NAME

glLightModelf, glLightModeli, glLightModelfv, glLightModeliv – set the lighting model parameters

C SPECIFICATION

void **glLightModelf**(GLenum *pname*,
GLfloat *param*)
void **glLightModeli**(GLenum *pname*,
GLint *param*)

delim $$

PARAMETERS

pname Specifies a single-valued lighting model parameter. **GL_LIGHT_MODEL_LOCAL_VIEWER** and **GL_LIGHT_MODEL_TWO_SIDE** are accepted.

param Specifies the value that *param* will be set to.

C SPECIFICATION

void **glLightModelfv**(GLenum *pname*,
const GLfloat **params*)
void **glLightModeliv**(GLenum *pname*,
const GLint **params*)

PARAMETERS

pname Specifies a lighting model parameter. **GL_LIGHT_MODEL_AMBIENT**, **GL_LIGHT_MODEL_LOCAL_VIEWER**, and **GL_LIGHT_MODEL_TWO_SIDE** are accepted.

params Specifies a pointer to the value or values that *params* will be set to.

DESCRIPTION

glLightModel sets the lighting model parameter. *pname* names a parameter and *params* gives the new value. There are three lighting model parameters:

GL_LIGHT_MODEL_AMBIENT

params contains four integer or floating-point values that specify the ambient RGBA intensity of the entire scene. Integer values are mapped linearly such that the most positive representable value maps to 1.0, and the most negative representable value maps to −1.0. Floating-point values are mapped directly. Neither integer nor floating-point values are clamped. The initial ambient scene intensity is (0.2, 0.2, 0.2, 1.0).

GL_LIGHT_MODEL_LOCAL_VIEWER

params is a single integer or floating-point value that specifies how specular reflection angles are computed. If *params* is 0 (or 0.0), specular reflection angles take the view direction to be parallel to and in the direction of the -*z* axis, regardless of the location of the vertex in eye coordinates. Otherwise, specular reflections are computed from the origin of the eye coordinate system. The initial value is 0.

GL_LIGHT_MODEL_TWO_SIDE

params is a single integer or floating-point value that specifies whether one- or two-sided lighting calculations are done for polygons. It has no effect on the lighting calculations for points, lines, or bitmaps. If *params* is 0 (or 0.0), one-sided lighting is specified, and only the *front* material parameters are used in the lighting equation. Otherwise, two-sided lighting is specified. In this case, vertices of back-facing polygons are lighted using the *back* material parameters, and have their normals reversed before the lighting equation is evaluated. Vertices of front-facing polygons are always lighted using the *front* material parameters, with no change to their normals. The initial value is 0.

In RGBA mode, the lighted color of a vertex is the sum of the material emission intensity, the product of the material ambient reflectance and the lighting model full-scene ambient intensity, and the contribution of each enabled light source. Each light source contributes the sum of three terms: ambient, diffuse, and specular. The ambient light source contribution is the product of the material ambient reflectance and the light's ambient intensity. The diffuse light source contribution is the product of the material diffuse reflectance, the light's diffuse intensity, and the dot product of the vertex's normal with the normalized vector from the vertex to the light source. The specular light source contribution is the product of the material specular reflectance, the light's specular intensity, and the dot product of the normalized vertex-to-eye and vertex-to-light vectors, raised to the power of the shininess of the material. All three light source contributions are attenuated equally based on the distance from the vertex to the light source and on light source direction, spread exponent, and spread cutoff angle. All dot products are replaced with 0 if they evaluate to a negative value.

The alpha component of the resulting lighted color is set to the alpha value of the material diffuse reflectance.

In color index mode, the value of the lighted index of a vertex ranges from the ambient to the specular values passed to **glMaterial** using **GL_COLOR_INDEXES**. Diffuse and specular coefficients, computed with a (.30, .59, .11) weighting of the lights' colors, the shininess of the material, and the same reflection and attenuation equations as in the RGBA case, determine how much above ambient the resulting index is.

ERRORS

GL_INVALID_ENUM is generated if *pname* is not an accepted value.

GL_INVALID_OPERATION is generated if **glLightModel** is executed between the execution of **glBegin** and the corresponding execution of **glEnd**.

ASSOCIATED GETS

glGet with argument **GL_LIGHT_MODEL_AMBIENT**
glGet with argument **GL_LIGHT_MODEL_LOCAL_VIEWER**
glGet with argument **GL_LIGHT_MODEL_TWO_SIDE**
glIsEnabled with argument **GL_LIGHTING**

SEE ALSO

glLight, **glMaterial**