EXAMPLE: SETTING UP TEXTURE

Name

glGenTextures — generate texture names

C Specification

```
void glGenTextures( GLsizei n,
                    GLuint * textures );
```

Parameters

n

Specifies the number of texture names to be generated.

textures

Specifies an array in which the generated texture names are stored.

EXAMPLE: SETTING UP TEXTURE

```
GLuint textureID;
glGenTextures(1, &textureID);

…. LOAD IMAGE OR GENERATE TEXTURE ....
```

Name

glBindTexture — bind a named texture to a texturing target

C Specification

```
void glBindTexture( GLenum target,
                    GLuint texture );
```

Parameters

target

Specifies the target to which the texture is bound. Must be one of `GL_TEXTURE_1D`, `GL_TEXTURE_2D`, `GL_TEXTURE_3D`, `GL_TEXTURE_1D_ARRAY`, `GL_TEXTURE_2D_ARRAY`, `GL_TEXTURE_RECTANGLE`, `GL_TEXTURE_CUBE_MAP`, `GL_TEXTURE_CUBE_MAP_ARRAY`, `GL_TEXTURE_BUFFER`, `GL_TEXTURE_2D_MULTISAMPLE` or `GL_TEXTURE_2D_MULTISAMPLE_ARRAY`.

texture

Specifies the name of a texture.

EXAMPLE: SETTING UP TEXTURE

```
GLuint textureID;
glGenTextures(1, &textureID);

…. LOAD IMAGE OR GENERATE TEXTURE ....

 glBindTexture(GL_TEXTURE_2D, textureID);
```

Name

glTexImage2D — specify a two-dimensional texture image

C Specification

```
void glTexImage2D( GLenum target,
                   GLint level,
                   GLint internalFormat,
                   GLsizei width,
                   GLsizei height,
                   GLint border,
                   GLenum format,
                   GLenum type,
                   const GLvoid * data );
```

EXAMPLE: SETTING UP TEXTURE

```
GLuint textureID;
glGenTextures(1, &textureID);

…. LOAD IMAGE OR GENERATE TEXTURE ....

 glBindTexture(GL_TEXTURE_2D, textureID);
 glTexImage2D(GL_TEXTURE_2D, 0, format, width, height, 0, format,
              GL_UNSIGNED_BYTE, data);
```

Name

glGenerateMipmap, glGenerateTextureMipmap — generate mipmaps for a specified texture object

C Specification

void **glGenerateMipmap**(GLenum *target*);

void **glGenerateTextureMipmap**(GLuint *texture*);

Parameters

target

Specifies the target to which the texture object is bound for **glGenerateMipmap**. Must be one of `GL_TEXTURE_1D`, `GL_TEXTURE_2D`, `GL_TEXTURE_3D`, `GL_TEXTURE_1D_ARRAY`, `GL_TEXTURE_2D_ARRAY`, `GL_TEXTURE_CUBE_MAP`, or `GL_TEXTURE_CUBE_MAP_ARRAY`.

texture

Specifies the texture object name for **glGenerateTextureMipmap**.

EXAMPLE: SETTING UP TEXTURE

```
GLuint textureID;
glGenTextures(1, &textureID);

…. LOAD IMAGE OR GENERATE TEXTURE ....

 glBindTexture(GL_TEXTURE_2D, textureID);
 glTexImage2D(GL_TEXTURE_2D, 0, format, width, height, 0, format,
              GL_UNSIGNED_BYTE, data);

// if want to use mip mapping
 glGenerateMipmap(GL_TEXTURE_2D);
```

Name

glTexParameter, glTextureParameter — set texture parameters

C Specification

```
void glTexParameterf( GLenum target,
                      GLenum pname,
                      GLfloat param );

void glTexParameteri( GLenum target,
                      GLenum pname,
                      GLint param );
```

EXAMPLE: SETTING UP TEXTURE

```
GLuint textureID;
glGenTextures(1, &textureID);

…. LOAD IMAGE OR GENERATE TEXTURE ....

 glBindTexture(GL_TEXTURE_2D, textureID);
 glTexImage2D(GL_TEXTURE_2D, 0, format, width, height, 0, format,
              GL_UNSIGNED_BYTE, data);

// if want to use mip mapping
 glGenerateMipmap(GL_TEXTURE_2D);

glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_S, GL_REPEAT);
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_T, GL_REPEAT);
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_LINEAR_MIPMAP_LINEAR);
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_LINEAR);
```

EXAMPLE: USING A TEXTURE

EXAMPLE: USING A TEXTURE (FRAGMENT SHADER)

```
#version 330 core



out vec4 fragColor;



void main() {
   fragColor = vec4(1,0,0,1);
}
```

EXAMPLE: USING A TEXTURE (FRAGMENT SHADER)

```
#version 330 core

in vec2 ourTexCoord;

out vec4 fragColor;



void main() {
   fragColor = vec4(1,0,0,1);
}
```

EXAMPLE: USING A TEXTURE (FRAGMENT SHADER)

```
#version 330 core

in vec2 ourTexCoord;

out vec4 fragColor;

uniform sampler2D ourTexture;

void main() {
   fragColor = vec4(1,0,0,1);
}
```

EXAMPLE: USING A TEXTURE (FRAGMENT SHADER)

```
#version 330 core

in vec2 ourTexCoord;

out vec4 fragColor;

uniform sampler2D ourTexture;

void main() {
   fragColor = texture(ourTexture, ourTexCoord);
}
```

EXAMPLE: USING A TEXTURE (MAIN)

```
GLuint textureID;
glGenTextures(1, &textureID);

…. LOAD IMAGE OR GENERATE TEXTURE ....

 glBindTexture(GL_TEXTURE_2D, textureID);
 glTexImage2D(GL_TEXTURE_2D, 0, format, width, height, 0, format,
              GL_UNSIGNED_BYTE, data);

// if want to use mip mapping
 glGenerateMipmap(GL_TEXTURE_2D);

// set the texture parameters
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_S, GL_REPEAT);
…
```

EXAMPLE: USING A TEXTURE (MAIN)

```
glActiveTexture(GL_TEXTURE0);

GLuint textureID;
glGenTextures(1, &textureID);

…. LOAD IMAGE OR GENERATE TEXTURE ....

 glBindTexture(GL_TEXTURE_2D, textureID);
 glTexImage2D(GL_TEXTURE_2D, 0, format, width, height, 0, format,
              GL_UNSIGNED_BYTE, data);

// if want to use mip mapping
 glGenerateMipmap(GL_TEXTURE_2D);

// set the texture parameters
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_S, GL_REPEAT);
…

glBindTexture(GL_TEXTURE_2D, textureQuestion);
```

EXAMPLE: USING A TEXTURE (SETTING THE SAMPLER)

```
glActiveTexture(GL_TEXTURE0);

…

// set the sampler for the fragment shader
Uniform::set(shader.id(), "ourTexture", 0)
```

EXAMPLE: USING A TEXTURE (FRAGMENT SHADER)

```
#version 330 core
in vec2 ourTexCoord;
out vec4 fragColor;

uniform sampler2D ourTexture;

void main() {
   fragColor = texture(ourTexture, ourTexCoord);
}
```

EXAMPLE: USING A TEXTURE (SETTING THE SAMPLER)

```
glActiveTexture(GL_TEXTURE1);

…

// set the sampler for the fragment shader
Uniform::set(shader.id(), "ourTexture", 1)
```

EXAMPLE: USING A TEXTURE (FRAGMENT SHADER)

```
#version 330 core
in vec2 ourTexCoord;
out vec4 fragColor;

uniform sampler2D ourTexture;

void main() {
   fragColor = texture(ourTexture, ourTexCoord);
}
```