I Wish I Knew How To …

Program OpenGL Core

32-bit with Xojo

November 2016 Edition (2.0)

By Eugene Dakin

Copyright © 2016. All rights reserved. No part of this material, may be reproduced or transmitted in any form, by any means (electronic, photocopying, recording or otherwise) without prior written permission.

Trademarks

Xojo, and Real Studio are a registered trademark or trademark of Xojo Software, Inc.. Microsoft, Windows XP, Windows Vista, Visual Basic, and Visual Studio are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries. All other trademarks are the property of their respective owners.

The examples, products and other information contained herein are fictitious. There is no association with any real product, person, place or name is intended or inferred.

About the Author

Eugene Dakin is an author, Professional Chemist, oilfield consultant and programmer in Canada. When he is in-between courses and oilfield projects, he and his wife spend valuable time raising their wonderful children. He has been hobby programming on computer's for over 25 years. He has created over 170 technical and industrial simulation programs for Windows in Xojo.

Eugene can be contacted via:

E-mail: eugenedakin@hotmail.com

Website: http://www.scispec.ca

Dedication

This writing is dedicated to my wife and children. Without their support this would not have been written.

Special Thanks

This book has been written with the helpful tips and tricks of fellow programmers in the Xojo community. Special thanks go to Alwyn Bester, Will Shank, Jim McKay. Additional thanks go to Jeff Quan for writing the many OpenGL library declares for Xojo.

Table of Contents

Chapter 1 – Introduction 9
What is OpenGL 10
OpenGL Suffix 11
What OpenGL Isnt! 12
Coordinates: 2-Dimensions Introduction 12
Draw a Square (Quad) 13
Draw a Triangle 18
Draw a Line Strip 20
Draw Lines 22
Draw a Line Loop 24
Draw a Polygon 26
Draw a Quad Strip 28
Draw a Triangle Strip 30
Draw a Filled Circle 32
Draw a Circle Outline 34
Draw a Sphere 36

Chapter 2 – Fundamentals 38
Math 38
Draw a Multi-Coloured Triangle 42
Manual Rotation 44
Automated Rotation 46
Move Triangle (Arrow Buttons) 48
Move Triangle (OpenGl Style) 50
Rotate Angle Triangle 52
Scaling 56
Load Lists 59
Coordinate System Convention 64
Rules Of Thumb 65
2D Rules of Thumb 65
3D Rules of Thumb 65
General Rules of Thumb 65

Chapter 3 – Working with Primitives 67
Points 67
Point Size 69
AntiAlias – Smoothing 71
WireFrame vs Fill 78
CullFace 82

Chapter 4 – OpenGL Feedback 87
3-D Coordinate (OpenGL) System 87
Frames Per Second(fps) 88
Adding Frames Per Second Measurement 92
OpenGL Version, Vendor, Renderer 95
Full Screen Mode 96

Chapter 5 – Graph and Chart 100
Simple Graph 100
ViewPort 102
Graph Axis 106
Line Chart 110
2D Pie Chart 113

Chapter 6 – Data Drawing 120
Rotating Cube – Vertex's 120
Rotating Cube – Array 127
Rotating Cube – Vertex Array 134
Rotating Cube – Colour Block Attributes 142
Rotating Cube – Interleaved Array 149
Rotating Pyramid – Interleaved Array 157
Load Lists 165

Chapter 7 – Colours 169
Colour Pyramid 169
Smooth vs Flat 180

Chapter 8 - Beginning Lighting 188
Normal 189
Angle Of Incidence 190
Position Light 191
Ambient Light 199
Diffused Light 206
Activate Colour 213
Material Colour 221

Chapter 9 – Texturing 230
Static 2D Quad Texture 231
What is a RAW File? 237
Textured Spinning Cube 238
InitTexture Method 238
OpenGLSurface1.Render 241
Load External Texture 244

Repeat Texture 252
Multiple Textures 260

Chapter 10 - Load Blender Models 267
Other Models 272
Basic Obj Model Loader 274
Add Material Lighting 286
Code Optimization - Polygon 289

Chapter 11 – Fog Shader 294

Chapter 12 – Push and Pop Matrix 300
Rotate and Stationary 300
Different Models 305

Chapter 13 – Transparency 311
Quad Transparency 312
Texture Transparency 316

Chapter 14 – Materials 319
Manual Material Data 320
Model Load Optimization 332
Create Blender Cube Material 346
Add Model Material 350
Create Blender Cube Texture 360
Render Cube Material 365

Chapter 15 – Viewing (Camera) 379
Projection View 380
Runtime Projection View Changes 387
First Person View 393
Skybox (Environment Cube) 405

Chapter 16 –Texture Text 415
Draw Slow Text 415
Text Refresh 421
OpenGLControl Pushbutton 429
Fast Text Refresh 432
Transparent Text Background 435

Chapter 17 – Threads 441
Double Threads 441

Chapter 18 – Terrain/Floor 449

Distant Floor Mipmap 449
Height Map 454

Chapter 19 – Collision/Detection 461
Ray-Plane Intersection 461
Ray-Sphere Intersection 473

Chapter 20 – Xojo Blocks 483
Pieces 485
Game Initialization 489
Timers 491
Default Colours/Settings 502

Chapter 21 – Sound 503
Sound Effects 503
Text-To-Speech (Windows) 504
Add Voices (Windows 10) 505
Text-To-Speech (Multiplatform) 506

Chapter 22 – Mouse Movement 507
Bounce Ball – Translate 507
Multiple Balls – Class/Object 511
Multiple Balls – Colour Change 517
Mouse Moving Ball Objects 519

Chapter 23 - First Person Viewer 525
Create the Floor 526
Adding a Cube 533
Build a Map 537
Wall Collisions 543
Rotating Enemy Billboard 546
Enemy Movement 551

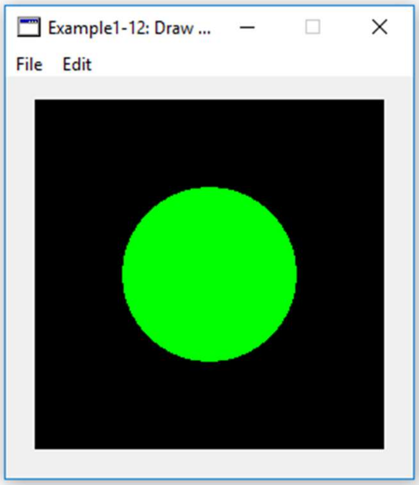
Dictionary 553

Index 557

Draw a Sphere

This example draws a sphere, which is a circle in 3-dimensions. Although the example looks like a circle, it is a true sphere. This example will be used in a later example where the scene will be viewed in 3-dimensions. Below is a screen grab of the running program.

Figure 14. Example 1-12: Sphere Screen Grab



A method has been created to help make the sphere which accepts two parameters, which is the radius of the sphere and the graduations (number of subsets) to make the sphere appear smooth. A higher graduation value (20) makes a smoother sphere than a sphere with 5 graduations.

To create this project, start Xojo and make a desktop and save the project as Example 1-12. Add the following control to the project.

Table 5. Added Controls

Control	Name	Settings
OpenGLSurface	OpenGLSurface 1	Width: 240, Height: 240

Add a new method to Window1 called DrawASphere and add the following code:

Code 20. Example 1-12: DrawCircleOutline

```
OpenGL.glColor3d(0, 1, 0)  //Green Colour
//Draw a green sphere
Dim x1, y1, z1, alpha, beta as Double
Const Pi = 3.1415926535
```

```
For alpha = 0 to Pi step (Pi/Graduations)
  OpenGL.glBegin(OpenGL.GL_TRIANGLE_STRIP)
  For beta = 0 to 2.01*Pi Step (Pi/Graduations)
    x1 = Radius*cos(beta)*sin(alpha)
    y1 = Radius*sin(beta)*sin(alpha)
    z1 = Radius*cos(alpha)
    Opengl.glVertex3f(x1, y1, z1)
    x1 = Radius*cos(beta)*sin(alpha + PI/Graduations)
    y1 = Radius*sin(beta)*sin(alpha + PI/Graduations)
    z1 = Radius*cos(alpha + PI/Graduations)
    Opengl.glVertex3f(x1, y1, z1)
  Next beta
  OpenGL.glEnd
Next alpha
```

Some variables are created and the Pi constant is defined. Two For-Next loops are created and x, y, and z vertices (corner coordinates) are calculated with the various math calculations, and the second set of vertices are calculated with the beta values.

A drawing colour was chosen to be green with the glColor3d, and the primitive type to be drawn for the sphere is a triangle strip.

To call the method, add the Render event handler and add the following code.

Code 21. Example 1-12: OpenGLSurface1 Render

```
//Draw a sphere with radius of 0.5
//Make 20 graduations to smooth the sphere
DrawASphere(0.5, 20)
```

Here is the call to the method which does the hard work of drawing the sphere. This example draws a filled sphere by drawing many triangle strips.

Index

2-D Intermediate, 38
3-D Coordinates, 86
Abs, 468, 478
ActionButton, 497
Activate Colour, 212
ActiveX Fundamentals, 42
Add Voices, 504
AddRows, 177
AlternateActionButton, 497
Angle of Incidence, 189
AntiAlias, 70
AoI, 189
Append, 176, 277
Array, 135, 485
Array Interleaved, 148, 156
Arrow Key Detection, 49, 52, 54
Arrow Movement, 48
Auto Rotate a Triangle, 46
Barycentric Coordinates, 462, 468, 472, 478
Blend, 71
Block Attributes, 141
Blue, 254
Byte, 254
Camera, 378
CancelButton, 496
Caption, 496
Colour Activate, 212
Colour Block Attributes, 141
Colour Pyramid, 168
Colours, 168
cos, 100
CType, 155, 279
Cube Array, 126
Cursors, 520
d, 351
dfactor, 73
DisplayName, 276, 324, 336
DoubleValue, 135
DoubleValue(0), 69
Draw a Line Strip, 20
Draw a Multi-Coloured Triangle, 42
Draw a Polygon, 26
Draw a Quad Strip, 28
Draw a Square (Quad), 13
Draw a Triangle, 18
Draw a Triangle Fan, 32
Draw a Triangle Strip, 30
Draw Line Loop, 24
Draw Lines, 22
DrawPicture, 495
Encodings, 277
Environment Cube, 404
EOF, 277
Examples
  01-01 Draw a 2-D Square (Quad), 13
  01-01 glColor4D Snippet, 16
  01-02 Draw a Triangle (2-D), 18
  01-03 Create a Line Strip (2-D), 20
  01-04 Background Colour Change, 22
  01-04 Create Two Lines, 23
  01-05 Draw a Line Loop, 24
  01-06 Draw a Polygon, 26
  01-07 Draw a Quad Strip, 28
  01-08 Draw a Triangle Strip, 30
  01-10 Draw a Filled Circle, 32
  01-11 Draw a Circle Outline, 34
  01-12 Draw a Filled Sphere, 36
  02-01 Draw a Multi-Coloured Triangle, 42
  02-01 Multi-Coloured Triangle, 43
  02-02 Rotate a Triangle, 45
  02-02 Rotate Snippet, 44
  02-03 Auto Rotate a Triangle, 47
  02-04 OpenGLSurface1.Render Method, 49
  02-04 Window1.KeyDown Method, 49
  02-04 Windows1.Open Method, 48

02-05 Translate OpenGLSurface1.Render Method, 51
02-05 Window1.KeyDown Method, 52
02-05 Windows1.Open Method, 51
02-06 OpenGLSurface1.Render Method, 54
02-06 Rotate Windows1 Open Method, 53
02-06 Window1.KeyDown Method, 54
02-07 OpenGLSurface1.Render Method, 57
02-07 Scale Windows1.open Method, 57
02-07 Window1.KeyDown Method, 58
02-08 LoadList, 62
03-01 Draw a point, 67
03-02 Change point size, 68
03-03 Smooth and Blend, 71
03-04 Wireframe or Filled Quads, 78
03-05 Face Cull, 83
04-01 Draw Cube Method, 89
04-01 fps Timer Event, 88
04-02 fps OpenGLSurface1.Render, 92
04-02 fps Timer Action, 91
04-03 Version, Vendor, Renderer, 94
04-04 ChangeFullScreen Method, 97
04-04 Full Screen Mode Key Capture, 96
05-01 Load Texture RAW File, 235
05-01 Simple Graph, 100
05-02 Viewport, 102
05-03 Graph Axis, 106
05-04 Simple Line Chart, 110
05-05 Pie Chart, 112
06-01 Draw Cube Vertex’s, 121
06-02 Draw Cube Array, 127
06-03 Cube Vertex Array, 135
06-04 Colour Array, 143
06-05 Interleaved Array, 151
06-06 Pyramid Array, 159
06-07 Load List, 165
07-01 Coloured Pyramid, 171
07-02 Smooth-Flat Shading, 184
08-01 Add Positional Light, 193
08-02 Ambient Light, 201
08-03 Diffuse Light, 208
08-04 Enable Colour Shading, 215
08-05 Colour Material Shading, 224
09-01 Texture in 2D, 231
09-02 OpenGLSurface1.Render Method, 240
09-03 External Texture, 245
09-04 Repeat Texture, 258
09-05 Multiple Textures, 261
10-01 Load WaveOBJ File Model, 277
10-02 Model Light Code, 285
10-03 Polygon Optimization, 289
11-01 Fog Shader, 295
12-01 Push-Pop Rotate-Stationary, 302
12-02 Draw Both Models, 307
13-01 Transparent Colours, 312
13-02 Transparent Textures, 316
14-01 Specular and Shininess, 320
14-02 OBJ Model Load Optimize, 333
14-03 Load Model Data, 355
15-01 Camera gluPerspective Command, 384
15-02 Adjust Camera Position, 387
15-03 FPV Render, 397
15-04 Initialize Skybox, 407
16-01 Slow Text, 416
16-02 Texture Dynamic Text, 422
16-03 Texture Pushbutton, 429
16-04 Optimized Picture to Texture, 431
17-01 Dual Threads, 441
18-01 Floor Mipmap, 450
18-02 HeightMap, 457
19-01 Ray-Plane Intersection, 467
19-02 Ray-Sphere Intersection, 478
20-01 Xojo Blocks, 485
21-01 Play Sound Effects, 503
21-02 Text-To-Speech (Win OS), 503
21-03 Text-To-Speech, 505
22-01 Ball Bounce, 508
22-02 Mutiple Ball Class, 512
22-03 Dynamic Primitive Colour, 517
22-04 Mouse Move Balls, 520
23-01a Setup Floor MipMap, 529

23-01b Setup Cube, 534
23-01c Add a Map, 539
23-01d Collision Detection, 544
23-01e Enemy Detection/Rotate, 549
23-01f Enemy Movement, 551
Explanation, 497
Feedback, 86
FingerPointer, 520
First Person Shooter Camera (FPS), 392
fps – Adding Frames per Second, 91
FPS – First Person Shooter Camera, 392
fps – Frames Per Second, 87
Full Screen Mode, 95
Gaussian Blur, 71
GetDescription, 504
GetVoices, 504
GL_ALPHA, 235
GL_ALPHA 12, 235
GL_ALPHA 16, 235
GL_ALPHA 8, 235
GL_ALPHA4, 235
GL_AMBIENT, 223
GL_AMBIENT_AND_DIFFUSE, 223
GL_BACK, 223
GL_BITMAP, 235
GL_BLEND, 72, 312
GL_BLEND, 233
GL_BLEND, 239
GL_BYTE, 235
GL_COLOR_ARRAY, 147
GL_COLOR_INDEXES, 223
GL_COLOR_MATERIAL, 215
GL_COMPILE, 62, 166
GL_COMPILE_AND_EXECUTE, 62, 166
GL_CONSTANT_ALPHA, 74, 75, 314
GL_CONSTANT_COLOR, 74, 75, 314
GL_CULL_FACE, 83
GL_DECAL, 233, 239
GL_DEPTH_TEST, 78
GL_DIFFUSE, 223
GL_DONT_CARE, 73
GL_DOUBLE, 147
GL_DST_ALPHA, 74, 75, 314
GL_DST_COLOR, 74, 75, 313
GL_EMISSION, 223
GL_EXP, 297
GL_EXP2, 297
GL_FASTEST, 73
GL_FILL, 79
GL_FLAT, 172, 184
GL_FLOAT, 235
GL_FOG, 295
GL_FOG_COLOR, 296
GL_FOG_END, 295
GL_FOG_MODE, 295
GL_FOG_START, 295
GL_FRAGMENT_SHADER_DERIVATIVE_HINT, 73
GL_FRONT, 223, 321
GL_FRONT_AND_BACK, 79, 223
GL_GENERATE_MIPMAP, 450
GL_INT, 235
GL_INTENSITY, 235
GL_INTENSITY12, 235
GL_INTENSITY16, 235
GL_INTENSITY4, 235
GL_INTENSITY8, 235
GL_LINE, 79
GL_LINE_LOOP, 15
GL_LINE_SMOOTH, 70, 72
GL_LINE_SMOOTH_HINT, 73
GL_LINE_STRIP, 15
GL_LINEAR, 297
GL_LINEAR, 234
GL_LINEAR_MIPMAP_LINEAR, 234
GL_LINEAR_MIPMAP_NEAREST, 234
GL_LINES, 15
GL_LUMINANCE, 235
GL_LUMINANCE_ALPHA, 235
GL_LUMINANCE12, 235
GL_LUMINANCE12_ALPHA12, 235
GL_LUMINANCE12_ALPHA4, 235
GL_LUMINANCE16, 235
GL_LUMINANCE16_ALPHA16, 235
GL_LUMINANCE4, 235
GL_LUMINANCE4_ALPHA4, 235

GL_LUMINANCE6_ALPHA2, 235
GL_LUMINANCE8, 235
GL_LUMINANCE8_ALPHA8, 235
GL_MINUS_CONSTANT_COLOR, 314
GL_MODELVIEW, 298
GL_MODULATE, 233, 239
GL_NEAREST, 234
GL_NICEST, 73
GL_NORMALIZE, 334
GL_ONE, 74, 75, 313
GL_ONE_MINUS_CONSTANT_ALPHA, 74, 75, 314
GL_ONE_MINUS_CONSTANT_COLOR, 74, 75
GL_ONE_MINUS_DST_ALPHA, 74, 75, 314
GL_ONE_MINUS_DST_COLOR, 74, 75, 313
GL_ONE_MINUS_SRC_ALPHA, 74, 75, 312, 314
GL_ONE_MINUS_SRC_COLOR, 74, 75, 313
GL_POINT_SIZE, 69
GL_POINT_SMOOTH, 70, 72
GL_POINT_SMOOTH_HINT, 73
GL_POINTS, 67, 69
GL_POINTS, 15
GL_POLYGON, 15, 278
GL_POLYGON_SMOOTH, 70, 72
GL_POLYGON_SMOOTH_HINT, 73
GL_PROJECTION, 298
GL_PROXY_TEXTURE_2D, 235, 239
GL_QUAD_STRIP, 15
GL_QUADS, 15, 147
GL_R3_G3_B2, 235
GL_RENDERER, 94
GL_REPLACE, 233, 239
GL_RGB, 235
GL_RGB10, 235
GL_RGB10_A2, 235
GL_RGB12, 235
GL_RGB16, 235
GL_RGB4, 235
GL_RGB5, 235
GL_RGB5_A1, 235
GL_RGB8, 235
GL_RGBA, 316
GL_RGBA, 235
GL_RGBA12, 235
GL_RGBA16, 235
GL_RGBA2, 235
GL_RGBA4, 235
GL_RGBA8, 235
GL_SHININESS, 223, 322
GL_SHORT, 235
GL_SMOOTH, 184
GL_SPECULAR, 223, 322
GL_SRC_ALPHA, 74, 75, 312, 314
GL_SRC_ALPHA_SATURATE, 74, 314
GL_SRC_COLOR, 74, 75, 313
GL_TEXTURE_1D, 233, 234
GL_TEXTURE_1D_ARRAY, 233, 234
GL_TEXTURE_2D, 233, 234, 235, 239
GL_TEXTURE_2D MULTISAMPLE, 233
GL_TEXTURE_2D_ARRAY, 234
GL_TEXTURE_2D_ARRAY, 233, 234
GL_TEXTURE_2D_MULTISAMPLE_ARRAY, 233
GL_TEXTURE_3D, 233, 234
GL_TEXTURE_COMPRESSION_HINT, 73
GL_TEXTURE_CUBE_MAP, 233
GL_TEXTURE_ENV, 233, 238
GL_TEXTURE_ENV_MODE, 233, 238
GL_TEXTURE_RECTANGLE, 233, 234
GL_TEXTURE_WRAP_S, 234, 239
GL_TEXTURE_WRAP_T, 234, 239
GL_TRIANGLE_FAN, 15, 33
GL_TRIANGLE_STRIP, 15, 457
GL_TRIANGLES, 15
GL_UNSIGNED_BYTE, 235, 255
GL_UNSIGNED_INT, 235
GL_UNSIGNED_SHORT, 235
GL_VENDOR, 94
GL_VERSION, 94
GL_VERTEX_ARRAY, 147
GL_ZERO, 74, 75, 313
glBegin, 14
glBlendFunc, 74, 312
glCallList, 63, 167
glColor, 14

glColor3b, 16
glColor3bv, 16
glColor3d, 16
glColor3dv, 16
glColor3f, 16
glColor3fv, 16
glColor3i, 16
glColor3iv, 16
glColor3s, 16
glColor3sv, 16
glColor3ub, 16
glColor3ubv, 16
glColor3ui, 16
glColor3uiv, 16
glColor3us, 16
glColor3usv, 16
glColor4b, 16
glColor4d, 16, 312
glColor4f, 16
glColor4i, 16
glColor4s, 16
glColor4ub, 16
glColor4ui, 16
glColor4us, 16
glColorPointer, 147
glDisableClientState, 139
glDrawArrays, 139
glEdgeFlag, 14
glEnable, 72
glEnableClientState, 139
glEnd, 14, 62, 166
glEvalCoord, 14
glEvalPoint, 14
glFlush, 33, 67
glFogf, 295
glFogfv, 296
glFogi, 295
glGenLists, 61, 165
glGetDoublev, 69
glGetString, 94
glGetString_RB, 94
glHint, 73
glIndex, 14
glMaterial, 14
glMaterialfv, 321
glNewList, 62, 166
glNormal, 14
glOrtho, 429, 491
glPixelZoom, 64
glPolygonMode, 79
glPopMatrix, 79, 124, 302
glPopMatrix, 50
glPushMatrix, 124, 302
glPushMatrix, 50
glRotated, 124
glRotateD Command, 52
glScaleD, 56
glShadeModel, 172
glTexCoord, 14
glTexCoord2i, 236
glTranslateD, 50
gluLookAt, 385
gluPerspective, 384
glVertex, 14
glVertex3d, 67, 69, 121, 191, 199, 206, 213
glVertexPointer, 139
Graph Axis, 105
GraphicQuestion, 496
Green, 254
Height, 254
Heron Formula, 463, 474
illum, 351
InRange, 489
Integer, 155
Interleaved Array, 148, 156
Intermediate 2-D, 38
Interpolated, 168
Ka, 351
Kd, 351
Ke, 351
Ks, 351
LeftB, 277
Light Position, 190
Lighting, 187
Line Chart, 109
Line Loop, 24

- Line Strip, 20
- Lines, 22
- Long, 254
- LowestPriority), 442
- MakeCurrent, 429
- MaskCol, 254
- Materil Colour, 220
- MemoryBlock, 68, 134
- Message, 497
- MessageDialog, 496
- MessageDialogButton, 496
- Microseconds, 137
- ModeMultiple, 497
- MouseDown, 521
- MouseDrag, 522
- MouseMove, 520
- MouseUp, 523
- Move Objects, 518
- Move Triangle (OpenGl Style), 50
- Multi-Coloured Triangle, 42
- newmtl, 351
- Ni, 351
- Normal, 188
- NormalPriority), 442
- Ns, 351
- OLEObject, 503
- OpenGL Suffix, 11
- PathTypeAbsolute, 276, 324, 337
- Period, 499
- Pixel, 254, 457
- Play, 503
- Polygon, 26
- Position Light, 190
- Projection View, 379
- Projection View at Runtime, 386
- Ptr, 155
- Quad Strip, 28
- Quad Texture, 230
- Random, 281
- Randomizer, 278, 281, 489
- Rate, 504
- RAW File, 236
- Ray-Plane Intersection, 460
- ReadLine, 277
- Red, 254
- ReDim, 135
- Render, 137, 430
- Renderer, 94
- ReplaceAll, 279
- RGBSurface, 254, 457
- RightB, 278
- Rotate, 52
- Rotate a Triangle, 44
- Runtime Projection View, 386
- SAPI, 503
- Scale, 56
- Select Case, 177
- sfactor, 73
- ShowModal, 497
- Simple Graph, 99
- Skybox, 404
- Smoothing, 70
- Sound, 502, 503
- Sound Effects, 502
- Speak, 504, 505
- Spinning Cube Texture, 237
- Split, 279
- SPObjectToken, 503
- SPVoice, 503
- Sqrt, 468
- Square, 13
- StandardPointer, 520
- Suffix
  - b, 11
  - d, 11
  - f, 11
  - i, 11
  - s, 11
  - ub, 11
  - ui, 11
  - us, 11
  - v, 11
- Text, 414
- Text Slow, 414
- TextInputStream, 277
- Text-To-Speech Win OS, 503

Texture Text, 414
Textured Spinning Cube, 237
Texturing, 229
Thread, 137
Triangle, 18
Triangle Arrow Movement, 48
Triangle Auto-Rotation, 46
Triangle Fan, 32
Triangle Rotation, 44
Triangle Strip, 30
TTS, 503
Ubound, 131, 277
UTF8, 277
Vendor, 94
Version, 94
Vertex Array, 133
Vertex Cube, 119
Viewing - Camera, 378
ViewPort, 101
Volume, 504
What is OpenGL, 10
Width, 254
Window Mode, 95
Xojo Blocks, 482

The ‘I Wish I Knew’ series contains technical data and advice that makes sense and contains practical and numerous examples with explanations to allow you to ease into the steep programming curve. You can create Xojo programs today!

The book “I Wish I Knew How to … Program OpenGL with Xojo” builds a strong foundation of programming graphics. This is an advanced graphics book and it is highly recommended to understand OpenGL and Xojo first. An intermediate book called “I Wish I Knew How to… Program the Canvas Control with Xojo Desktop” is a good book to study before learning OpenGL. The examples have been created and tested on Windows 10 and OSX El Capitan 32-bit Xojo settings, as some of the examples do not work when compiled as a 64-bit program.

Creating static data such as graphs can use most of the common numerical types in Xojo. If you wish to create motion and games, then this requires the knowledge of MemoryBlocks and OpenGl. The Timer control limitation for drawing on Windows OS has been enhanced by implementing a timer thread, which allows high Frames-Per-Second rates!! A partial OBJ model loader has been created to render and animate more complicated Blender models.

There are 23 chapters, more than 550 pages, and over 80 example programs. Source code is included with this book and is not heavily optimized because it was built for educational purposes. OpenGL animation speed is very impressive.

This is one of many books by Eugene with others available at XojoLibrary. Other books can be purchased at http://XojoLibrary.com where many great Xojo resources are available.

Happy programming!

Eugene

Eugene Dakin MBA, Ph.D., P.Chem., is an author of Xojo reference materials and has many years of experience in the programming industry. More great reference books are “*I Wish I Knew How To … Program Raspberry Pi 2B and 3B with Xojo”, “I Wish I Knew How to … Program the Canvas with Xojo in Windows”*.

ISBN: *978-1-927924-01-3*