GL: 3-D Graphics

Version 6.1.1

Scott Owens

November 4, 2014

The `sgl` libraries provide access to the rendering functions of OpenGL 1.5 and GLU 1.3 libraries. The `sgl` libraries to not address system-level concerns, such as the attachment of GL rendering contexts to displays. Instead, the libraries should work with any Racket extension that provides GL with access to the system (such as a binding for `glx`). Notably, the `racket/gui/base` library provides support for rendering contexts via the `canvas%` class and its `with-gl-context` method.

Contents

Index **53**

1 Using OpenGL

The `sgl/gl` library provides direct access to the C-style OpenGL API, whereas the `sgl` library provides a more Racket-like interface. The `sgl/gl` library provides a binding for each `#defined` constant and for most functions in OpenGL 1.5 and GLU 1.3. The functions perform comparable checking to their C-language counterparts; they check the types of their arguments, but do not check the length of array arguments. The `sgl` library provides wrappers around many of the functions in the `sgl/gl` library to present a more Racket-friendly interface, including function names that follow Racket conventions, and checked, symbolic enumeration arguments, and array-length checks.

Warning on Safety: OpenGL programming is inherently unsafe, even when using only the `sgl` library. Although `sgl` checks the arguments to each function call, violation of higher-level assumptions of the system's OpenGL library can cause it to crash, bringing the entire Racket system down. For example, sending a large number of vertices in a single `glBegin` causes at least some GL implementations to crash.

Some examples are available in the `"examples"` directory of the `"sgl"` collection in the Racket installation. For `"alpha.rkt"`, try choosing the `"sk.jpg"` image distritbuted with Racket in the `"icons"` collection; you may have to press the `t` key a few times if the spinning cube is blank.

2 C-Style OpenGL

`(require sgl/gl)` package: `sgl`

The `sgl/gl` module provides a direct interface to the system's GL library closely following the conventions of the C-language OpenGL API. It provides a binding for each `#defined` constant (these start with `GL_`) and for the functions in the GL 1.5 and GLU 1.3 specifications, except for the following:

- Vertex arrays (GL 1.5, Section 2.8)
- Buffer objects (GL 1.5, Section 2.9)
- `glGetPointerv` (GL 1.5, Section 6.1.11)
- Buffer object queries (GL 1.5, Section 6.1.13)
- Polygon tessellation (GLU 1.3, Section 5)
- `gluQuadricCallback` (GLU 1.3, Section 6.2)
- NURBS callbacks (GLU 1.3, Section 7.2)

If one of the provided functions is not present on your system (e.g. if your system supports only GL 1.3), then the corresponding `sgl/gl` function raises a run-time exception when invoked.

The functions provided by `sgl/gl` perform comparable checking to their C-language counterparts; they check the types of their arguments, but do not check the length of array arguments. The following details the kinds of Racket values that can be provided for each primitive OpenGL type:

- `GLbyte`, `GLshort`, `GLint`: exact integer in the proper range
- `GLubyte`, `GLushort`, `GLuint`: exact non-negative integer in the proper range
- `GLsizei`, `GLenum`, `GLbitfield`: exact non-negative integer in the proper range
- `GFfloat`, `GLdouble`: real number
- `GFclampf`, `GLclampd`: real number
- `GLboolean`: any value, where `#f` means `GL_FALSE` and all other values mean `GL_TRUE`; do not use `GL_FALSE` or `GL_TRUE`, since they are bound to integers, both will end up being converted to GL_TRUE.

OpenGL functions that take vector arguments accept `cvector` values. The type of the `cvector` is checked; for example, `glVertex3fv` expects a vector of `GLfloats`, so `glVertex3fv` accepts only a `cvector` containing reals. See also `sgl/gl-vectors`. Functions that accept arrays of type `void*` accept any `cvector`; you must ensure that you supply the proper kind of vector, as in the C-language OpenGL API.

Examples:

```
(require sgl/gl
         sgl/gl-vectors)
(glBegin GL_TRIANGLES)
(glVertex3i 1 2 3)
(glVertex4fv (gl-float-vector 1 2 3 4))
(glEnd)
```

```
glPixelMapfv : procedure?
glPixelMapuiv : procedure?
glPixelMapusv : procedure?
glDeleteTextures : procedure?
glDeleteQueries : procedure?
```

These functions do not take a size argument, because it is derived from the length of the argument vector.

```
glGenTextures : procedure?
glGenQueries : procedure?
```

These functions do not take vector arguments. Instead, they allocate a vector of the requested size and return it.

```
glAreTexturesResident : procedure?
```

This function takes in a `GLuint` vector and textures, and it returns 2 values: the specified boolean and a boolean vector of residences.

```
glGetBooleanv : procedure?
glGetIntegerv : procedure?
glGetFloatv : procedure?
glGetDoublev : procedure?
glGetLightfv : procedure?
glGetLightiv : procedure?
glGetMaterialfv : procedure?
glGetMaterialiv : procedure?
glGetTexEnvfv : procedure?
glGetTexEnviv : procedure?
glGetTexGendv : procedure?
```

```
glGetTexGenfv : procedure?
glGetTexGeniv : procedure?
glGetTexParameterfv : procedure?
glGetTexParameteriv : procedure?
glGetTexLevelParameterfv : procedure?
glGetTexLevelParameteriv : procedure?
glGetPixelMapfv : procedure?
glGetPixelMapuiv : procedure?
glGetPixelMapusv : procedure?
glGetMapdv : procedure?
glGetMapfv : procedure?
glGetMapiv : procedure?
glGetBufferParameteriv : procedure?
glGetConvolutionParameterfv : procedure?
glGetConvolutionParameteriv : procedure?
glGetHistogramParameterfv : procedure?
glGetHistogramParameteriv : procedure?
glGetMinmaxParameterfv : procedure?
glGetMinmaxParameteriv : procedure?
glGetQueryiv : procedure?
glGetQueryObjectiv : procedure?
glGetQueryObjectuiv : procedure?
```

Instead of taking a vector argument, these function take an integer argument that specifies the size of the vector that is returned.

```
glGetClipPlane : procedure?
```

This function does not take a vector argument and returns a `GLdouble` vector of length 4.

```
glGetString : procedure?
gluCheckExtension : procedure?
gluErrorString : procedure?
gluGetString : procedure?
```

These functions deal with strings instead of `GLubyte` vectors.

```
gluProject : procedure?
gluUnProject : procedure?
gluUnProject4 : procedure?
```

Instead of taking pointers to `GLdoubles` for return values, these function directly return `GLdouble` vectors.

```
glSelectBuffer : procedure?
glFeedbackBuffer : procedure?
```

These functions do not take vectors, instead they return a `selection-buffer-object` or `feedback-buffer-object`. The `select-buffer->gl-uint-vector` and `feedback-buffer->gl-float-vector` functions copy the contents of the buffer into a vector. Because the OpenGL library writes to the buffer-object on OpenGL function calls after `glSelectBuffer` or `glFeedbackBuffer` has returned, if the buffer is garbage collected before OpenGL is finished writing to it, the entire Racket system can crash. The `gl-process-selection` function in `sgl` helps interpret the results of `glSelectBuffer` in a Racket-friendly format.

```
glAccum : procedure?
glActiveTexture : procedure?
glAlphaFunc : procedure?
glBegin : procedure?
glBeginQuery : procedure?
glBindTexture : procedure?
glBitmap : procedure?
glBlendColor : procedure?
glBlendEquation : procedure?
glBlendFunc : procedure?
glBlendFuncSeparate : procedure?
glCallList : procedure?
glCallLists : procedure?
glClear : procedure?
glClearAccum : procedure?
glClearColor : procedure?
glClearDepth : procedure?
glClearIndex : procedure?
glClearStencil : procedure?
glClipPlane : procedure?
glColor3b : procedure?
glColor3bv : procedure?
glColor3d : procedure?
glColor3dv : procedure?
glColor3f : procedure?
glColor3fv : procedure?
glColor3i : procedure?
glColor3iv : procedure?
glColor3s : procedure?
glColor3sv : procedure?
glColor3ub : procedure?
glColor3ubv : procedure?
```

```
glColor3ui : procedure?
glColor3uiv : procedure?
glColor3us : procedure?
glColor3usv : procedure?
glColor4b : procedure?
glColor4bv : procedure?
glColor4d : procedure?
glColor4dv : procedure?
glColor4f : procedure?
glColor4fv : procedure?
glColor4i : procedure?
glColor4iv : procedure?
glColor4s : procedure?
glColor4sv : procedure?
glColor4ub : procedure?
glColor4ubv : procedure?
glColor4ui : procedure?
glColor4uiv : procedure?
glColor4us : procedure?
glColor4usv : procedure?
glColorMask : procedure?
glColorMaterial : procedure?
glColorSubTable : procedure?
glColorTable : procedure?
glColorTableParameterfv : procedure?
glColorTableParameteriv : procedure?
glCompressedTexImage1D : procedure?
glCompressedTexImage2D : procedure?
glCompressedTexImage3D : procedure?
glCompressedTexSubImage1D : procedure?
glCompressedTexSubImage2D : procedure?
glCompressedTexSubImage3D : procedure?
glConvolutionFilter1D : procedure?
glConvolutionFilter2D : procedure?
glConvolutionParameterf : procedure?
glConvolutionParameterfv : procedure?
glConvolutionParameteri : procedure?
glConvolutionParameteriv : procedure?
glCopyColorSubTable : procedure?
glCopyColorTable : procedure?
glCopyConvolutionFilter1D : procedure?
glCopyConvolutionFilter2D : procedure?
glCopyPixels : procedure?
glCopyTexImage1D : procedure?
```

```
glCopyTexImage2D : procedure?
glCopyTexSubImage1D : procedure?
glCopyTexSubImage2D : procedure?
glCopyTexSubImage3D : procedure?
glCullFace : procedure?
glDeleteLists : procedure?
glDepthFunc : procedure?
glDepthMask : procedure?
glDepthRange : procedure?
glDisable : procedure?
glDrawBuffer : procedure?
glDrawPixels : procedure?
glEdgeFlag : procedure?
glEdgeFlagv : procedure?
glEnable : procedure?
glEnd : procedure?
glEndList : procedure?
glEndQuery : procedure?
glEvalCoord1d : procedure?
glEvalCoord1dv : procedure?
glEvalCoord1f : procedure?
glEvalCoord1fv : procedure?
glEvalCoord2d : procedure?
glEvalCoord2dv : procedure?
glEvalCoord2f : procedure?
glEvalCoord2fv : procedure?
glEvalMesh1 : procedure?
glEvalMesh2 : procedure?
glEvalPoint1 : procedure?
glEvalPoint2 : procedure?
glFinish : procedure?
glFlush : procedure?
glFogCoordd : procedure?
glFogCoorddv : procedure?
glFogCoordf : procedure?
glFogCoordfv : procedure?
glFogf : procedure?
glFogfv : procedure?
glFogi : procedure?
glFogiv : procedure?
glFrontFace : procedure?
glFrustum : procedure?
glGenLists : procedure?
glGetColorTable : procedure?
```

```
glGetCompressedTexImage : procedure?
glGetConvolutionFilter : procedure?
glGetError : procedure?
glGetHistogram : procedure?
glGetMinmax : procedure?
glGetPolygonStipple : procedure?
glGetSeparableFilter : procedure?
glGetTexImage : procedure?
glHint : procedure?
glHistogram : procedure?
glIndexMask : procedure?
glIndexd : procedure?
glIndexdv : procedure?
glIndexf : procedure?
glIndexfv : procedure?
glIndexi : procedure?
glIndexiv : procedure?
glIndexs : procedure?
glIndexsv : procedure?
glIndexub : procedure?
glIndexubv : procedure?
glInitNames : procedure?
glIsBuffer : procedure?
glIsEnabled : procedure?
glIsList : procedure?
glIsQuery : procedure?
glIsTexture : procedure?
glLightModelf : procedure?
glLightModelfv : procedure?
glLightModeli : procedure?
glLightModeliv : procedure?
glLightf : procedure?
glLightfv : procedure?
glLighti : procedure?
glLightiv : procedure?
glLineStipple : procedure?
glLineWidth : procedure?
glListBase : procedure?
glLoadIdentity : procedure?
glLoadMatrixd : procedure?
glLoadMatrixf : procedure?
glLoadName : procedure?
glLoadTransposeMatrixd : procedure?
glLoadTransposeMatrixf : procedure?
```

```
glLogicOp : procedure?
glMap1d : procedure?
glMap1f : procedure?
glMap2d : procedure?
glMap2f : procedure?
glMapGrid1d : procedure?
glMapGrid1f : procedure?
glMapGrid2d : procedure?
glMapGrid2f : procedure?
glMaterialf : procedure?
glMaterialfv : procedure?
glMateriali : procedure?
glMaterialiv : procedure?
glMatrixMode : procedure?
glMinmax : procedure?
glMultMatrixd : procedure?
glMultMatrixf : procedure?
glMultTransposeMatrixd : procedure?
glMultTransposeMatrixf : procedure?
glMultiTexCoord1d : procedure?
glMultiTexCoord1dv : procedure?
glMultiTexCoord1f : procedure?
glMultiTexCoord1fv : procedure?
glMultiTexCoord1i : procedure?
glMultiTexCoord1iv : procedure?
glMultiTexCoord1s : procedure?
glMultiTexCoord1sv : procedure?
glMultiTexCoord2d : procedure?
glMultiTexCoord2dv : procedure?
glMultiTexCoord2f : procedure?
glMultiTexCoord2fv : procedure?
glMultiTexCoord2i : procedure?
glMultiTexCoord2iv : procedure?
glMultiTexCoord2s : procedure?
glMultiTexCoord2sv : procedure?
glMultiTexCoord3d : procedure?
glMultiTexCoord3dv : procedure?
glMultiTexCoord3f : procedure?
glMultiTexCoord3fv : procedure?
glMultiTexCoord3i : procedure?
glMultiTexCoord3iv : procedure?
glMultiTexCoord3s : procedure?
glMultiTexCoord3sv : procedure?
glMultiTexCoord4d : procedure?
```

```
glMultiTexCoord4dv : procedure?
glMultiTexCoord4f : procedure?
glMultiTexCoord4fv : procedure?
glMultiTexCoord4i : procedure?
glMultiTexCoord4iv : procedure?
glMultiTexCoord4s : procedure?
glMultiTexCoord4sv : procedure?
glNewList : procedure?
glNormal3b : procedure?
glNormal3bv : procedure?
glNormal3d : procedure?
glNormal3dv : procedure?
glNormal3f : procedure?
glNormal3fv : procedure?
glNormal3i : procedure?
glNormal3iv : procedure?
glNormal3s : procedure?
glNormal3sv : procedure?
glOrtho : procedure?
glPassThrough : procedure?
glPixelStoref : procedure?
glPixelStorei : procedure?
glPixelTransferf : procedure?
glPixelTransferi : procedure?
glPixelZoom : procedure?
glPointParameterf : procedure?
glPointParameterfv : procedure?
glPointParameteri : procedure?
glPointParameteriv : procedure?
glPointSize : procedure?
glPolygonMode : procedure?
glPolygonOffset : procedure?
glPolygonStipple : procedure?
glPopAttrib : procedure?
glPopClientAttrib : procedure?
glPopMatrix : procedure?
glPopName : procedure?
glPushAttrib : procedure?
glPushClientAttrib : procedure?
glPushMatrix : procedure?
glPushName : procedure?
glRasterPos2d : procedure?
glRasterPos2dv : procedure?
glRasterPos2f : procedure?
```

```
glRasterPos2fv : procedure?
glRasterPos2i : procedure?
glRasterPos2iv : procedure?
glRasterPos2s : procedure?
glRasterPos2sv : procedure?
glRasterPos3d : procedure?
glRasterPos3dv : procedure?
glRasterPos3f : procedure?
glRasterPos3fv : procedure?
glRasterPos3i : procedure?
glRasterPos3iv : procedure?
glRasterPos3s : procedure?
glRasterPos3sv : procedure?
glRasterPos4d : procedure?
glRasterPos4dv : procedure?
glRasterPos4f : procedure?
glRasterPos4fv : procedure?
glRasterPos4i : procedure?
glRasterPos4iv : procedure?
glRasterPos4s : procedure?
glRasterPos4sv : procedure?
glReadBuffer : procedure?
glReadPixels : procedure?
glRectd : procedure?
glRectdv : procedure?
glRectf : procedure?
glRectfv : procedure?
glRecti : procedure?
glRectiv : procedure?
glRects : procedure?
glRectsv : procedure?
glRenderMode : procedure?
glResetHistogram : procedure?
glResetMinmax : procedure?
glRotated : procedure?
glRotatef : procedure?
glSampleCoverage : procedure?
glScaled : procedure?
glScalef : procedure?
glScissor : procedure?
glSecondaryColor3b : procedure?
glSecondaryColor3bv : procedure?
glSecondaryColor3d : procedure?
glSecondaryColor3dv : procedure?
```

```
glSecondaryColor3f : procedure?
glSecondaryColor3fv : procedure?
glSecondaryColor3i : procedure?
glSecondaryColor3iv : procedure?
glSecondaryColor3s : procedure?
glSecondaryColor3sv : procedure?
glSecondaryColor3ub : procedure?
glSecondaryColor3ubv : procedure?
glSecondaryColor3ui : procedure?
glSecondaryColor3uiv : procedure?
glSecondaryColor3us : procedure?
glSecondaryColor3usv : procedure?
glSeparableFilter2D : procedure?
glShadeModel : procedure?
glStencilFunc : procedure?
glStencilMask : procedure?
glStencilOp : procedure?
glTexCoord1d : procedure?
glTexCoord1dv : procedure?
glTexCoord1f : procedure?
glTexCoord1fv : procedure?
glTexCoord1i : procedure?
glTexCoord1iv : procedure?
glTexCoord1s : procedure?
glTexCoord1sv : procedure?
glTexCoord2d : procedure?
glTexCoord2dv : procedure?
glTexCoord2f : procedure?
glTexCoord2fv : procedure?
glTexCoord2i : procedure?
glTexCoord2iv : procedure?
glTexCoord2s : procedure?
glTexCoord2sv : procedure?
glTexCoord3d : procedure?
glTexCoord3dv : procedure?
glTexCoord3f : procedure?
glTexCoord3fv : procedure?
glTexCoord3i : procedure?
glTexCoord3iv : procedure?
glTexCoord3s : procedure?
glTexCoord3sv : procedure?
glTexCoord4d : procedure?
glTexCoord4dv : procedure?
glTexCoord4f : procedure?
```

```
glTexCoord4fv : procedure?
glTexCoord4i : procedure?
glTexCoord4iv : procedure?
glTexCoord4s : procedure?
glTexCoord4sv : procedure?
glTexEnvf : procedure?
glTexEnvfv : procedure?
glTexEnvi : procedure?
glTexEnviv : procedure?
glTexGend : procedure?
glTexGendv : procedure?
glTexGenf : procedure?
glTexGenfv : procedure?
glTexGeni : procedure?
glTexGeniv : procedure?
glTexImage1D : procedure?
glTexImage2D : procedure?
glTexImage3D : procedure?
glTexParameterf : procedure?
glTexParameterfv : procedure?
glTexParameteri : procedure?
glTexParameteriv : procedure?
glTexSubImage1D : procedure?
glTexSubImage2D : procedure?
glTexSubImage3D : procedure?
glTranslated : procedure?
glTranslatef : procedure?
glVertex2d : procedure?
glVertex2dv : procedure?
glVertex2f : procedure?
glVertex2fv : procedure?
glVertex2i : procedure?
glVertex2iv : procedure?
glVertex2s : procedure?
glVertex2sv : procedure?
glVertex3d : procedure?
glVertex3dv : procedure?
glVertex3f : procedure?
glVertex3fv : procedure?
glVertex3i : procedure?
glVertex3iv : procedure?
glVertex3s : procedure?
glVertex3sv : procedure?
glVertex4d : procedure?
```

```
glVertex4dv : procedure?
glVertex4f : procedure?
glVertex4fv : procedure?
glVertex4i : procedure?
glVertex4iv : procedure?
glVertex4s : procedure?
glVertex4sv : procedure?
glViewport : procedure?
glWindowPos2d : procedure?
glWindowPos2dv : procedure?
glWindowPos2f : procedure?
glWindowPos2fv : procedure?
glWindowPos2i : procedure?
glWindowPos2iv : procedure?
glWindowPos2s : procedure?
glWindowPos2sv : procedure?
glWindowPos3d : procedure?
glWindowPos3dv : procedure?
glWindowPos3f : procedure?
glWindowPos3fv : procedure?
glWindowPos3i : procedure?
glWindowPos3iv : procedure?
glWindowPos3s : procedure?
glWindowPos3sv : procedure?
gluBuild1DMipmapLevels : procedure?
gluBuild1DMipmaps : procedure?
gluBuild2DMipmapLevels : procedure?
gluBuild2DMipmaps : procedure?
gluBuild3DMipmapLevels : procedure?
gluBuild3DMipmaps : procedure?
gluCylinder : procedure?
gluDisk : procedure?
gluLookAt : procedure?
gluNewQuadric : procedure?
gluOrtho2D : procedure?
gluPartialDisk : procedure?
gluPerspective : procedure?
gluPickMatrix : procedure?
gluQuadricDrawStyle : procedure?
gluQuadricNormals : procedure?
gluQuadricOrientation : procedure?
gluQuadricTexture : procedure?
gluScaleImage : procedure?
gluSphere : procedure?
```

These functions are all direct translations of the C OpenGL API.

```
GL_FALSE : exact-integer?
GL_TRUE : exact-integer?
GL_BYTE : exact-integer?
GL_UNSIGNED_BYTE : exact-integer?
GL_SHORT : exact-integer?
GL_UNSIGNED_SHORT : exact-integer?
GL_INT : exact-integer?
GL_UNSIGNED_INT : exact-integer?
GL_FLOAT : exact-integer?
GL_DOUBLE : exact-integer?
GL_2_BYTES : exact-integer?
GL_3_BYTES : exact-integer?
GL_4_BYTES : exact-integer?
GL_POINTS : exact-integer?
GL_LINES : exact-integer?
GL_LINE_LOOP : exact-integer?
GL_LINE_STRIP : exact-integer?
GL_TRIANGLES : exact-integer?
GL_TRIANGLE_STRIP : exact-integer?
GL_TRIANGLE_FAN : exact-integer?
GL_QUADS : exact-integer?
GL_QUAD_STRIP : exact-integer?
GL_POLYGON : exact-integer?
GL_VERTEX_ARRAY : exact-integer?
GL_NORMAL_ARRAY : exact-integer?
GL_COLOR_ARRAY : exact-integer?
GL_INDEX_ARRAY : exact-integer?
GL_TEXTURE_COORD_ARRAY : exact-integer?
GL_EDGE_FLAG_ARRAY : exact-integer?
GL_VERTEX_ARRAY_SIZE : exact-integer?
GL_VERTEX_ARRAY_TYPE : exact-integer?
GL_VERTEX_ARRAY_STRIDE : exact-integer?
GL_NORMAL_ARRAY_TYPE : exact-integer?
GL_NORMAL_ARRAY_STRIDE : exact-integer?
GL_COLOR_ARRAY_SIZE : exact-integer?
GL_COLOR_ARRAY_TYPE : exact-integer?
GL_COLOR_ARRAY_STRIDE : exact-integer?
GL_INDEX_ARRAY_TYPE : exact-integer?
GL_INDEX_ARRAY_STRIDE : exact-integer?
GL_TEXTURE_COORD_ARRAY_SIZE : exact-integer?
GL_TEXTURE_COORD_ARRAY_TYPE : exact-integer?
GL_TEXTURE_COORD_ARRAY_STRIDE : exact-integer?
GL_EDGE_FLAG_ARRAY_STRIDE : exact-integer?
```

```
GL_VERTEX_ARRAY_POINTER : exact-integer?
GL_NORMAL_ARRAY_POINTER : exact-integer?
GL_COLOR_ARRAY_POINTER : exact-integer?
GL_INDEX_ARRAY_POINTER : exact-integer?
GL_TEXTURE_COORD_ARRAY_POINTER : exact-integer?
GL_EDGE_FLAG_ARRAY_POINTER : exact-integer?
GL_V2F : exact-integer?
GL_V3F : exact-integer?
GL_C4UB_V2F : exact-integer?
GL_C4UB_V3F : exact-integer?
GL_C3F_V3F : exact-integer?
GL_N3F_V3F : exact-integer?
GL_C4F_N3F_V3F : exact-integer?
GL_T2F_V3F : exact-integer?
GL_T4F_V4F : exact-integer?
GL_T2F_C4UB_V3F : exact-integer?
GL_T2F_C3F_V3F : exact-integer?
GL_T2F_N3F_V3F : exact-integer?
GL_T2F_C4F_N3F_V3F : exact-integer?
GL_T4F_C4F_N3F_V4F : exact-integer?
GL_MATRIX_MODE : exact-integer?
GL_MODELVIEW : exact-integer?
GL_PROJECTION : exact-integer?
GL_TEXTURE : exact-integer?
GL_POINT_SMOOTH : exact-integer?
GL_POINT_SIZE : exact-integer?
GL_POINT_SIZE_GRANULARITY : exact-integer?
GL_POINT_SIZE_RANGE : exact-integer?
GL_LINE_SMOOTH : exact-integer?
GL_LINE_STIPPLE : exact-integer?
GL_LINE_STIPPLE_PATTERN : exact-integer?
GL_LINE_STIPPLE_REPEAT : exact-integer?
GL_LINE_WIDTH : exact-integer?
GL_LINE_WIDTH_GRANULARITY : exact-integer?
GL_LINE_WIDTH_RANGE : exact-integer?
GL_POINT : exact-integer?
GL_LINE : exact-integer?
GL_FILL : exact-integer?
GL_CW : exact-integer?
GL_CCW : exact-integer?
GL_FRONT : exact-integer?
GL_BACK : exact-integer?
GL_POLYGON_MODE : exact-integer?
GL_POLYGON_SMOOTH : exact-integer?
```

```
GL_POLYGON_STIPPLE : exact-integer?
GL_EDGE_FLAG : exact-integer?
GL_CULL_FACE : exact-integer?
GL_CULL_FACE_MODE : exact-integer?
GL_FRONT_FACE : exact-integer?
GL_POLYGON_OFFSET_FACTOR : exact-integer?
GL_POLYGON_OFFSET_UNITS : exact-integer?
GL_POLYGON_OFFSET_POINT : exact-integer?
GL_POLYGON_OFFSET_LINE : exact-integer?
GL_POLYGON_OFFSET_FILL : exact-integer?
GL_COMPILE : exact-integer?
GL_COMPILE_AND_EXECUTE : exact-integer?
GL_LIST_BASE : exact-integer?
GL_LIST_INDEX : exact-integer?
GL_LIST_MODE : exact-integer?
GL_NEVER : exact-integer?
GL_LESS : exact-integer?
GL_EQUAL : exact-integer?
GL_LEQUAL : exact-integer?
GL_GREATER : exact-integer?
GL_NOTEQUAL : exact-integer?
GL_GEQUAL : exact-integer?
GL_ALWAYS : exact-integer?
GL_DEPTH_TEST : exact-integer?
GL_DEPTH_BITS : exact-integer?
GL_DEPTH_CLEAR_VALUE : exact-integer?
GL_DEPTH_FUNC : exact-integer?
GL_DEPTH_RANGE : exact-integer?
GL_DEPTH_WRITEMASK : exact-integer?
GL_DEPTH_COMPONENT : exact-integer?
GL_LIGHTING : exact-integer?
GL_LIGHT0 : exact-integer?
GL_LIGHT1 : exact-integer?
GL_LIGHT2 : exact-integer?
GL_LIGHT3 : exact-integer?
GL_LIGHT4 : exact-integer?
GL_LIGHT5 : exact-integer?
GL_LIGHT6 : exact-integer?
GL_LIGHT7 : exact-integer?
GL_SPOT_EXPONENT : exact-integer?
GL_SPOT_CUTOFF : exact-integer?
GL_CONSTANT_ATTENUATION : exact-integer?
GL_LINEAR_ATTENUATION : exact-integer?
GL_QUADRATIC_ATTENUATION : exact-integer?
```

```
GL_AMBIENT : exact-integer?
GL_DIFFUSE : exact-integer?
GL_SPECULAR : exact-integer?
GL_SHININESS : exact-integer?
GL_EMISSION : exact-integer?
GL_POSITION : exact-integer?
GL_SPOT_DIRECTION : exact-integer?
GL_AMBIENT_AND_DIFFUSE : exact-integer?
GL_COLOR_INDEXES : exact-integer?
GL_LIGHT_MODEL_TWO_SIDE : exact-integer?
GL_LIGHT_MODEL_LOCAL_VIEWER : exact-integer?
GL_LIGHT_MODEL_AMBIENT : exact-integer?
GL_FRONT_AND_BACK : exact-integer?
GL_SHADE_MODEL : exact-integer?
GL_FLAT : exact-integer?
GL_SMOOTH : exact-integer?
GL_COLOR_MATERIAL : exact-integer?
GL_COLOR_MATERIAL_FACE : exact-integer?
GL_COLOR_MATERIAL_PARAMETER : exact-integer?
GL_NORMALIZE : exact-integer?
GL_CLIP_PLANE0 : exact-integer?
GL_CLIP_PLANE1 : exact-integer?
GL_CLIP_PLANE2 : exact-integer?
GL_CLIP_PLANE3 : exact-integer?
GL_CLIP_PLANE4 : exact-integer?
GL_CLIP_PLANE5 : exact-integer?
GL_ACCUM_RED_BITS : exact-integer?
GL_ACCUM_GREEN_BITS : exact-integer?
GL_ACCUM_BLUE_BITS : exact-integer?
GL_ACCUM_ALPHA_BITS : exact-integer?
GL_ACCUM_CLEAR_VALUE : exact-integer?
GL_ACCUM : exact-integer?
GL_ADD : exact-integer?
GL_LOAD : exact-integer?
GL_MULT : exact-integer?
GL_RETURN : exact-integer?
GL_ALPHA_TEST : exact-integer?
GL_ALPHA_TEST_REF : exact-integer?
GL_ALPHA_TEST_FUNC : exact-integer?
GL_BLEND : exact-integer?
GL_BLEND_SRC : exact-integer?
GL_BLEND_DST : exact-integer?
GL_ZERO : exact-integer?
GL_ONE : exact-integer?
```

```
GL_SRC_COLOR : exact-integer?
GL_ONE_MINUS_SRC_COLOR : exact-integer?
GL_SRC_ALPHA : exact-integer?
GL_ONE_MINUS_SRC_ALPHA : exact-integer?
GL_DST_ALPHA : exact-integer?
GL_ONE_MINUS_DST_ALPHA : exact-integer?
GL_DST_COLOR : exact-integer?
GL_ONE_MINUS_DST_COLOR : exact-integer?
GL_SRC_ALPHA_SATURATE : exact-integer?
GL_FEEDBACK : exact-integer?
GL_RENDER : exact-integer?
GL_SELECT : exact-integer?
GL_2D : exact-integer?
GL_3D : exact-integer?
GL_3D_COLOR : exact-integer?
GL_3D_COLOR_TEXTURE : exact-integer?
GL_4D_COLOR_TEXTURE : exact-integer?
GL_POINT_TOKEN : exact-integer?
GL_LINE_TOKEN : exact-integer?
GL_LINE_RESET_TOKEN : exact-integer?
GL_POLYGON_TOKEN : exact-integer?
GL_BITMAP_TOKEN : exact-integer?
GL_DRAW_PIXEL_TOKEN : exact-integer?
GL_COPY_PIXEL_TOKEN : exact-integer?
GL_PASS_THROUGH_TOKEN : exact-integer?
GL_FEEDBACK_BUFFER_POINTER : exact-integer?
GL_FEEDBACK_BUFFER_SIZE : exact-integer?
GL_FEEDBACK_BUFFER_TYPE : exact-integer?
GL_SELECTION_BUFFER_POINTER : exact-integer?
GL_SELECTION_BUFFER_SIZE : exact-integer?
GL_FOG : exact-integer?
GL_FOG_MODE : exact-integer?
GL_FOG_DENSITY : exact-integer?
GL_FOG_COLOR : exact-integer?
GL_FOG_INDEX : exact-integer?
GL_FOG_START : exact-integer?
GL_FOG_END : exact-integer?
GL_LINEAR : exact-integer?
GL_EXP : exact-integer?
GL_EXP2 : exact-integer?
GL_LOGIC_OP : exact-integer?
GL_INDEX_LOGIC_OP : exact-integer?
GL_COLOR_LOGIC_OP : exact-integer?
GL_LOGIC_OP_MODE : exact-integer?
```

```
GL_CLEAR : exact-integer?
GL_SET : exact-integer?
GL_COPY : exact-integer?
GL_COPY_INVERTED : exact-integer?
GL_NOOP : exact-integer?
GL_INVERT : exact-integer?
GL_AND : exact-integer?
GL_NAND : exact-integer?
GL_OR : exact-integer?
GL_NOR : exact-integer?
GL_XOR : exact-integer?
GL_EQUIV : exact-integer?
GL_AND_REVERSE : exact-integer?
GL_AND_INVERTED : exact-integer?
GL_OR_REVERSE : exact-integer?
GL_OR_INVERTED : exact-integer?
GL_STENCIL_TEST : exact-integer?
GL_STENCIL_WRITEMASK : exact-integer?
GL_STENCIL_BITS : exact-integer?
GL_STENCIL_FUNC : exact-integer?
GL_STENCIL_VALUE_MASK : exact-integer?
GL_STENCIL_REF : exact-integer?
GL_STENCIL_FAIL : exact-integer?
GL_STENCIL_PASS_DEPTH_PASS : exact-integer?
GL_STENCIL_PASS_DEPTH_FAIL : exact-integer?
GL_STENCIL_CLEAR_VALUE : exact-integer?
GL_STENCIL_INDEX : exact-integer?
GL_KEEP : exact-integer?
GL_REPLACE : exact-integer?
GL_INCR : exact-integer?
GL_DECR : exact-integer?
GL_NONE : exact-integer?
GL_LEFT : exact-integer?
GL_RIGHT : exact-integer?
GL_FRONT_LEFT : exact-integer?
GL_FRONT_RIGHT : exact-integer?
GL_BACK_LEFT : exact-integer?
GL_BACK_RIGHT : exact-integer?
GL_AUX0 : exact-integer?
GL_AUX1 : exact-integer?
GL_AUX2 : exact-integer?
GL_AUX3 : exact-integer?
GL_COLOR_INDEX : exact-integer?
GL_RED : exact-integer?
```

```
GL_GREEN : exact-integer?
GL_BLUE : exact-integer?
GL_ALPHA : exact-integer?
GL_LUMINANCE : exact-integer?
GL_LUMINANCE_ALPHA : exact-integer?
GL_ALPHA_BITS : exact-integer?
GL_RED_BITS : exact-integer?
GL_GREEN_BITS : exact-integer?
GL_BLUE_BITS : exact-integer?
GL_INDEX_BITS : exact-integer?
GL_SUBPIXEL_BITS : exact-integer?
GL_AUX_BUFFERS : exact-integer?
GL_READ_BUFFER : exact-integer?
GL_DRAW_BUFFER : exact-integer?
GL_DOUBLEBUFFER : exact-integer?
GL_STEREO : exact-integer?
GL_BITMAP : exact-integer?
GL_COLOR : exact-integer?
GL_DEPTH : exact-integer?
GL_STENCIL : exact-integer?
GL_DITHER : exact-integer?
GL_RGB : exact-integer?
GL_RGBA : exact-integer?
GL_MAX_LIST_NESTING : exact-integer?
GL_MAX_ATTRIB_STACK_DEPTH : exact-integer?
GL_MAX_MODELVIEW_STACK_DEPTH : exact-integer?
GL_MAX_NAME_STACK_DEPTH : exact-integer?
GL_MAX_PROJECTION_STACK_DEPTH : exact-integer?
GL_MAX_TEXTURE_STACK_DEPTH : exact-integer?
GL_MAX_EVAL_ORDER : exact-integer?
GL_MAX_LIGHTS : exact-integer?
GL_MAX_CLIP_PLANES : exact-integer?
GL_MAX_TEXTURE_SIZE : exact-integer?
GL_MAX_PIXEL_MAP_TABLE : exact-integer?
GL_MAX_VIEWPORT_DIMS : exact-integer?
GL_MAX_CLIENT_ATTRIB_STACK_DEPTH : exact-integer?
GL_ATTRIB_STACK_DEPTH : exact-integer?
GL_CLIENT_ATTRIB_STACK_DEPTH : exact-integer?
GL_COLOR_CLEAR_VALUE : exact-integer?
GL_COLOR_WRITEMASK : exact-integer?
GL_CURRENT_INDEX : exact-integer?
GL_CURRENT_COLOR : exact-integer?
GL_CURRENT_NORMAL : exact-integer?
GL_CURRENT_RASTER_COLOR : exact-integer?
```

```
GL_CURRENT_RASTER_DISTANCE : exact-integer?
GL_CURRENT_RASTER_INDEX : exact-integer?
GL_CURRENT_RASTER_POSITION : exact-integer?
GL_CURRENT_RASTER_TEXTURE_COORDS : exact-integer?
GL_CURRENT_RASTER_POSITION_VALID : exact-integer?
GL_CURRENT_TEXTURE_COORDS : exact-integer?
GL_INDEX_CLEAR_VALUE : exact-integer?
GL_INDEX_MODE : exact-integer?
GL_INDEX_WRITEMASK : exact-integer?
GL_MODELVIEW_MATRIX : exact-integer?
GL_MODELVIEW_STACK_DEPTH : exact-integer?
GL_NAME_STACK_DEPTH : exact-integer?
GL_PROJECTION_MATRIX : exact-integer?
GL_PROJECTION_STACK_DEPTH : exact-integer?
GL_RENDER_MODE : exact-integer?
GL_RGBA_MODE : exact-integer?
GL_TEXTURE_MATRIX : exact-integer?
GL_TEXTURE_STACK_DEPTH : exact-integer?
GL_VIEWPORT : exact-integer?
GL_AUTO_NORMAL : exact-integer?
GL_MAP1_COLOR_4 : exact-integer?
GL_MAP1_GRID_DOMAIN : exact-integer?
GL_MAP1_GRID_SEGMENTS : exact-integer?
GL_MAP1_INDEX : exact-integer?
GL_MAP1_NORMAL : exact-integer?
GL_MAP1_TEXTURE_COORD_1 : exact-integer?
GL_MAP1_TEXTURE_COORD_2 : exact-integer?
GL_MAP1_TEXTURE_COORD_3 : exact-integer?
GL_MAP1_TEXTURE_COORD_4 : exact-integer?
GL_MAP1_VERTEX_3 : exact-integer?
GL_MAP1_VERTEX_4 : exact-integer?
GL_MAP2_COLOR_4 : exact-integer?
GL_MAP2_GRID_DOMAIN : exact-integer?
GL_MAP2_GRID_SEGMENTS : exact-integer?
GL_MAP2_INDEX : exact-integer?
GL_MAP2_NORMAL : exact-integer?
GL_MAP2_TEXTURE_COORD_1 : exact-integer?
GL_MAP2_TEXTURE_COORD_2 : exact-integer?
GL_MAP2_TEXTURE_COORD_3 : exact-integer?
GL_MAP2_TEXTURE_COORD_4 : exact-integer?
GL_MAP2_VERTEX_3 : exact-integer?
GL_MAP2_VERTEX_4 : exact-integer?
GL_COEFF : exact-integer?
GL_DOMAIN : exact-integer?
```

```
GL_ORDER : exact-integer?
GL_FOG_HINT : exact-integer?
GL_LINE_SMOOTH_HINT : exact-integer?
GL_PERSPECTIVE_CORRECTION_HINT : exact-integer?
GL_POINT_SMOOTH_HINT : exact-integer?
GL_POLYGON_SMOOTH_HINT : exact-integer?
GL_DONT_CARE : exact-integer?
GL_FASTEST : exact-integer?
GL_NICEST : exact-integer?
GL_SCISSOR_TEST : exact-integer?
GL_SCISSOR_BOX : exact-integer?
GL_MAP_COLOR : exact-integer?
GL_MAP_STENCIL : exact-integer?
GL_INDEX_SHIFT : exact-integer?
GL_INDEX_OFFSET : exact-integer?
GL_RED_SCALE : exact-integer?
GL_RED_BIAS : exact-integer?
GL_GREEN_SCALE : exact-integer?
GL_GREEN_BIAS : exact-integer?
GL_BLUE_SCALE : exact-integer?
GL_BLUE_BIAS : exact-integer?
GL_ALPHA_SCALE : exact-integer?
GL_ALPHA_BIAS : exact-integer?
GL_DEPTH_SCALE : exact-integer?
GL_DEPTH_BIAS : exact-integer?
GL_PIXEL_MAP_S_TO_S_SIZE : exact-integer?
GL_PIXEL_MAP_I_TO_I_SIZE : exact-integer?
GL_PIXEL_MAP_I_TO_R_SIZE : exact-integer?
GL_PIXEL_MAP_I_TO_G_SIZE : exact-integer?
GL_PIXEL_MAP_I_TO_B_SIZE : exact-integer?
GL_PIXEL_MAP_I_TO_A_SIZE : exact-integer?
GL_PIXEL_MAP_R_TO_R_SIZE : exact-integer?
GL_PIXEL_MAP_G_TO_G_SIZE : exact-integer?
GL_PIXEL_MAP_B_TO_B_SIZE : exact-integer?
GL_PIXEL_MAP_A_TO_A_SIZE : exact-integer?
GL_PIXEL_MAP_S_TO_S : exact-integer?
GL_PIXEL_MAP_I_TO_I : exact-integer?
GL_PIXEL_MAP_I_TO_R : exact-integer?
GL_PIXEL_MAP_I_TO_G : exact-integer?
GL_PIXEL_MAP_I_TO_B : exact-integer?
GL_PIXEL_MAP_I_TO_A : exact-integer?
GL_PIXEL_MAP_R_TO_R : exact-integer?
GL_PIXEL_MAP_G_TO_G : exact-integer?
GL_PIXEL_MAP_B_TO_B : exact-integer?
```

```
GL_PIXEL_MAP_A_TO_A : exact-integer?
GL_PACK_ALIGNMENT : exact-integer?
GL_PACK_LSB_FIRST : exact-integer?
GL_PACK_ROW_LENGTH : exact-integer?
GL_PACK_SKIP_PIXELS : exact-integer?
GL_PACK_SKIP_ROWS : exact-integer?
GL_PACK_SWAP_BYTES : exact-integer?
GL_UNPACK_ALIGNMENT : exact-integer?
GL_UNPACK_LSB_FIRST : exact-integer?
GL_UNPACK_ROW_LENGTH : exact-integer?
GL_UNPACK_SKIP_PIXELS : exact-integer?
GL_UNPACK_SKIP_ROWS : exact-integer?
GL_UNPACK_SWAP_BYTES : exact-integer?
GL_ZOOM_X : exact-integer?
GL_ZOOM_Y : exact-integer?
GL_TEXTURE_ENV : exact-integer?
GL_TEXTURE_ENV_MODE : exact-integer?
GL_TEXTURE_1D : exact-integer?
GL_TEXTURE_2D : exact-integer?
GL_TEXTURE_WRAP_S : exact-integer?
GL_TEXTURE_WRAP_T : exact-integer?
GL_TEXTURE_MAG_FILTER : exact-integer?
GL_TEXTURE_MIN_FILTER : exact-integer?
GL_TEXTURE_ENV_COLOR : exact-integer?
GL_TEXTURE_GEN_S : exact-integer?
GL_TEXTURE_GEN_T : exact-integer?
GL_TEXTURE_GEN_MODE : exact-integer?
GL_TEXTURE_BORDER_COLOR : exact-integer?
GL_TEXTURE_WIDTH : exact-integer?
GL_TEXTURE_HEIGHT : exact-integer?
GL_TEXTURE_BORDER : exact-integer?
GL_TEXTURE_COMPONENTS : exact-integer?
GL_TEXTURE_RED_SIZE : exact-integer?
GL_TEXTURE_GREEN_SIZE : exact-integer?
GL_TEXTURE_BLUE_SIZE : exact-integer?
GL_TEXTURE_ALPHA_SIZE : exact-integer?
GL_TEXTURE_LUMINANCE_SIZE : exact-integer?
GL_TEXTURE_INTENSITY_SIZE : exact-integer?
GL_NEAREST_MIPMAP_NEAREST : exact-integer?
GL_NEAREST_MIPMAP_LINEAR : exact-integer?
GL_LINEAR_MIPMAP_NEAREST : exact-integer?
GL_LINEAR_MIPMAP_LINEAR : exact-integer?
GL_OBJECT_LINEAR : exact-integer?
GL_OBJECT_PLANE : exact-integer?
```

```
GL_EYE_LINEAR : exact-integer?
GL_EYE_PLANE : exact-integer?
GL_SPHERE_MAP : exact-integer?
GL_DECAL : exact-integer?
GL_MODULATE : exact-integer?
GL_NEAREST : exact-integer?
GL_REPEAT : exact-integer?
GL_CLAMP : exact-integer?
GL_S : exact-integer?
GL_T : exact-integer?
GL_R : exact-integer?
GL_Q : exact-integer?
GL_TEXTURE_GEN_R : exact-integer?
GL_TEXTURE_GEN_Q : exact-integer?
GL_VENDOR : exact-integer?
GL_RENDERER : exact-integer?
GL_VERSION : exact-integer?
GL_EXTENSIONS : exact-integer?
GL_NO_ERROR : exact-integer?
GL_INVALID_VALUE : exact-integer?
GL_INVALID_ENUM : exact-integer?
GL_INVALID_OPERATION : exact-integer?
GL_STACK_OVERFLOW : exact-integer?
GL_STACK_UNDERFLOW : exact-integer?
GL_OUT_OF_MEMORY : exact-integer?
GL_CURRENT_BIT : exact-integer?
GL_POINT_BIT : exact-integer?
GL_LINE_BIT : exact-integer?
GL_POLYGON_BIT : exact-integer?
GL_POLYGON_STIPPLE_BIT : exact-integer?
GL_PIXEL_MODE_BIT : exact-integer?
GL_LIGHTING_BIT : exact-integer?
GL_FOG_BIT : exact-integer?
GL_DEPTH_BUFFER_BIT : exact-integer?
GL_ACCUM_BUFFER_BIT : exact-integer?
GL_STENCIL_BUFFER_BIT : exact-integer?
GL_VIEWPORT_BIT : exact-integer?
GL_TRANSFORM_BIT : exact-integer?
GL_ENABLE_BIT : exact-integer?
GL_COLOR_BUFFER_BIT : exact-integer?
GL_HINT_BIT : exact-integer?
GL_EVAL_BIT : exact-integer?
GL_LIST_BIT : exact-integer?
GL_TEXTURE_BIT : exact-integer?
```

```
GL_SCISSOR_BIT : exact-integer?
GL_ALL_ATTRIB_BITS : exact-integer?
GL_PROXY_TEXTURE_1D : exact-integer?
GL_PROXY_TEXTURE_2D : exact-integer?
GL_TEXTURE_PRIORITY : exact-integer?
GL_TEXTURE_RESIDENT : exact-integer?
GL_TEXTURE_BINDING_1D : exact-integer?
GL_TEXTURE_BINDING_2D : exact-integer?
GL_TEXTURE_INTERNAL_FORMAT : exact-integer?
GL_ALPHA4 : exact-integer?
GL_ALPHA8 : exact-integer?
GL_ALPHA12 : exact-integer?
GL_ALPHA16 : exact-integer?
GL_LUMINANCE4 : exact-integer?
GL_LUMINANCE8 : exact-integer?
GL_LUMINANCE12 : exact-integer?
GL_LUMINANCE16 : exact-integer?
GL_LUMINANCE4_ALPHA4 : exact-integer?
GL_LUMINANCE6_ALPHA2 : exact-integer?
GL_LUMINANCE8_ALPHA8 : exact-integer?
GL_LUMINANCE12_ALPHA4 : exact-integer?
GL_LUMINANCE12_ALPHA12 : exact-integer?
GL_LUMINANCE16_ALPHA16 : exact-integer?
GL_INTENSITY : exact-integer?
GL_INTENSITY4 : exact-integer?
GL_INTENSITY8 : exact-integer?
GL_INTENSITY12 : exact-integer?
GL_INTENSITY16 : exact-integer?
GL_R3_G3_B2 : exact-integer?
GL_RGB4 : exact-integer?
GL_RGB5 : exact-integer?
GL_RGB8 : exact-integer?
GL_RGB10 : exact-integer?
GL_RGB12 : exact-integer?
GL_RGB16 : exact-integer?
GL_RGBA2 : exact-integer?
GL_RGBA4 : exact-integer?
GL_RGB5_A1 : exact-integer?
GL_RGBA8 : exact-integer?
GL_RGB10_A2 : exact-integer?
GL_RGBA12 : exact-integer?
GL_RGBA16 : exact-integer?
GL_CLIENT_PIXEL_STORE_BIT : exact-integer?
GL_CLIENT_VERTEX_ARRAY_BIT : exact-integer?
```

```
GL_ALL_CLIENT_ATTRIB_BITS : exact-integer?
GL_CLIENT_ALL_ATTRIB_BITS : exact-integer?
GL_UNSIGNED_BYTE_3_3_2 : exact-integer?
GL_UNSIGNED_SHORT_4_4_4_4 : exact-integer?
GL_UNSIGNED_SHORT_5_5_5_1 : exact-integer?
GL_UNSIGNED_INT_8_8_8_8 : exact-integer?
GL_UNSIGNED_INT_10_10_10_2 : exact-integer?
GL_RESCALE_NORMAL : exact-integer?
GL_TEXTURE_BINDING_3D : exact-integer?
GL_PACK_SKIP_IMAGES : exact-integer?
GL_PACK_IMAGE_HEIGHT : exact-integer?
GL_UNPACK_SKIP_IMAGES : exact-integer?
GL_UNPACK_IMAGE_HEIGHT : exact-integer?
GL_TEXTURE_3D : exact-integer?
GL_PROXY_TEXTURE_3D : exact-integer?
GL_TEXTURE_DEPTH : exact-integer?
GL_TEXTURE_WRAP_R : exact-integer?
GL_MAX_3D_TEXTURE_SIZE : exact-integer?
GL_UNSIGNED_BYTE_2_3_3_REV : exact-integer?
GL_UNSIGNED_SHORT_5_6_5 : exact-integer?
GL_UNSIGNED_SHORT_5_6_5_REV : exact-integer?
GL_UNSIGNED_SHORT_4_4_4_4_REV : exact-integer?
GL_UNSIGNED_SHORT_1_5_5_5_REV : exact-integer?
GL_UNSIGNED_INT_8_8_8_8_REV : exact-integer?
GL_UNSIGNED_INT_2_10_10_10_REV : exact-integer?
GL_BGR : exact-integer?
GL_BGRA : exact-integer?
GL_MAX_ELEMENTS_VERTICES : exact-integer?
GL_MAX_ELEMENTS_INDICES : exact-integer?
GL_CLAMP_TO_EDGE : exact-integer?
GL_TEXTURE_MIN_LOD : exact-integer?
GL_TEXTURE_MAX_LOD : exact-integer?
GL_TEXTURE_BASE_LEVEL : exact-integer?
GL_TEXTURE_MAX_LEVEL : exact-integer?
GL_LIGHT_MODEL_COLOR_CONTROL : exact-integer?
GL_SINGLE_COLOR : exact-integer?
GL_SEPARATE_SPECULAR_COLOR : exact-integer?
GL_SMOOTH_POINT_SIZE_RANGE : exact-integer?
GL_SMOOTH_POINT_SIZE_GRANULARITY : exact-integer?
GL_SMOOTH_LINE_WIDTH_RANGE : exact-integer?
GL_SMOOTH_LINE_WIDTH_GRANULARITY : exact-integer?
GL_ALIASED_POINT_SIZE_RANGE : exact-integer?
GL_ALIASED_LINE_WIDTH_RANGE : exact-integer?
GL_CONSTANT_COLOR : exact-integer?
```

```
GL_ONE_MINUS_CONSTANT_COLOR : exact-integer?
GL_CONSTANT_ALPHA : exact-integer?
GL_ONE_MINUS_CONSTANT_ALPHA : exact-integer?
GL_BLEND_COLOR : exact-integer?
GL_FUNC_ADD : exact-integer?
GL_MIN : exact-integer?
GL_MAX : exact-integer?
GL_BLEND_EQUATION : exact-integer?
GL_FUNC_SUBTRACT : exact-integer?
GL_FUNC_REVERSE_SUBTRACT : exact-integer?
GL_CONVOLUTION_1D : exact-integer?
GL_CONVOLUTION_2D : exact-integer?
GL_SEPARABLE_2D : exact-integer?
GL_CONVOLUTION_BORDER_MODE : exact-integer?
GL_CONVOLUTION_FILTER_SCALE : exact-integer?
GL_CONVOLUTION_FILTER_BIAS : exact-integer?
GL_REDUCE : exact-integer?
GL_CONVOLUTION_FORMAT : exact-integer?
GL_CONVOLUTION_WIDTH : exact-integer?
GL_CONVOLUTION_HEIGHT : exact-integer?
GL_MAX_CONVOLUTION_WIDTH : exact-integer?
GL_MAX_CONVOLUTION_HEIGHT : exact-integer?
GL_POST_CONVOLUTION_RED_SCALE : exact-integer?
GL_POST_CONVOLUTION_GREEN_SCALE : exact-integer?
GL_POST_CONVOLUTION_BLUE_SCALE : exact-integer?
GL_POST_CONVOLUTION_ALPHA_SCALE : exact-integer?
GL_POST_CONVOLUTION_RED_BIAS : exact-integer?
GL_POST_CONVOLUTION_GREEN_BIAS : exact-integer?
GL_POST_CONVOLUTION_BLUE_BIAS : exact-integer?
GL_POST_CONVOLUTION_ALPHA_BIAS : exact-integer?
GL_HISTOGRAM : exact-integer?
GL_PROXY_HISTOGRAM : exact-integer?
GL_HISTOGRAM_WIDTH : exact-integer?
GL_HISTOGRAM_FORMAT : exact-integer?
GL_HISTOGRAM_RED_SIZE : exact-integer?
GL_HISTOGRAM_GREEN_SIZE : exact-integer?
GL_HISTOGRAM_BLUE_SIZE : exact-integer?
GL_HISTOGRAM_ALPHA_SIZE : exact-integer?
GL_HISTOGRAM_LUMINANCE_SIZE : exact-integer?
GL_HISTOGRAM_SINK : exact-integer?
GL_MINMAX : exact-integer?
GL_MINMAX_FORMAT : exact-integer?
GL_MINMAX_SINK : exact-integer?
GL_TABLE_TOO_LARGE : exact-integer?
```

```
GL_COLOR_MATRIX : exact-integer?
GL_COLOR_MATRIX_STACK_DEPTH : exact-integer?
GL_MAX_COLOR_MATRIX_STACK_DEPTH : exact-integer?
GL_POST_COLOR_MATRIX_RED_SCALE : exact-integer?
GL_POST_COLOR_MATRIX_GREEN_SCALE : exact-integer?
GL_POST_COLOR_MATRIX_BLUE_SCALE : exact-integer?
GL_POST_COLOR_MATRIX_ALPHA_SCALE : exact-integer?
GL_POST_COLOR_MATRIX_RED_BIAS : exact-integer?
GL_POST_COLOR_MATRIX_GREEN_BIAS : exact-integer?
GL_POST_COLOR_MATRIX_BLUE_BIAS : exact-integer?
GL_POST_COLOR_MATRIX_ALPHA_BIAS : exact-integer?
GL_COLOR_TABLE : exact-integer?
GL_POST_CONVOLUTION_COLOR_TABLE : exact-integer?
GL_POST_COLOR_MATRIX_COLOR_TABLE : exact-integer?
GL_PROXY_COLOR_TABLE : exact-integer?
GL_PROXY_POST_CONVOLUTION_COLOR_TABLE : exact-integer?
GL_PROXY_POST_COLOR_MATRIX_COLOR_TABLE : exact-integer?
GL_COLOR_TABLE_SCALE : exact-integer?
GL_COLOR_TABLE_BIAS : exact-integer?
GL_COLOR_TABLE_FORMAT : exact-integer?
GL_COLOR_TABLE_WIDTH : exact-integer?
GL_COLOR_TABLE_RED_SIZE : exact-integer?
GL_COLOR_TABLE_GREEN_SIZE : exact-integer?
GL_COLOR_TABLE_BLUE_SIZE : exact-integer?
GL_COLOR_TABLE_ALPHA_SIZE : exact-integer?
GL_COLOR_TABLE_LUMINANCE_SIZE : exact-integer?
GL_COLOR_TABLE_INTENSITY_SIZE : exact-integer?
GL_CONSTANT_BORDER : exact-integer?
GL_REPLICATE_BORDER : exact-integer?
GL_CONVOLUTION_BORDER_COLOR : exact-integer?
GL_TEXTURE0 : exact-integer?
GL_TEXTURE1 : exact-integer?
GL_TEXTURE2 : exact-integer?
GL_TEXTURE3 : exact-integer?
GL_TEXTURE4 : exact-integer?
GL_TEXTURE5 : exact-integer?
GL_TEXTURE6 : exact-integer?
GL_TEXTURE7 : exact-integer?
GL_TEXTURE8 : exact-integer?
GL_TEXTURE9 : exact-integer?
GL_TEXTURE10 : exact-integer?
GL_TEXTURE11 : exact-integer?
GL_TEXTURE12 : exact-integer?
GL_TEXTURE13 : exact-integer?
```

```
GL_TEXTURE14 : exact-integer?
GL_TEXTURE15 : exact-integer?
GL_TEXTURE16 : exact-integer?
GL_TEXTURE17 : exact-integer?
GL_TEXTURE18 : exact-integer?
GL_TEXTURE19 : exact-integer?
GL_TEXTURE20 : exact-integer?
GL_TEXTURE21 : exact-integer?
GL_TEXTURE22 : exact-integer?
GL_TEXTURE23 : exact-integer?
GL_TEXTURE24 : exact-integer?
GL_TEXTURE25 : exact-integer?
GL_TEXTURE26 : exact-integer?
GL_TEXTURE27 : exact-integer?
GL_TEXTURE28 : exact-integer?
GL_TEXTURE29 : exact-integer?
GL_TEXTURE30 : exact-integer?
GL_TEXTURE31 : exact-integer?
GL_ACTIVE_TEXTURE : exact-integer?
GL_CLIENT_ACTIVE_TEXTURE : exact-integer?
GL_MAX_TEXTURE_UNITS : exact-integer?
GL_TRANSPOSE_MODELVIEW_MATRIX : exact-integer?
GL_TRANSPOSE_PROJECTION_MATRIX : exact-integer?
GL_TRANSPOSE_TEXTURE_MATRIX : exact-integer?
GL_TRANSPOSE_COLOR_MATRIX : exact-integer?
GL_MULTISAMPLE : exact-integer?
GL_SAMPLE_ALPHA_TO_COVERAGE : exact-integer?
GL_SAMPLE_ALPHA_TO_ONE : exact-integer?
GL_SAMPLE_COVERAGE : exact-integer?
GL_SAMPLE_BUFFERS : exact-integer?
GL_SAMPLES : exact-integer?
GL_SAMPLE_COVERAGE_VALUE : exact-integer?
GL_SAMPLE_COVERAGE_INVERT : exact-integer?
GL_MULTISAMPLE_BIT : exact-integer?
GL_NORMAL_MAP : exact-integer?
GL_REFLECTION_MAP : exact-integer?
GL_TEXTURE_CUBE_MAP : exact-integer?
GL_TEXTURE_BINDING_CUBE_MAP : exact-integer?
GL_TEXTURE_CUBE_MAP_POSITIVE_X : exact-integer?
GL_TEXTURE_CUBE_MAP_NEGATIVE_X : exact-integer?
GL_TEXTURE_CUBE_MAP_POSITIVE_Y : exact-integer?
GL_TEXTURE_CUBE_MAP_NEGATIVE_Y : exact-integer?
GL_TEXTURE_CUBE_MAP_POSITIVE_Z : exact-integer?
GL_TEXTURE_CUBE_MAP_NEGATIVE_Z : exact-integer?
```

```
GL_PROXY_TEXTURE_CUBE_MAP : exact-integer?
GL_MAX_CUBE_MAP_TEXTURE_SIZE : exact-integer?
GL_COMPRESSED_ALPHA : exact-integer?
GL_COMPRESSED_LUMINANCE : exact-integer?
GL_COMPRESSED_LUMINANCE_ALPHA : exact-integer?
GL_COMPRESSED_INTENSITY : exact-integer?
GL_COMPRESSED_RGB : exact-integer?
GL_COMPRESSED_RGBA : exact-integer?
GL_TEXTURE_COMPRESSION_HINT : exact-integer?
GL_TEXTURE_COMPRESSED_IMAGE_SIZE : exact-integer?
GL_TEXTURE_COMPRESSED : exact-integer?
GL_NUM_COMPRESSED_TEXTURE_FORMATS : exact-integer?
GL_COMPRESSED_TEXTURE_FORMATS : exact-integer?
GL_CLAMP_TO_BORDER : exact-integer?
GL_COMBINE : exact-integer?
GL_COMBINE_RGB : exact-integer?
GL_COMBINE_ALPHA : exact-integer?
GL_SOURCE0_RGB : exact-integer?
GL_SOURCE1_RGB : exact-integer?
GL_SOURCE2_RGB : exact-integer?
GL_SOURCE0_ALPHA : exact-integer?
GL_SOURCE1_ALPHA : exact-integer?
GL_SOURCE2_ALPHA : exact-integer?
GL_OPERAND0_RGB : exact-integer?
GL_OPERAND1_RGB : exact-integer?
GL_OPERAND2_RGB : exact-integer?
GL_OPERAND0_ALPHA : exact-integer?
GL_OPERAND1_ALPHA : exact-integer?
GL_OPERAND2_ALPHA : exact-integer?
GL_RGB_SCALE : exact-integer?
GL_ADD_SIGNED : exact-integer?
GL_INTERPOLATE : exact-integer?
GL_SUBTRACT : exact-integer?
GL_CONSTANT : exact-integer?
GL_PRIMARY_COLOR : exact-integer?
GL_PREVIOUS : exact-integer?
GL_DOT3_RGB : exact-integer?
GL_DOT3_RGBA : exact-integer?
GL_BLEND_DST_RGB : exact-integer?
GL_BLEND_SRC_RGB : exact-integer?
GL_BLEND_DST_ALPHA : exact-integer?
GL_BLEND_SRC_ALPHA : exact-integer?
GL_POINT_SIZE_MIN : exact-integer?
GL_POINT_SIZE_MAX : exact-integer?
```

```
GL_POINT_FADE_THRESHOLD_SIZE : exact-integer?
GL_POINT_DISTANCE_ATTENUATION : exact-integer?
GL_GENERATE_MIPMAP : exact-integer?
GL_GENERATE_MIPMAP_HINT : exact-integer?
GL_DEPTH_COMPONENT16 : exact-integer?
GL_DEPTH_COMPONENT24 : exact-integer?
GL_DEPTH_COMPONENT32 : exact-integer?
GL_MIRRORED_REPEAT : exact-integer?
GL_FOG_COORDINATE_SOURCE : exact-integer?
GL_FOG_COORDINATE : exact-integer?
GL_FRAGMENT_DEPTH : exact-integer?
GL_CURRENT_FOG_COORDINATE : exact-integer?
GL_FOG_COORDINATE_ARRAY_TYPE : exact-integer?
GL_FOG_COORDINATE_ARRAY_STRIDE : exact-integer?
GL_FOG_COORDINATE_ARRAY_POINTER : exact-integer?
GL_FOG_COORDINATE_ARRAY : exact-integer?
GL_COLOR_SUM : exact-integer?
GL_CURRENT_SECONDARY_COLOR : exact-integer?
GL_SECONDARY_COLOR_ARRAY_SIZE : exact-integer?
GL_SECONDARY_COLOR_ARRAY_TYPE : exact-integer?
GL_SECONDARY_COLOR_ARRAY_STRIDE : exact-integer?
GL_SECONDARY_COLOR_ARRAY_POINTER : exact-integer?
GL_SECONDARY_COLOR_ARRAY : exact-integer?
GL_MAX_TEXTURE_LOD_BIAS : exact-integer?
GL_TEXTURE_FILTER_CONTROL : exact-integer?
GL_TEXTURE_LOD_BIAS : exact-integer?
GL_INCR_WRAP : exact-integer?
GL_DECR_WRAP : exact-integer?
GL_TEXTURE_DEPTH_SIZE : exact-integer?
GL_DEPTH_TEXTURE_MODE : exact-integer?
GL_TEXTURE_COMPARE_MODE : exact-integer?
GL_TEXTURE_COMPARE_FUNC : exact-integer?
GL_COMPARE_R_TO_TEXTURE : exact-integer?
GL_BUFFER_SIZE : exact-integer?
GL_BUFFER_USAGE : exact-integer?
GL_QUERY_COUNTER_BITS : exact-integer?
GL_CURRENT_QUERY : exact-integer?
GL_QUERY_RESULT : exact-integer?
GL_QUERY_RESULT_AVAILABLE : exact-integer?
GL_ARRAY_BUFFER : exact-integer?
GL_ELEMENT_ARRAY_BUFFER : exact-integer?
GL_ARRAY_BUFFER_BINDING : exact-integer?
GL_ELEMENT_ARRAY_BUFFER_BINDING : exact-integer?
GL_VERTEX_ARRAY_BUFFER_BINDING : exact-integer?
```

```
GL_NORMAL_ARRAY_BUFFER_BINDING : exact-integer?
GL_COLOR_ARRAY_BUFFER_BINDING : exact-integer?
GL_INDEX_ARRAY_BUFFER_BINDING : exact-integer?
GL_TEXTURE_COORD_ARRAY_BUFFER_BINDING : exact-integer?
GL_EDGE_FLAG_ARRAY_BUFFER_BINDING : exact-integer?
GL_SECONDARY_COLOR_ARRAY_BUFFER_BINDING : exact-integer?
GL_FOG_COORDINATE_ARRAY_BUFFER_BINDING : exact-integer?
GL_WEIGHT_ARRAY_BUFFER_BINDING : exact-integer?
GL_VERTEX_ATTRIB_ARRAY_BUFFER_BINDING : exact-integer?
GL_READ_ONLY : exact-integer?
GL_WRITE_ONLY : exact-integer?
GL_READ_WRITE : exact-integer?
GL_BUFFER_ACCESS : exact-integer?
GL_BUFFER_MAPPED : exact-integer?
GL_BUFFER_MAP_POINTER : exact-integer?
GL_STREAM_DRAW : exact-integer?
GL_STREAM_READ : exact-integer?
GL_STREAM_COPY : exact-integer?
GL_STATIC_DRAW : exact-integer?
GL_STATIC_READ : exact-integer?
GL_STATIC_COPY : exact-integer?
GL_DYNAMIC_DRAW : exact-integer?
GL_DYNAMIC_READ : exact-integer?
GL_DYNAMIC_COPY : exact-integer?
GL_SAMPLES_PASSED : exact-integer?
GL_FOG_COORD_SRC : exact-integer?
GL_FOG_COORD : exact-integer?
GL_CURRENT_FOG_COORD : exact-integer?
GL_FOG_COORD_ARRAY_TYPE : exact-integer?
GL_FOG_COORD_ARRAY_STRIDE : exact-integer?
GL_FOG_COORD_ARRAY_POINTER : exact-integer?
GL_FOG_COORD_ARRAY : exact-integer?
GL_FOG_COORD_ARRAY_BUFFER_BINDING : exact-integer?
GL_SRC0_RGB : exact-integer?
GL_SRC1_RGB : exact-integer?
GL_SRC2_RGB : exact-integer?
GL_SRC0_ALPHA : exact-integer?
GL_SRC1_ALPHA : exact-integer?
GL_SRC2_ALPHA : exact-integer?
GLU_FALSE : exact-integer?
GLU_TRUE : exact-integer?
GLU_VERSION : exact-integer?
GLU_EXTENSIONS : exact-integer?
GLU_INVALID_ENUM : exact-integer?
```

```
GLU_INVALID_VALUE : exact-integer?
GLU_OUT_OF_MEMORY : exact-integer?
GLU_INVALID_OPERATION : exact-integer?
GLU_OUTLINE_POLYGON : exact-integer?
GLU_OUTLINE_PATCH : exact-integer?
GLU_NURBS_ERROR : exact-integer?
GLU_ERROR : exact-integer?
GLU_NURBS_BEGIN : exact-integer?
GLU_NURBS_BEGIN_EXT : exact-integer?
GLU_NURBS_VERTEX : exact-integer?
GLU_NURBS_VERTEX_EXT : exact-integer?
GLU_NURBS_NORMAL : exact-integer?
GLU_NURBS_NORMAL_EXT : exact-integer?
GLU_NURBS_COLOR : exact-integer?
GLU_NURBS_COLOR_EXT : exact-integer?
GLU_NURBS_TEXTURE_COORD : exact-integer?
GLU_NURBS_TEX_COORD_EXT : exact-integer?
GLU_NURBS_END : exact-integer?
GLU_NURBS_END_EXT : exact-integer?
GLU_NURBS_BEGIN_DATA : exact-integer?
GLU_NURBS_BEGIN_DATA_EXT : exact-integer?
GLU_NURBS_VERTEX_DATA : exact-integer?
GLU_NURBS_VERTEX_DATA_EXT : exact-integer?
GLU_NURBS_NORMAL_DATA : exact-integer?
GLU_NURBS_NORMAL_DATA_EXT : exact-integer?
GLU_NURBS_COLOR_DATA : exact-integer?
GLU_NURBS_COLOR_DATA_EXT : exact-integer?
GLU_NURBS_TEXTURE_COORD_DATA : exact-integer?
GLU_NURBS_TEX_COORD_DATA_EXT : exact-integer?
GLU_NURBS_END_DATA : exact-integer?
GLU_NURBS_END_DATA_EXT : exact-integer?
GLU_NURBS_ERROR1 : exact-integer?
GLU_NURBS_ERROR2 : exact-integer?
GLU_NURBS_ERROR3 : exact-integer?
GLU_NURBS_ERROR4 : exact-integer?
GLU_NURBS_ERROR5 : exact-integer?
GLU_NURBS_ERROR6 : exact-integer?
GLU_NURBS_ERROR7 : exact-integer?
GLU_NURBS_ERROR8 : exact-integer?
GLU_NURBS_ERROR9 : exact-integer?
GLU_NURBS_ERROR10 : exact-integer?
GLU_NURBS_ERROR11 : exact-integer?
GLU_NURBS_ERROR12 : exact-integer?
GLU_NURBS_ERROR13 : exact-integer?
```

```
GLU_NURBS_ERROR14 : exact-integer?
GLU_NURBS_ERROR15 : exact-integer?
GLU_NURBS_ERROR16 : exact-integer?
GLU_NURBS_ERROR17 : exact-integer?
GLU_NURBS_ERROR18 : exact-integer?
GLU_NURBS_ERROR19 : exact-integer?
GLU_NURBS_ERROR20 : exact-integer?
GLU_NURBS_ERROR21 : exact-integer?
GLU_NURBS_ERROR22 : exact-integer?
GLU_NURBS_ERROR23 : exact-integer?
GLU_NURBS_ERROR24 : exact-integer?
GLU_NURBS_ERROR25 : exact-integer?
GLU_NURBS_ERROR26 : exact-integer?
GLU_NURBS_ERROR27 : exact-integer?
GLU_NURBS_ERROR28 : exact-integer?
GLU_NURBS_ERROR29 : exact-integer?
GLU_NURBS_ERROR30 : exact-integer?
GLU_NURBS_ERROR31 : exact-integer?
GLU_NURBS_ERROR32 : exact-integer?
GLU_NURBS_ERROR33 : exact-integer?
GLU_NURBS_ERROR34 : exact-integer?
GLU_NURBS_ERROR35 : exact-integer?
GLU_NURBS_ERROR36 : exact-integer?
GLU_NURBS_ERROR37 : exact-integer?
GLU_AUTO_LOAD_MATRIX : exact-integer?
GLU_CULLING : exact-integer?
GLU_SAMPLING_TOLERANCE : exact-integer?
GLU_DISPLAY_MODE : exact-integer?
GLU_PARAMETRIC_TOLERANCE : exact-integer?
GLU_SAMPLING_METHOD : exact-integer?
GLU_U_STEP : exact-integer?
GLU_V_STEP : exact-integer?
GLU_NURBS_MODE : exact-integer?
GLU_NURBS_MODE_EXT : exact-integer?
GLU_NURBS_TESSELLATOR : exact-integer?
GLU_NURBS_TESSELLATOR_EXT : exact-integer?
GLU_NURBS_RENDERER : exact-integer?
GLU_NURBS_RENDERER_EXT : exact-integer?
GLU_OBJECT_PARAMETRIC_ERROR : exact-integer?
GLU_OBJECT_PARAMETRIC_ERROR_EXT : exact-integer?
GLU_OBJECT_PATH_LENGTH : exact-integer?
GLU_OBJECT_PATH_LENGTH_EXT : exact-integer?
GLU_PATH_LENGTH : exact-integer?
GLU_PARAMETRIC_ERROR : exact-integer?
```

```
GLU_DOMAIN_DISTANCE : exact-integer?
GLU_MAP1_TRIM_2 : exact-integer?
GLU_MAP1_TRIM_3 : exact-integer?
GLU_POINT : exact-integer?
GLU_LINE : exact-integer?
GLU_FILL : exact-integer?
GLU_SILHOUETTE : exact-integer?
GLU_SMOOTH : exact-integer?
GLU_FLAT : exact-integer?
GLU_NONE : exact-integer?
GLU_OUTSIDE : exact-integer?
GLU_INSIDE : exact-integer?
GLU_TESS_BEGIN : exact-integer?
GLU_BEGIN : exact-integer?
GLU_TESS_VERTEX : exact-integer?
GLU_VERTEX : exact-integer?
GLU_TESS_END : exact-integer?
GLU_END : exact-integer?
GLU_TESS_ERROR : exact-integer?
GLU_TESS_EDGE_FLAG : exact-integer?
GLU_EDGE_FLAG : exact-integer?
GLU_TESS_COMBINE : exact-integer?
GLU_TESS_BEGIN_DATA : exact-integer?
GLU_TESS_VERTEX_DATA : exact-integer?
GLU_TESS_END_DATA : exact-integer?
GLU_TESS_ERROR_DATA : exact-integer?
GLU_TESS_EDGE_FLAG_DATA : exact-integer?
GLU_TESS_COMBINE_DATA : exact-integer?
GLU_CW : exact-integer?
GLU_CCW : exact-integer?
GLU_INTERIOR : exact-integer?
GLU_EXTERIOR : exact-integer?
GLU_UNKNOWN : exact-integer?
GLU_TESS_WINDING_RULE : exact-integer?
GLU_TESS_BOUNDARY_ONLY : exact-integer?
GLU_TESS_TOLERANCE : exact-integer?
GLU_TESS_ERROR1 : exact-integer?
GLU_TESS_ERROR2 : exact-integer?
GLU_TESS_ERROR3 : exact-integer?
GLU_TESS_ERROR4 : exact-integer?
GLU_TESS_ERROR5 : exact-integer?
GLU_TESS_ERROR6 : exact-integer?
GLU_TESS_ERROR7 : exact-integer?
GLU_TESS_ERROR8 : exact-integer?
```

```
GLU_TESS_MISSING_BEGIN_POLYGON : exact-integer?
GLU_TESS_MISSING_BEGIN_CONTOUR : exact-integer?
GLU_TESS_MISSING_END_POLYGON : exact-integer?
GLU_TESS_MISSING_END_CONTOUR : exact-integer?
GLU_TESS_COORD_TOO_LARGE : exact-integer?
GLU_TESS_NEED_COMBINE_CALLBACK : exact-integer?
GLU_TESS_WINDING_ODD : exact-integer?
GLU_TESS_WINDING_NONZERO : exact-integer?
GLU_TESS_WINDING_POSITIVE : exact-integer?
GLU_TESS_WINDING_NEGATIVE : exact-integer?
GLU_TESS_WINDING_ABS_GEQ_TWO : exact-integer?
GLU_TESS_MAX_COORD : real?
```

All OpenGL-defined constants.

```
(feedback-buffer->gl-float-vector buf) → gl-float-vector?
  buf : feedback-buffer-object?
```

Converts a result from `glFeedbackBuffer` to a vector.

```
(select-buffer->gl-uint-vector buf) → gl-uint-vector?
  buf : select-buffer-object?
```

Converts a result from `glSelectBuffer` to a vector.

3 Racket-Style OpenGL

`(require sgl)` package: `sgl`

The functions in `sgl` use Racket style names instead of C style names. To convert a C OpenGL name to a Racket OpenGL name, change the `gl` prefix to `gl-`, separate adjacent words with hyphens, and convert to all lower case. Functions that have several variants to accommodate different numbers and types of arguments are collapsed into one or two functions in `sgl`. For example, `sgl` provides two vertex functions: `vertex` and `vertex-v`. The `vertex` function accepts 2, 3 or 4 numerical arguments, and the `vertex-v` function accepts `gl-vector`s of length 2, 3 or 4. The C language OpenGL interface, in contrast, has 24 vertex functions: `glVertex3i`, `glVertex4fv`, etc.

Functions in `sgl` take symbols instead of integers for `GLenum` arguments. Each function checks that the given symbol is an acceptable argument and raises an exception if it is not. Given the name of a C-language `#define` constant, determine the corresponding symbolic argument by removing the leading `GL_`, converting the letters to lower-case and replacing each `_` with `-`. For example, `GL_TRIANGLES` becomes `'triangles`, and `GL_TRIANGLE_STRIP` becomes `'triangle-strip`. Additionally, the functions check the length of any array arguments to ensure that OpenGL does not attempt to write or read after the array.

The `sgl` module is not as complete as the `sgl/gl` module.

Examples:

```
(require sgl sgl/gl-vectors)
(gl-begin 'triangles)
(gl-vertex 1 2 3)
(gl-vertex-v (gl-float-vector 1 2 3 4))
(gl-end)
```

```
(struct gl-selection-record (min-z max-z stack)
    #:extra-constructor-name make-gl-selection-record)
  min-z : real?
  max-z : real?
  stack : ....
```

Represents a selection.

```
gl-accum : procedure?
gl-active-texture : procedure?
gl-alpha-func : procedure?
gl-begin : procedure?
gl-begin-query : procedure?
gl-blend-color : procedure?
```

```
gl-blend-equation : procedure?
gl-blend-func : procedure?
gl-blend-func-separate : procedure?
gl-call-list : procedure?
gl-check-extension : procedure?
gl-clear : procedure?
gl-clear-accum : procedure?
gl-clear-color : procedure?
gl-clear-depth : procedure?
gl-clear-index : procedure?
gl-clear-stencil : procedure?
gl-clip-plane : procedure?
gl-color : procedure?
gl-color-mask : procedure?
gl-color-material : procedure?
gl-color-v : procedure?
gl-copy-pixels : procedure?
gl-cull-face : procedure?
gl-cylinder : procedure?
gl-delete-lists : procedure?
gl-delete-queries : procedure?
gl-depth-func : procedure?
gl-depth-mask : procedure?
gl-depth-range : procedure?
gl-disable : procedure?
gl-disk : procedure?
gl-edge-flag : procedure?
gl-enable : procedure?
gl-end : procedure?
gl-end-list : procedure?
gl-end-query : procedure?
gl-eval-coord : procedure?
gl-eval-coord-v : procedure?
gl-eval-mesh : procedure?
gl-eval-point : procedure?
gl-feedback-buffer->gl-float-vector : procedure?
gl-finish : procedure?
gl-flush : procedure?
gl-front-face : procedure?
gl-frustum : procedure?
gl-gen-lists : procedure?
gl-gen-queries : procedure?
gl-get-error : procedure?
gl-get-string : procedure?
```

```
gl-hint : procedure?
gl-index : procedure?
gl-index-mask : procedure?
gl-index-v : procedure?
gl-init-names : procedure?
gl-is-buffer : procedure?
gl-is-enabled : procedure?
gl-is-list : procedure?
gl-is-query : procedure?
gl-light : procedure?
gl-light-model : procedure?
gl-light-model-v : procedure?
gl-light-v : procedure?
gl-line-stipple : procedure?
gl-line-width : procedure?
gl-list-base : procedure?
gl-load-identity : procedure?
gl-load-matrix : procedure?
gl-load-name : procedure?
gl-load-transpose-matrix : procedure?
gl-look-at : procedure?
gl-map-grid : procedure?
gl-material : procedure?
gl-material-v : procedure?
gl-matrix-mode : procedure?
gl-mult-matrix : procedure?
gl-mult-transpose-matrix : procedure?
gl-multi-tex-coord : procedure?
gl-multi-tex-coord-v : procedure?
gl-new-list : procedure?
gl-new-quadric : procedure?
gl-normal : procedure?
gl-normal-v : procedure?
gl-ortho : procedure?
gl-ortho-2d : procedure?
gl-partial-disk : procedure?
gl-pass-through : procedure?
gl-perspective : procedure?
gl-pick-matrix : procedure?
gl-pixel-store : procedure?
gl-point-parameter : procedure?
gl-point-parameter-v : procedure?
gl-point-size : procedure?
gl-polygon-mode : procedure?
```

```
gl-polygon-offset : procedure?
gl-pop-attrib : procedure?
gl-pop-client-attrib : procedure?
gl-pop-matrix : procedure?
gl-pop-name : procedure?
gl-project : procedure?
gl-push-matrix : procedure?
gl-push-name : procedure?
gl-quadric-draw-style : procedure?
gl-quadric-normals : procedure?
gl-quadric-orientation : procedure?
gl-quadric-texture : procedure?
gl-raster-pos : procedure?
gl-raster-pos-v : procedure?
gl-rect : procedure?
gl-rect-v : procedure?
gl-render-mode : procedure?
gl-rotate : procedure?
gl-sample-coverage : procedure?
gl-scale : procedure?
gl-scissor : procedure?
gl-secondary-color : procedure?
gl-secondary-color-v : procedure?
gl-select-buffer->gl-uint-vector : procedure?
gl-shade-model : procedure?
gl-sphere : procedure?
gl-stencil-func : procedure?
gl-stencil-mask : procedure?
gl-stencil-op : procedure?
gl-tex-coord : procedure?
gl-tex-coord-v : procedure?
gl-tex-gen : procedure?
gl-tex-gen-v : procedure?
gl-translate : procedure?
gl-u-get-string : procedure?
gl-un-project : procedure?
gl-un-project4 : procedure?
gl-vertex : procedure?
gl-vertex-v : procedure?
gl-viewport : procedure?
gl-window-pos : procedure?
gl-window-pos-v : procedure?
```

Racket-style variants of the OpenGL functions.

```
(gl-process-selection vec hits) → (listof gl-selection-record?)
  vec : gl-uint-vector?
  hits : exact-nonnegative-integer?
```

Parses the contents of *vec* from the format used by `glSelectBuffer`. The second argument should be the number of hits as returned by `glRenderMode`.

```
(gl-get-gl-version-number) → exact-nonnegative-integer?
```

Returns the run-time OpenGL version number as an integer: `10`, `11`, `12`, `13`, `14`, `15`, or `20`.

```
(gl-get-glu-version-number) → exact-nonnegative-integer?
```

Returns the run-time GLU version number as an integer: `10`, `11`, `12`, or `13`.

4 OpenGL Vectors

```
(require sgl/gl-vectors)
```
package: sgl

The sgl/gl-vectors module supports OpenGL programming with cvectors. In this document and in the error messages, a "gl-vector" is just a cvector, while a "gl-⟨*type*⟩-vector" is a cvector with an appropriate type. Use the sgl/gl-vectors module vectors instead of a Racket cvector directly, because they are specialized to handle the OpenGL types correctly.

```
(gl-vector? v) → boolean?
  v : any/c
(gl-vector->vector vec) → vector?
  vec : cvector?
(gl-vector->list vec) → list?
  vec : cvector?
(gl-vector-length vec) → exact-nonnegative-integer?
  vec : cvector?
(gl-vector-ref vec pos) → any/v
  vec : cvector?
  pos : exact-nonnegative-integer?
(gl-vector-set! vec pos v) → void?
  vec : cvector?
  pos : exact-nonnegative-integer?
  v : any/v
```

Synonyms for cvector?, cvector->vector, cvector-length, etc.

```
(gl-byte-vector? v) → boolean?
  v : any/c
(make-gl-byte-vector pos) → gl-byte-vector?
  pos : exact-nonnegative-integer?
(gl-byte-vector v ...) → gl-byte-vector?
  v : byte?
(vector->gl-byte-vector v ...) → gl-byte-vector?
  v : (vectorof byte?)
(list->gl-byte-vector v ...) → gl-byte-vector?
  v : (listof byte?)
(gl-byte-vector+ vec ...+) → gl-byte-vector?
  vec : gl-byte-vector?
(gl-byte-vector- vec ...+) → gl-byte-vector?
  vec : gl-byte-vector?
(gl-byte-vector* x vec) → gl-byte-vector?
  x : real?
  vec : gl-byte-vector?
```

Operations on vectors of byte elements. The gl-byte-vector+ and gl-byte-vector- functions compute the element-by-element sum and difference of the given vectors, respectively. The gl-byte-vector* function multiplies each element of *vec* by *x*.

```
(gl-ubyte-vector? v) → boolean?
  v : any/c
(make-gl-ubyte-vector pos) → gl-ubyte-vector?
  pos : exact-nonnegative-integer?
(gl-ubyte-vector v ...) → gl-ubyte-vector?
  v : ubyte?
(vector->gl-ubyte-vector v ...) → gl-ubyte-vector?
  v : (vectorof ubyte?)
(list->gl-ubyte-vector v ...) → gl-ubyte-vector?
  v : (listof ubyte?)
(gl-ubyte-vector+ vec ...+) → gl-ubyte-vector?
  vec : gl-ubyte-vector?
(gl-ubyte-vector- vec ...+) → gl-ubyte-vector?
  vec : gl-ubyte-vector?
(gl-ubyte-vector* x vec) → gl-ubyte-vector?
  x : real?
  vec : gl-ubyte-vector?
```

Operations on vectors of ubyte elements. The gl-ubyte-vector+ and gl-ubyte-vector- functions compute the element-by-element sum and difference of the given vectors, respectively. The gl-ubyte-vector* function multiplies each element of *vec* by *x*.

```
(gl-short-vector? v) → boolean?
  v : any/c
(make-gl-short-vector pos) → gl-short-vector?
  pos : exact-nonnegative-integer?
(gl-short-vector v ...) → gl-short-vector?
  v : short?
(vector->gl-short-vector v ...) → gl-short-vector?
  v : (vectorof short?)
(list->gl-short-vector v ...) → gl-short-vector?
  v : (listof short?)
(gl-short-vector+ vec ...+) → gl-short-vector?
  vec : gl-short-vector?
(gl-short-vector- vec ...+) → gl-short-vector?
  vec : gl-short-vector?
(gl-short-vector* x vec) → gl-short-vector?
  x : real?
  vec : gl-short-vector?
```

Operations on vectors of short elements. The gl-short-vector+ and gl-short-

vector- functions compute the element-by-element sum and difference of the given vectors, respectively. The gl-short-vector* function multiplies each element of *vec* by *x*.

```
(gl-ushort-vector? v) → boolean?
  v : any/c
(make-gl-ushort-vector pos) → gl-ushort-vector?
  pos : exact-nonnegative-integer?
(gl-ushort-vector v ...) → gl-ushort-vector?
  v : ushort?
(vector->gl-ushort-vector v ...) → gl-ushort-vector?
  v : (vectorof ushort?)
(list->gl-ushort-vector v ...) → gl-ushort-vector?
  v : (listof ushort?)
(gl-ushort-vector+ vec ...+) → gl-ushort-vector?
  vec : gl-ushort-vector?
(gl-ushort-vector- vec ...+) → gl-ushort-vector?
  vec : gl-ushort-vector?
(gl-ushort-vector* x vec) → gl-ushort-vector?
  x : real?
  vec : gl-ushort-vector?
```

Operations on vectors of ushort elements. The gl-ushort-vector+ and gl-ushort-vector- functions compute the element-by-element sum and difference of the given vectors, respectively. The gl-ushort-vector* function multiplies each element of *vec* by *x*.

```
(gl-int-vector? v) → boolean?
  v : any/c
(make-gl-int-vector pos) → gl-int-vector?
  pos : exact-nonnegative-integer?
(gl-int-vector v ...) → gl-int-vector?
  v : int?
(vector->gl-int-vector v ...) → gl-int-vector?
  v : (vectorof int?)
(list->gl-int-vector v ...) → gl-int-vector?
  v : (listof int?)
(gl-int-vector+ vec ...+) → gl-int-vector?
  vec : gl-int-vector?
(gl-int-vector- vec ...+) → gl-int-vector?
  vec : gl-int-vector?
(gl-int-vector* x vec) → gl-int-vector?
  x : real?
  vec : gl-int-vector?
```

Operations on vectors of int elements. The gl-int-vector+ and gl-int-vector- functions compute the element-by-element sum and difference of the given vectors, respectively.

The gl-int-vector* function multiplies each element of vec by x.

```
(gl-uint-vector? v) → boolean?
  v : any/c
(make-gl-uint-vector pos) → gl-uint-vector?
  pos : exact-nonnegative-integer?
(gl-uint-vector v ...) → gl-uint-vector?
  v : uint?
(vector->gl-uint-vector v ...) → gl-uint-vector?
  v : (vectorof uint?)
(list->gl-uint-vector v ...) → gl-uint-vector?
  v : (listof uint?)
(gl-uint-vector+ vec ...+) → gl-uint-vector?
  vec : gl-uint-vector?
(gl-uint-vector- vec ...+) → gl-uint-vector?
  vec : gl-uint-vector?
(gl-uint-vector* x vec) → gl-uint-vector?
  x : real?
  vec : gl-uint-vector?
```

Operations on vectors of uint elements. The gl-uint-vector+ and gl-uint-vector- functions compute the element-by-element sum and difference of the given vectors, respectively. The gl-uint-vector* function multiplies each element of vec by x.

```
(gl-float-vector? v) → boolean?
  v : any/c
(make-gl-float-vector pos) → gl-float-vector?
  pos : exact-nonnegative-integer?
(gl-float-vector v ...) → gl-float-vector?
  v : float?
(vector->gl-float-vector v ...) → gl-float-vector?
  v : (vectorof float?)
(list->gl-float-vector v ...) → gl-float-vector?
  v : (listof float?)
(gl-float-vector+ vec ...+) → gl-float-vector?
  vec : gl-float-vector?
(gl-float-vector- vec ...+) → gl-float-vector?
  vec : gl-float-vector?
(gl-float-vector* x vec) → gl-float-vector?
  x : real?
  vec : gl-float-vector?
```

Operations on vectors of float elements. The gl-float-vector+ and gl-float-vector- functions compute the element-by-element sum and difference of the given vectors, respectively. The gl-float-vector* function multiplies each element of vec by x.

```
(gl-double-vector? v) → boolean?
  v : any/c
(make-gl-double-vector pos) → gl-double-vector?
  pos : exact-nonnegative-integer?
(gl-double-vector v ...) → gl-double-vector?
  v : double?
(vector->gl-double-vector v ...) → gl-double-vector?
  v : (vectorof double?)
(list->gl-double-vector v ...) → gl-double-vector?
  v : (listof double?)
(gl-double-vector+ vec ...+) → gl-double-vector?
  vec : gl-double-vector?
(gl-double-vector- vec ...+) → gl-double-vector?
  vec : gl-double-vector?
(gl-double-vector* x vec) → gl-double-vector?
  x : real?
  vec : gl-double-vector?
```

Operations on vectors of `double` elements. The `gl-double-vector+` and `gl-double-vector-` functions compute the element-by-element sum and difference of the given vectors, respectively. The `gl-double-vector*` function multiplies each element of *vec* by *x*.

```
(gl-boolean-vector? v) → boolean?
  v : any/c
(make-gl-boolean-vector pos) → gl-boolean-vector?
  pos : exact-nonnegative-integer?
(gl-boolean-vector v ...) → gl-boolean-vector?
  v : boolean?
(vector->gl-boolean-vector v ...) → gl-boolean-vector?
  v : (vectorof boolean?)
(list->gl-boolean-vector v ...) → gl-boolean-vector?
  v : (listof boolean?)
(gl-boolean-vector+ vec ...+) → gl-boolean-vector?
  vec : gl-boolean-vector?
(gl-boolean-vector- vec ...+) → gl-boolean-vector?
  vec : gl-boolean-vector?
(gl-boolean-vector* x vec) → gl-boolean-vector?
  x : real?
  vec : gl-boolean-vector?
```

Operations on vectors of `boolean` elements. The `gl-boolean-vector+` and `gl-boolean-vector-` functions compute the element-by-element sum and difference of the given vectors, respectively. The `gl-boolean-vector*` function multiplies each element of *vec* by *x*.

```
(gl-vector-norm vec) → real?
  vec : gl-vector?
```

Returns the square root of the sum of the squares of the elements of *vec*.

5 Bitmaps

```
(require sgl/bitmap)
```
package: sgl

```
(bitmap->gl-list  bitmap
                  [#:with-gl with-gl-proc
                   #:mask mask])          → exact-integer?
  bitmap : (is-a?/c bitmap%)
  with-gl-proc : ((-> any) . -> . any) = (lambda (f) (f))
  mask : (or/c (is-a?/c bitmap%) false/c)
       = (send bitmap get-loaded-mask)
```

Converts the given bitmap into an OpenGL list that can be rendered with `gl-call-list` or `glCallList`. The rendered object is a square on the z=0 plane with corners at (0,0) and (1,1).

The *with-gl-proc* must accept a thunk and call it while the relevant OpenGL context is selected. Otherwise, the relevant OpenGL context must be selected already.

If *mask* is not `#f`, it is used as the mask bitmap for extracting alpha values.

6 Initialization

`(require sgl/init)` package: `sgl`

Requiring the `sgl/init` library initializes platform-specific OpenGL state to help avoid crashes when OpenGL commands are incorrectly used without a current context. This library is `required` by `sgl` and `sgl/gl`, so it normally does not need to be `required` explicitly.

On Mac OS X, `sgl/init` checks whether any GL context is current, and if not, it creates a dummy context and sets it as the current context.

Index