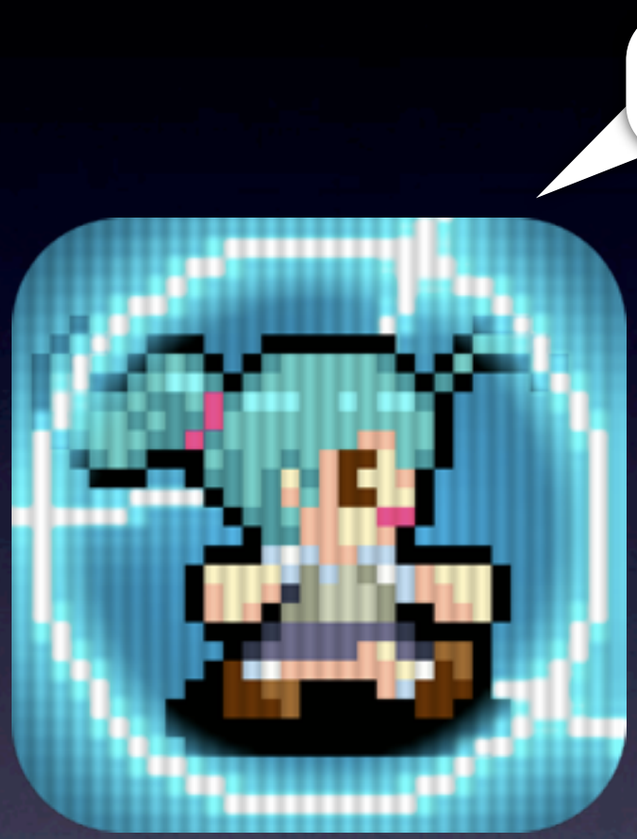
COCOS2D SHADERS

Use of shader on cocos2d

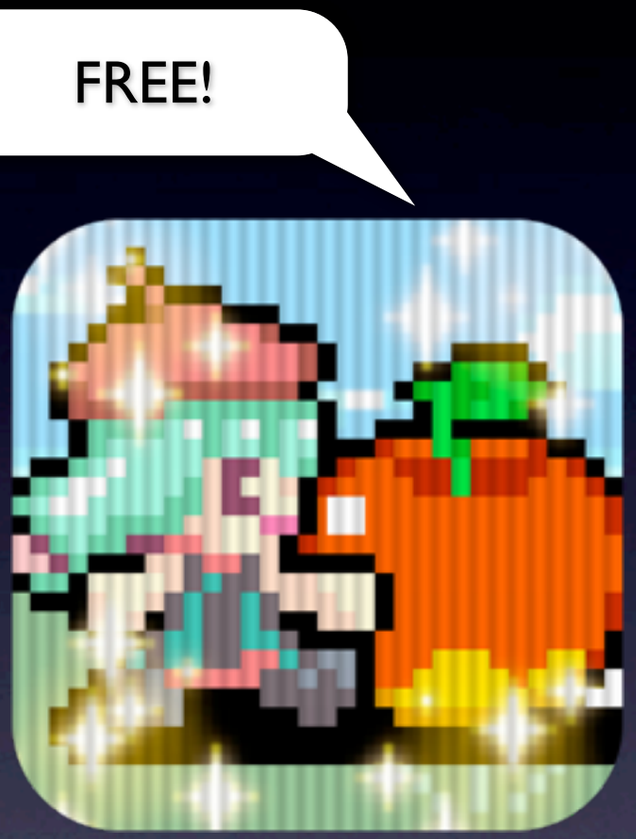
version cocos2d v2.0-rc1

My Profile

- xionchannel
- @ajinotataki
- Technical Artist



ElectroMaster
0.18M DL

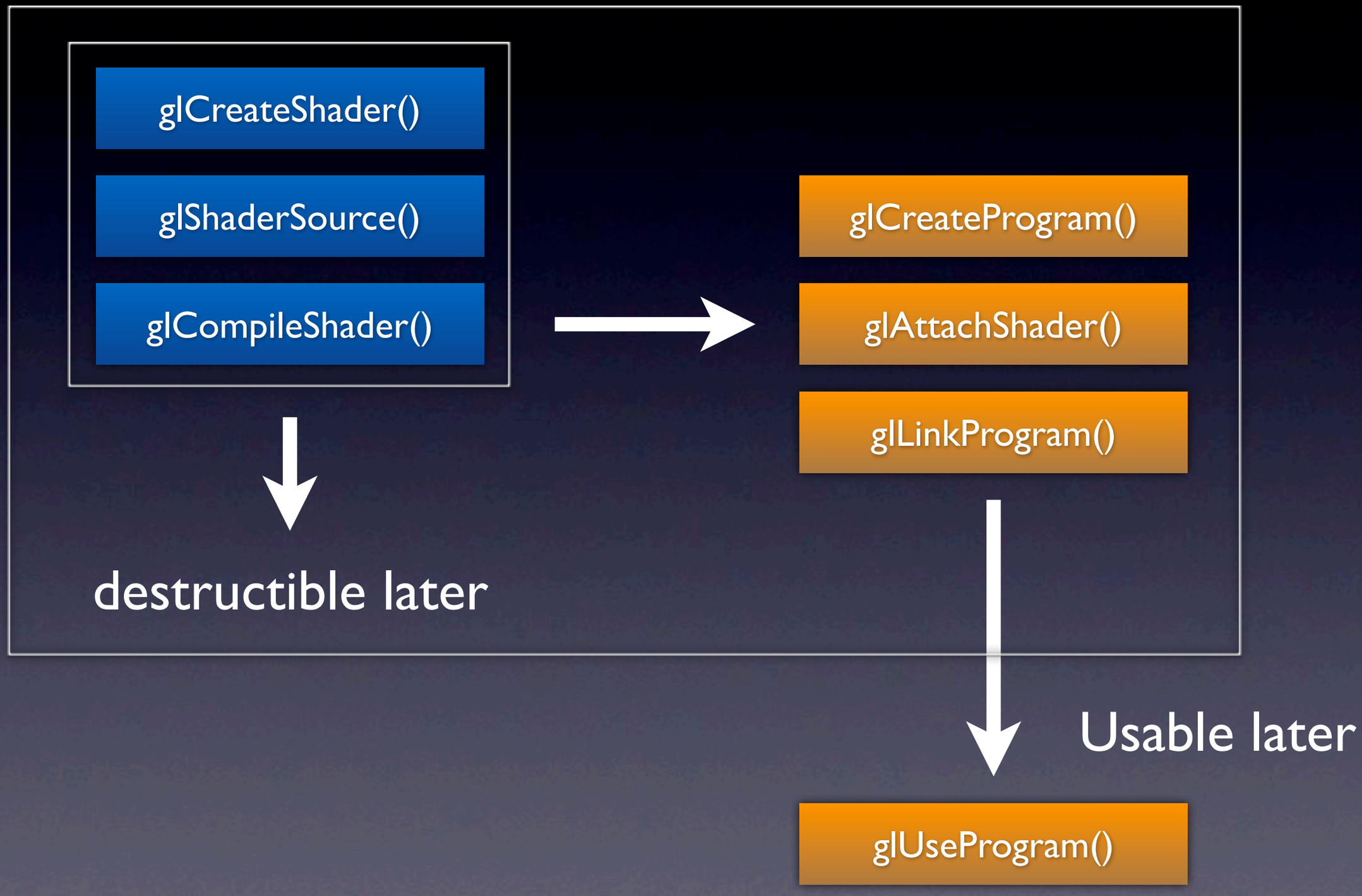


HungryMaster
0.05M DL

What does shader do?

- To calculate Rendering effects by GPU
- VertexShader and FragmentShader
- Use of GLSL language on OpenGL
- Shader arguments (e.g. Attribute, Uniform)

Flow on OpenGL ES 2.0



1. Create VertexShader, FragmentShader objects by glCreateShader()
2. Load shader program into object by glShaderSource()
3. Compile shader program by glCompileShader()

4. Create program object by glCreateProgram()

5. Attach shader program to program object by glAttachShader()
   (At this time, you can destroy shader as object)

6. Link shader program by glLinkProgram()
   (These steps are preprocessing)

7. Apply shader program by glUseProgram()

Use in cocos2d

Flow on cocos2d 2.0

```
self.shaderProgram = [[CCShaderCache sharedShaderCache]
                       programForKey:kCCShader_PositionTextureColor];
```

glCreateShader()

glShaderSource()

glCompileShader()

glCreateProgram()

glAttachShader()

glLinkProgram()

Destroy

Initialize

```
CC_NODE_DRAW_SETUP();
```

glUseProgram()

draw

Flow of initialization

```
self.shaderProgram = [[CCShaderCache sharedShaderCache]
                       programForKey:kCCShader_PositionTextureColor];
```

```
[CCShaderCache sharedShaderCache]
```

```
[[CCShaderCache alloc] init];
```

```
[self loadDefaultShaders];
```

```
[[CCGLProgram alloc]
initWithVertexShaderByteArray:fragmentShaderByteArray:];
```

glCreateShader()

glShaderSource()

glCompileShader()

glCreateProgram()

glAttachShader()

Preparation of Attribute, Uniform

glLinkProgram()

Destroy

Flow of initialization

1. Call [CCShaderCache sharedShaderCache]
2. In step 1, call [[CCShaderCache alloc] init]
3. In step 2, call [self loadDefaultShaders]
4. In step 3, call [[CCGLProgram alloc] initWithVertexShaderByteArray: fragmentShaderByteArray:], to compile shader and keep them in array.

5. Prepare identifiers of Attribute for Uniform
   (see updateUniforms())

6. Link shader program and destroy programs that
   are created by step4.

7. Identifiers (e.g. attribute, uniform) pass
   self.shaderProgram which is a part of CCNode.
   This is due to cocos2d using shaders for
   CCNode that are kept in array by step4.

Flow of drawing

```
CC_NODE_DRAW_SETUP();
```

glUseProgram()

glUniform*() Update dynamic uniform

glBindTexture2d() Specify texture

glVertexAttribPointer() Specify attribute positions

glDrawArrays() Draw polygons

Flow of drawing

1. In CC_NODE_DRAW_SETUP() macro, execute glUseProgram(). Enable to use shader.
2. Update dynamic Uniform parameters.
3. Bind texture.
4. Setup positions of Attribute parameters.
5. Draw polygons.

Attribute, Uniform

- Attribute is parameter for VertexShader.
  - ★ Requires the number of vertexes data.
    (vertex positions, normal vectors, vertex colors)
- Uniform is parameter of both sides.
  - ★ does not require the number of pixels data.
  - ★ Same data for all pixels.
    (Texture, value of calculation)

How to use our original shader on cocos2d?

Modify-initialize method

```
self.shaderProgram = [[CCShaderCache sharedShaderCache]
                       programForKey:kCCShader_PositionTextureColor];
```

```
[CCShaderCache sharedShaderCache]
```

```
[[CCShaderCache alloc] init];
```

```
[self loadDefaultShaders];
```

Replace with our original code.

```
[[CCGLProgram alloc]
initWithVertexShaderByteArray:fragmentShaderByteArray:];
```

glCreateShader()

glShaderSource()

glCompileShader()

Destroy

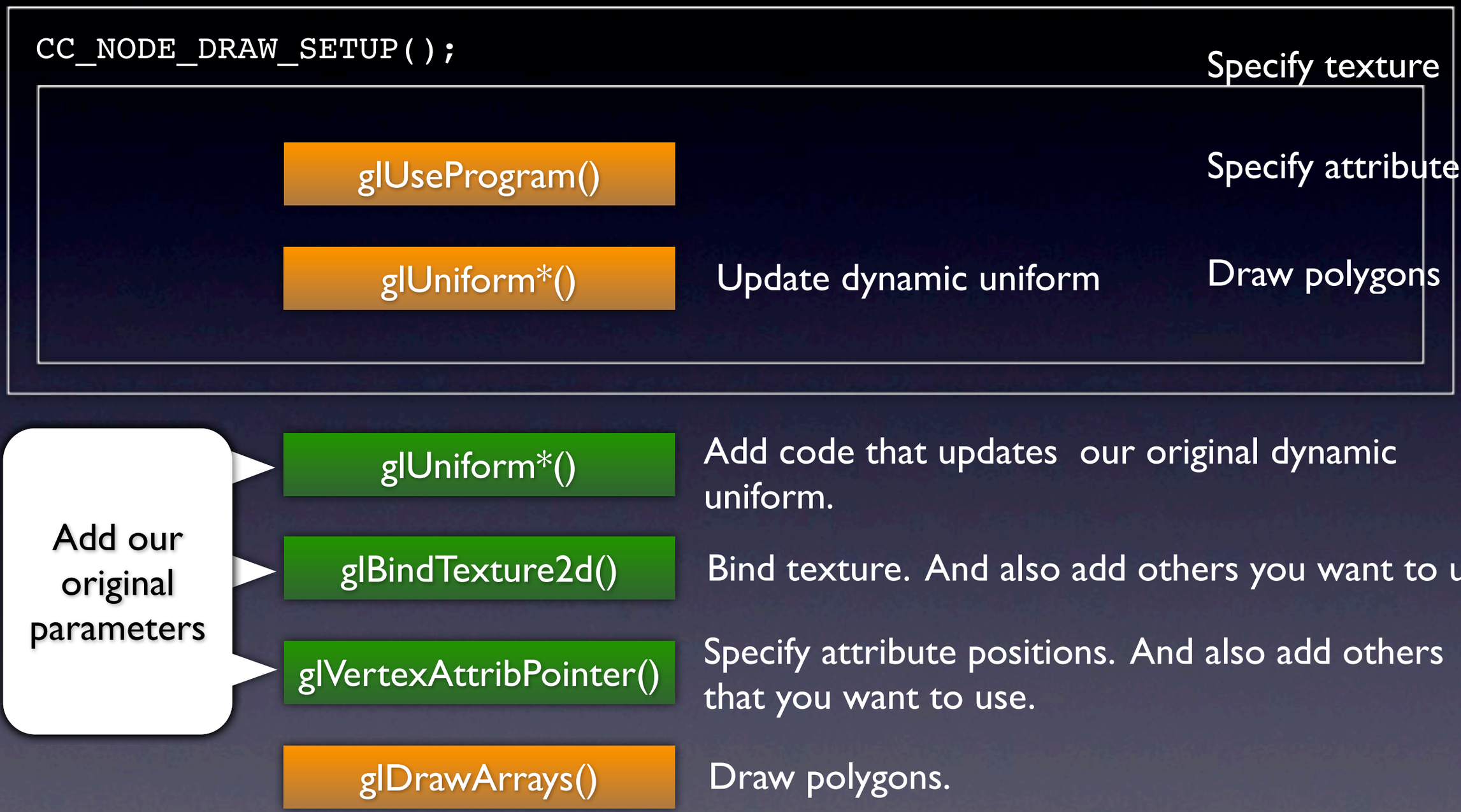
glCreateProgram()

glAttachShader()

Preparation of Attribute, Uniform

glLinkProgram()

Modify-draw method



Demo

e.g. Directional lighting by using normal mapping.

e.g. Point lighting by using normal mapping.

GLSL references

- Reference site(床井研究室)

  http://marina.sys.wakayama-u.ac.jp/~tokoi/?date=20051006

- Today's source code
  http://xionchannel.no-ip.org/cocos2d_shaderTest.zip

- Today's keynote
  http://xionchannel.no-ip.org/cocos2dShader20120621.pdf

Special Thanks(Translation correction)

Tomohisa Takaoka http://twitter.com/tomohisa
Nicholas Salerno at http://salernodesignstudio.com/