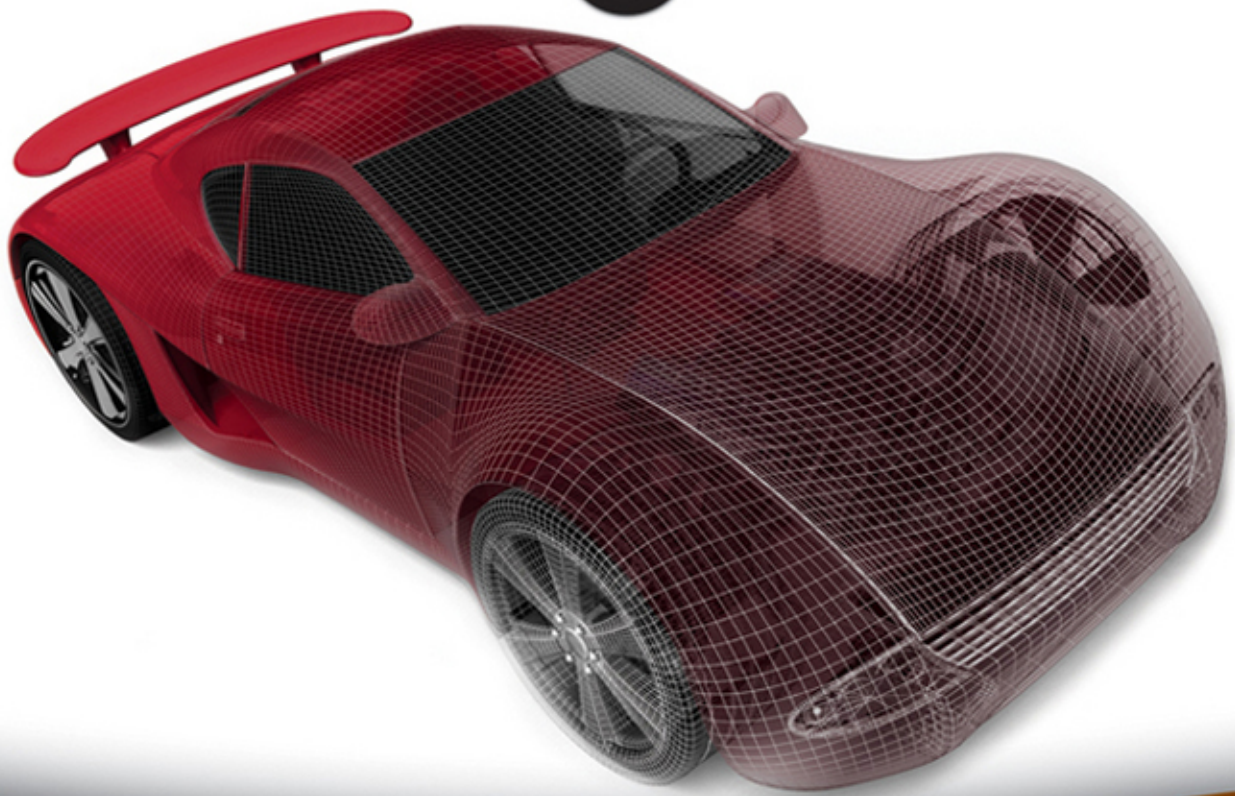
Join the discussion @ p2p.wrox.com

Wrox Programmer to Programmer™



Professional WebGL Programming

Developing 3D Graphics for the Web

Andreas Anyuru

PROFESSIONAL WEBGL™ PROGRAMMING

PROFESSIONAL

WebGL™ Programming

PROFESSIONAL

WebGL™ Programming

DEVELOPING 3D GRAPHICS FOR THE WEB

Andreas Anyuru

John Wiley & Sons, Ltd.

This edition first published 2012

Registered office

John Wiley & Sons Ltd, The Atrium, Southern Gate, Chichester, West Sussex, PO19 8SQ, United Kingdom

For details of our global editorial offices, for customer services and for information about how to apply for permission to reuse the copyright material in this book please see our website at www.wiley.com.

978-1-119-96886-3
978-1-119-94058-6 (ebk)
978-1-119-94059-3 (ebk)
978-1-119-94060-9 (ebk)

10 9 8 7 6 5 4 3 2 1

A catalogue record for this book is available from the British Library.

Printed in the United States by Bind-Rite.

To Jessica, Tilde, and Victor

ABOUT THE AUTHOR

ANDREAS ANYURU has extensive experience with various web technologies, including WebGL. He has worked with web browser integration and optimizations for mobile devices for many years. Andreas is one of the contributors to the V8 JavaScript engine that is used in Google Chrome for desktops and in Android. He has developed new courses and taught at the Information and Communications Engineering division at Lund University.

Andreas is a Senior Member of Technical Staff at ST-Ericsson, where his primary area of expertise is web technologies. In this position, he is responsible for ensuring that existing and forthcoming web technologies work in an optimal way on ST-Ericsson's Linux-based mobile platforms.

Andreas holds a M.Sc. in Electrical Engineering from Lund University, Faculty of Engineering, Sweden.

ABOUT THE TECHNICAL EDITOR

PAUL BRUNT was working with JavaScript early on in the emergence of HTML5 to create some early games and applications that made extensive use of SVG, canvas, and a new generation of fast JavaScript engines. This work included a proof-of-concept platform game demonstration called "Berts Breakdown." With a keen interest in computer art and an extensive knowledge of Blender, combined with knowledge of real-time graphics, the introduction of WebGL was the catalyst for the creation of GLGE. He began work on GLGE in 2009 when WebGL was still in its infancy, gearing it heavily towards the development of online games. Paul has also contributed to other WebGL frameworks and projects, as well as porting the jiglib physics library to javascript in 2010, demoing 3D physics within a browser for the first time.

CREDITS

VP CONSUMER AND TECHNOLOGY PUBLISHING DIRECTOR
Michelle Leete

ASSOCIATE DIRECTOR—BOOK CONTENT MANAGEMENT
Martin Tribe

ASSOCIATE PUBLISHER
Chris Webb

EXECUTIVE COMMISSIONING EDITOR
Birgit Gruber

ASSISTANT EDITOR
Ellie Scott

ASSOCIATE MARKETING DIRECTOR
Louise Breinholt

SENIOR MARKETING EXECUTIVE
Kate Parrett

EDITORIAL MANAGER
Jodi Jensen

SENIOR PROJECT EDITOR
Sara Shlaer

DEVELOPMENT EDITOR
Box Twelve Communications, Inc.

TECHNICAL EDITOR
Paul Brunt

COPY EDITOR
Marylouise Wiack

SENIOR PRODUCTION EDITOR
Debra Banninger

PROOFREADER
Nancy Carrasco

INDEXER
Robert Swanson

COVER DESIGNER
LeAndra Young

COVER IMAGE
© iStock / Mark Evans

ACKNOWLEDGMENTS

A BIG THANK YOU — goes to the Khronos WebGL Working Group, which created the WebGL specification and the browser vendors, including open source projects, which have implemented the specification. Without your work, this book would not exist. In addition, I want to thank all developers in the WebGL community who have contributed to WebGL as a technology with their discussions, experiments, and examples.

I want to thank the people at Wiley, especially Jeff Riley, Sara Shlaer, Ellie Scott, and Chris Webb. You have all done a great job supporting me.

I want to say thank you to all my great colleagues at ST-Ericsson — you make ST-Ericsson a great place to work. I especially want to thank Kent Olsson, Christian Bejram, Marco Cornero, Göran Roth, and Patrik Åberg for believing in me, supporting me, and coaching me.

I want to thank my family for their support and understanding while writing this book. Most of all, I want to say thanks to my wonderful wife Jessica, whom I love very much. Thanks for everything, Jessica!

CONTENTS

INTRODUCTION

TODAY USERS SPEND A MAJORITY OF THEIR TIME WITH THEIR COMPUTERS, tablets, and smartphones surfing with web browsers. They are writing documents, e-mailing, chatting, reading news, watching videos, listening to music, playing games, or buying things. The web browser has become the most important application on most devices. As a result, the innovation and development of web technologies are evolving very fast. Web browsers are getting faster, more stable, more secure, and more capable of handling new technologies every day.

This means very exciting times for web developers. If you want to develop an application that you want to distribute to as many users as possible, the web is definitely the platform to choose. But even if web applications have evolved tremendously in recent years, one disadvantage of web applications compared to native applications has been that it has not been possible to create real-time 3D graphics that games and other applications might need. This has changed completely with WebGL.

WebGL lets you hardware-accelerate your 2D and 3D graphics in a way that has not been possible on the web before now. You can get all the advantages that building a web application in HTML, CSS and JavaScript provides and still take advantage of the powerful graphics processor unit (GPU) that devices have. This book will start with the basics and teach you everything you need to know to start developing 2D or 3D graphics applications based on WebGL.

WHO THIS BOOK IS FOR

This book is primarily for web developers who have a basic understanding of HTML, CSS and JavaScript and want to get a deeper understanding of how to use WebGL to create hardware-accelerated 2D or 3D graphics for their web applications or web pages.

The second group of developers who can benefit from this book is composed of those who have some previous experience from a desktop 3D API (such as OpenGL or Direct3D) and want to make use of this knowledge on the web by using WebGL.

The book is also useful for students who study a 3D graphics course and want to use WebGL as an easy way to prototype and test things. Since you need only a web browser that supports WebGL and a text editor to get started to write your applications, the threshold to get started is much lower than with other 3D graphics APIs (such as desktop OpenGL or Direct3D), where you often need to set up a complete toolchain.

WHAT THIS BOOK COVERS

This book will teach you how to develop web applications based on WebGL. Even though the WebGL API can be used to hardware-accelerate both 2D graphics and 3D graphics, it is primarily an API used to create 3D graphics. Some books about other 3D graphics APIs describe only the API; they do not

really teach you much about 3D graphics or how you use the API. This book does not require any previous knowledge about 3D graphics. Instead you learn about 3D graphics and using the WebGL API to build applications.

In addition, this book contains some fundamental linear algebra that is good to understand to get a deeper understanding of 3D graphics and WebGL. By having the relevant linear algebra available in the same book, you can concentrate on the parts of linear algebra that are important for 3D graphics. You do not need to dive into other generic linear algebra books consisting of hundreds of pages and wherein topics often are described in a very general and abstract way. If you're the type of person who just wants to get started writing code, you don't have to read all the linear algebra sections (which is mainly part of Chapter 1). You can skim them, and when you later notice that there is something related to linear algebra that you want to understand, you can go back and read them more thoroughly.

You should note that this book does not include all available methods of the WebGL API. For a complete reference of the API, you should have a look at the latest WebGL specification from Khronos at `www.khronos.org/registry/webgl/specs/latest/`

HOW THIS BOOK IS STRUCTURED

This book is organized in a logical order whereby most readers will benefit from reading the chapters in the order they appear. However, since all readers have different backgrounds, feel free to choose an order that suits you. The following overview might help you in this planning.

Chapter 1 is a theoretical chapter that gives you some background of WebGL and compares WebGL to some other graphics technologies. It introduces you to the WebGL graphics pipeline and also contains an overview of graphics hardware. Further, the chapter contains some linear algebra that is important for 3D graphics and gives you a deeper understanding of 3D graphics and WebGL. If you feel that you have enough knowledge about the topics presented in this chapter, feel free to skim the content and continue with Chapter 2.

Chapter 2 provides a lot of practical knowledge. You create your first complete WebGL application and learn how to write a very basic vertex shader and a fragment shader. To make your journey towards learning WebGL as smooth as possible, this chapter introduces some useful development and debugging tools. You also learn what to think of when you troubleshoot your WebGL application.

Chapter 3 teaches you the details about which different options you have for drawing in WebGL. You learn about the different WebGL methods and primitives you can use to handle drawing.

Chapter 4 introduces you to three small JavaScript libraries that are useful to handle vector and matrix mathematics in your WebGL application. You also learn how to use transformations to position and orient the objects in your 3D world. This is a very important chapter, and if you have no previous experience of 3D graphics, you should be prepared to spend some extra time on it.

Chapter 5 walks through how you perform texturing in WebGL. Texturing is an important step towards making your 3D objects look more realistic. In addition, this chapter describes how to make

your applications more robust by handling what is referred to as *lost context* in WebGL. Make sure you read this part before you start to build real-world applications that have high requirements on robustness.

Chapter 6 teaches you how you use animation to create motion in your WebGL scenes. You will also learn the details of how event-handling in JavaScript works and how you can use key events and mouse events to affect your WebGL scene.

Chapter 7 teaches you a lot about the exciting topic of lighting in WebGL. You learn about different lighting models and how to write vertex shaders and fragment shaders for these. Lighting is a very important ingredient in making your WebGL scenes look more realistic.

Chapter 8 contains useful guidelines, tips, and tricks that make sure your WebGL application gets the best possible performance. You also learn how to analyze performance problems in a WebGL application, how to find the bottleneck, and how to try to eliminate it. This chapter also features a description of how WebGL works under the hood. You learn about the key software components and the key hardware components. An example from the mobile industry is used.

WHAT YOU NEED TO USE THIS BOOK

To run the examples in this book, you need a web browser that supports WebGL. If you don't have a browser that supports WebGL, you can download one for free. For a current list of which web browsers that support WebGL, visit `www.khronos.org/webgl/wiki/`. At the time of this writing, the following browsers support WebGL:

- Apple Safari
- Google Chrome
- Mozilla Firefox
- Opera

To write your own WebGL applications, simply use your favorite text editor. The book also describes some other tools that are helpful for debugging and trouble shooting of WebGL applications. These are, for example, Chrome Developer Tools, Firebug, and WebGL Inspector. These tools are all open source and can be downloaded for free.

CONVENTIONS

To help you get the most from the text and keep track of what's happening, we've used a number of conventions throughout the book.

The pencil icon indicates notes, tips, hints, tricks, or and asides to the current discussion.

As for styles in the text:

- We *highlight* new terms and important words when we introduce them.
- We show keyboard strokes like this: Ctrl+A.
- We show file names, URLs, and code within the text like so: `persistence.properties`.
- We present code in two different ways:

```
We use a monofont type with no highlighting for most code examples.
```

```
We use bold to emphasize code that's particularly important in the present context.
```

SOURCE CODE

As you work through the examples in this book, you may choose either to type in all the code manually or to use the source code files that accompany the book. All of the source code corresponding to the code listings in this book is available for download at `www.wrox.com`. You will find the code listings from the source code are accompanied by a download icon and listing number so you know it's available for download and can easily locate it in the download file. Once at the site, simply locate the book's title (either by using the Search box or by using one of the title lists) and click the Download Code link on the book's detail page to obtain all the source code for the book.

Because many books have similar titles, you may find it easiest to search by ISBN; this book's ISBN is 978-1-119-96886-3.

Once you download the code, just decompress it with your favorite compression tool. Alternately, you can go to the main Wrox code download page at `www.wrox.com/dynamic/books/download.aspx` to see the code available for this book and all other Wrox books.

ERRATA

We make every effort to ensure that there are no errors in the text or in the code. However, no one is perfect, and mistakes do occur. If you find an error in one of our books, like a spelling mistake or faulty piece of code, we would be very grateful for your feedback. By sending in errata you may save another reader hours of frustration and at the same time you will be helping us provide even higher quality information.

To find the errata page for this book, go to `www.wrox.com` and locate the title using the Search box or one of the title lists. Then, on the book details page, click the Book Errata link. On this page you can view all errata that has been submitted for this book and posted by Wrox editors.

A complete book list including links to each book's errata is also available at `www.wrox.com/misc-pages/booklist.shtml`.

If you don't spot "your" error on the Book Errata page, go to `www.wrox.com/contact/techsupport.shtml` and complete the form there to send us the error you have found. We'll check the information and, if appropriate, post a message to the book's errata page and fix the problem in subsequent editions of the book.

P2P.WROX.COM

For author and peer discussion, join the P2P forums at `p2p.wrox.com`. The forums are a web-based system for you to post messages relating to Wrox books and related technologies and interact with other readers and technology users. The forums offer a subscription feature to e-mail you topics of interest of your choosing when new posts are made to the forums. Wrox authors, editors, other industry experts, and your fellow readers are present on these forums.

At `http://p2p.wrox.com` you will find a number of different forums that will help you not only as you read this book, but also as you develop your own applications. To join the forums, just follow these steps:

1. Go to `p2p.wrox.com` and click the Register link.
2. Read the terms of use and click Agree.
3. Complete the required information to join as well as any optional information you wish to provide and click Submit.
4. You will receive an e-mail with information describing how to verify your account and complete the joining process.

You can read messages in the forums without joining P2P, but in order to post your own messages, you must join.

Once you join, you can post new messages and respond to messages other users post. You can read messages at any time on the web. If you would like to have new messages from a particular forum e-mailed to you, click the Subscribe to this Forum icon by the forum name in the forum listing.

For more information about how to use the Wrox P2P, be sure to read the P2P FAQs for answers to questions about how the forum software works as well as many common questions specific to P2P and Wrox books. To read the FAQs, click the FAQ link on any P2P page.

1

Introducing WebGL

WHAT'S IN THIS CHAPTER?

- The basics of WebGL
- Why 3D graphics in the browser offer great possibilities
- How to work with an immediate-mode API
- The basics of graphics hardware
- The WebGL graphics pipeline
- How WebGL compares to other graphics technologies
- Some basic linear algebra needed for WebGL

In this chapter you will be introduced to WebGL and learn some important theory behind WebGL, as well as general theory behind 3D graphics. You will learn what WebGL is and get an understanding of the graphics pipeline that is used for WebGL.

You will also be given an overview of some other related graphics standards that will give you a broader understanding of WebGL and how it compares to these other technologies.

The chapter concludes with a review of some basic linear algebra that is useful to understand if you really want to master WebGL on a deeper level.

THE BASICS OF WEBGL

WebGL is an application programming interface (API) for advanced 3D graphics on the web. It is based on OpenGL ES 2.0, and provides similar rendering functionality, but in an HTML and JavaScript context. The rendering surface that is used for WebGL is the HTML5 canvas element, which was originally introduced by Apple in the WebKit open-source browser engine.

The reason for introducing the HTML5 canvas was to be able to render 2D graphics in applications such as Dashboard widgets and in the Safari browser on the Apple Mac OS X operating system.

Based on the canvas element, Vladimir Vukićević at Mozilla started experimenting with 3D graphics for the canvas element. He called the initial prototype Canvas 3D. In 2009 the Khronos Group started a new WebGL working group, which now consists of several major browser vendors, including Apple, Google, Mozilla, and Opera. The Khronos Group is a non-profit industry consortium that creates open standards and royalty-free APIs. It was founded in January 2000 and is behind a number of other APIs and technologies such as OpenGL ES for 3D graphics for embedded devices, OpenCL for parallel programming, OpenVG for low-level acceleration of vector graphics, and OpenMAX for accelerated multimedia components. Since 2006 the Khronos Group has also controlled and promoted OpenGL, which is a 3D graphics API for desktops.

The final WebGL 1.0 specification was frozen in March 2011, and WebGL support is implemented in several browsers, including Google Chrome, Mozilla Firefox, and (at the time of this writing) in the development releases of Safari and Opera.

For the latest information about which versions of different browsers support WebGL, visit `www.khronos.org/webgl/`.

SO WHY IS WEBGL SO GREAT?

In the early days of the web, the content consisted of static documents of text. The web browser was used to retrieve and display these static pages. Over time the web technology has evolved tremendously, and today many websites are actually full-featured applications. They support two-way communication between the server and the client, users can register and log in, and web applications now feature a rich user interface that includes graphics as well as audio and video.

The fast evolution of web applications has led to them becoming an attractive alternative to native applications. Some advantages of web applications include the following:

- They are cheap and easy to distribute to a lot of users. A compatible web browser is all that the user needs.
- Maintenance is easy. When you find a bug in your application or when you have added some nice new features that you want your users to benefit from, you only have to upgrade the application on the web server and your users are able to benefit from your new application immediately.
- At least in theory, it is easier to have cross-platform support (i.e., to support several operating systems such as Windows, Mac OS, Linux, and so on) since the application is executing inside the web browser.

However, to be honest, web applications also have had (and still have) some limitations compared to native applications. One limitation has been that the user interface of web applications has not been as rich as for their native application counterparts. This changed a lot with the introduction of the HTML5 canvas tag, which made it possible to create really advanced 2D graphics for your web