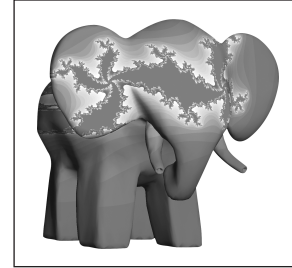


Index



- , (comma) sequence operator, 83, 87
- . (period), selection operator
 - components, selecting, 84–85
 - order of precedence, 83
 - swizzling, 84–85
 - (period), swizzle operator
 - components, rearranging, 84–85
 - order of precedence, 83
- ?: (question mark colon), ternary selection operator, 83, 87
- (minus signs), decrement operator
 - matrix decrement, 86
 - order of precedence, 83
 - vector decrement, 86
- / (forward slash), division operator
 - matrix division, 86
 - order of precedence, 83
 - preprocessor expression, 89
 - vector division, 86
- /= (forward slash equal sign), assignment operator, 83, 87
- (minus sign), negation operator
 - matrix negation, 86
 - order of precedence, 83
 - preprocessor expression, 89
 - vector negation, 86
- (minus sign), subtraction operator
 - matrix subtraction, 86
 - order of precedence, 83
 - preprocessor expression, 89
 - vector subtraction, 86
- = (minus sign equal sign), assignment operator, 83, 87
- [] (square brackets), index operator, 83–84
- & (ampersand) bit-wise expression, preprocessor, 83
- && (ampersands), logical and operator
 - Boolean logical and, 86
 - order of precedence, 83
 - preprocessor expression, 89
- * (asterisk), multiplication operator
 - matrix/matrix multiplication, 85–86
 - order of precedence, 83
 - preprocessor expression, 89
 - vector/matrix multiplication, 85–86
- *= (asterisk equal sign), assignment operator, 83, 87
- | (bar) bit-wise expression, preprocessor, 83
- || (bars), logical inclusive or operator
 - Boolean logical inclusive, 86
 - order of precedence, 83
 - preprocessor expression, 89
- ^ (caret) bit-wise expression, preprocessor, 89
- ^^ (carets), logical exclusive or operator
 - Boolean logical or, 86
 - order of precedence, 83
- = (equal sign), assignment operator, 83, 87
- == (equal signs), equality operator
 - Boolean equality comparison, 86
 - order of precedence, 83
 - preprocessor expression, 89
- != (exclamation equal sign), equality operator
 - Boolean inequality comparison, 86
 - order of precedence, 83
 - preprocessor expression, 89
- ! (exclamation mark), logical not operator
 - Boolean logical not, 86
 - order of precedence, 83
 - preprocessor expression, 89
- >= (greater than or equal), relational operators
 - Boolean greater than or equal, 86
 - order of precedence, 83
 - preprocessor expression, 89

- > (greater than sign), relational operators
 - Boolean greater than, 86
 - order of precedence, 83
 - preprocessor expression, 89
 - >> (greater than signs) bit-wise shift, preprocessor, 89
 - <= (less than or equal), relational operators
 - Boolean less than or equal, 86
 - order of precedence, 83
 - preprocessor expression, 89
 - < (less than sign), relational operators
 - Boolean less than, 86
 - order of precedence, 83
 - << (less than signs) bit-wise shift, preprocessor, 89
 - % (percent), preprocessor expression, 89
 - + (plus sign), addition operator
 - matrix addition, 86
 - order of precedence, 83
 - preprocessor expression, 89
 - vector addition, 86
 - += (plus sign equal sign), assignment operator, 83, 87
 - ++ (plus signs), increment operator
 - matrix increment, 86
 - order of precedence, 83
 - vector increment, 86
 - # (pound sign), hash-based operator, 90
 - ## (pound signs), hash-based operator, 90
 - _ (underscore), in variable names, 72
 - __ (underscores), in variable names, 73
 - ~ (tilde), unary preprocessor expression, 89
 - 1D noise function, 294–297
 - 1D texture, 26, 513
 - 1D texture maps, 383–384
 - 2D noise function, 297–298
 - 2D texture, 26, 513
 - 3D and 4D noise function, 298–299
 - 3D texture, 26, 513
 - 3Dlabs logo, Color Plate 21
- A**
- abs**
 - description, 122
 - hatching example, 360
 - purpose, 124, 125
 - Absolute value functions, 122
 - Accumulation buffer, 7, 513
 - acos**, 119
 - Active attribute variables, 181, 513. *See also* attribute variables.
 - Active samplers, 192, 513
 - Active texture unit, 26
 - Active uniform variables, 189–190, 513. *See also* uniform variables.
 - Adaptive analytic prefiltering, 342–345
 - Add blend mode, 393, Color Plate 30
 - Addition operator
 - matrix addition, 86
 - order of precedence, 83
 - preprocessor expression, 89
 - vector addition, 86
 - Advanced RenderMan: ...CGI for Motion Pictures*, 346
 - Air-brush pen effects, 365. *See also* Gooch shading.
 - Akeley, Kurt, 3
 - Algorithm analysis, developing shaders, 205
 - Aliasing, 335–339, 513. *See also* antialiasing.
 - all**, 86–87, 134
 - Alpha, 514
 - Alpha test, 16, 514
 - Ambient lighting value, modifying, 13
 - Ampersand (&) bit-wise expression, preprocessor, 83
 - Ampersands (&&), logical and operator
 - Boolean logical and, 86
 - order of precedence, 83
 - preprocessor expression, 89
 - Amplitude, 514
 - Analytic integration, 345–348, Color Plate 25
 - Analytic prefiltering, 341–342
 - Angle functions, 118–120
 - Animated shaders. *See* shaders, animated.
 - Animation. *See* shaders, animated.
 - Anisotropic, 252, 514
 - Anisotropic reflection, 252
 - Antialiasing
 - adaptive analytic prefiltering, 342–345
 - analytic integration, 345–348, Color Plate 25
 - analytic prefiltering, 341–342
 - area sampling, 346
 - avoiding aliasing, 337–338
 - brick pattern, 157, 348–349
 - checkerboard shader, 350–351
 - convolution, 345–348
 - convolution filter, 345–348
 - convolution kernel, 345–348
 - defined, 15, 514
 - enabling, 15
 - frequency clamping, 349–351
 - increasing resolution, 338–339
 - sources of aliasing, 335–337
 - strip example, 339–349
 - supersampling, 338–339
 - any**, 86–87, 134
 - API (application programming interface), 1, 514
 - Application programming interface (API), 1, 514
 - ARB (OpenGL Architecture Review Board), 2
 - ARB prefix, 3
 - Arc cosine functions, 119
 - Arc sine functions, 119
 - Arc tangent functions, 120
 - Area sampling, 346, 514
 - Arrays, 71–72
 - Articles of interest, 529–542
 - Artistic effects. *See* NPR (non-photorealistic rendering).
 - asin**, 119
 - Aspect ratio, setting, 24

- Assignment operator, 83, 87
 - Asterisk (*), multiplication operator
 - matrix/matrix multiplication, 85–86
 - order of precedence, 83
 - preprocessor expression, 89
 - vector/matrix multiplication, 85–86
 - Asterisk equal sign (*=), assignment operator, 83, 87
 - atan**, 120
 - Attached shader objects, listing, 172–173
 - Attaching shader objects, 160, 164–167. *See also* **glAttachObjectARB**.
 - Attenuation, 219, 514
 - attribute**
 - example, 65
 - shader input/output, 49
 - shader interface, 76–77, 78
 - variable qualifier, 76–77, 78
 - Attribute aliasing, 178, 514
 - Attribute variables
 - active, 181, 513
 - active, querying, 181–183, 453–455
 - associating generic vertex attributes with, 177–181, 438–440
 - built-in vertex, 38–39, 95–96, 173
 - defined, 38–39, 77, 514
 - gl_Color*, 39, 96, 173
 - gl_Normal*, 39, 77, 96, 173
 - gl_Vertex*, 39, 77, 96, 173
 - location, querying, 178–179, 462–463
 - user-defined, 38–39, 77, 174–176
 - value, modifying, 173–176, 500–504
 - value, querying, 183–184, 477–479
 - vertex array state, modifying, 176–177, 451–452, 505–507
 - vertex array state, querying, 183–184, 480–481
 - vertex arrays, enabling, 177, 451–452
- Attributes
 - generic vertex, 95–96, 160, 174–181
 - vertex, 10, 527
- Average blend mode, 389, Color Plate 30
- B**
- Baldwin, Dave, 406
- Ball, example, 271–279, Color Plate 16, Color Plate 17
- Bar (|) bit-wise expression, preprocessor, 83
- Bars (||), logical inclusive or operator
 - Boolean logical inclusive, 86
 - order of precedence, 83
 - preprocessor expression, 89
- Behind blend mode, 390, Color Plate 30
- Biasing attributes, setting, 19
- Bidirectional reflectance distribution function (BRDF), 252, 515
- Bit-wise operators, 51
- Bitfields in structures, 51
- Black curves as edges, 365. *See also* Gooch shading.
- Blend modes, Color Plate 30
 - add, 393
 - average, 389
 - behind, 390
 - clear, 390
 - color burn, 391
 - color dodge, 391
 - darken, 390
 - difference, 393
 - dissolve, 389–390
 - examples, Color Plate 30
 - exclusion, 393–394
 - hard light, 392–393
 - inverse difference, 393
 - lighten, 390–391
 - multiply, 391
 - normal, 389
 - opacity, 394
 - overlay, 392
 - screen, 391
 - soft light, 392
 - subtract, 393
- Blending, 16, 388–394
- Blinking on/off, 316
- Blue elephant, Color Plate 1
- Books of interest, 529–542
- bool** data type, 66–67
- Borders, convolution, 395–396
- BRDF (bidirectional reflectance distribution function), 252–262, 515, Color Plate 11
- break**, 80
- Brick shader
 - aliasing, 157
 - antialiasing, 348–349, Color Plate 25
 - application source code, 194–199
 - brick/mortar pattern, 151–156
 - color computation, 151–156
- diffuse reflection, 148–150
- fragment shader, 151–156
- homogeneous vertex position, 150–151
- lighting calculations, 146–147
- overview, 143–145
- reflection vector, 148
- scaling, 156
- vertex shader, 145–151
- Brightness, interpolation operations, 386
- Buffers
 - frame buffer components, 17
 - memory objects, 11–12
- Built-in attribute variables, 38–39, 95–96
- Built-in constants, 109–110
- Built-in functions. *See* functions, built-in.
- Built-in uniform variables, 40, 97, 102, 104–108
- Built-in varying variables, 101
- Bump map, 515
- Bump mapping
 - defined, 280–288, 515



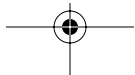
- Bump mapping defined (*continued*)
 - examples, Color Plate 9, Color Plate 18
- Bust, wood shaded, Color Plate 22
- bvecn** data type, 67–68
- C**
- C language, and OpenGL, 48–51
- Call by value-return, 51, 81–82, 515
- Caret (^) bit-wise expression, preprocessor, 89
- Carets (^), logical exclusive or operator
 - Boolean logical or, 86
 - order of precedence, 83
- ceil**, 122, 125–126
- Cg, 415–416
- Character support, 50–51
- Checkerboard shader, 350–351
- Chessboard, Color Plate 12
- CIE colors, 383–385
- clamp**, 123, 127–128, 337–338
- class**, 71
- Clear blend mode, 390, Color Plate 30
- Client-side operations, 11–12
- Clip space, 24, 515
- Clipping
 - defined, 14, 112, 515
 - frustum, 24–25, 518
 - user, 25, 231, 526
 - with vertex shaders, 98
- Clipping coordinate system, 24, 515
- Clipping planes, 24–25
- Cloud shader, 303–306, Color Plate 20
- Cohen, Elaine, 365
- Color
 - gradients, 367. *See also* Gooch shading.
 - light sources, 13
 - secondary, setting, 16
 - space conversions, 383–385
- Color burn blend mode, 391, Color Plate 30
- Color computation, 151–156, 367
- Color dodge blend mode, 391, Color Plate 30
- Color Plates. *See also color insert section.*
 - 2, 239
 - 3, 241
 - 4, 242
 - 5, 242, 246
 - 6, 246–247
 - 7, 247, 251
 - 8, 251
 - 9, 251
 - 10, 253
 - 11, 261
 - 12, 268
 - 13, 267, 271
 - 14, 271, 276
 - 15, 280
 - 16, 276, 280
 - 17, 277
 - 18, 287–288
 - 19, 288
 - 20, 21, 305, 307, 308
 - 21, 330
 - 22, 311, 312
 - 23, 367
 - 24, 327
 - 25, 156, 349
 - 26, 384
 - 27, 385
 - 28, 385
 - 29, 386
 - 30, 387
- Color sum, 16, 515
- Comet tails, 321
- Comma (,) sequence operator, 83, 87
- Commands. *See* functions.
- Common functions, 121–129
- Compiler, 542–55
- Compiler front end, 211–212, 515
- Compilers, error handling, 90
- Compiling
 - OpenGL shaders, 160, 162–163, 441–442
 - shader objects, 162–163, 441–442
 - source code, 52
- Complex numbers, 370. *See also* Mandelbrot set.
- Component-wise operations, 85–87
- Composite images, 7, 388–394
- Computational frequency, 204–205
- Confetti cannon shader, 322–327
- const**, 76, 78–79, 81–82
- Constants, 109–110
- Constructors
 - defined, 50, 73–75, 515
 - type conversions, 75
- continue**, 80
- Contrast, interpolation operations, 386–387
- Control texture, 267, 388, 515
- Converting data types, 75
- Convolution
 - borders, 395–396
 - defined, 345–348, 394, 516
 - edge detection, 400
 - equation, 394
 - filter, 345–348, 394, 516
 - fragment shaders, 395
 - Gaussian smoothing filter, 399
 - general computation, 398–400
 - kernel, 345–348, 394, 516
 - neighborhood averaging, 396–397, 521
 - noise reduction, 397–398
 - sharpness, 400–402, Color Plate 29
 - smoothing, 396–400
 - vertex shaders, 395
- Convolution filter, 345–348, 394, 516
- Convolution kernel, 345–348, 394, 516
- Convolution operations, 394–402
- Cook, Rob, 405



- Cool colors, 367. *See also* Gooch shading.
- Coordinate systems
 - complex numbers, 373–374. *See also* Mandelbrot sets.
 - OpenGL, 20–25
 - surface-local coordinate space, 281
 - tangent space, 282
- Coordinate transforms
 - aspect ratio, setting, 24
 - clip space, 24
 - clipping planes, 24–25
 - eye coordinates, 22
 - eye space, 22
 - field of view, setting, 24
 - frustum clipping, 24–25
 - light source, 24
 - model space, 21
 - model transformation matrix, 22
 - modeling coordinate system, 21
 - modeling transformation, 22
 - modelview matrix, 22–24
 - normalized device coordinate space, 25
 - object space, 21
 - parallel projection, setting, 24
 - perspective projection, setting, 24
 - projection matrix, 24
 - projection transformation, 24
 - user clipping, 25
 - viewing matrix, 22
 - viewing transformation, 22
 - viewport transformation, 25
 - window coordinates, 25
 - world coordinate system, 21
 - world space, 21
- Copying images, 18–19
- cos, 119
- Cosine functions, 119
- Cow, Color Plate 15
- cross, 130
- Cross product functions, 130
- Cube map, 26–27, 516
- Cube-map-access functions, 137, 237–238
- Cube map sampler type, 49, 69–70, 192, 237–238
- Cube map texture, 26, 229, 235, 516
- Cube mapping, 26–27, 247, 516
- Culling
 - defined, 14, 516
 - enabling/disabling, 14
 - primitives, 14
- D**
- Darken blend mode, 390, Color Plate 30
- Data binding, 4
- Data types
 - arrays, 71–72
 - automatic promotion, 50
 - bool, 66–67
 - bvecn, 67–68
 - declaring, 72–73
 - enumerated types, 51
 - float, 66–67
 - int, 66–67
 - ivecn, 67–68
 - matn, 67–68
 - matrices, 68–69
 - promotion, 73
 - samplerCube, 69–70
 - samplerND, 69–70
 - samplerNDShadow, 69–70, 115
 - samplers, 69–70
 - scalars, 65–67
 - scope, 72–73
 - structures, 70–71
 - type-casting, 51
 - type conversions, 75
 - type matching, 73
 - vecn, 67–68
 - vectors, 67–68
 - void, 72
- Data values, passing, 40–41
- Debugging shaders, 206–208
- Declaring data types, 72–73
- Decrement operator
 - matrix decrement, 86
 - order of precedence, 83
 - scalar decrement, 86
 - vector decrement, 86
- #define, 87
- defined, 87, 89
- degrees, 119
- Degrees, converting to radians, 119
- Dependent texture read, 44, 516
- Depth buffer, 6, 516
- Depth comparison lookup functions, 137–138
- Depth-cuing, 226, 516
- Depth test, 16, 516
- Depth texture, 29, 69, 115, 137, 235
- Depth-texture-access function, 69, 115, 137, 237–238
- Derivative functions, 138–139
- Detaching shader objects, 168
- Developer forums, 31–32
- Developing shaders
 - algorithm analysis, 205
 - built-in functions, 205
 - compiler front end, 211–212
 - computational frequency, 204–205
 - debugging, 206–208
 - general principles, 201–204
 - geometry, 208
 - information log review, 206
 - iterative testing, 203
 - modularity, 203–204
 - performance, 204–206



- Developing shaders (*continued*)
 - problem analysis, 202
 - progressive complexity, 202–203
 - RenderMonkey, 208–211
 - simplicity, 203
 - textures for complex functions, 206
 - tools for, 208–212
 - vectors, 206
- dFdx**
 - antialiasing, 342–345
 - defined, 138
 - hatching example, 359–360
- dFdy**
 - antialiasing, 342–345
 - defined, 138
 - hatching example, 359–360
- Difference blend mode, 393, Color Plate 30
- Diffuse reflection factor, 367
- Dinosaur, Color Plate 8
- Directional lights, 217–219
- discard**
 - debugging shaders, 207–208
 - fading images in/out, 320
 - flow control, 80
 - fragment shader output, 103–104
 - lattice effect, 279–280
- Discontinuous functions, 124–126
- Discontinuous jumps at arbitrary points, 124, 129
- Displacement mapping algorithms, 40
- Display list, 11, 516
- Display list mode, 11, 516
- Display memory, 5, 516–517
- Dissolve blend mode, 389–390, Color Plate 30
- distance**, 130–131
- Distance functions, 130–131
- Dithering, 16
- Division operator
 - matrix division, 86
 - order of precedence, 83
 - preprocessor expression, 89
 - scalar division, 86
 - vector division, 86
- do-while** statement, 80
- Donut shape. *See* torus.
- dot**, 130, 383
- Dot product functions, 130
- Double buffering, 6, 517
- Drawing. *See* images, drawing; primitives, drawing.
- Driver, 52, 412, 415, 517
- Driver model, 52–54
- E**
- Earth
 - cloud cover, 242–243
 - color image, Color Plate 2
 - daytime/nighttime, 241–247, Color Plate 5, Color Plate 6
- shader, 238–247
- texture-mapped sphere, Color Plate 3, Color Plate 6
- texture maps, 239, Color Plate 2, Color Plate 5
- Edges
 - black curves as, 365
 - detecting with convolution, 400
 - generating, 365–366
- Elephant, Color Plate 1
- #elif**, 87
- #else**, 87
- #endif**, 87
- enum**, 71
- Environment mapping
 - defined, 247, 517
 - shader, 247–251, Color Plates 7–9
- equal**, 86, 134
- Equal sign (=), assignment operator, 83, 87
- Equal signs (==), equality operator
 - Boolean equality comparison, 86
 - order of precedence, 83
 - preprocessor expression, 89
- Equality operator
 - Boolean equality comparison, 86
 - Boolean inequality comparison, 86
 - order of precedence, 83
 - preprocessor expression, 89
- #error**, 88
- Error conditions, OpenGL, 8
- Error handling, OpenGL Shading Language compiler, 90
- Euclidean distance function, 131
- Exclamation equal sign (!=), equality operator
 - Boolean inequality comparison, 86
 - order of precedence, 83
 - preprocessor expression, 89
- Exclamation mark (!), logical not operator
 - Boolean logical not, 86
 - order of precedence, 83
 - preprocessor expression, 89
- Exclusion blend mode, 393–394, Color Plate 30
- Executables, 34, 54, 60, 164–167, 193, 517
- exp2**, 120–121
- Exponential functions, 120–121
- EXT prefix, 3
- Extensions
 - description, 3–4
 - imaging, querying, 19
 - listing, 3
 - registry Web site, 3
 - supported, determining, 3
- Extrapolation shaders for imaging, 385–388
- Eye coordinate system, 22, 517
- Eye space, 22, 517
- F**
- faceforward**, 131
- Fading in/out, 320, 327



- Field of view, setting, 24
 - `__FILE__` macro, 88
 - File name support, 51
 - Filtering, 394, 517. *See also* convolution and antialiasing.
 - Finishing primitives, 17
 - Fire effect, 321
 - Fireworks effect, 321
 - Fixed functionality, 8, 517
 - Flat shading, 15, 517
 - float** data type, 66–67
 - Floating-point values
 - double precision, 51
 - matrix types, 48
 - querying implementation-dependent limits, 194
 - scalar data types, 66–67
 - floor**, 122, 125–126
 - Flow-control
 - calling conventions, 81–82
 - functions, 80–81, 82
 - overview, 79–80
 - type conversions, 75
 - Flushing primitives, 17
 - Fog, 16, 226–228, 517
 - for** statement, 80
 - Forward slash (/), division operator
 - matrix division, 86
 - order of precedence, 83
 - preprocessor expression, 89
 - scalar division, 86
 - vector division, 86
 - Forward slash equal sign (/=), assignment operator, 83, 87
 - fract**, 122, 125–126
 - Fractals. *See* Mandelbrot set.
 - Fragment, 15, 33, 517
 - Fragment depth value, 151–156
 - Fragment processing, 15–16, 518
 - Fragment processing functions, 138–139
 - Fragment processor. *See also* programmable processors.
 - defined, 41, 518
 - description, 41–45, 100–104
 - Fragment shaders. *See* shaders, fragment.
 - Fragments, 15, 33, 517
 - Frame buffer, 5–7, 17, 19, 518
 - Frame buffer operations, 17, 518
 - Frequency, 518
 - Frequency clamping, 349–351
 - Freudenberg, Bert, 356
 - Frustum. *See* view frustum.
 - Frustum clipping, 24–25, 518
 - fttransform**
 - description, 130
 - example, 151
 - positional invariance, 113
 - purpose of, 131
 - syntax, 130
 - Functions. *See also* entries for specific functions.
 - 1D noise, 294–297
 - 2D noise, 297–298
 - 3D and 4D noise, 298–299
 - gradient noise, 297
 - overloading, 50, 140
 - processing order, 4
 - state query, 168–173
 - value noise, 293–297
 - Functions, built-in
 - angle, 118–120
 - common, 121–129
 - cube-map access, 135–138, 238
 - depth-texture access, 135–138, 238
 - developing shaders, 205
 - exponential, 120–121
 - fragment processing, 138–139
 - geometric, 130–131
 - matrix, 132–133
 - noise, 139–140, 302–303
 - texture access, 135–138
 - trigonometry, 118–120
 - vector relational, 133–135
 - Fuzzy objects, 321–328
 - fwidth**, 139, 343, 347
- ## G
- Gaussian smoothing filter, 399
 - Generic vertex attribute arrays, 176–177, 183–184, 451–452, 505–507
 - Generic vertex attributes
 - associating with named attribute variables, 176–181, 438–440
 - parameters, querying, 183–184, 477–479
 - pointers, querying, 183–184, 480–481
 - value, modifying, 173–175, 500–504
 - Geometric functions, 130–131
 - Geometric image transformations, 382
 - Geometric primitives, 4, 518. *See also* primitives.
 - Geometry, developing shaders, 208
 - Geometry data, sending. *See* primitives, drawing.
 - glActiveTexture**, 26, 236, 239, 244, 301
 - GL_ADD**, 231, 233
 - glAlphaFunc**, 16
 - glAttachObjectARB**
 - application code example, 196, 198
 - defined, 163–164
 - overview, 52–54, 55, 61, 160, 199
 - syntax, 436–437
 - undoing, 168
 - gl_BackColor**, 99, 101, 110, 113, 225
 - gl_BackLightModelProduct**, 107, 225
 - gl_BackMaterial**, 106, 225
 - gl_BackSecondaryColor**, 99, 101, 110, 113, 225
 - glBegin**, 10, 14, 38, 40, 173, 179
 - glBindAttribLocationARB**
 - application code example, 179–181, 255–256, 283, 324

- `glBindAttribLocationARB` (*continued*)
 - defined, 177–178
 - overview, 39, 57, 160, 199
 - syntax, 438–440
- `glBindBuffer`, 11
- `glBindTexture`, 27, 236, 239, 244, 301
- `glBitmap`, 18, 382
- `GL_BLEND`, 231–232
- `glBlendColor`, 16
- `glBlendEquation`, 16
- `glBlendFunc`, 16
- `glBufferData`, 11
- `glBufferSubData`, 11
- `glCallList`, 11
- `glCallLists`, 11
- `GL_CLAMP_TO_BORDER`, 396
- `GL_CLAMP_TO_EDGE`, 396
- `glClear`, 17
- `glClearAccum`, 17
- `glClearColor`, 17
- `glClearDepth`, 17
- `glClearStencil`, 17
- `glClientActiveTexture`, 30
- `gl_ClipPlane[i]`, 105, 108
- `glClipPlane`, 14, 25, 112
- `gl_ClipVertex`, 40, 98, 112, 231
- `gl_Color`
 - attribute variable defined, 96, 173
 - attribute variable, example shader code, 179–180, 225, 231
 - attribute variable, overview, 39
 - varying variable, defined, 101
 - varying variable, example shader code, 326, 382, 386–388
 - varying variable, front facing computation, 99
- `glColor`, 10, 13, 38, 95, 173, 175, 179–180, 318, 366
- `glColorMask`, 17
- `glColorPointer`, 10–11, 324
- `glCompileShaderARB`
 - application code example, 196, 198
 - defined, 162–163
 - overview, 52, 55, 61, 160, 199
 - syntax, 441–442
- `glCompressedTexImage1D/2D/3D`, 27
- `glCompressedTexSubImage1D/2D/3D`, 27
- `GL_CONSTANT_BORDER`, 396
- `glCopyPixels`, 18, 20
- `glCopyTexImage`, 18, 20, 27
- `glCopyTexSubImage`, 18, 20, 27
- `glCreateProgramObjectARB`
 - application code example, 196, 198
 - defined, 163
 - overview, 55, 61, 160, 199
 - syntax, 443–444
 - undoing, 167
- `glCreateShaderObjectARB`
 - application code example, 195, 197
 - defined, 161
 - overview, 52, 55, 61, 159, 199
 - syntax, 445–446
 - undoing, 167
- `glCullFace`, 14, 366
- `GL_CURRENT_VERTEX_ATTRIB_ARB`, 184
- `GL_DECAL`, 231–232
- `glDeleteObjectARB`
 - defined, 167
 - overview, 55
 - syntax, 447–448
- `glDepthFunc`, 16, 366
- `glDepthMask`, 17
- `gl_DepthRange`, 105
- `glDepthRange`, 14, 25
- `gl_DepthRangeParameters`, 105
- `glDetachObjectARB`
 - defined, 168
 - overview, 55
 - syntax, 449–450
- `glDisable`, 8, 12–13, 27, 110, 111, 217
- `glDisableClientState`, 8, 325
- `glDisableVertexAttribArrayARB`
 - application code example, 325
 - defined, 177
 - overview, 57
 - syntax, 451–452
- `glDrawArrays`, 10, 95, 176–177, 324–325
- `glDrawBuffer`, 17
- `glDrawElements`, 10, 177
- `glDrawPixels`, 18, 27, 382, 388
- `glDrawRangeElements`, 10, 177
- `glEnable`, 7–8, 12–15, 25, 26, 27, 110, 111, 217, 236, 366
- `glEnableClientState`, 8, 324
- `glEnableVertexArrayPointer`, 160
- `glEnableVertexAttribArrayARB`
 - application code example, 324–325
 - defined, 177
 - overview, 57, 199
 - syntax, 451–452
- `glEnd`, 10, 38, 40, 173, 179
- `glEndList`, 11
- `GL_EYE_LINEAR`, 229–230
- `gl_EyePlaneS/T/R/Q[i]`, 107, 230
- `GL_FILL`, 111, 366
- `glFinish`, 17
- `glFlush`, 17
- `glFog`, 16, 99
- `gl_Fog`, 49, 107, 108, 226–228
- `gl_FogCoord`, 96, 99, 173, 180, 226
- `gl_FogFragCoord`, 99, 101, 112, 226–228
- `gl_FogParameters`, 107
- `gl_FragColor`
 - defined, 103–104
 - overview, 44, 49
 - shader code example, 65, 155–156, 241, 247, 250, 261, 270, 277, 279, 287, 305–308, 310, 318, 327, 332, 344, 349, 351, 364, 369, 375, 386–388, 397–398, 401

- use in debugging, 207–208
- gl_FragCoord*, 43, 102–104
- gl_FragDepth*, 44, 49, 103–104
- GL_FRAGMENT_SHADER_ARB, 161, 164–166, 169, 195, 197
- gl_FrontColor*, 99, 101, 110, 113, 224–225
- glFrontFace**, 15, 103
- gl_FrontFacing*, 43, 102–103, 110
- gl_FrontLightModelProduct*, 107, 224
- gl_FrontMaterial*, 106, 108, 218, 220, 222, 224
- gl_FrontSecondaryColor*, 99, 101, 110, 113, 224–225
- glFrustum**, 24
- glGet**, 8, 13, 26, 94, 96, 100, 102, 109, 114, 192, 194
- glGetActiveAttribARB**
 - defined, 181–183
 - overview, 57
 - syntax, 453–455
- glGetActiveUniformARB**
 - defined, 189–191
 - overview, 56
 - syntax, 456–459
- glGetAttachedObjectsARB**
 - defined, 172–173
 - overview, 56
 - syntax, 460–461
- glGetAttribLocationARB**
 - defined, 178–179
 - overview, 160, 199
 - syntax, 462–463
- glGetClipPlane**, 8
- glGetHandleARB**
 - defined, 172
 - overview, 56
 - syntax, 464
- glGetInfoLogARB**
 - application code example, 172
 - defined, 171–172
 - information log for compilation, 162–163
 - information log for linking, 166
 - overview, 56
 - syntax, 465–466
- glGetLight**, 8
- glGetMaterial**, 8
- glGetObjectParameterfv/ivARB**
 - application code example, 172, 196–198
 - defined, 168–170
 - overview, 56
 - querying compilation status, 162–163
 - querying deletion status, 167
 - querying length of longest attribute variable name, 182
 - querying length of longest uniform variable name, 190
 - querying link status, 164, 166
 - querying number of active attribute variables, 182
 - querying number of active uniform variables, 190
 - querying number of attached shaders, 173
 - querying shader source string length, 170
 - querying validation status, 193
 - syntax, 467–470
 - uniform variables, getting number of, 190
- glGetShaderSourceARB**
 - defined, 170
 - overview, 56
 - syntax, 471–472
- glGetString**, 3, 19, 159
- glGetUniformARB**
 - obtaining the location of a uniform variable, 186
 - overview, 56
 - syntax, 464
- glGetUniformLocationARB**
 - application code example, 188–189, 195, 240, 244, 325
 - defined, 185–186
 - purpose, 56
 - querying link results, 165
 - overview, 40, 57, 160, 199
 - syntax, 475–476
- glGetVertexAttribfv/iv/dvARB**
 - defined, 183–184
 - overview, 57
 - syntax 477–479
- glGetVertexAttribPointervARB**
 - defined, 184
 - overview, 57
 - syntax 480–481
- glInterleavedArrays**, 10–11
- glLight**, 12–13, 24
- gl_LightModel*, 107, 224
- glLightModel**, 12–13
- gl_LightModelParameters*, 106–107
- gl_LightModelProducts*, 107
- gl_LightSource[i]*, 49, 97, 106, 108, 218–223, 225
- gl_LightSourceParameters*, 106
- GL_LINE_SMOOTH, 15
- GL_LINEAR, 239, 302, 384
- glLineStipple**, 15
- glLineWidth**, 15, 366
- glLinkProgramARB**
 - application code example, 196–198
 - blocking until compilation is complete, 163
 - defined, 163–167
 - determining active attribute variables, 182
 - determining active uniform variables, 190
 - effect on attribute bindings, 179, 324
 - overview, 53, 56, 62, 160, 199
 - syntax, 482–485
- glLoadIdentity**, 23
- glLoadMatrix**, 12, 23
- glLogicOp**, 16
- glMapBuffer**, 11
- glMaterial**, 12–13
- gl_MaterialParameters*, 106
- glMatrixMode**, 12, 23–24, 30

- GL_MAX_COMBINED_TEXTURE_IMAGE_UNITS_ARB, 114–115, 194
- GL_MAX_FRAGMENT_UNIFORM_COMPONENTS_ARB, 102, 194
- GL_MAX_TEXTURE_COORDS_ARB, 115, 194
- GL_MAX_TEXTURE_IMAGE_UNITS_ARB, 114–115, 192, 194
- GL_MAX_TEXTURE_UNITS, 26, 114–115
- GL_MAX_VARYING_FLOATS_ARB, 100, 194
- GL_MAX_VERTEX_ATTRIBS_ARB, 96, 194
- GL_MAX_VERTEX_TEXTURE_IMAGE_UNITS_ARB, 114–115, 192, 194
- GL_MAX_VERTEX_UNIFORM_COMPONENTS_ARB, 97, 194
- gl_MaxClipPlanes*, 105, 109
- gl_MaxCombinedTextureImageUnits*, 109
- gl_MaxFragmentUniformComponents*, 109
- gl_MaxLights*, 106–107, 109
- gl_MaxTextureCoords*, 96, 99, 101, 105, 107, 109
- gl_MaxTextureImageUnits*, 107, 109
- gl_MaxTextureUnits*, 109
- gl_MaxVaryingFloats*, 109
- gl_MaxVertexAttribs*, 109
- gl_MaxVertexTextureImageUnits*, 109
- gl_MaxVertexUniformComponents*, 109
- gl_ModelViewMatrix*
defined, 97, 105
example shader code, 112, 146–147, 151, 216, 231, 240, 245, 249, 268, 273, 285, 304, 320, 357, 368, 373
overview, 49
- gl_ModelViewProjectionMatrix*
defined, 105
example shader code, 64, 108, 215, 320, 327
- GL_MODULATE, 231–232
- glMultiDrawArrays**, 10, 12, 177
- glMultiDrawElements**, 10, 177
- gl_MultiTexCoord*
built-in attribute variables, 180
defined, 96, 173
example shader code, 217, 241, 245, 269, 285, 320, 358, 374
- glMultiTexCoord**, 29–30, 180
- glMultMatrix**, 12, 23
- GL_NEAREST, 384, 396
- glNewList**, 11
- gl_Normal*
built-in attribute variables, 179–180
defined, 77, 96, 173
example shader code, 108, 146–147, 151, 156, 216, 240, 245, 249, 255, 258, 268, 285, 304, 320, 357, 368, 373
overview, 39
- glNormal**, 10, 38, 95, 173, 175, 179–180, 318
- GL_NORMAL_MAP, 229
- gl_NormalMatrix*
defined, 105, 108
example shader code, 108, 146–148, 151, 216, 240, 245, 249, 268, 285, 304, 320, 357, 368, 373
- glNormalPointer**, 10–11
- gl_NormalScale*, 105, 217
- GL_OBJECT_ACTIVE_ATTRIBUTE_MAX_LENGTH_ARB, 170, 182
- GL_OBJECT_ACTIVE_ATTRIBUTES_ARB, 169, 182
- GL_OBJECT_ACTIVE_UNIFORM_MAX_LENGTH_ARB, 170, 190
- GL_OBJECT_ACTIVE_UNIFORMS_ARB, 170, 190
- GL_OBJECT_ATTACHED_OBJECTS_ARB, 169, 173
- GL_OBJECT_COMPILE_STATUS_ARB, 162, 169, 196, 198
- GL_OBJECT_DELETE_STATUS_ARB, 167, 169
- GL_OBJECT_INFO_LOG_LENGTH_ARB, 169, 171, 172
- GL_OBJECT_LINEAR, 229–230
- GL_OBJECT_LINK_STATUS_ARB, 164, 169, 197–198
- GL_OBJECT_SHADER_SOURCE_LENGTH_ARB, 170
- GL_OBJECT_SUBTYPE_ARB, 161, 169
- GL_OBJECT_TYPE_ARB, 161, 169
- GL_OBJECT_VALIDATE_STATUS_ARB, 169
- gl_ObjectPlaneS/T/R/Q[i]*, 107, 230
- glOrtho**, 24
- Gloss map, 242–243, 518
- glPixelMap**, 19
- glPixelStore**, 18, 20
- glPixelTransfer**, 19
- glPixelZoom**, 19
- gl_Point*, 106
- GL_POINT, 111
- GL_POINTS, 325
- GL_POINT_SMOOTH, 15
- glPointParameter**, 15
- gl_PointParameters*, 105
- glPointSize**, 15, 111, 324
- gl_PointSize*, 40, 98, 111
- glPolygonMode**, 15, 366
- glPolygonOffset**, 15
- GL_POLYGON_SMOOTH, 15
- glPolygonStipple**, 15
- glPopAttrib**, 8
- glPopClientAttrib**, 8
- glPopMatrix**, 24
- gl_Position*
defined, 76, 97–98
example shader code, 64–65, 108, 146, 151, 156, 215–216, 241, 245, 249, 257–258, 269, 273, 285, 304, 320, 326–327, 357–358, 368, 374
invariance, 113, 130–131
overview, 40, 49
use in clipping, 112
use in computing raster position, 112
use in debugging, 207
- gl_ProjectionMatrix*, 105
- glPushAttrib**, 8

- glPushClientAttrib**, 8
- glPushMatrix**, 23
- glRasterPos**, 19, 112–113
- glReadBuffer**, 20
- glReadPixels**, 18, 20
- GL_REDUCE, 396
- GL_REFLECTION_MAP, 229
- GL_REPEAT, 239, 251, 299, 301–302
- GL_REPLACE, 231–232
- GL_REPLICATE_BORDER, 396
- glRotate**, 12, 23
- glScale**, 12, 23
- glScissor**, 16
- gl_SecondaryColor*, 96, 99, 101, 173, 180, 225
- glSecondaryColor**, 13, 16, 180**glShadeMode**, 15
- GL_SHADER_OBJECT_ARB, 161, 169
- glShaderSourceARB**
 - application code example, 196–198
 - defined, 161
 - overview, 52, 55, 61, 160, 199
 - querying results of, 170
 - syntax, 486–487
- GL_SPHERE_MAP, 229
- glStencilFunc**, 16
- glStencilMask**, 17
- glStencilOp**, 16
- gl_TexCoord[i]*
 - defined, 71–72, 99, 101
 - example shader code, 217, 230–233, 240–241, 269, 279, 285, 331–332, 388, 390, 397, 398, 401
 - use in computing raster position, 113
- glTexCoord**, 10, 29–30, 95
- glTexCoordPointer**, 29–30
- glTexEnv**, 29–30, 236
- glTexGen**, 27, 30
- glTexImage1D/2D/3D**, 18–19, 27, 236, 239
- glTexParameter**, 27–29, 236, 238, 239, 301–302
- glTexSubImage1D/2D/3D**, 18–19, 27
- gl_TextureEnvColor[i]*, 107, 232
- gl_TextureMatrix[i]*, 105, 108, 217
- GL_TEXTURE_WRAP_S, 239, 301, 396
- GL_TEXTURE_WRAP_T, 239, 302, 396
- GL_TEXTURE_WRAP_R, 302
- glTranslate**, 12, 23
- gluLookAt**, 23
- glUniformARB**
 - application code example, 189, 197–199, 240, 244
 - defined, 186–189
 - obtaining location of a uniform variable, 186
 - overview, 40, 56, 160, 199
 - syntax, 488–494
 - use with samplers, 192
- glUniformMatrixARB**
 - defined, 188
 - syntax, 490–494
- glUnmapBuffer**, 11
- gluPerspective**, 24
- glUseProgramObjectARB**
 - application code example, 197–198, 366
 - defined, 166
 - effect on processing vertex attributes, 173
 - implicitly called as a side-effect of linking, 164
 - overview, 54, 56, 62, 160, 199
 - syntax, 495–497
- glValidateProgramARB**
 - defined, 193
 - overview, 56
 - syntax, 498–499
- glVertex**, 10, 38, 96, 112, 173, 176, 179–180, 318
- gl_Vertex*
 - built-in attribute variables, 179–181
 - defined, 77, 96, 173
 - differences between vertex shaders and fragment shaders, 156
 - example shader code, 64, 108, 112, 113, 146–151, 215–216, 230, 231, 240, 245, 249, 255, 258, 268, 273, 285, 304, 320, 325–327, 357, 368, 373
 - invariant transformation of, 131
 - overview, 39
- glVertexAttribARB**
 - application code example, 179, 255–256
 - defined, 174–176
 - overview, 39, 57, 160, 199
 - signal the end of a vertex, 96
 - specifying generic attributes, 179–181
 - syntax, 500–504
- GL_VERTEX_ATTRIB_ARRAY_ENABLED_ARB, 183
- GL_VERTEX_ATTRIB_ARRAY_NORMALIZED_ARB, 184
- GL_VERTEX_ATTRIB_ARRAY_POINTER_ARB, 184
- GL_VERTEX_ATTRIB_ARRAY_SIZE_ARB, 183
- GL_VERTEX_ATTRIB_ARRAY_STRIDE_ARB, 184
- GL_VERTEX_ATTRIB_ARRAY_TYPE_ARB, 184
- GL_VERTEX_PROGRAM_POINT_SIZE_ARB, 111
- GL_VERTEX_PROGRAM_TWO_SIDE_ARB, 110
- GL_VERTEX_SHADER_ARB, 161, 164–166, 169, 195, 197
- glVertexAttribPointerARB**
 - application code example, 324–325
 - defined, 176
 - overview, 57, 160, 199
 - syntax, 505–507
- glVertexPointer**, 10–11, 324
- glViewport**, 14, 25
- glWindowPos**, 19
- Gooch, Amy, 365
- Gooch, Bruce, 365
- Gooch shading
 - characteristics of, 365
 - color computations, 367
 - defined, 365, 518

- Gooch shading (*continued*)
 diffuse reflection factor, 367
 edges, generating, 365–366
 further reading, 378–379
 highlights, 366
 history of, 365
 luminance value range, 366–367
 technical illustration example, 364–369
- goto**, 80
- Gouraud shading. *See* smooth shading.
- Gradient, 343, 359–360, 518–519
- Gradient noise, 297, 519. *See also* Perlin noise
- Gradient vector, 342, 519
- Grammar, language, 421–433
- Granite shader, 308, Color Plate 20
- Graphics accelerator, 5, 519
- Graphics context, 5, 8, 519
- Graphics processing pipeline, 8–9, 519
- Graphics shading languages. *See* OpenGL Shading Language.
- Greater than or equal (\geq), relational operators
 Boolean greater than or equal, 86
 order of precedence, 83
 preprocessor expression, 89
- Greater than sign ($>$), relational operators
 Boolean greater than, 86
 order of precedence, 83
 preprocessor expression, 89
- Greater than signs ($>>$) bit-wise shift, preprocessor, 89
- greaterThan**, 133
- greaterThanEqual**, 134
- Gruschel, Jens, 389
- ## H
- Haeberli, Paul, 385
- Hand-drawn effects. *See* NPR (non-photorealistic rendering).
- Hanrahan, Pat, 405
- Hard light blend mode, 392–393, Color Plate 30
- Hash-based operator, 90
- Hatching shader, 356–364
- HDTV standard, color conversions, 383, 385
- Hermite interpolation functions, 124
- High-Level Shader Language (HLSL), 412–414
- Highlights, Gooch shading, 366
- HLSL (High-Level Shader Language), 412–414
- Homogeneous vertex position, calculating, 40, 76, 146, 150–151
- House exterior, Color Plate 7
- HP logo, Color Plate 10
- ## I
- #if**, 87
- if-else** statement, 80
- if** statement, 80
- #ifdef**, 87
- #ifndef**, 87
- Image filtering, 394. *See also* convolution.
- Images
 adjusting brightness, 386
 adjusting contrast, 386–387
 adjusting saturation, 387
 adjusting sharpness, 387–388
 blending, 388–394
 combining. *See* composite images.
 edge detection, 400
 sharpening, 400–402
 size requirements, 396
 smoothing, 396–400
 subset support, querying, 19
 warping, 382
- Images, drawing. *See also* primitives, drawing.
 biasing attributes, setting, 19
 copying, 18–19
 extensions supported, querying, 19
 frame buffer reads, enabling, 19
 image subset support, querying, 19
 imaging subset, 19
 lookup table, specifying, 19
 modeling, 20
 pixel packing, 19
 pixel rectangles, 18
 pixel transfer, 18
 pixel unpacking, 18–19
 raster position, 19
 rasterization, enabling, 19
 read control, 19
 rectangles, rendering, 18
 scaling attributes, setting, 19
 storage values, specifying, 18, 20
 texture memory, specifying images for, 18
 transferring, 18–19
 zoom factor, 19
- Imaginary numbers, 369. *See also* Mandelbrot set.
- Imaging shaders. *See* shaders, imaging.
- Imaging subset, 19, 519
- Immediate mode, 11, 519
- Imperfections, hatching example, 361–362
- Imperfections in surfaces. *See* noise.
- Implementation-dependent API values, 194
- in**, 81–82
- #include** directives, 51
- Incoming values, mapping by sign, 124–125
- Increment operator
 matrix increment, 86
 order of precedence, 83
 vector increment, 86
- Index operator
 arrays, indexing, 83–84
 matrices, indexing, 83–84
 order of precedence, 83
 vectors, indexing, 83–84
- Indexing, 83–84
- information logs, 56, 88, 162–163, 165–166, 169, 171–172, 193, 196, 206

- Information log
 - application code example, 172, 196
 - getting, 171–172
 - length, 169
 - preprocessor #error message, 88
 - overview, 56
 - result of compiling, 162–163
 - result of linking, 165–166
 - result of validating, 193
 - use in debugging shaders, 206
- Initializers, 73–75
- inout**, 81–82
- Input values, mapping to multiple output values, 383–384
- Installing shaders, 54, 163–167
- Integers
 - OpenGL Shading Language support for, 51
 - scalar data types, 66–67
- Intensity-to-RGB lookup, 384
- Interpolation shaders for imaging
 - about, 385–386
 - brightness, 386, Color Plate 26
 - contrast, 386–387, Color Plate 27
 - sharpness, 387–388, Color Plate 29
- Inverse difference blend mode, 393, Color Plate 30
- inversesqrt**, 120–121
- Iris GL, 1–2
- Irregular surfaces, 280–288
- ISL (OpenGL Shader), 409–411
- Isotropic, 519
- Iterative testing, 203
- ivec*n*** data type, 67–68

- J**
- Jigsaw, Color Plate 4
- Julia sets, 376–377

- K**
- Kessenich, John, 299
- Key-frame interpolation, 318–320, 519

- L**
- L-value, 84, 519
- Lander, Jeff, 366
- Language comparisons. *See* OpenGL Shading Language.
- Language grammar, 421–433
- Laplacian operator, 400–401
- Lattice shader, 279–280, Color Plate 15
- length**, 130
- Length of vector, determining, 130
- Less than or equal (<=), relational operators
 - Boolean less than or equal, 86
 - order of precedence, 83
 - preprocessor expression, 89
- Less than sign (<), relational operators
 - Boolean less than, 86
 - order of precedence, 83
 - Less than signs (<<) bit-wise shift, preprocessor, 89
- lessThan**, 133
- lessThanEqual**, 86, 133
- Level-of-detail, 28–29, 135, 238, 261, 520
- Lexical analysis, 54, 211, 520
- Licea-Kane, Bill, 271
- Light sources
 - ambient lighting value, modifying, 13
 - attributes, 12–13
 - colors, setting, 13
 - coordinate transforms, 24
 - directional lights, 217–219
 - enabling/disabling, 12–13
 - global parameters, 13
 - number supported, querying, 12–13
 - point lights, 219–220
 - position, specifying, 24
 - primitives, 12–13
 - spotlights, 220–222
 - type, setting, 13
- Lighten blend mode, 390–391, Color Plate 30
- Lighting
 - brick pattern, 146–147
 - errors, 284
 - Gooch shading, 366
 - hatching example, 361
 - inconsistent tangents, 284
 - and material properties, 222–224
 - none, 225
 - two-sided, 224–225
- #line**, 88–89
- __LINE__** macro, 88
- Linear blend functions, 124
- Lines, 15
- Linker, 54–55
- Linking, 53–54, 164–166
- Liquid spray effect, 321
- Lod** (level of detail) suffix, 135–136
- log2**, 120–121
- Logical and operator
 - Boolean logical and, 86
 - order of precedence, 83
 - preprocessor expression, 89
- Logical exclusive or operator
 - Boolean logical or, 86
 - order of precedence, 83
- Logical inclusive or operator
 - Boolean logical inclusive, 86
 - order of precedence, 83
 - preprocessor expression, 89
- Logical not operator
 - Boolean logical not, 86
 - order of precedence, 83
 - preprocessor expression, 89
- Logical operations, 16
- Logos
 - 3Dlabs, Color Plate 21
 - HP, Color Plate 10



Lookup table operations, 383–384
 Looping. *See* flow-control.
 Low dynamic range artistic tone algorithm, 365. *See also* Gooch shading.
 Low-pass filtering, 342, 520
 Luminance value, computing, 383
 Luminance value range, 366–367

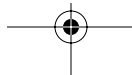
M

Macros, 88
Making Noises, 293
 Mandelbrot set, 369–379
 Marble shader, 307–308, Color Plate 20
 Masking, 6
 Matching data types, 73
 Material properties, 12–13, 222–224
 Mathematical mappings, 383
matn data type, 48, 67–68
 Matrices, 68–69
 Matrix functions, 132–133
 Matrix multiplication functions, 132–133
matrixcompmult, 132–133
max, 123, 127–128
 Maximum component functions, 123
 Microsoft High-Level Shader Language (HLSL), 412–414
min, 123, 127–128
 Minimum component functions, 123
 Minus sign (-), negation operator
 matrix negation, 86
 order of precedence, 83
 preprocessor expression, 89
 vector negation, 86
 Minus sign (-), subtraction operator
 matrix subtraction, 86
 order of precedence, 83
 preprocessor expression, 89
 vector subtraction, 86
 Minus sign equal sign (=), assignment operator, 83, 87
 Minus signs (--), decrement operator
 matrix decrement, 86
 order of precedence, 83
 scalar decrement, 86
 vector decrement, 86
 Mipmap level, 28, 520
 Mipmap texture, 28–29, 40, 135, 237–238, 261, 266, 301, 337, 520
 Mitchell, Jason, 365
mix, 124, 337–338, 392–393
mod, 122, 127
 Model space, 21, 520
 Model transformation matrix, 22, 520
 Modeling, 20, 520
 Modeling coordinate system, 21, 520
 Modeling transformation, 22, 520

Modelview matrix
 defined, 22–23, 520
 loading, 23
 multiplying, 23
 replacing, 23
 selecting, 23
 stack manipulation, 23–24
 values, modifying, 12
 viewing transformation, specifying, 23
 Modelview-projection matrix, 521
 Modes. *See* state settings.
 Modulus functions, 122
 Morphing. *See* key-frame interpolation.
 Molten lava effect, 306–307
 Motion blur, 327
 Multiple textures, 241–247
 Multiplication operator
 matrix/matrix multiplication, 85–86
 order of precedence, 83
 preprocessor expression, 89
 vector/matrix multiplication, 85–86
 Multiply blend mode, 391, Color Plate 30
 Multisample buffer, 7, 521

N

Named attribute variables, 173–184, 438–440. *See also* attribute variables.
 Negation operator
 matrix negation, 86
 order of precedence, 83
 preprocessor expression, 89
 vector negation, 86
 Negative numbers, square roots of, 370. *See also* Mandelbrot set.
 Negative results, preventing, 124
 Neighborhood averaging, 396–397, 521
 Newell, Martin, 21
 Noise
 1D noise function, 294–297
 2D noise function, 297–298
 3D and 4D noise functions, 298–299
 built-in **noise** function, 139–140, 302–303
 defined, 291–299, 521
 gradient noise, 297, 519
 granite, 308, Color Plate 20
 hatching example, 362
 marble, 307–308, Color Plate 20
 molten lava, 306–307
 octaves, 293–297, 521
 Perlin noise, 297, 522
 shader for, 303–306, Color Plate 20
 sun surface, 306–307, Color Plate 20
 from texture maps, 302–303
 textures, 299–302
 turbulence, 306–308, 526
 value noise, 293–297, 526
 wood, 308–312, Color Plate 22



- noise**, 139–140, 302–303
- Noise functions, 139–140
- Noise reduction, convolution, 397–398
- noisier**, 139–140
- Non-photorealistic rendering (NPR). *See* NPR (non-photorealistic rendering).
- Normal blend mode, 389, Color Plate 30
- Normal map, 288, 521, Color Plate 19
- normalize**, 130–131
- Normalized device coordinate space, 25, 521
- not**, 86, 135
- notEqual**, 86, 134
- NPR (non-photorealistic rendering)
 - defined, 355, 521
 - Gooch shader, 364–369
 - further reading, 378–379
 - hatching shader, 356–364
 - Julia set shader, 376–377
 - Mandelbrot set shader, 369–377
 - technical illustration shader, 364–369
- NVIDIA Cg, 415–416
- O**
- Object coordinate system, 21, 521. *See* modeling coordinate system.
- Object space, 21, 521. *See also* modeling coordinate system.
- Octaves, 293–297, 521
- Offscreen memory, 5, 521
- Olano, Marc, 406
- 1D noise function, 294–297
- 1D texture, 26, 383–384, 513
- 1D texture maps, 383–384
- Opacity blend mode, 394, Color Plate 30
- OpenGL
 - basic architecture, 8–9
 - evolution, 3–4
 - execution model, 4
 - fixed functionality, 8–9
 - history, 1–2
- OpenGL-managed objects, deleting, 447–448
- OpenGL Shader (ISL), 409–411
- OpenGL shaders. *See* shaders, OpenGL.
- OpenGL Shading Language
 - absent qualifier, 79
 - arrays, 71–72
 - attribute qualifier, 77
 - benefits, 58–60
 - C additions, 48–50
 - C basis, 48
 - C features not supported, 50–51
 - call by value-return, 51
 - calling conventions, 81–82
 - characters, 54, 72–73
 - chronology of, 405–406
 - compared to Cg, 415–416
 - compared to HLSL, 412–414
 - compared to OpenGL Shader (ISL), 409–411
 - compared to RenderMan, 406–409
 - compiler, 54–55
 - compiler front-end, 211–212
 - compiling source code, 52, 162–163
 - component-wise operation, 85–87
 - constant qualifier, 78–79
 - constructors, 50, 73–75
 - data structures. *See* shader objects.
 - data types, 65–73
 - declarations, 72–73
 - defined, 33, 521
 - design considerations, 45–47
 - design goals, 46–47
 - driver model, 52–54
 - error handling, 90
 - floating-point matrix types, 48
 - floating-point values, 48
 - flow control, 79–82
 - functions, built-in, 82. *See also* Functions, built-in
 - functions, user-defined, 80–81
 - further reading, 417–418
 - indexing, 83–84
 - initializers, 73–75
 - inputs and outputs, 49
 - installing executables, 54, 166–167
 - interface to a shader, 76–78
 - introduction, 33–35
 - language comparisons, 405–418
 - linker, 54–55
 - linking source code, 53–54, 163–166
 - matrices, 68–69
 - modifying. *See* extensions.
 - operations, 83–87
 - overloading functions, 50
 - overview, 45–51
 - preprocessor, 87–89
 - preprocessor expressions, 89–90
 - promotion, 73
 - qualifiers, 76–79
 - samplers, 49, 69–70
 - scalars, 66–67
 - scope, 72–73
 - shader objects, 52
 - state access, 49
 - structures, 70–71
 - summary, 90
 - swizzling, 84–85
 - system overview, 52–58
 - type-matching, 73
 - type conversions, 75
 - using a shader, 166–167
 - uniform qualifier, 77
 - unions, 51
 - variable declaration, 50
 - varying qualifier, 77
 - vectors, 67–68
 - void, 72

- OpenGL Shading Language API
 - ARB_fragment_shader*, 57–58, 159
 - ARB_shader_objects*, 55–56, 159
 - ARB_vertex_shader*, 56–57, 159
 - defined, 159, 522
 - description, 159–194, 435–507
 - extensions, 55–58
- Operations
 - component-wise, 85–87
 - indexing, 83–84
 - operator order of precedence, 83
 - rearranging components. *See* swizzling.
 - swizzling, 84–85
- Operators
 - , (comma) sequence operator, 83, 87
 - . (period) selection operator
 - components, selecting, 84–85
 - order of precedence, 83
 - swizzling, 84–85
 - . (period) swizzle operator
 - components, rearranging, 84–85
 - order of precedence, 83
 - ?: (question mark colon) selection operator, 83, 87
 - ?: (question mark colon) ternary selection operator, 83, 87
 - (minus signs) decrement operator
 - matrix decrement, 86
 - order of precedence, 83
 - scalar decrement, 86
 - vector decrement, 86
 - / (forward slash) division operator
 - matrix division, 86
 - order of precedence, 83
 - preprocessor expression, 89
 - scalar division, 86
 - vector division, 86
 - /= (forward slash equal sign) assignment operator, 83, 87
 - = (minus sign equal sign) assignment operator, 83, 87
 - (minus sign) negation operator
 - matrix negation, 86
 - order of precedence, 83
 - preprocessor expression, 89
 - scalar negation, 86
 - vector negation, 86
 - (minus sign) subtraction operator
 - matrix subtraction, 86
 - order of precedence, 83
 - preprocessor expression, 89
 - scalar subtraction, 86
 - vector subtraction, 86
 - [] (square brackets) index operator
 - arrays, indexing, 83–84
 - matrices, indexing, 83–84
 - order of precedence, 83
 - vectors, indexing, 83–84
 - & (ampersand) bit-wise expression, preprocessor, 83
 - && (ampersands) logical and operator
 - Boolean logical and, 86
 - order of precedence, 83
 - preprocessor expression, 89
 - *= (asterisk equal sign) assignment operator, 83, 87
 - * (asterisk) multiplication, 132–133
 - matrix/matrix multiplication, 132–133
 - * (asterisk) multiplication operator
 - matrix/matrix multiplication, 85–86
 - order of precedence, 83
 - preprocessor expression, 89
 - vector/matrix/scalar multiplication, 85–86
 - | (bar) bit-wise expression, preprocessor, 83
 - || (bars) logical inclusive or operator
 - Boolean logical inclusive, 86
 - order of precedence, 83
 - preprocessor expression, 89
 - ^ (caret) bit-wise expression, preprocessor, 89
 - ^^ (carets) logical exclusive or operator
 - Boolean logical or, 86
 - order of precedence, 83
 - = (equal sign) assignment operator, 83, 87
 - == (equal signs) equality operator
 - Boolean equality comparison, 86
 - order of precedence, 83
 - preprocessor expression, 89
 - != (exclamation equal sign) equality operator
 - Boolean inequality comparison, 86
 - order of precedence, 83
 - preprocessor expression, 89
 - ! (exclamation mark) logical not operator
 - Boolean logical not, 86
 - order of precedence, 83
 - preprocessor expression, 89
 - >= (greater than or equal) relational operators
 - Boolean greater than or equal, 86
 - order of precedence, 83
 - preprocessor expression, 89
 - > (greater than sign) relational operators
 - Boolean greater than, 86
 - order of precedence, 83
 - preprocessor expression, 89
 - >> (greater than signs) bit-wise shift, preprocessor, 89
 - <= (less than or equal) relational operators
 - Boolean less than or equal, 86
 - order of precedence, 83
 - preprocessor expression, 89
 - < (less than sign) relational operators
 - Boolean less than, 86
 - order of precedence, 83
 - << (less than signs) bit-wise shift, preprocessor, 89

- % (percent)
 - preprocessor expression, 89
- + (plus sign) addition operator
 - matrix addition, 86
 - order of precedence, 83
 - preprocessor expression, 89
 - scalar addition, 86
 - vector addition, 86
- += (plus sign equal sign) assignment operator, 83, 87
- ++ (plus signs) increment operator
 - matrix increment, 86
 - order of precedence, 83
 - scalar increment, 86
 - vector increment, 86
- # (pound sign), hash-based operator, 90
- ## (pound signs), hash-based operator, 90
- ~ (tilde) unary preprocessor expression, 89
- #define**, 87
- #elif**, 87
- #else**, 87
- #endif**, 87
- #error**, 88
- #if**, 87
- #ifdef**, 87
- #ifndef**, 87
- #line**, 88–89
- #pragma**, 88
- preprocessor expressions, 89–90
- preprocessor support, 87
- sizeof**, 90
- #undef**, 87
- out**, 81–82
- Overlay blend mode, 392, Color Plate 30
- Overloading functions, 50, 140
- P**
- Papers of interest, 529–542
- Parallel projection, setting, 24
- Parsing, 522
- Particle color change, 327
- Particle lifespan, 327
- Particle system shader, 321–328, Color Plate 24
- Particle systems, 321–322, 522
- Particle Systems—A Technique for Modeling...Fuzzy Objects*, 321
- Peachey, Darwyn, 293
- Pen and ink effects. *See* NPR (non-photorealistic rendering).
- Per-fragment operations, 16, 522
- Per-texture-object level-of-detail bias, 28–29
- Per-texture-unit level-of-detail bias, 29
- Per-vertex operations, 12–13
- Percent (%), preprocessor expression, 89
- Performance, 11, 204–206
- Period (.), selection operator
 - components, selecting, 84–85
 - order of precedence, 83
 - swizzling, 84–85
- Period (.), swizzle operator
 - components, rearranging, 84–85
 - order of precedence, 83
- Perlin, Ken, 139, 291–292, 405
- Perlin noise, 297, 522
- Perspective projection, 14, 24
- Photorealism, 355, 522
- Pixar RenderMan, 406–409
- Pixel-by-pixel blending, 388–394
- Pixel group, 18, 522
- Pixel ownership test, 16, 522
- Pixel packing, 19, 522
- Pixel rectangle, 18, 522. *See also* images, drawing.
- Pixel transfer, 18, 522
- Pixel unpacking, 18–19, 522
- Pixel zoom, 382
- PixelFlow architecture, 405–406
- Plus sign (+), addition operator
 - matrix addition, 86
 - order of precedence, 83
 - preprocessor expression, 89
 - scalar addition, 86
 - vector addition, 86
- Plus sign equal sign (+=), assignment operator, 83, 87
- Plus signs (++), increment operator
 - matrix increment, 86
 - order of precedence, 83
 - vector increment, 86
- Point lights, 219–220
- Point primitives, size, 12, 98, 111, 324
- Point sampling, 336, 522–523
- Point size mode, 111
- Points, 15, 130, 324
- Polygons, 15
- Polynomial texture maps (PTMs), 252–262, 523
- Position invariance, 113
- Pound sign (#), hash-based operator, 90
- Pound signs (##), hash-based operator, 90
- pow**, 120–121
- #pragma**, 88
- Pre-fragment operations, 16
- Preprocessor, 87–90
- Primitive assembly, 13–14, 523
- Primitive processing, 14
- Primitives. *See also* geometric primitives.
 - defined, 4, 523
 - type, specifying, 13–14
- Primitives, drawing. *See also* images, drawing.
 - alpha test, 16
 - antialiasing, 15
 - antialiasing, enabling, 15
 - blending, 16
 - buffer objects, 11–12
 - buffers, setting, 17
 - client-side operations, 11–12

Primitives, drawing (*continued*)

- clipping, 14
- color sum, 16
- culling, 14
- depth test, 16
- display list, 11
- display list method, 11–12
- display list mode, 11
- dithering, 16
- enabling, 16
- finishing, 17
- flat shading, 15
- flushing, 17
- fog, 16
- fog parameters, setting, 16
- fragment processing, 15–16
- fragments, 15
- frame buffer operations, 17
- immediate mode, 11
- light source, 12–13
- lines, 15
- logical operations, 16
- material properties, 12–13
- modelview matrix, 12
- per-fragment operations, 16
- per-vertex operations, 12–13
- performance, 11
- perspective projection, 14
- pixel ownership test, 16
- points, 15
- polygons, 15
- pre-fragment operations, 16
- primitive type, specifying, 13–14
- rasterization, 15–16
- rasterization values, specifying, 15
- rendering, 11
- scissor test, 16
- secondary colors, setting, 16
- server-side, 11–12
- smooth shading, 15
- specifying, 9–12
- state settings, 12–13
- stencil test, 16
- stipple pattern, 15
- supported by OpenGL, 9–10
- texture memory, 16
- texturing, 15–16
- transformation and lighting (T&L), 12–13
- type, specifying, 14
- vertex arrays, 10–11
- vertex arrays method, 10–11
- vertex-at-a-time method, 10, 14
- vertex attributes, 10
- vertex processing, 12
- view volume, 14
- viewport transformation, 14
- window coordinates, 14

Problem analysis, 202

Procedural texture shaders. *See* shaders, procedural texture.

Procedural texturing, 265–267, 523. *See also* antialiasing.

Processing pipeline. *See* graphics processing pipeline.

Program object, getting, 172

Program objects

- active attribute variables, querying 181–183, 453–455
- active uniform variables, querying 189–191, 456–459
- attached objects, querying 172–173, 460–461
- attaching shader objects to, 163–164, 436–437
- attribute locations, querying 178–179, 462–463
- binding attribute locations to, 177–181, 438–440
- creating, 160, 163–167, 443–444
- defined, 52, 523
- deleting, 167, 447–448
- detaching shader objects from, 168, 449–450
- handles, querying, 464
- information log, querying, 171–172, 465–466
- installing in current rendering state, 495–497
- linking, 164–166, 482–485
- parameters, querying, 168–170, 467–470
- uniform variable values, querying 473–474
- uniform variable locations, querying 185–186, 475–476
- using, 166–167, 495–497

validating, 498–499

Programmable pipeline. *See also* shaders, fragment; shaders, vertex.

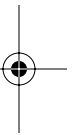
- clipping, 98, 112
- constants, 109–110
- fragment processor, 100–104
- OpenGL interaction, 110–115
- point primitives, size, 98
- point size mode, 111
- position invariance, 113
- raster position, 112
- state access, 97
- texturing, 113–115
- two-sided color mode, 110
- vertex attributes, 95–96
- vertex processor, 94–100

Programmable pipeline, variables

- special input, 102–103
- special output, 97–98, 103–104
- uniform, 97, 102, 104–108
- varying
 - built-in, 98–99
 - fragment processor, 101
 - user-defined, 100
 - vertex processor, 98–100

Programmable processors. *See also* fragment processor; shaders; vertex processor.

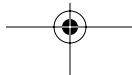
- accessing texture memory, 44



- attribute values, 38–39
 - built-in attribute values, 38–39
 - built-in uniform variables, 40
 - passing data values, 40–41
 - uniform variables, 40
 - user-defined attribute values, 38–39
 - user-defined uniform variables, 40
 - varying variables, 41
 - Proj** (projective texturing) suffix, 136
 - Projection matrix, 24, 523. *See also* projection transformation.
 - Projection transformation, 24, 523
 - Promoting data types, 73
 - PTMs (polynomial texture maps), 252–262, 523
 - Pulse train, 346, 523
- Q**
- Qualifiers. *See also* variables.
 - absent, 79
 - attribute**, 76–77, 78
 - const**, 76, 78–79
 - uniform**, 76, 77, 78
 - varying**, 76, 77–78
 - Question mark colon (:?), ternary selection operator, 83, 87
- R**
- R-value, 84, 523
 - radians**, 119
 - Radians, converting to degrees, 119
 - Range of values, ensuring, 123, 127–128
 - Raster position, 19, 112–113, 523
 - Rasterization
 - defined, 15, 523
 - of images, 19
 - imaging shaders and, 381
 - as input to imaging shaders, 381
 - point width, 15
 - primitives, 15–16
 - of primitives, 15
 - types of, 381
 - values, specifying, 15
 - Rate-of-change functions, 138–139
 - Read control, 19, 524
 - Realistic surfaces. *See* noise.
 - Reciprocal functions, 121
 - Rectangles, rendering, 18
 - Recursive functions. *See* Mandelbrot set.
 - Reeves, Bill, 321
 - reflect**, 131
 - Reflection, 148–150, 252–262
 - Reflection direction functions, 131
 - Reflection vector, 148
 - Regular patterns, 267–271
 - Rendering, 5, 11, 524. *See also* geometric primitives.
 - Rendering pipeline. *See* graphics processing pipeline.
 - RenderMan, 406–409
 - RenderMonkey, 208–211, Color Plate 1
 - Resolution, increasing, 338–339
 - return**, 80
 - RGB colors, converting to/from CIE, 384–385
 - RGBA-to-RGBA lookup, 384
 - Rocking motions, 120
- S**
- Sampler, 49, 69–70, 192, 524
 - Sampler variables, 135
 - samplerCube** data type, 69–70
 - samplernD**, 49
 - samplernD** data type, 69–70
 - samplernDShadow** data type, 69–70, 115
 - Samplers, 49, 69–70, 192, 524
 - Saturation, 387, Color Plate 28
 - Scalar data types, 65–67
 - Scale and bias operation, 383
 - Scaling, 19
 - Scanning. *See* lexical analysis.
 - Scissor test, 16, 524
 - Scope of data types, 72–73
 - Screen blend mode, 391, Color Plate 30
 - Segal, Mark, 3
 - Selection operator
 - components, selecting, 84–85
 - order of precedence, 83
 - swizzling, 84–85
 - Semantic analysis, 211, 524
 - Sequence operator, 83, 87
 - SGL OpenGL Shader (ISL), 409–411
 - Shader objects
 - attaching to program object, 160, 164–167, 436–437
 - compiling, 162–163, 441–442
 - creating, 159–162, 167–168, 445–446
 - defined, 52, 524
 - detaching from program object, 168, 449–450
 - handles, querying, 172–173, 460–461
 - information log, querying, 171–172, 465–466
 - parameters, querying, 168–170, 467–470
 - source code, specifying, 486–487
 - source code, querying, 170–171, 471–472
 - Shaders. *See also* programmable processors.
 - defined, 34, 524
 - developing. *See* developing shaders.
 - noise, 303–306, Color Plate 20
 - purpose of, 35–36
 - stored textures. *See* texture mapping.
 - Shaders, animated. *See also* programmable pipeline; shaders, fragment; shaders, imaging; shaders, vertex.
 - blending objects, 318–320
 - blinking on/off, 316
 - confetti cannon, 322–327
 - fading in/out, 320, 327
 - hatching, 357–358



- Shaders, animated (*continued*)
 key-frame interpolation, 318–320
 motion blur, 327
 particle color change, 327
 particle lifespan, 327
 particle systems, 321–328, Color Plate 24
 threshold, 317
 translation, 317–318
 wobble, 328–332, Color Plate 21
- Shaders, fragment. *See also* programmable pipeline;
 shaders, animated; shaders, imaging; shaders,
 vertex.
 brick, 151–156
 brightness, 386, Color Plate 26
 contrast, 386–387, Color Plate 27
 defined, 34, 518
 examples, 63–65, Color Plate 2, Color Plate 3,
 Color Plate 5
 general convolution, 398–400
 hatching, 362–364
 inputs and outputs, 43
 Mandelbrot set, 374–376
 neighborhood averaging convolution, 396–397
 noise reduction convolution, 397–398
 purpose of, 42
 saturation, 387, Color Plate 28
 sharpness, 387–388, 400–402, Color Plate 29
 smoothing, 396–400
 technical illustration, 368–369
 unsharp masking, 401–402
- Shaders, imaging. *See also* programmable pipeline;
 shaders, animated; shaders, fragment; shaders,
 imaging.
 blend modes, 388–394, Color Plate 30
 color space conversions, 384–385
 convolution operations, 394–402
 extrapolation operations, 385–388
 further reading, 402–403
 geometric image transformations, 382
 image combining, 388–394
 image warping, 382
 interpolation operations, 385–388
 lookup table operations, 383–384
 mathematical mappings, 383
 pixel zoom, 382
 rasterization and, 381
 scale and bias operation, 383
- Shaders, OpenGL
 attached shader objects, listing, 172–173
 brick shader source code, 194–199
 cleaning up, 167–168
 compiling, 160, 162–163
 current program object, getting, 172
 defined, 34, 521
 development aids, 193
 executables, validating, 193
 implementation-dependent API values, 194
 information log, getting, 171–172
 linking, 163–167
 overview, 159–160
 program objects, creating, 160, 163–167
 querying object state, 168–173
 samplers, 192
 shader objects
 attaching, 160, 164–167
 compiling, 162–163
 creating, 159–162, 167–168
 detaching, 168
 listing, 172–173
 source code, getting, 170
 source code buffer size, getting, 170
 uniform variables, specifying, 185–191
 using (installing), 163–167
 vertex attributes, specifying, 173–184
- Shaders, procedural texture
 advantages/disadvantages, 265–267
 ball, 271–279, Color Plate 16, Color Plate 17
 bump mapping, 280–288, Color Plate 9, Color
 Plate 18
 defined, 265–267, 523
 examples, Color Plate 12
 irregular surfaces, 280–288
 lattice, 279–280, Color Plate 15
 normal maps, 288, Color Plate 19
 regular patterns, 267–271
 sphere, 271–279
 stripes, 268–271
 surface details, 280–288
 surface-local coordinate space, 281
 tangent space, 282
 vs. stored textures, 265–267
- Shaders, traditional
 depth-cuing, 226
 fog, 226–228
 light sources, 217–222
 lighting, 222–225
 textures, 228–233
 transformation, 215–217
 user clipping, 231
- Shaders, vertex. *See also* programmable pipeline;
 shaders, animated; shaders, fragment; shaders,
 imaging.
 attribute values, 38–39
 brick, 145–151
 clipping, 112
 convolution, 395
 defined, 34, 527
 example, 63–65
 hatching, 357–358
 inputs and outputs, 39
 Mandelbrot set, 373–374
 point size mode, 111
 position invariance, 113
 purpose of, 38



- raster position, 112
 - technical illustration example, 368
 - texturing, 113–115
 - two-sided color mode, 110
 - Shading language. *See* OpenGL shading language.
 - shadownD**, 137
 - shadownDLod**, 137
 - shadownDProj**, 137
 - shadownDProjLod**, 137
 - Sharpness
 - convolution, 400–402, Color Plate 29
 - fragment shaders, 387–388, 400–402
 - interpolation operations, 387–388
 - Laplacian operator, 400–401
 - unsharp masking, 388
 - Shirley, Peter, 365
 - sign**, 122, 124–125
 - Sign of integer, determining, 122
 - Silhouette edges, 366
 - Silicon Graphics, Inc., 1–2
 - sin**, 119
 - Sine functions, 119
 - sizeof**, 90
 - Smoke effect, 321
 - Smooth shading, 15, 524
 - Smoothing, 396–400
 - smoothstep**
 - antialiasing, 337–338, 341–342, 343–345
 - example, 157
 - hatching example, 361
 - purpose, 129
 - syntax, 124
 - Soft light blend mode, 392, Color Plate 30
 - Source code, getting, 170
 - Source code buffer size, getting, 170
 - Special input variables, 102–103
 - Special output variables, 97–98, 103–104
 - Spheres
 - example, 271–279
 - Gooch shading, Color Plate 23
 - with stripe pattern, 358
 - Sphere mapping, 247, 524
 - Spotlights, 220–222
 - Spray effect, 321
 - sqrt**, 120–121
 - Square brackets ([]), index operator, 83–84
 - Square root functions, 121
 - Stanford Real-Time Shading Language, 406
 - State
 - access, 49
 - description, 7–8
 - graphics context, 7
 - query functions, 168–173
 - querying, 168–173
 - State settings
 - client-side, 8
 - server-side, 7–8
 - values, modifying, 12–13
 - values, querying, 8
 - Stencil buffer, 6, 524–525
 - Stencil test, 16, 525
 - step**
 - antialiasing, 341, 346
 - description, 129
 - example, 153–154, 157
 - hatching example, 358, 361
 - purpose, 129
 - syntax, 124, 129
 - Stereo viewing, 6
 - Stipple pattern, 15
 - Stochastic functions, 139–140
 - Storage values, specifying, 18, 20
 - Stripes
 - antialiasing, 339–349
 - examples, 268–271, Color Plate 13, Color Plate 14
 - Structures, data types, 70–71
 - Subtract blend mode, 393, Color Plate 30
 - Subtraction operator
 - matrix subtraction, 86
 - order of precedence, 83
 - preprocessor expression, 89
 - scalar subtraction, 86
 - vector subtraction, 86
 - Sun surface effect, 306–307, Color Plate 20
 - Supersampling, 338–339, 525
 - Surface details, 280–288
 - Surface-local coordinate space, 281, 525
 - switch**, 80
 - Swizzle, 84–85, 525
 - Swizzle operator
 - components, rearranging, 84–85
 - order of precedence, 83
 - Swizzling, 84–85, 525
 - Syntactic analysis, 211, 525
- ## T
- tan**, 119
 - Tangent functions, 119
 - Tangent space, 282, 525
 - Teapots
 - Gooch shading, Color Plate 23
 - hatching example, 363
 - noise shaders, Color Plate 20
 - wood shaded, Color Plate 22
 - Technical illustration example, 364–369
 - Tejada, Antonio, 328
 - Temporal aliasing, 337, 525
 - Ternary selection operator, 83, 87
 - Testing
 - alpha test, 16
 - depth test, 16
 - developing shaders, 203
 - iterative testing, 203



- Testing (*continued*)
 pixel ownership test, 16
 scissor test, 16
 stencil test, 16
- Texel, 28, 525
- Texture mapping. *See also* textures; texturing.
 accessing from shaders, 236–238
 anisotropic reflection, 252–262
 BRDF (bidirectional reflectance distribution function), 252–262, Color Plate 11
 bump mapping, 280–288, Color Plate 9, Color Plate 18
 creating noise, 302–303
 defined, 16, 526
 examples
 conventional vs. polynomial, Color Plate 10
 earth's surface, cloud cover, 242–243
 earth's surface, cloudless, 239, Color Plate 2, Color Plate 5
 earth's surface, daytime/nighttime, 241–247
 earth's surface, texture mapped, Color Plate 3, Color Plate 6
 environment mapping, 247–251, Color Plates 7, Color Plates 8, Color Plates 9
 gloss map, 242–243
 mirror surface, Color Plate 9
 multiple textures, 241–247
 simple texture, 238–241
 sphere mapping, 247–251
 texture on a sphere, 241
 two-dimensional map, 239
 hardware units, defining, 194
 normal maps, 288, Color Plate 19
 overview, 235–236
 PTMs (polynomial texture maps), 252–262
 reflection, 252–262
 stored textures vs. procedural texture shaders, 265–267
- Texture matrix stack, 30
- Texture memory, 16, 526
- Texture objects, 27–29, 526
- Texture unit, 26, 526
- textureCube**, 137
- texturenD**, 136–137
- texturenDLod**, 136–137
- texturenDProj**, 136–137
- texturenDProjLod**, 136–137
- Textures. *See also* texture mapping; texturing.
 access, 30, 525
 access functions, 135–138
 application, 30, 231–233, 525
 bump maps, 280–288, 515
 for complex functions, 206
 control, 267, 515
 coordinate generation, 228–231
 coordinate sets, defining, 194
 coordinates, 29–30
 environment, 28–29
 image size requirements, 396
 lookup functions, 136–137
 memory, 16, 18, 526
 mipmap, 28, 520
 multiple, 241–247
 multiple, examples, 241–247
 noise, 299–302
 simple, 238–241
 on a sphere, 241
 transformation matrix, 30
 units, 26–27
- Texturing. *See also* texture mapping; textures.
 1D, 26, 513
 2D, 26, 513
 3D, 26, 513
 active texture unit, 26
 bump map texture, 288, 515
 cube map texture, 26, 516
 defined, 16, 526
 developer forums, 31–32
 level-of-detail, 28
 limits, 114–115
 mipmap level, 28, 520
 mipmap textures, 28, 520
 per-texture-object level-of-detail bias, 28–29
 per-texture-unit level-of-detail bias, 29
 programmable pipeline, 113–115
 texels, 28
 texture units, 26–27
 web sites of interest, 31–32
- Texturing and Modeling: A Procedural Approach*, 293
- 3D and 4D noise function, 298–299
- 3D texture, 26, 513
- 3Dlabs logo, Color Plate 21
- Threshold, animation, 317
- Threshold functions with smooth transitions, 124, 129
- Tilde (~), unary preprocessor expression, 89
- T&L (transformation and lighting), 12–13, 525
- Torus, Color Plate 11, Color Plate 13, Color Plate 18, Color Plate 23
- Traditional shaders. *See* shaders, traditional.
- Transferring images, 18–19
- Transform function, 130
- Transformation, 20–25, 215–217
- Transformation and lighting (T&L), 12–13, 526
- Transforming coordinates. *See* coordinate transforms.
- Translation, animation, 317–318
- Triceratops, Color Plate 8
- Trigonometry functions, 118–120
- Tufte, Edward, 364
- Turbulence, 306–308, 526
- Two-dimensional images. *See* shaders, imaging.
- Two-dimensional texture map, 239
- Two-sided color mode, 110



Two-sided lighting, 224–225

2D noise function, 297–298

2D texture, 26, 513

typedef, 70–71

U

#undef, 87

Underscore (`_`), in variable names, 72

Underscores (`_`), in variable names, 73

uniform

defined, 49

example, 65

shader input/output, 49

shader interface, 76, 77, 78

variable qualifier, 76, 77, 78

Uniform line density, obtaining, 359–360

Uniform variables

active, 189–190, 513

active, querying, 456–459

defined, 40, 77, 526

floating-point values, defining, 194

location, querying, 185–186, 475–476

programmable pipeline, 97, 102, 104–108

specifying, 185–191

value, modifying, 186–189, 488–494

value, querying, 473–474

union, 71

Unsharp masking, 388, 401–402, 526

URLs of interest, 529–542

User clipping, 25, 112, 231, 526

User-defined attribute values, 38–39

User-defined uniform variables, 40

Using (installing) shaders, 163–167

V

Value noise, 293–297, 526

Value noise function, 293–297

Value range, ensuring, 123, 127–128

Variables. *See also* qualifiers.

attribute, 38, 77, 514

declaring, 50

name case sensitivity, 72

uniform, 40, 77, 97, 102, 104–108, 526

varying, 41, 77–78, 98–101, 527

Variables, programmable pipeline

special input, 102–103

special output, 97–98, 103–104

uniform, 40, 77, 97, 102, 104–108, 526

varying

built-in, 98–99

defined, 41, 77–78, 527

fragment processor, 101

user-defined, 100

vertex processor, 98–100

varying

example, 65

shader input/output, 49

shader interface, 76, 77–78

variable qualifier, 76, 77–78

Varying variables

built-in, 98–99

defined, 41, 77–78, 527

floating-point values, defining, 98–99

fragment processor, 101

user-defined, 100

vertex processor, 98–100

vecn, 48, 67–68

Vectors

comparison functions, 133–135

data types, 67–68

developing shaders, 206

length, determining, 130

relational functions, 133–135

`__VERSION__` macro, 88

Vertex, 10–11, 527

Vertex arrays method, 10–11

Vertex-at-a-time method, 10, 14

Vertex attributes, 10, 95–96, 173–184, 194, 527

Vertex processing, 12, 527

Vertex processor, 37–41, 94–100, 527. *See also*

programmable processors.

Vertex shaders. *See* shaders, vertex.

View frustum, 527

View volume, 14, 527

Viewing matrix, 22, 527

Viewing transformation, 22, 528

Viewing volume. *See* projection transformation.

Viewport transformation, 14, 25, 528

Visual complexity, increasing, 139–140

void data type, 72

Voorhies, Douglas, 385

W

Warm colors, 367. *See also* Gooch shading.

Waves, modeling on surface, 120

Web sites of interest, 529–542

while statement, 80

Wildcat VP, Color Plate 22

Window coordinates, 14, 25, 528

Wobble shader, 328–332, Color Plate 21

Wood shader, 308–312, Color Plate 22

Woodblock printing shader. *See* hatching shader.

World coordinate system, 21, 528

World space, 21, 528

Z

Zoom factor, 19