
 - Interpolation patch, 36
 - LFO patch options, 39
 - Replicate in Space, 99–100
 - Virtual Macros, 50

S

- sawtooth waveforms, 38
- scale
 - Audio Input, 87
 - Master Scale, 117–118
- Screen Capture, 90
- screensavers
 - adding options, 135–136
 - challenges, 136
 - making, 133–135
 - QC outputs as, 5–6
 - summary, 136
- setting up QC, 5
- Shadows, 93
- sharing, 7
- Shininess, 94
- Side Texture Image, 114
- Simulator
 - creating app with Leopard/Snow Leopard, 124
 - creating Cocoa app with Lion, 127
- sin waveforms, 38
- Size Delta, 54
- Smooth
 - MIDI Notes to control Sprites, 78
 - overview, 61–64
- Snow Leopard
 - creating Cocoa app with, 122–124
 - OpenCL, 142
 - publishing inputs with, 127–129
 - QC installation, 5
 - Rendering Mode, 46

- Sound, 86. *See also* Audio Input
- Soundflower, 86
- source code editor, 121
- Spectrum, 87–88
- Specularity
 - 3D cities, 119
 - Lighting, 94
- Speed
 - adding in MIDI controllers, 82–83
 - adding options to screensaver, 135
 - MIDI Notes to control Sprites, 78
- Sprite
 - drag-and-drop interaction, 64–67
 - MIDI Notes to control, 76–81
 - starting point composition, 51
 - video walls, 115
- square waveforms, 38
- starting point composition, 51
- String, 43–45
- String Truncate, 45
- Structure Index Member, 89

T

- Teapot
 - Lighting, 92–94
 - Timelines, 94–96
- templates for screensavers, 133–135
- Tension slider, 34–35
- terms, 3
- texturing
 - 3D cities, 110–112
 - City Block macro, 113
 - floor, 118
- three-dimensional (3D) coordinates, 16, 18
- three-dimensional (3D) environments, 7. *See also* modeling complex environments
- three-dimensional (3D) Transformation. *See* 3D (three-dimensional) Transformation
- Tiger, 9
- Timelines
 - challenges, 96
 - overview, 94–96
 - summary, 96
- Top and Btm Texture Image, 114

- Trackball
 - hierarchies with Environment patches, 39–41
 - Lighting, 93
 - starting point composition, 51
- Transformation, 3D. *See* 3D (three-dimensional) Transformation
- Translation X, Y, Z, 41–42
- treble, 85
- triangle waveforms, 38
- Tv Stand
 - Master Scale, 117–118
 - video walls, 115
- Tv Wall
 - Master Scale, 117–118
 - video walls, 115–116
- Twirl Distortion filter, 133–135
- Type, 38

U

- unsafe patches, 137–138

V

- variables, patch
 - Iterator Variables, 101–104
 - overview, 16, 17
- velocity
 - MIDI Notes, 71
 - Particle System parameters, 53
- Video Input
 - Core Image Filters, 140
 - Kaleidoscope control, 68–69
- video introduction, 3–4
- Video Jockey, 6
- video walls
 - 3D cities, 115–116
 - Master Scale, 117–118
- Viewer
 - Editor vs. WYSIWYG, 11
 - Environment patches, 39–42
 - overview, 13, 14

- Virtual Macros
 - City Block, 113
 - defined, 50
 - MIDI, 73–76
- visual effects
 - adding filter, 21–23
 - challenges, 32
 - Core Image Filters, 139–142
 - Core Image FX and FPS, 30–32
 - filter chains and layering, 23–24
 - filter tools, 24–30
 - summary, 32
- visual programming language, 11
- Volume, 87
- Volume Peak, 87

W

- waves, 38–39
- wrapping texture, 110
- WYSIWYG (What You See Is What You Get) vs. Editor, 11–12

X

- X, Y, Z Min and Max Velocity, 53
- Xcode
 - creating app with Leopard/Snow Leopard, 122–124
 - creating app with Lion, 124–127
 - overview, 121
 - .xib files, 123
- XPos Env, 102–103

Y

- YouTube, 6

Z

- Zoom Blur, 30–31
- zoom controls, 11