

# OpenGL® ES™ 3.0

## Programming Guide

*Second Edition*



**Dan Ginsburg** ■ **Budirijanto Purnomo**

With Earlier Contributions from **Dave Shreiner** and **Aaftab Munshi**

Foreword by **Neil Trevett**, President, Khronos Group

---

## **Praise for *OpenGL® ES™ 3.0 Programming Guide, Second Edition***

“As a graphics technologist and intense OpenGL ES developer, I can honestly say that if you buy only one book on OpenGL ES 3.0 programming, then this should be the book. Dan and Budirijanto have written a book clearly by programmers for programmers. It is simply required reading for anyone interested in OpenGL ES 3.0. It is informative, well organized, and comprehensive, but best of all practical. You will find yourself reaching for this book over and over again instead of the actual OpenGL ES specification during your programming sessions. I give it my highest recommendation.”

—Rick Tewell, Graphics Technology Architect, Freescale

“This book provides outstanding coverage of the latest version of OpenGL ES, with clear, comprehensive explanations and extensive examples. It belongs on the desk of anyone developing mobile applications.”

—Dave Astle, Graphics Tools Lead, Qualcomm Technologies, Inc., and Founder, GameDev.net

“The second edition of *OpenGL® ES™ 3.0 Programming Guide* provides a solid introduction to OpenGL ES 3.0 specifications, along with a wealth of practical information and examples to help any level of developer begin programming immediately. We’d recommend this guide as a primer on OpenGL ES 3.0 to any of the thousands of developers creating apps for the many mobile and embedded products using our PowerVR Rogue graphics.”

—Kristof Beets, Business Development, Imagination Technologies

“This is a solid OpenGL ES 3.0 reference book. It covers all aspects of the API and will help any developer get familiar with and understand the API, including specifically the new ES 3.0 functionality.”

—Jed Fisher, Managing Partner, 4D Pipeline

“This is a clear and thorough reference for OpenGL ES 3.0, and an excellent presentation of the concepts present in all modern OpenGL programming. This is the guide I’d want by my side when diving into embedded OpenGL.”

—Todd Furlong, President & Principal Engineer, Inv3rsion LLC

---

*This page intentionally left blank*

---

# OpenGL<sup>®</sup> ES<sup>™</sup> 3.0 Programming Guide

Second Edition

# OpenGL Series

from Addison-Wesley



Visit [informit.com/opengl](http://informit.com/opengl) for a complete list of available products.

The OpenGL graphics system is a software interface to graphics hardware. (“GL” stands for “Graphics Library.”) It allows you to create interactive programs that produce color images of moving, three-dimensional objects. With OpenGL, you can control computer-graphics technology to produce realistic pictures, or ones that depart from reality in imaginative ways.

The **OpenGL Series** from Addison-Wesley Professional comprises tutorial and reference books that help programmers gain a practical understanding of OpenGL standards, along with the insight needed to unlock OpenGL’s full potential.



Make sure to connect with us!  
[informit.com/socialconnect](http://informit.com/socialconnect)

**informit.com**  
the trusted technology learning source

◆◆Addison-Wesley

**Safari**  
Books Online

---

# OpenGL<sup>®</sup> ES<sup>™</sup> 3.0 Programming Guide

Second Edition

*Dan Ginsburg*  
*Budirijanto Purnomo*

With Earlier Contributions From  
*Dave Shreiner*  
*Aaftab Munshi*

◆ 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.

Front cover image is from Snapdragon Game Studio's *Fortress: Fire OpenGL® ES™ 3.0 demo*, courtesy of Qualcomm Technologies Inc.

OpenGL® is a registered trademark and the OpenGL® ES™ logo is a trademark of Silicon Graphics Inc. used by permission by Khronos.

The OpenGL® ES™ shading language built-in functions described in Appendix B are copyrighted by Khronos and are reprinted with permission from the *OpenGL® ES™ 3.00.4 Shading Language Specification*.

The OpenGL® ES™ 3.0 Reference Card is copyrighted by Khronos and reprinted with permission.

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.

For information about buying this title in bulk quantities, or for special sales opportunities (which may include electronic versions; custom cover designs; and content particular to your business, training goals, marketing focus, or branding interests), please contact our corporate sales department at [corpsales@pearsoned.com](mailto:corpsales@pearsoned.com) or (800) 382-3419.

For government sales inquiries, please contact [governmentsales@pearsoned.com](mailto:governmentsales@pearsoned.com).

For questions about sales outside the U.S., please contact [international@pearsoned.com](mailto:international@pearsoned.com).

Visit us on the Web: [informit.com/aw](http://informit.com/aw)

*Library of Congress Cataloging-in-Publication Data*  
Ginsburg, Dan.

OpenGL ES 3.0 programming guide / Dan Ginsburg, Budirijanto Purnomo ; with earlier contributions from Dave Shreiner, Aaftab Munshi.—Second edition.

pages cm  
Revised edition of: The OpenGL ES 2.0 programming guide / Aaftab Munshi, Dan Ginsburg, Dave Shreiner. 2009.

Includes bibliographical references and index.

ISBN 978-0-321-93388-1 (paperback : alk. paper)

1. OpenGL. 2. Computer graphics—Specifications. 3. Application program interfaces (Computer software) 4. Computer programming. I. Purnomo, Budirijanto. II. Shreiner, Dave. III. Munshi, Aaftab. IV. Title.

T385.G5426 2014

006.6'6—dc23

2013049233

Copyright © 2014 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: 978-0-321-93388-1

ISBN-10: 0-321-93388-5

Text printed in the United States on recycled paper at RR Donnelley in Crawfordsville, Indiana.

First printing, March 2014

**Editor-in-Chief**

Mark L. Taub

**Executive Editor**

Laura Lewin

**Development Editor**

Sheri Cain

**Managing Editor**

John Fuller

**Project Editor**

Elizabeth Ryan

**Copy Editor**

Jill Hobbs

**Indexer**

Infodex Indexing Services,  
Inc.

**Proofreader**

Linda Begley

**Technical Reviewers**

Emmanuel Agu

Peter Lohrmann

Maurice Ribble

**Editorial Assistant**

Olivia Basegio

**Cover Designer**

Chuti Prasertsith

**Compositor**

diacriTech

---

# Contents

<b>List of Figures</b> .....	<b>xvii</b>
<b>List of Examples</b> .....	<b>xxi</b>
<b>List of Tables</b> .....	<b>xxv</b>
<b>Foreword</b> .....	<b>xxix</b>
<b>Preface</b> .....	<b>xxxii</b>
Intended Audience .....	xxxii
Organization of This Book.....	xxxii
Example Code and Shaders .....	xxxvi
Errata.....	xxxvi
<b>Acknowledgments</b> .....	<b>xxxvii</b>
<b>About the Authors</b> .....	<b>xxxix</b>
<b>1. Introduction to OpenGL ES 3.0</b> .....	<b>1</b>
OpenGL ES 3.0.....	3
Vertex Shader.....	4
Primitive Assembly.....	7
Rasterization .....	7
Fragment Shader.....	8
Per-Fragment Operations .....	9
What's New in OpenGL ES 3.0.....	11
Texturing .....	11
Shaders.....	13



---

Geometry.....	15
Buffer Objects.....	16
Framebuffer.....	17
OpenGL ES 3.0 and Backward Compatibility.....	17
EGL.....	19
Programming with OpenGL ES 3.0.....	20
Libraries and Include Files.....	20
EGL Command Syntax.....	20
OpenGL ES Command Syntax.....	21
Error Handling.....	22
Basic State Management.....	23
Further Reading.....	25
<b>2. Hello Triangle: An OpenGL ES 3.0 Example.....</b>	<b>27</b>
Code Framework.....	28
Where to Download the Examples.....	28
Hello Triangle Example.....	29
Using the OpenGL ES 3.0 Framework.....	34
Creating a Simple Vertex and Fragment Shader.....	35
Compiling and Loading the Shaders.....	36
Creating a Program Object and Linking the Shaders.....	38
Setting the Viewport and Clearing the Color Buffer.....	39
Loading the Geometry and Drawing a Primitive.....	40
Displaying the Back Buffer.....	41
Summary.....	42
<b>3. An Introduction to EGL.....</b>	<b>43</b>
Communicating with the Windowing System.....	44
Checking for Errors.....	45
Initializing EGL.....	46
Determining the Available Surface Configurations.....	46
Querying EGLConfig Attributes.....	48
Letting EGL Choose the Configuration.....	51
Creating an On-Screen Rendering Area: The EGL Window.....	53
Creating an Off-Screen Rendering Area: EGL Pbuffers.....	56
Creating a Rendering Context.....	60

---

Making an EGLContext Current .....	62
Putting All Our EGL Knowledge Together.....	63
Synchronizing Rendering .....	66
Summary.....	67
<b>4. Shaders and Programs .....</b>	<b>69</b>
Shaders and Programs.....	69
Creating and Compiling a Shader.....	70
Creating and Linking a Program.....	74
Uniforms and Attributes.....	80
Getting and Setting Uniforms.....	81
Uniform Buffer Objects .....	87
Getting and Setting Attributes .....	92
Shader Compiler.....	93
Program Binaries.....	94
Summary.....	95
<b>5. OpenGL ES Shading Language.....</b>	<b>97</b>
OpenGL ES Shading Language Basics.....	98
Shader Version Specification .....	98
Variables and Variable Types .....	99
Variable Constructors .....	100
Vector and Matrix Components.....	101
Constants.....	102
Structures .....	103
Arrays .....	104
Operators .....	104
Functions .....	106
Built-In Functions.....	107
Control Flow Statements .....	107
Uniforms.....	108
Uniform Blocks.....	109
Vertex and Fragment Shader Inputs/Outputs .....	111
Interpolation Qualifiers .....	114
Preprocessor and Directives.....	115
Uniform and Interpolator Packing.....	117

Precision Qualifiers .....	119
Invariance .....	121
Summary .....	123
<b>6. Vertex Attributes, Vertex Arrays, and Buffer Objects.....</b>	<b>125</b>
Specifying Vertex Attribute Data .....	126
Constant Vertex Attribute .....	126
Vertex Arrays .....	126
Declaring Vertex Attribute Variables in a Vertex Shader.....	135
Binding Vertex Attributes to Attribute Variables	
in a Vertex Shader .....	137
Vertex Buffer Objects .....	140
Vertex Array Objects .....	150
Mapping Buffer Objects.....	154
Flushing a Mapped Buffer .....	158
Copying Buffer Objects .....	159
Summary .....	160
<b>7. Primitive Assembly and Rasterization.....</b>	<b>161</b>
Primitives .....	161
Triangles .....	162
Lines .....	163
Point Sprites.....	164
Drawing Primitives .....	165
Primitive Restart .....	168
Provoking Vertex .....	169
Geometry Instancing.....	169
Performance Tips.....	172
Primitive Assembly .....	174
Coordinate Systems.....	175
Perspective Division .....	178
Viewport Transformation .....	178
Rasterization .....	179
Culling.....	180
Polygon Offset.....	181
Occlusion Queries.....	183
Summary.....	185

---

<b>8. Vertex Shaders .....</b>	<b>187</b>
Vertex Shader Overview .....	188
Vertex Shader Built-In Variables.....	189
Precision Qualifiers.....	192
Number of Uniforms Limitations in a Vertex Shader.....	193
Vertex Shader Examples .....	196
Matrix Transformations.....	196
Lighting in a Vertex Shader.....	199
Generating Texture Coordinates .....	205
Vertex Skinning .....	207
Transform Feedback.....	211
Vertex Textures .....	214
OpenGL ES 1.1 Vertex Pipeline as an ES 3.0 Vertex Shader.....	215
Summary.....	223
<b>9. Texturing.....</b>	<b>225</b>
Texturing Basics .....	226
2D Textures.....	226
Cubemap Textures.....	228
3D Textures.....	229
2D Texture Arrays.....	230
Texture Objects and Loading Textures.....	230
Texture Filtering and Mipmapping .....	237
Automatic Mipmap Generation .....	242
Texture Coordinate Wrapping.....	243
Texture Swizzles.....	244
Texture Level of Detail .....	245
Depth Texture Compare (Percentage Closest Filtering).....	245
Texture Formats.....	246
Using Textures in a Shader .....	255
Example of Using a Cubemap Texture.....	258
Loading 3D Textures and 2D Texture Arrays .....	260
Compressed Textures.....	262
Texture Subimage Specification.....	266
Copying Texture Data from the Color Buffer.....	269

Sampler Objects .....	273
Immutable Textures .....	276
Pixel Unpack Buffer Objects .....	277
Summary .....	278
<b>10. Fragment Shaders .....</b>	<b>279</b>
Fixed-Function Fragment Shaders .....	280
Fragment Shader Overview .....	282
Built-In Special Variables .....	283
Built-In Constants .....	284
Precision Qualifiers .....	285
Implementing Fixed-Function Techniques Using Shaders .....	286
Multitexturing .....	286
Fog .....	288
Alpha Test (Using Discard) .....	291
User Clip Planes .....	293
Summary .....	295
<b>11. Fragment Operations .....</b>	<b>297</b>
Buffers .....	298
Requesting Additional Buffers .....	299
Clearing Buffers .....	299
Using Masks to Control Writing to Framebuffers .....	301
Fragment Tests and Operations .....	303
Using the Scissor Test .....	304
Stencil Buffer Testing .....	305
Blending .....	311
Dithering .....	314
Multisampled Anti-Aliasing .....	314
Centroid Sampling .....	316
Reading and Writing Pixels to the Framebuffer .....	316
Pixel Pack Buffer Objects .....	320
Multiple Render Targets .....	320
Summary .....	324

---

<b>12. Framebuffer Objects .....</b>	<b>325</b>
Why Framebuffer Objects? .....	325
Framebuffer and Renderbuffer Objects .....	327
Choosing a Renderbuffer Versus a Texture as a Framebuffer Attachment.....	328
Framebuffer Objects Versus EGL Surfaces .....	329
Creating Framebuffer and Renderbuffer Objects .....	329
Using Renderbuffer Objects.....	330
Multisample Renderbuffers .....	333
Renderbuffer Formats .....	333
Using Framebuffer Objects .....	335
Attaching a Renderbuffer as a Framebuffer Attachment .....	337
Attaching a 2D Texture as a Framebuffer Attachment.....	338
Attaching an Image of a 3D Texture as a Framebuffer Attachment .....	339
Checking for Framebuffer Completeness.....	341
Framebuffer Blits.....	342
Framebuffer Invalidation.....	344
Deleting Framebuffer and Renderbuffer Objects.....	346
Deleting Renderbuffer Objects That Are Used as Framebuffer Attachments.....	347
Reading Pixels and Framebuffer Objects.....	347
Examples.....	348
Performance Tips and Tricks.....	354
Summary.....	355
<b>13. Sync Objects and Fences .....</b>	<b>357</b>
Flush and Finish .....	357
Why Use a Sync Object?.....	358
Creating and Deleting a Sync Object .....	358
Waiting for and Signaling a Sync Object .....	359
Example .....	360
Summary.....	361

---

<b>14. Advanced Programming with OpenGL ES 3.0 .....</b>	<b>363</b>
Per-Fragment Lighting.....	363
Lighting with a Normal Map .....	364
Lighting Shaders.....	366
Lighting Equations .....	369
Environment Mapping .....	370
Particle System with Point Sprites.....	374
Particle System Setup.....	374
Particle System Vertex Shader .....	375
Particle System Fragment Shader .....	377
Particle System Using Transform Feedback.....	380
Particle System Rendering Algorithm .....	381
Particle Emission with Transform Feedback .....	381
Rendering the Particles.....	385
Image Postprocessing.....	387
Render-to-Texture Setup.....	387
Blur Fragment Shader .....	388
Projective Texturing.....	390
Projective Texturing Basics.....	391
Matrices for Projective Texturing .....	392
Projective Spotlight Shaders.....	394
Noise Using a 3D Texture .....	397
Generating Noise.....	397
Using Noise.....	402
Procedural Texturing .....	404
A Procedural Texture Example .....	405
Anti-Aliasing of Procedural Textures.....	407
Further Reading on Procedural Textures .....	410
Rendering Terrain with Vertex Texture Fetch .....	410
Generating a Square Terrain Grid .....	411
Computing Vertex Normal and Fetching Height Value in Vertex Shader.....	412
Further Reading on Large Terrain Rendering.....	413
Shadows Using a Depth Texture.....	414
Rendering from the Light Position Into a Depth Texture .....	415
Rendering from the Eye Position with the Depth Texture .....	418
Summary.....	420

<b>15. State Queries .....</b>	<b>421</b>
OpenGL ES 3.0 Implementation String Queries .....	421
Querying Implementation-Dependent Limits .....	423
Querying OpenGL ES State.....	429
Hints .....	435
Entity Name Queries.....	436
Nonprogrammable Operations Control and Queries .....	436
Shader and Program State Queries .....	438
Vertex Attribute Queries .....	440
Texture State Queries .....	441
Sampler Queries.....	442
Asynchronous Object Queries.....	442
Sync Object Queries.....	443
Vertex Buffer Queries.....	444
Renderbuffer and Framebuffer State Queries .....	445
Summary.....	446
<b>16. OpenGL ES Platforms.....</b>	<b>447</b>
Building for Microsoft Windows with Visual Studio .....	447
Building for Ubuntu Linux.....	449
Building for Android 4.3+ NDK (C++).....	450
Prerequisites.....	451
Building the Example Code with Android NDK.....	452
Building for Android 4.3+ SDK (Java).....	452
Building for iOS 7 .....	453
Prerequisites.....	453
Building the Example Code with Xcode 5.....	453
Summary.....	455
<b>A. GL_HALF_FLOAT .....</b>	<b>457</b>
16-Bit Floating-Point Number .....	458
Converting a Float to a Half-Float.....	459
<b>B. Built-In Functions .....</b>	<b>463</b>
Angle and Trigonometry Functions .....	465
Exponential Functions .....	466
Common Functions.....	467



---

Floating-Point Pack and Unpack Functions .....	471
Geometric Functions .....	472
Matrix Functions .....	474
Vector Relational Functions .....	475
Texture Lookup Functions.....	476
Fragment Processing Functions.....	483
<b>C. ES Framework API.....</b>	<b>485</b>
Framework Core Functions .....	485
Transformation Functions .....	490
<b>Index.....</b>	<b>495</b>

---

## List of Figures

<b>Figure 1-1</b>	OpenGL ES 3.0 Graphics Pipeline .....	4
<b>Figure 1-2</b>	OpenGL ES 3.0 Vertex Shader.....	5
<b>Figure 1-3</b>	OpenGL ES 3.0 Rasterization Stage.....	7
<b>Figure 1-4</b>	OpenGL ES 3.0 Fragment Shader.....	8
<b>Figure 1-5</b>	OpenGL ES 3.0 Per-Fragment Operations .....	10
<b>Figure 2-1</b>	Hello Triangle Example.....	33
<b>Figure 5-1</b>	Z Fighting Artifacts Due to Not Using Invariance.....	121
<b>Figure 5-2</b>	Z Fighting Avoided Using Invariance .....	122
<b>Figure 6-1</b>	Triangle with a Constant Color Vertex and Per-Vertex Position Attributes.....	125
<b>Figure 6-2</b>	Position, Normal, and Two Texture Coordinates Stored as an Array .....	128
<b>Figure 6-3</b>	Selecting Constant or Vertex Array Vertex Attribute .....	133
<b>Figure 6-4</b>	Specifying and Binding Vertex Attributes for Drawing One or More Primitives.....	138
<b>Figure 7-1</b>	Triangle Primitive Types .....	162
<b>Figure 7-2</b>	Line Primitive Types .....	163
<b>Figure 7-3</b>	gl_PointCoord Values .....	165
<b>Figure 7-4</b>	Cube .....	167
<b>Figure 7-5</b>	Connecting Triangle Strips .....	173
<b>Figure 7-6</b>	OpenGL ES Primitive Assembly Stage.....	175
<b>Figure 7-7</b>	Coordinate Systems .....	175
<b>Figure 7-8</b>	Viewing Volume.....	176
<b>Figure 7-9</b>	OpenGL ES Rasterization Stage.....	179
<b>Figure 7-10</b>	Clockwise and Counterclockwise Triangles.....	180
<b>Figure 7-11</b>	Polygon Offset.....	182

<b>Figure 8-1</b>	OpenGL ES 3.0 Programmable Pipeline .....	188
<b>Figure 8-2</b>	OpenGL ES 3.0 Vertex Shader.....	189
<b>Figure 8-3</b>	Geometric Factors in Computing Lighting Equation for a Directional Light.....	199
<b>Figure 8-4</b>	Geometric Factors in Computing Lighting Equation for a Spotlight.....	202
<b>Figure 9-1</b>	2D Texture Coordinates.....	227
<b>Figure 9-2</b>	3D Texture Coordinate for Cubemap .....	228
<b>Figure 9-3</b>	3D Texture.....	229
<b>Figure 9-4</b>	MipMap2D: Nearest Versus Trilinear Filtering .....	241
<b>Figure 9-5</b>	GL_REPEAT, GL_CLAMP_TO_EDGE, and GL_MIRRORED_REPEAT Modes .....	243
<b>Figure 10-1</b>	OpenGL ES 3.0 Programmable Pipeline .....	280
<b>Figure 10-2</b>	OpenGL ES 3.0 Fragment Shader.....	283
<b>Figure 10-3</b>	Multitextured Quad .....	287
<b>Figure 10-4</b>	Linear Fog on Torus in PVRShaman .....	289
<b>Figure 10-5</b>	Alpha Test Using Discard .....	292
<b>Figure 10-6</b>	User Clip Plane Example.....	294
<b>Figure 11-1</b>	The Post-Shader Fragment Pipeline .....	297
<b>Figure 12-1</b>	Framebuffer Objects, Renderbuffer Objects, and Textures.....	328
<b>Figure 12-2</b>	Render to Color Texture.....	350
<b>Figure 12-3</b>	Render to Depth Texture.....	353
<b>Figure 14-1</b>	Per-Fragment Lighting Example .....	364
<b>Figure 14-2</b>	Environment Mapping Example .....	370
<b>Figure 14-3</b>	Particle System Sample .....	374
<b>Figure 14-4</b>	Particle System with Transform Feedback .....	380
<b>Figure 14-5</b>	Image Postprocessing Example.....	387
<b>Figure 14-6</b>	Light Bloom Effect .....	389
<b>Figure 14-7</b>	Light Bloom Stages.....	390
<b>Figure 14-8</b>	Projective Spotlight Example.....	391
<b>Figure 14-9</b>	2D Texture Projected onto Object .....	392
<b>Figure 14-10</b>	Fog Distorted by 3D Noise Texture .....	397
<b>Figure 14-11</b>	2D Slice of Gradient Noise.....	402
<b>Figure 14-12</b>	Checkerboard Procedural Texture.....	407

---

<b>Figure 14-13</b>	Anti-Aliased Checkerboard Procedural Texture.....	409
<b>Figure 14-14</b>	Terrain Rendered with Vertex Texture Fetch .....	411
<b>Figure 14-15</b>	Shadow Rendering with a Depth Texture and 6 × 6 PCF .....	414
<b>Figure 16-1</b>	Building Samples with CMake GUI on Windows .....	448
<b>Figure 16-2</b>	VertexArrayObjects Sample in Xcode Running on iOS 7 Simulator.....	454
<b>Figure A-1</b>	A 16-Bit Floating-Point Number .....	458

---

*This page intentionally left blank*

---

## List of Examples

<b>Example 1-1</b>	A Vertex Shader Example.....	6
<b>Example 1-2</b>	A Fragment Shader Example.....	9
<b>Example 2-1</b>	Hello_Triangle.c Example .....	29
<b>Example 3-1</b>	Initializing EGL.....	44
<b>Example 3-2</b>	Specifying EGL Attributes.....	51
<b>Example 3-3</b>	Querying EGL Surface Configurations.....	52
<b>Example 3-4</b>	Creating an EGL Window Surface .....	55
<b>Example 3-5</b>	Creating an EGL Pixel Buffer .....	59
<b>Example 3-6</b>	Creating an EGL Context.....	62
<b>Example 3-7</b>	A Complete Routine for Creating an EGL Window .....	64
<b>Example 3-8</b>	Creating a Window Using the <code>esUtil</code> Library.....	65
<b>Example 4-1</b>	Loading a Shader.....	73
<b>Example 4-2</b>	Create, Attach Shaders to, and Link a Program.....	79
<b>Example 4-3</b>	Querying for Active Uniforms .....	86
<b>Example 5-1</b>	Sample Vertex Shader .....	112
<b>Example 5-2</b>	Vertex and Fragment Shaders with Matching Output/Input Declarations .....	113
<b>Example 6-1</b>	Array of Structures .....	129
<b>Example 6-2</b>	Structure of Arrays .....	130
<b>Example 6-3</b>	Using Constant and Vertex Array Attributes.....	133
<b>Example 6-4</b>	Creating and Binding Vertex Buffer Objects .....	141
<b>Example 6-5</b>	Drawing with and without Vertex Buffer Objects.....	146
<b>Example 6-6</b>	Drawing with a Buffer Object per Attribute .....	149
<b>Example 6-7</b>	Drawing with a Vertex Array Object.....	152
<b>Example 6-8</b>	Mapping a Buffer Object for Writing.....	157
<b>Example 8-1</b>	Vertex Shader with Matrix Transform for the Position .....	196

---

<b>Example 8-2</b>	Directional Light.....	200
<b>Example 8-3</b>	Spotlight.....	203
<b>Example 8-4</b>	Sphere Map Texture Coordinate Generation.....	206
<b>Example 8-5</b>	Cubemap Texture Coordinate Generation .....	206
<b>Example 8-6</b>	Vertex Skinning Shader with No Check of Whether Matrix Weight = 0.....	208
<b>Example 8-7</b>	Vertex Skinning Shader with Checks of Whether Matrix Weight = 0 .....	210
<b>Example 8-8</b>	Displacement Mapping Vertex Shader .....	214
<b>Example 8-9</b>	OpenGL ES 1.1 Fixed-Function Vertex Pipeline .....	216
<b>Example 9-1</b>	Generating a Texture Object, Binding It, and Loading Image Data.....	234
<b>Example 9-2</b>	Loading a 2D Mipmap Chain .....	238
<b>Example 9-3</b>	Vertex and Fragment Shaders for Performing 2D Texturing .....	255
<b>Example 9-4</b>	Loading a Cubemap Texture.....	258
<b>Example 9-5</b>	Vertex and Fragment Shader Pair for Cubemap Texturing .....	259
<b>Example 10-1</b>	Multitexture Fragment Shader.....	287
<b>Example 10-2</b>	Vertex Shader for Computing Distance to Eye.....	289
<b>Example 10-3</b>	Fragment Shader for Rendering Linear Fog.....	290
<b>Example 10-4</b>	Fragment Shader for Alpha Test Using Discard .....	292
<b>Example 10-5</b>	User Clip Plane Vertex Shader .....	294
<b>Example 10-6</b>	User Clip Plane Fragment Shader .....	295
<b>Example 11-1</b>	Setting up Multiple Render Targets .....	322
<b>Example 11-2</b>	Fragment Shader with Multiple Render Targets .....	324
<b>Example 12-1</b>	Copying Pixels Using Framebuffer Blits .....	343
<b>Example 12-2</b>	Render to Texture.....	348
<b>Example 12-3</b>	Render to Depth Texture.....	351
<b>Example 13-1</b>	Inserting a Fence Command and Waiting for Its Result in Transform Feedback Example .....	361
<b>Example 14-1</b>	Per-Fragment Lighting Vertex Shader.....	366
<b>Example 14-2</b>	Per-Fragment Lighting Fragment Shader.....	367
<b>Example 14-3</b>	Environment Mapping Vertex Shader .....	371
<b>Example 14-4</b>	Environment Mapping Fragment Shader .....	372
<b>Example 14-5</b>	Particle System Vertex Shader.....	375

---

<b>Example 14-6</b>	Update Function for Particle System Sample .....	376
<b>Example 14-7</b>	Particle System Fragment Shader.....	377
<b>Example 14-8</b>	Draw Function for Particle System Sample.....	378
<b>Example 14-9</b>	Particle Emission Vertex Shader.....	382
<b>Example 14-10</b>	Emit Particles with Transform Feedback.....	384
<b>Example 14-11</b>	Particle Rendering Vertex Shader.....	386
<b>Example 14-12</b>	Blur Fragment Shader .....	388
<b>Example 14-13</b>	Projective Texturing Vertex Shader.....	394
<b>Example 14-14</b>	Projective Texturing Fragment Shader.....	396
<b>Example 14-15</b>	Generating Gradient Vectors .....	398
<b>Example 14-16</b>	3D Noise.....	400
<b>Example 14-17</b>	Noise-Distorted Fog Fragment Shader .....	402
<b>Example 14-18</b>	Checker Vertex Shader.....	405
<b>Example 14-19</b>	Checker Fragment Shader with Conditional Checks .....	406
<b>Example 14-20</b>	Checker Fragment Shader without Conditional Checks .....	406
<b>Example 14-21</b>	Anti-Aliased Checker Fragment Shader .....	407
<b>Example 14-22</b>	Terrain Rendering Flat Grid Generation.....	411
<b>Example 14-23</b>	Terrain Rendering Vertex Shader .....	412
<b>Example 14-24</b>	Set up a MVP Matrix from the Light Position .....	415
<b>Example 14-25</b>	Create a Depth Texture and Attach It to a Framebuffer Object.....	416
<b>Example 14-26</b>	Rendering to Depth Texture Shaders .....	417
<b>Example 14-27</b>	Rendering from the Eye Position Shaders .....	418



- [Diary of a Sexpot pdf, azw \(kindle\)](#)
- [read online The Essential Job Interview Handbook pdf, azw \(kindle\), epub, doc, mobi](#)
- [Producing Great Sound for Film and Video \(3rd Edition\) \(DV Expert Series\) book](#)
- [read online Lutheran Theology \(Doing Theology\) pdf, azw \(kindle\), epub](#)
- [click Roger Zelazny's Visual Guide to Castle Amber online](#)
- [click Kvinnan som inte fanns pdf, azw \(kindle\)](#)
  
- <http://www.netc-bd.com/ebooks/The-Working-Garde-Manger.pdf>
- <http://chelseaprintandpublishing.com/?freebooks/Mr--Putin--Operative-in-the-Kremlin--Brookings-FOCUS-Book-.pdf>
- <http://weddingcellist.com/lib/Producing-Great-Sound-for-Film-and-Video--3rd-Edition---DV-Expert-Series-.pdf>
- <http://flog.co.id/library/Lutheran-Theology--Doing-Theology-.pdf>
- <http://dadhoc.com/lib/Roger-Zelazny-s-Visual-Guide-to-Castle-Amber.pdf>
- <http://musor.ruspb.info/?library/A-Theology-of-Luke-and-Acts--God-s-Promised-Program--Realized-for-All-Nations--Biblical-Theology-of-the-New-Te>