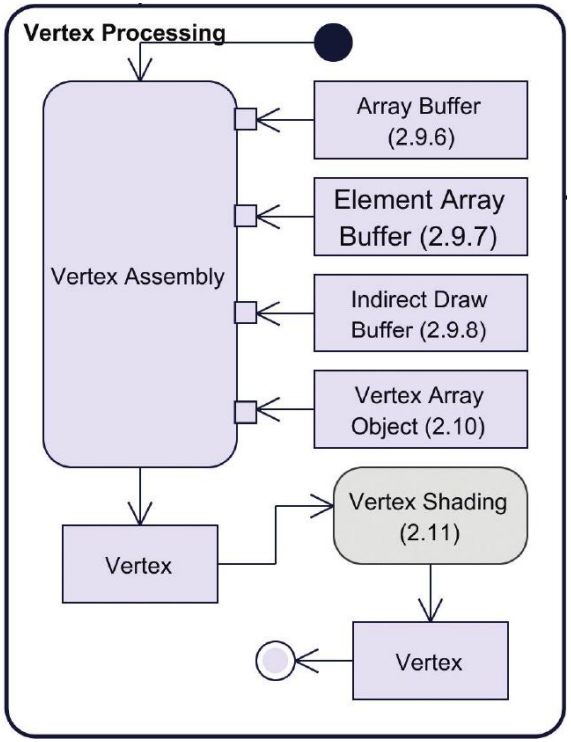
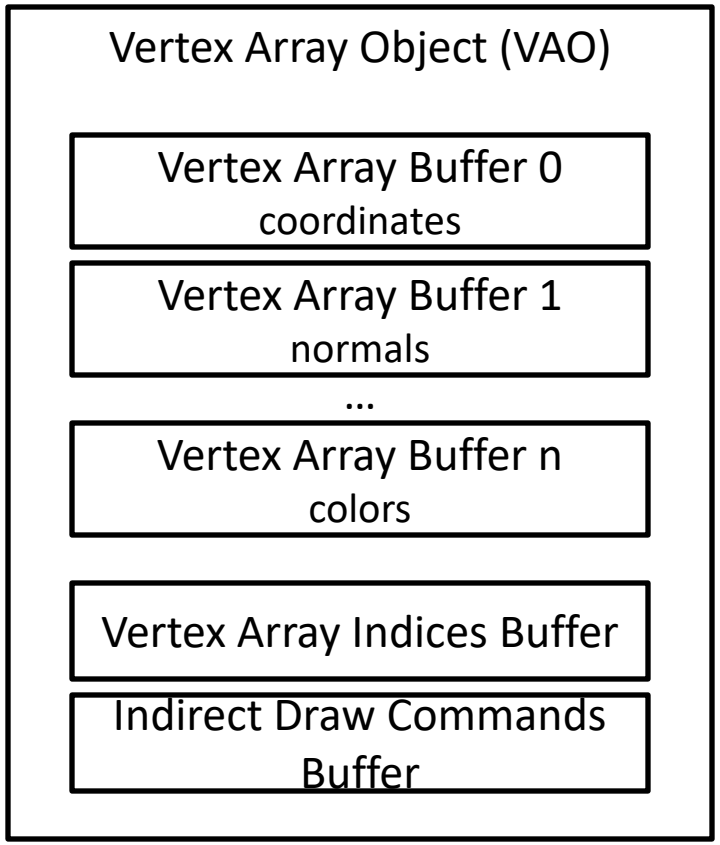
Računarska grafika 2

13M111RG2

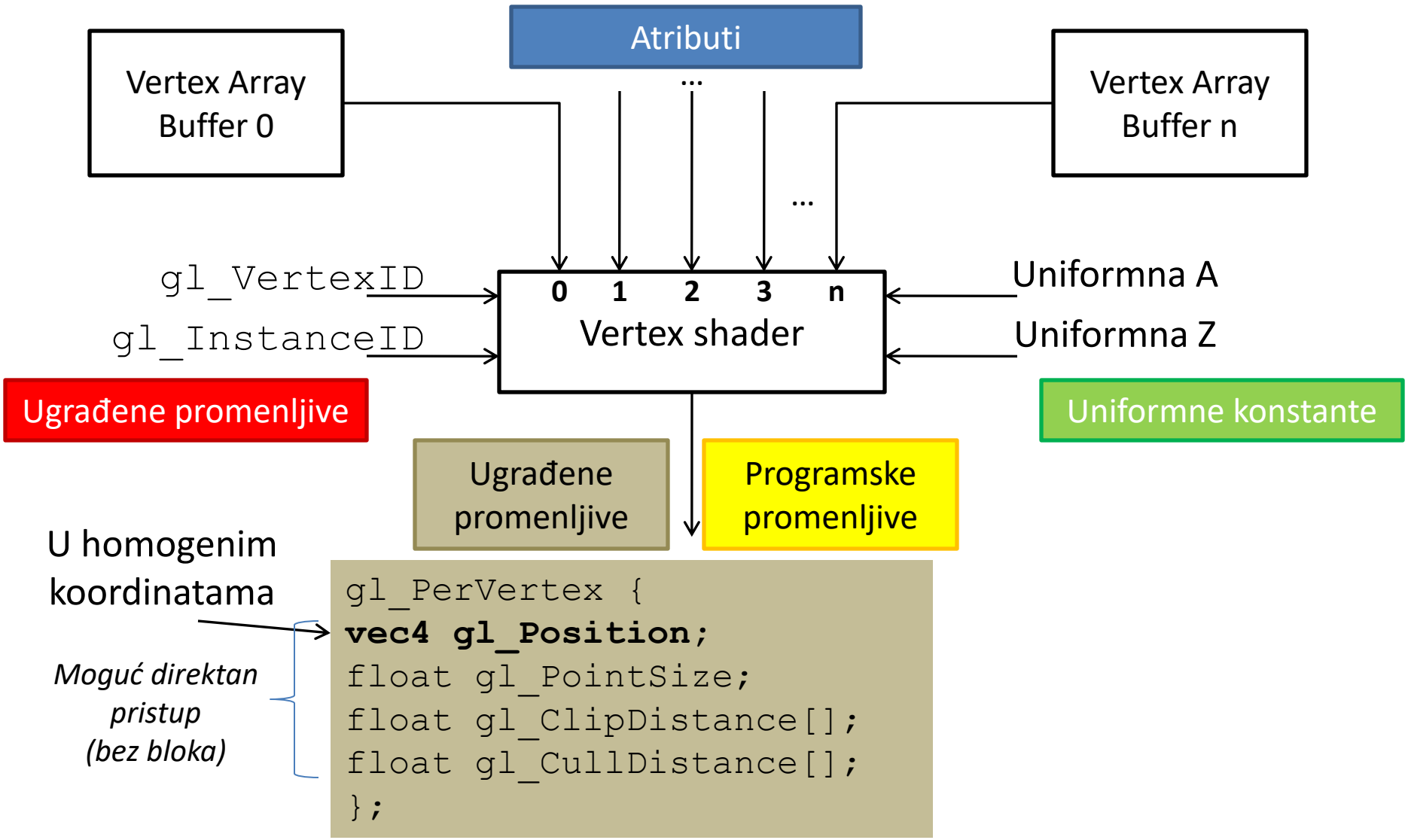
4 Programi za senčenje

Obrada temena (geometrije)

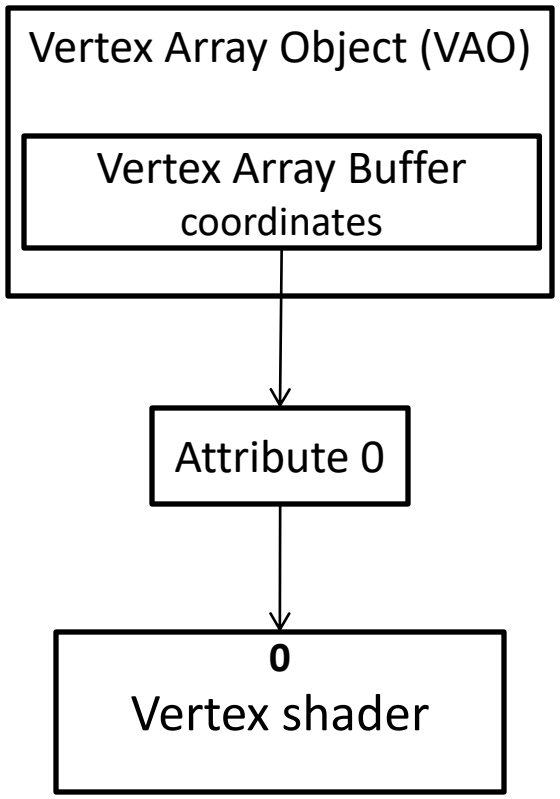




Vertex shading



Klijentski kod + Vertex shading

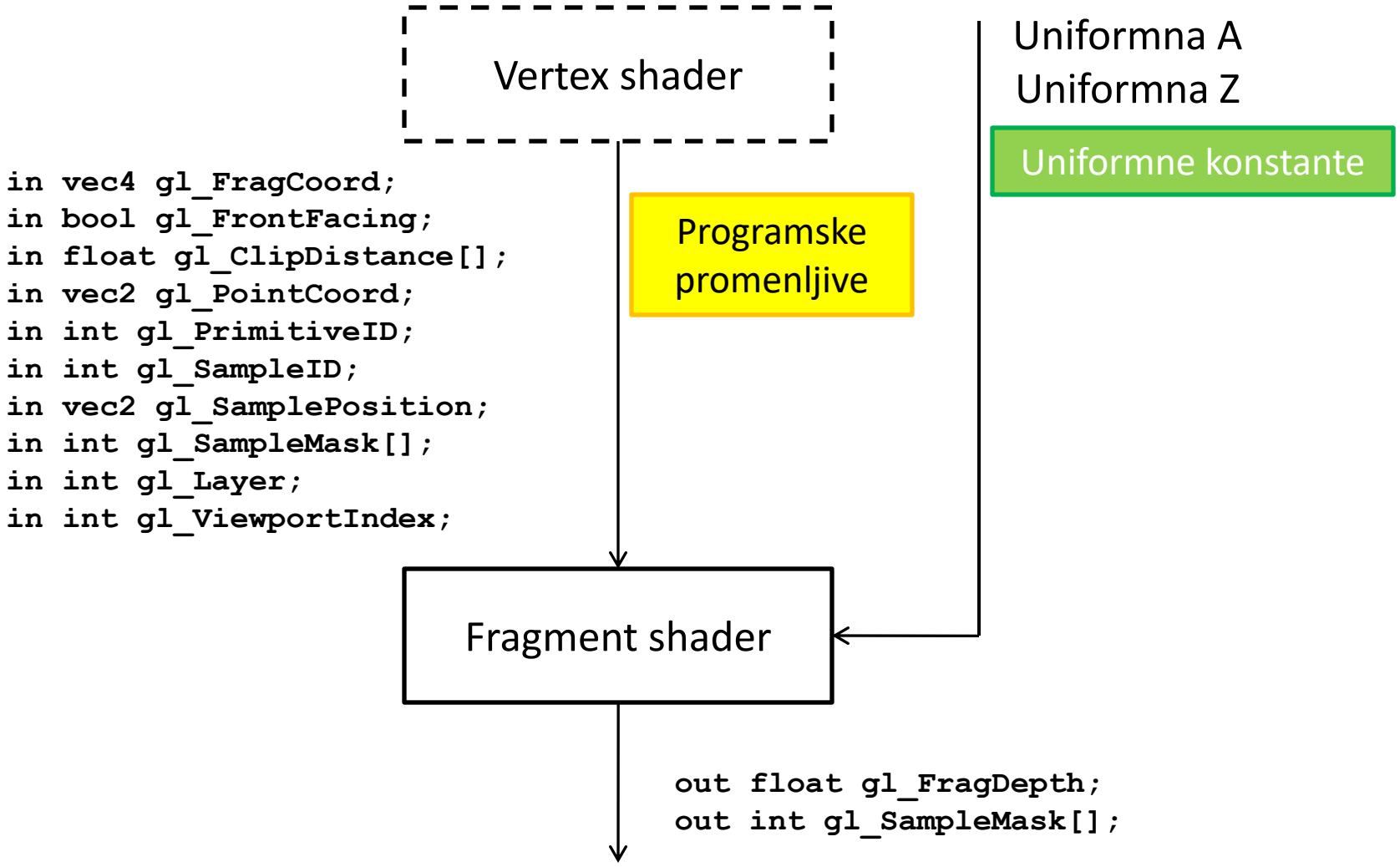



```
int VAO_ID;
glGenVertexArrays(1, &VAO_ID);
void GenVertexArrays( sizei n, uint *arrays );

glBindVertexArray(VAO_ID);

glGenBuffers(1, &buffer_ID);
glBindBuffer(GL_ARRAY_BUFFER, bufferID);
glBufferData(GL_ARRAY_BUFFER, ...);
void BufferData( enum target, sizeiptr size,
                 const void *data, enum usage );
glVertexAttribPointer(0, 3, GL_FLOAT,...);
glEnableVertexAttribArray(0);
```

Fragment shading



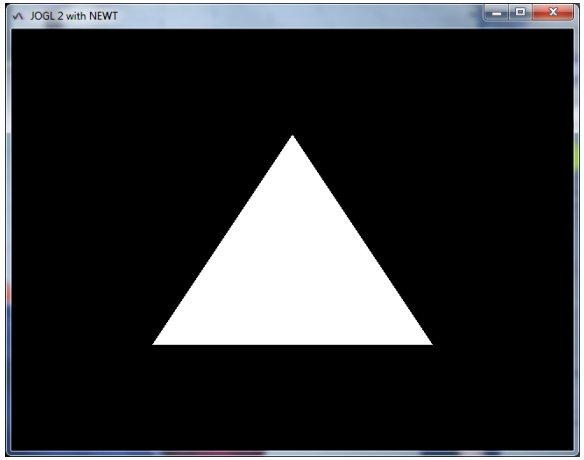
Primer izvornog koda programa za senčenje

Vertex shader

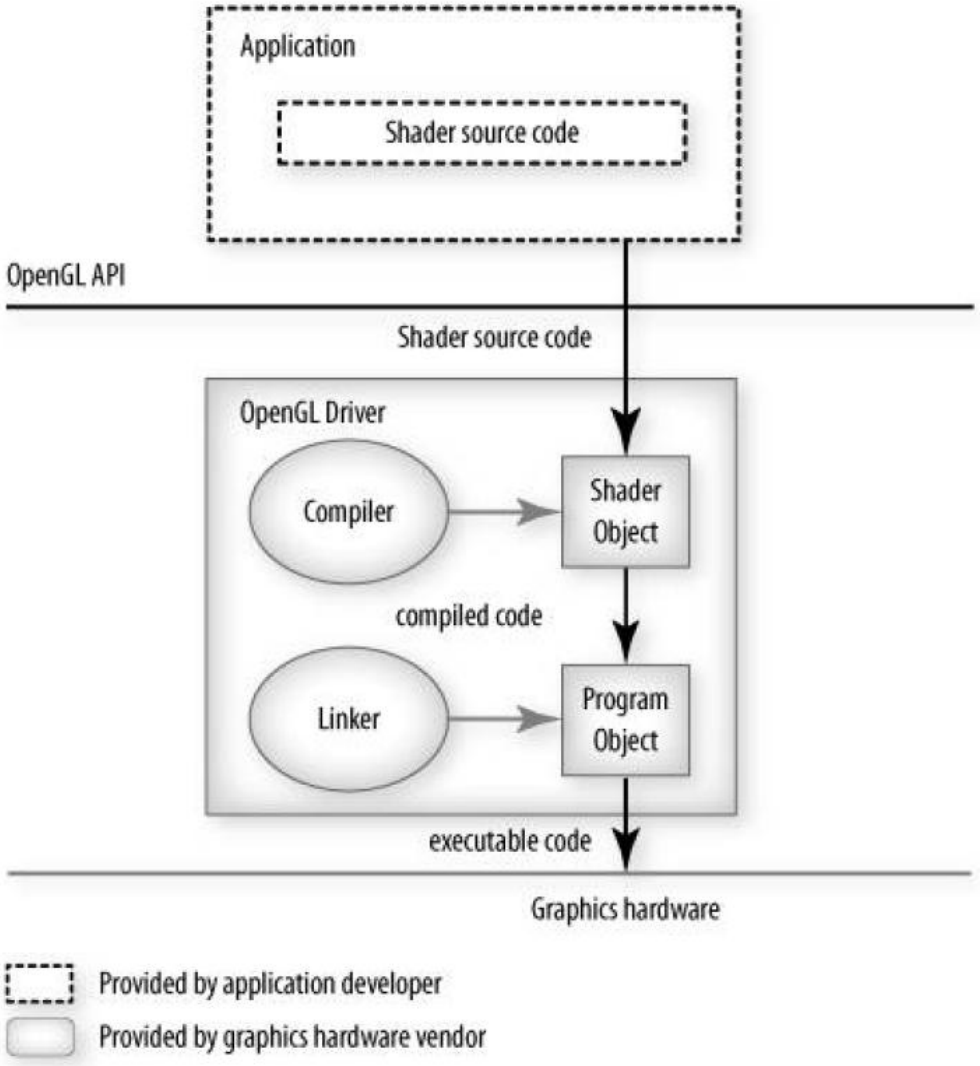
```
#version 430 core
layout(location = 0) in vec3 vertexPosition;
void main()
{
  gl_Position = vec4(vertexPosition, 1.0);
}
```

Fragment shader

```
#version 430
out vec4 outColor;
void main()
{
  outColor = vec4(1.0, 1.0, 1.0, 1.0);
}
```



Model prevođenja i izvršenja



Prevođenje programa za senčenje (1)


```
#version 430 core
layout(location = 0) in vec3 vertexPosition;
void main()
{
  gl_Position = vec4(vertexPosition, 1.0);
}
```

```
int vsShaderObjID = glCreateShader(GL_VERTEX_SHADER);
glShaderSource(vsShaderObjID, vsSource.length, vsSource, null);
glCompileShader(vsShaderObjID);

int []params = new int[1];
glGetShaderiv(vsShaderObjID, GL_COMPILE_STATUS, params, 0);

if(params[0] == 0 ) { … } // compilation failure

glGetShaderiv(vsShaderObjID, GL_INFO_LOG_LENGTH, params, 0);
int infoLogLen = params[0];
ByteBuffer shaderLog = ByteBuffer.allocate(infoLogLen);
glGetShaderInfoLog(vsShaderObjID, infoLogLen, null, shaderLog);
```

Prevođenje programa za senčenje (2)


```
#version 430
out vec4 outColor;
void main()
{
  outColor = vec4(1.0, 1.0, 1.0, 1.0);
}
```

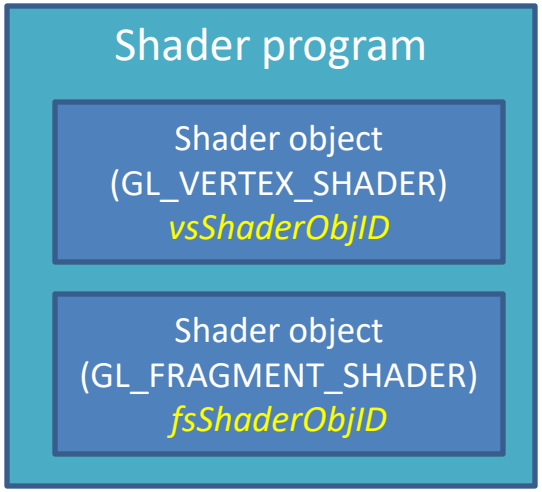
```
int fsShaderObjID = glCreateShader(GL_FRAGMENT_SHADER);
glShaderSource(fsShaderObjID, vsSource.length, vsSource, null);
glCompileShader(fsShaderObjID);

int []params = new int[1];
glGetShaderiv(fsShaderObjID, GL_COMPILE_STATUS, params, 0);

if(params[0] == 0 ) { … } // compilation failure

glGetShaderiv(fsShaderObjID, GL_INFO_LOG_LENGTH, params, 0);
int infoLogLen = params[0];
ByteBuffer shaderLog = ByteBuffer.allocate(infoLogLen);
glGetShaderInfoLog(fsShaderObjID, infoLogLen, null, shaderLog);
```

Prevođenje programa za senčenje (3)




```
int programID = glCreateProgram();
glAttachShader(programID, vsShaderObjID);
glAttachShader(programID, fsShaderObjID);

glLinkProgram(programID);
```

```
glGetProgramiv(programID, GL_LINK_STATUS, params, 0);
if( params[0] == 0 ) { … }// link error

glGetProgramiv(programID, GL_INFO_LOG_LENGTH, params, 0);
int infoLogLen = params[0];
ByteBuffer programLog = ByteBuffer.allocate(infoLogLen);
glGetProgramInfoLog(programID, infoLogLen, null, programLog);

glDeleteShader(vsShaderObjID);
glDeleteShader(fsShaderObjID);
…
glDeleteProgram(programID);
```

Crtanje

```
glClearColor(0.0f, 0.0f, 0.0f, 0.0f);
glClear(GL_COLOR_BUFFER_BIT |
        GL_STENCIL_BUFFER_BIT |
        GL_DEPTH_BUFFER_BIT);
…
glBindVertexArray(VAO_ID);

glUseProgram(shader_program_ID);
glDrawArrays(GL_TRIANGLES, 0, 3);

…
glUseProgram(0);
glBindVertexArray(0);
```