



Oculus Rift

IN ACTION

Bradley Austin Davis
Karen Bryla
Phillips Alexander Benton
FOREWORD BY Philip Rosedale

 MANNING

Oculus Rift in Action

Bradley Austin Davis, Karen Bryla, and Phillips Alexander Benton



Copyright

For online information and ordering of this and other Manning books, please visit www.manning.com. The publisher offers discounts on this book when ordered in quantity. For more information, please contact

Special Sales Department
Manning Publications Co.
20 Baldwin Road
PO Box 761
Shelter Island, NY 11964
Email: orders@manning.com

©2015 by Manning Publications Co. All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by means electronic, mechanical, photocopying, or otherwise, without prior written permission of the publisher.

Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in the book, and Manning Publications was aware of a trademark claim, the designations have been printed in initial caps or all caps.

♻️ Recognizing the importance of preserving what has been written, it is Manning's policy to have all books we publish printed on acid-free paper, and we exert our best efforts to that end. Recognizing also our responsibility to conserve the resources of our planet, Manning books are printed on paper that is at least 15 percent recycled and processed without the use of elemental chlorine.



Manning Publications Co.
20 Baldwin Road
PO Box 761
Shelter Island, NY 11964

Development editor: Dan Maharry
Technical development editor Justin Chase
Copyeditor: Liz Welch
Proofreader: Elizabeth Martin
Technical proofreader: Frederik Vanhoutte
Typesetter: Dennis Dalinnik
Cover designer: Marija Tudor

ISBN 9781617292194

Printed in the United States of America

1 2 3 4 5 6 7 8 9 10 – EBM – 20 19 18 17 16 15

Dedication

For Leo and Kesten

B.D.

For Sam, Ted, and Max

K.B.

For Antonia

A.B.

Brief Table of Contents

[Copyright](#)

[Brief Table of Contents](#)

[Table of Contents](#)

[Foreword](#)

[Preface](#)

[Acknowledgments](#)

[About this Book](#)

[About the Authors](#)

[Author Online](#)

[About the Cover Illustration](#)

[1. Getting started](#)

[Chapter 1. Meet the Oculus Rift](#)

[2. Using the Oculus C API](#)

[Chapter 2. Creating your first Rift interactions](#)

[Chapter 3. Pulling data out of the Rift: working with the head tracker](#)

[Chapter 4. Sending output to the Rift: working with the display](#)

[Chapter 5. Putting it all together: integrating head tracking and 3D rendering](#)

[Chapter 6. Performance and quality](#)

[3. Using Unity](#)

[Chapter 7. Unity: creating applications that run on the Rift](#)

[Chapter 8. Unity: tailoring your application for the Rift](#)

[4. The VR user experience](#)

[Chapter 9. UI design for VR](#)

[Chapter 10. Reducing motion sickness and discomfort](#)

[5. Advanced Rift integrations](#)

[Chapter 11. Using the Rift with Java and Python](#)

[Chapter 12. Case study: a VR shader editor](#)

[Chapter 13. Augmenting virtual reality](#)

[Appendix A. Setting up the Rift in a development environment](#)

[Appendix B. Mathematics and software patterns for 3D graphics](#)

[D. Glossary](#)

[Index](#)

[List of Figures](#)

[List of Tables](#)

[List of Listings](#)

Table of Contents

[Copyright](#)

[Brief Table of Contents](#)

[Table of Contents](#)

[Foreword](#)

[Preface](#)

[Acknowledgments](#)

[About this Book](#)

[About the Authors](#)

[Author Online](#)

[About the Cover Illustration](#)

[1. Getting started](#)

[Chapter 1. Meet the Oculus Rift](#)

[1.1. Why support the Rift?](#)

[1.1.1. The call of virtual reality](#)

[1.1.2. But what about the Rift?](#)

[1.2. How is the Rift being used today?](#)

[1.3. Get to know the Rift hardware](#)

[1.3.1. The DK2](#)

[1.3.2. The DK1](#)

[1.3.3. The GPU](#)

[1.4. How the Rift works](#)

[Conventional applications](#)

[Rift applications](#)

[1.4.1. Using head tracking to change the point of view](#)

[1.4.2. Rendering an immersive view](#)

[1.5. Setting up the Rift for development](#)

[1.6. Dealing with motion sickness](#)

[1.7. Development paths](#)

[1.8. Summary](#)

[2. Using the Oculus C API](#)

2.1. SDK interfaces

2.1.1. Oculus runtime

2.1.2. Oculus SDK

2.2. Working with the SDK

2.2.1. SDK management

2.2.2. Managing the HMD

2.3. Getting input from the head tracker

2.3.1. Reserving a pointer to the device manager and locating the headset

2.3.2. Fetching tracker data

2.3.3. Reporting tracker data to the console

2.3.4. Exiting and cleaning up

2.3.5. Understanding the output

2.4. A framework for demo code: the GlfwApp base class

2.5. Rendering output to the display

2.5.1. The constructor: accessing the Rift

2.5.2. Creating the OpenGL window

2.5.3. Rendering two rectangles, one for each eye

2.6. What's next?

2.7. Summary

Chapter 3. Pulling data out of the Rift: working with the head tracker

3.1. The head tracker API

3.1.1. Enabling and resetting head tracking

3.1.2. Receiving head tracker data

3.2. Receiving and applying the tracker data: an example

3.2.1. Initial setup and binding

3.2.2. Fetching orientation

3.2.3. Applying the orientation to the rendered scene

3.3. Additional features: drift correction and prediction

3.3.1. Drift correction

[3.3.2. Prediction](#)

[3.3.3. Using drift correction and prediction](#)

[3.4. Summary](#)

[Chapter 4. Sending output to the Rift: working with the display](#)

[4.1. Targeting the Rift display](#)

[4.1.1. Extended vs. Direct HMD mode](#)

[4.1.2. Creating the OpenGL window: choosing the display mode](#)

[4.1.3. Creating the OpenGL window: Extended Desktop mode](#)

[4.1.4. Creating the OpenGL window: Direct HMD mode](#)

[4.1.5. Full screen vs. windowed: extensions with glfwCreateWindow\(\)](#)

[4.1.6. Dispensing with the boilerplate](#)

[4.2. How the Rift display is different: why it matters to you](#)

[4.2.1. Each eye sees a distinct half of the display panel](#)

[4.2.2. How the lenses affect the view](#)

[4.3. Generating output for the Rift](#)

[4.4. Correcting for lens distortion](#)

[4.4.1. The nature of the distortion](#)

[4.4.2. SDK distortion correction support](#)

[4.4.3. Example of distortion correction](#)

[4.5. Summary](#)

[Chapter 5. Putting it all together: integrating head tracking and 3D rendering](#)

[5.1. Setting the scene](#)

[5.2. Our sample scene in monoscopic 3D](#)

[5.3. Adding stereoscopy](#)

[5.3.1. Verifying your scene by inspection](#)

[5.4. Rendering to the Rift](#)

[5.4.1. Enhanced data for each eye](#)

[5.4.2. Improved user settings](#)

[5.4.3. Setting up the SDK for distortion rendering](#)

[5.4.4. The offscreen framebuffer targets](#)

[5.4.5. The Oculus texture description](#)

[5.4.6. Projection and modelview offset](#)

[5.4.7. The Rift's rendering loop](#)

[5.5. Enabling sensors](#)

[5.5.1. Implications of prediction](#)

[5.5.2. Getting your matrices in order](#)

[5.6. Summary](#)

[Chapter 6. Performance and quality](#)

[6.1. Understanding VR performance requirements](#)

[Higher performance requirements](#)

[Stricter performance requirements](#)

[Higher rendering cost](#)

[6.2. Detecting and preventing performance issues](#)

[Follow the SDK guidelines](#)

[Optimizing your rendering pipeline](#)

[Detecting performance issues](#)

[6.3. Using timewarp: catching up to the user](#)

[The problem: pose and prediction diverge](#)

[The solution: timewarp](#)

[6.3.1. Using timewarp in your code](#)

[6.3.2. How timewarp works](#)

[6.3.3. Limitations of timewarp](#)

[6.4. Advanced uses of timewarp](#)

[6.4.1. When you're running early](#)

[6.4.2. When you're running late](#)

[6.5. Dynamic framebuffer scaling](#)

[6.6. Summary](#)

[3. Using Unity](#)

[Chapter 7. Unity: creating applications that run on the Rift](#)

[7.1. Creating a basic Unity project for the Rift](#)

[7.1.1. Use real-life scale for Rift scenes](#)

[7.1.2. Creating an example scene](#)

[7.2. Importing the Oculus Unity 4 Integration package](#)

[7.3. Using the Oculus player controller prefab: getting a scene on the Rift, no scripting required](#)

[7.3.1. Adding the OVRPlayerController prefab to your scene](#)

[7.3.2. Doing a test run: the Unity editor workflow for Rift applications](#)

[7.3.3. The OVRPlayerController prefab components](#)

[7.4. Using the Oculus stereo camera prefab: getting a scene on the Rift using your own character controller](#)

[Create a scene](#)

[Add a character controller](#)

[Add the OVRCameraRig prefab to the character controller](#)

[Change the mouselook script for use with the Rift](#)

[7.4.1. The OVRCameraRig prefab components](#)

[7.5. Using player data from the user's profile](#)

[7.5.1. Ensuring the user has created a profile](#)

[7.6. Building your application as a full screen standalone application](#)

[7.7. Summary](#)

[Chapter 8. Unity: tailoring your application for the Rift](#)

[8.1. Creating a Rift-friendly UI](#)

[8.1.1. Using the Unity GUI tools to create a UI](#)

[8.1.2. Creating an in-world UI](#)

[8.2. Using Rift head tracking to interact with objects](#)

[8.2.1. Setting up objects for detection](#)

[8.2.2. Selecting and moving objects](#)

[8.2.3. Using collision to put the selected object down](#)

[8.3. Easing the user into VR](#)

[8.3.1. Knowing when the health and safety warning has been dismissed](#)

[8.3.2. Re-centering the user's avatar](#)

[8.3.3. Creating splash scenes](#)

[8.4. Quality and performance considerations](#)

[8.4.1. Measuring quality: looking at application frame rates](#)

[8.4.2. Using timewarp](#)

[8.4.3. \(Not\) Mirroring to the display](#)

[8.4.4. Using the Unity project quality settings](#)

[8.5. Summary](#)

[4. The VR user experience](#)

[Chapter 9. UI design for VR](#)

[9.1. New UI paradigms for VR](#)

[9.1.1. UI conventions that won't work in VR and why](#)

[9.1.2. Can your world tell your story?](#)

[9.1.3. Getting your user from the desktop to VR](#)

[9.1.4. Cutscenes](#)

[9.2. Designing 3D user interfaces](#)

[9.2.1. Criteria for a good UI](#)

[9.2.2. Guidelines for 3D scene and UI design](#)

[9.2.3. The mouse is mightier than the sword](#)

[9.2.4. Using the Rift as an input device](#)

[9.3. Animations and avatars](#)

[9.3.1. Cockpits and torsos: context in the first person](#)

[9.3.2. Character animations](#)

[9.4. Tracking devices and gestural interfaces](#)

[9.4.1. Beyond the gamepad](#)

[9.4.2. Gestural interfaces](#)

[9.5. Summary](#)

[Chapter 10. Reducing motion sickness and discomfort](#)

[10.1. What does causing motion sickness and discomfort mean?](#)

[10.2. Strategies and guidelines for creating a comfortable VR environment](#)

[10.2.1. Start with a solid foundation for your VR application](#)

[10.2.2. Give your user a comfortable start](#)

[10.2.3. The golden rule of VR comfort: the user is in control of the camera](#)

[10.2.4. Rethink your camera work: new approaches for favorite techniques](#)

[10.2.5. Make navigation as comfortable as possible: character movement and speed](#)

[10.2.6. Design your world with VR constraints in mind](#)

[10.2.7. Pay attention to ergonomics: eyestrain, neck strain, and fatigue](#)

[10.2.8. Use sound to increase immersion and orient the user to action](#)

[10.2.9. Don't forget your user: give the player the option of an avatar body](#)

[10.2.10. Account for human variation](#)

[10.2.11. Help your users help themselves](#)

[10.2.12. Evaluate your content for use in the VR environment](#)

[10.2.13. Experiment as much as possible](#)

[10.3. Testing your VR application for motion sickness potential](#)

[10.3.1. Use standardized motion and simulator sickness questionnaires](#)

[10.3.2. Test with a variety of users and as many as you can](#)

[10.3.3. Test with new users](#)

[10.3.4. Test with users who have set their personal profile](#)

[10.3.5. Test in stages](#)

[10.3.6. Test in different display modes](#)

[10.4. Summary](#)

[5. Advanced Rift integrations](#)

[Chapter 11. Using the Rift with Java and Python](#)

[11.1. Using the Java bindings](#)

[Requirements](#)

[JNA vs. JNI vs. Homebrew](#)

[11.1.1. Meet our Java binding: JOVR](#)

[11.1.2. The Jocular-examples project](#)

[11.1.3. The RiftApp class](#)

[11.1.4. The RiftDemo class](#)

[11.2. Using the Python bindings](#)

[11.2.1. Meet our Python binding: PyOVR](#)

[11.2.2. Development environment](#)

[11.2.3. The pyovr-examples project](#)

[11.2.4. The RiftApp class](#)

[11.2.5. The RiftDemo class](#)

[11.3. Working with other languages](#)

[11.4. Summary](#)

[Chapter 12. Case study: a VR shader editor](#)

[12.1. The starting point: Shadertoy](#)

[12.2. The destination: ShadertoyVR](#)

[12.3. Making the jump from 2D to 3D](#)

[12.3.1. UI layout](#)

[12.3.2. User inputs](#)

[12.3.3. Project planning](#)

[12.3.4. Picking our feature set](#)

[12.3.5. UI design](#)

[12.3.6. Windowing and UI libraries](#)

[12.4. Implementation](#)

[12.4.1. Supporting the Rift in Qt](#)

[12.4.2. Offscreen rendering and input processing](#)

[12.5. Dealing with performance issues](#)

[Finding your target and reacting when you're not hitting it](#)

[Eye-per-frame mode and timewarp](#)

[Dynamic framebuffer scaling](#)

[Scaling texture in the VR scene, not the UI](#)

[12.6. Building virtual worlds on the GPU](#)

[12.6.1. Raycasting: building 3D scenes one pixel at a time](#)

[12.6.2. Finding the ray direction in 2D](#)

[12.6.3. Finding the ray direction in VR](#)

[12.6.4. Handling the ray origin: stereopsis and head tracking](#)

[12.6.5. Adapting an existing Shadertoy shader to run in ShadertoyVR](#)

[12.7. Summary](#)

[Chapter 13. Augmenting virtual reality](#)

[13.1. Real-world images for VR: panoramic photography](#)

[13.1.1. Panorama photos](#)

[13.1.2. Photo spheres](#)

[13.1.3. Photo spheres...in space!](#)

[13.2. Using live webcam video in the Rift](#)

[13.2.1. Threaded frame capture from a live image feed](#)

[13.2.2. Image enhancement](#)

[13.2.3. Proper scaling: webcam aspect ratio](#)

[13.2.4. Proper ranging: field of view](#)

[13.2.5. Image stabilization](#)

[13.3. Stereo vision](#)

[13.3.1. Stereo vision in our example code](#)

[13.3.2. Quirks of stereo video from inside the Rift](#)

[13.4. The Leap Motion hand sensor](#)

[13.4.1. Developing software for the Leap Motion and the Rift](#)

[13.4.2. The Leap, the Rift, and their respective coordinate systems](#)

[13.4.3. Demo: integrating Leap and Rift](#)

[13.5. Summary](#)

[Appendix A. Setting up the Rift in a development environment](#)

[A.1. Selecting a display mode: Direct HMD Access or Extended Desktop mode](#)

[A.2. Configuring the displays in your OS for Extended Desktop mode](#)

[A.2.1. Extending or cloning \(mirroring\): which should you choose?](#)

[A.3. Improving your development environment](#)

[A.3.1. Fix it](#)

[A.3.2. Fix it cheaply](#)

[A.3.3. Clone it with a gadget](#)

[A.3.4. Remote development](#)

[A.4. Configuring the Rift for your use](#)

[Adjusting the headset for a proper fit](#)

[A.4.1. Create a user profile](#)

[A.5. Verifying your setup and troubleshooting](#)

[Appendix B. Mathematics and software patterns for 3D graphics](#)

[B.1. Coordinate systems](#)

[B.2. Introduction to matrices](#)

[B.3. Matrix transforms](#)

[B.4. Representing rotation](#)

[B.4.1. Euler angles](#)

[B.4.2. Quaternions](#)

[B.4.3. Spherical linear interpolation \(“slerp”\)](#)

[B.5. The scene graph software design pattern](#)

[B.6. The matrix stack software design pattern](#)

[B.7. The modelview software design pattern](#)

[Appendix C. Suggested books and resources](#)

[Books, research papers, and websites](#)

[3D graphics programming](#)

[OpenGL](#)

[Developing for the Rift](#)

[Motion sickness/simulator sickness](#)

[UI design for VR](#)

[Unity](#)

[Demos, games, and apps](#)

[VR demos, games, and applications worth a view](#)

[Oculus Share](#)

[D. Glossary](#)

[Index](#)

[List of Figures](#)

[List of Tables](#)

[List of Listings](#)

Foreword

Two amazing advances from the smartphone arms race have come together to create the head-mounted display (HMD): light, cheap, high-resolution displays, and a new generation of super accurate and fast-motion sensor chips. Rather than display information or graphics on the surface in front of you, these displays rest on your head and update quickly enough to convince you that what you're seeing is as real as the place you left behind. Although HMDs have existed for decades, they've never worked well enough to be more than aspirational prototypes destined for science museums. But with the Oculus Rift, the sensation of having a stable 3D world surround you as you move your head is a game-changing shift from peering into a 3D space through a desktop screen or handheld device.

Within the next 10 years, improved devices like the Oculus Rift will replace many of the screens we surround ourselves with today as their resolution scales to eclipse our TVs and monitors. Ultimately, we will use them to replace as much or as little of the world around us as we choose, with digital content that is indistinguishable from reality. The impact of these first-generation devices on gaming and virtual worlds will be incredible.

But along this road there are many changes to UI, experience, and computing paradigms that you'll need to understand, and the authors of *Oculus Rift in Action* take you on a comprehensive overview of them. As a developer getting started with the Rift, you get a complete walkthrough of connecting to and rendering to the device.

Beyond this, you'll learn the important differences raised by such devices: How do you type without a keyboard? If Microsoft and the Mac revolutionized computing by putting things in windows, what will we do in an HMD? Why do we get sick using these devices, and how can we fix that? This book gives you a complete and grounded overview of the specific technology and operation of the Oculus Rift, as well as the big picture topics that you'll need to survive in a new world without monitors. Finally, it dives into the new and complex design factors around how to correctly control things, navigate, and build in the virtual world as an "avatar" given the capabilities and limitations of these new input devices for the head and body.

PHILIP ROSEDALE CREATOR OF SECOND LIFE

Preface

No matter what people have, they always dream of something more: more power, more influence, more knowledge, but perhaps most importantly, more possibilities. This drive is part of the human condition and is responsible for our going from the Wright brothers to Apollo 11 in a single century.

If you want the future, you have to build it yourself. But the future I want, the one I think many of us want, isn't something we can each build on our own, if only for lack of time and resources.

We've written this book to lend a hand to those who want to help build the future in virtual reality (VR) but perhaps don't know where to start.

BRAD DAVIS SEATTLE, WA

Virtual reality was not something I'd expected to ever get involved in. As fun as it was to daydream about having my own holodeck to simulate an environment as if I were really there, the technology never seemed to be there, and so I pursued other work. My coauthor Brad, though, paid more attention and spotted the Oculus Rift on Kickstarter. As an early backer, he was very enthusiastic about its potential to create truly commercial VR. Brad made it sound interesting enough that I ordered my own DK1 development kit. While I waited the two months for it to ship, I researched what others were doing and watched YouTube videos. When it finally arrived, nothing I'd seen or read could do justice to the actual experience.

Like many people, my first experience was the *Oculus World* (also known as *Tuscany*) demo. In it you can meander around an old Tuscan villa. The graphics aren't spectacular, and the low resolution on the DK1 made it appear as though I was looking through a screen door, but those things didn't matter one little bit when I tilted my head to look up and the scene changed to match where I was looking. I was overcome by giggly delight, looking up at the wooden rafters of the house. When I moved my avatar outside, I looked up to see the sky. This was immersion as I'd never felt before, and it was amazing.

That first experience sent my mind racing with thoughts about the potential of VR. I could see the Rift being used for gaming, virtual tourism, storytelling, and science. But to me, education was the most interesting, and it's where I first saw the Rift's potential turned into reality. When my younger son came home from school telling me he was learning about Paris and the Eiffel Tower for multicultural day, I downloaded the *Tower Eiffel* demo by Didier They and let him see what it's like to stand beneath the tower's impressive metal arches.

When my boys and I watched the *Nova* television series with Neil deGrasse Tyson, I downloaded *Titans of Space* by DrashVR so that they could take their own trip through the solar system and feel how grand and vast the universe truly is. They, of course, now want to visit Paris and work for NASA and I'm truly excited to see what the future brings.

KAREN BRYLA TINTON FALLS, NJ

A long time ago, I noticed that people are always looking *around* but they rarely look *up*. I guess it's because there's not usually a lot of stuff overhead to see. I thought that if I could help people learn to

look up as often as they look around, then we would go to space sooner, because people would look up at the stars and the moon and think, “Hey, let’s go check that out.” And I want to go to the moon. Not just as a one-off thing where you leave your lander behind when you go home—I want humans to have real cities in space, with shops and streets and hot dog stands.

So I got into computer graphics because of space. I figured that the best way I can get there (short of becoming an astronaut, which seems too much like real work) is to make virtual reality happen. I want to put people into virtual worlds that train them to expect more from the real one. In VR, there’s no reason for the world not to stretch as far above you as it does to either side. In VR, we can make worlds where all the best stuff is overhead, and you’ll always have to look up to find it. After a while looking up will get to be a habit.

And if we can teach people to look up, then someday I’ll eat a hot dog on the moon.

ALEX BENTON LONDON, ENGLAND

Acknowledgments

Creating this book was not an isolated effort by the writers. In fact, it took tremendous effort, patience, and support from a great many wonderful and talented people to make this book possible.

Thank you to Dan Maharry, our development editor at Manning, who has the patience of a saint and excellent taste in '90s sci-fi. His guidance throughout this process was invaluable. Thanks to Robin de Jongh for getting the ball rolling for us. And thank you to Mary Piergies, Liz Welch, Elizabeth Martini, Kevin Sullivan, Justin Chase, and Frederik Vanhoutte, and the rest of the team at Manning for all of their help getting this book to completion, something that at times felt like a Sisyphean task. It takes an amazing team to get that rock over the hill.

We need to give special thanks to Iñigo Quilez and Pol Jeremias, authors of Shadertoy.com, for their advice and support on ShadertoyVR. And we also want to thank Philip Rosedale, creator of Second Life, for writing the foreword.

We'd also like to thank our MEAP readers for their comments and corrections to our early draft chapters, and to the following reviewers who read the manuscript at various stages during development: Alex Lucas, Andrew Henderson, Bas van Oerle, Behram Patel, Çağatay Çatal, Daniel Walters, George Freeman, Jan-Jaap Severs, Joaquin Gracia, Jose San Leandro, Kathleen Estrada, Ker Fricklas, Mackenzie Zastrow, Marco Massenzio, and Scott Chaussée.

Finally, none of us could have done this without the support of our families.

BRAD DAVIS wishes to thank Leo and Kesten, for all their understanding and devotion.

KAREN BRYLA wishes to thank her husband Sam Kass and her children, Ted and Max, for their patience and support. She particularly wants to thank them for so willingly testing out early versions of demos to help her better understand what triggers motion sickness in VR. She also needs to thank Sam for taking many of the photos used in this book and for his invaluable feedback on the text and examples.

ALEX BENTON wishes to thank his amazing wife, Dr. Antonia Benton, for her constant support and encouragement, and Verna Coulson for her unwavering enthusiasm.

About this Book

Oculus Rift in Action is designed to help you create comfortable and usable virtual reality (VR) applications that run on the Oculus Rift head-mounted display.

How this book is organized

This book is organized into five parts:

- **[Part 1: Getting started](#)**— [Part 1](#) introduces you to VR and the Oculus Rift hardware. We'll cover why you'd want to support the Rift in your software and how the Rift works.
- **[Part 2: Using the Oculus C API](#)**— [Part 2](#) covers how to develop Rift applications using the Oculus C API. Whether you're looking to write applications using the API directly, to integrate Rift support into your own game engine, or simply to better understand how Rift support works your game engine of choice (Unity, for example), this part of the book is for you.
- **[Part 3: Using Unity](#)**— [Part 3](#) covers how to use Unity, a popular development IDE and 3D graphics engine, to develop Rift applications. Unity is a great way to jump-start creating 3D games as it handles just about every aspect of game development, such as graphics, audio, physics, interaction and networking. With the Unity integration package from Oculus, you can quickly get your application running on the Rift. If you want to use Unity for your VR development, you'll find much value in [part 3](#).
- **[Part 4: The VR user experience](#)**— In [part 4](#), we turn our attention to the VR experience. No matter how you've created your VR application, you're going to want to design your application so that it's comfortable and easy to use in the VR environment. In this part of the book we look at the challenges of creating a usable UI for the VR environment. We cover some of the common pitfalls of designing a UI for VR along with the latest research into the key components to an immersive virtual experience. We also take a look at what you can do to maximize user comfort including guidelines and examples of how to mitigate motion sickness triggers and other causes of physical discomfort such as fatigue and eyestrain.
- **[Part 5: Advanced Rift integrations](#)**— In the final chapters, we provide information and examples for work that goes beyond the core integration of the Rift APIs. Here you'll learn to work with the Oculus C API using Java or Python, along with the basics of how to use the C APIs with any language. We also provide an example of creating a complete VR experience by building a VR version of an existing web application for use on the Rift. Finally, we cover integrating additional inputs into Rift apps, using modern hardware like web cameras and the Leap Motion.

Wondering where to start? Every reader should start with [part 1](#) because it introduces you to the hardware and to the virtual reality concepts we'll be using throughout the book. After that, where you go depends on how you plan to develop your application. C/C++ developers will want to turn to [part 2](#) and Unity developers to [part 3](#). No matter how you're going to develop, your next stop should be [part 4](#), to learn how to ensure your users get the most out of your application. When you're ready to move on to advanced Rift integrations and see a full-fledged VR app in action, turn to [part 5](#).

What this book doesn't do

This book doesn't cover how to use OpenGL, nor does it discuss the basics of 3D programming. It also doesn't cover C or C++ or how to use any particular development environment. If you're unfamiliar with these topics, you'll find some good references listed in [appendix C](#).

Code conventions and downloads

All source code in the book is in a `fixed-width font` like this, which sets it off from the surrounding text. In many listings, the code is annotated to point out the key concepts, and numbered bullets are sometimes used in the text to provide additional information about the code.

We have tried to format the code so that it fits within the available page space in the book by adding line breaks and using indentation carefully. Sometimes, however, very long lines include line-continuation markers. **Fixed-width font** like this in listings indicates new code.

Source code for all the working examples in this book is available online in our GitHub repository at github.com/OculusRiftInAction/OculusRiftInAction.

If you're using Unity ([part 3](#) of this book), you don't need to download the entire example repository. The scripts and the example scenes for [part 3](#) are in `/examples/unity`.

If you're using the C API ([part 2](#) of this book), details of how to download and build the C++ and Java example applications are discussed next.

- **[download online Abstract Space: Beneath the Media Surface](#)**
- [Honey and Jam pdf, azw \(kindle\)](#)
- [download online Action Plan for Diabetes: Your Guide to Controlling Blood Sugar \(Action Plan for Health series\) pdf, azw \(kindle\), epub](#)
- [read online Betty Crocker Vegetarian Cooking here](#)

- <http://diy-chirol.com/lib/Ice-World--Techniques-and-Experiences-of-Modern-Ice-Climbing.pdf>
- <http://diy-chirol.com/lib/Bitter-Sweet-Love--The-Dark-Elements--Book-0-5-.pdf>
- <http://kamallubana.com/?library/Shadow-of-the-Thylacine--One-Man-s-Epic-Search-for-the-Tasmanian-Tiger.pdf>
- <http://creativebeard.ru/freebooks/Much-Ado-About-Murder--Shakespeare---Smythe--Book-3-.pdf>