Федеральное агентство связи  
Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Сибирский государственный университет

телекоммуникаций и информатики»

**Лабораторная работа по теме:**

**«JNI»**

Выполнили:

студентки 4 курса

ИВТ, гр. ИП-712

Гервас А.В.

Онищенко А.В.

Новосибирск 2020

Оглавление

[Задание 3](#_Toc58544273)

[Скриншоты 3](#_Toc58544274)

[Листинг кода 3](#_Toc58544275)

**Задание**

Написать программу, рисующую куб с текстурой. Вся прорисовка должна быть реализована в JNI.

**Скриншоты**

 

**Листинг кода**

Приложение написано на языке Java.

**MainActivity.java**

package com.example.user.lab5;

import android.app.Activity;

import android.app.ActivityManager;

import android.content.Context;

import android.content.pm.ConfigurationInfo;

import android.opengl.GLSurfaceView;

import android.os.Build;

import android.os.Bundle;

import android.widget.Toast;

public class MainActivity extends Activity {

private GLSurfaceView glSurfaceView;

private boolean rendererSet;

private boolean isProbablyEmulator() {

return Build.VERSION.SDK\_INT >= Build.VERSION\_CODES.ICE\_CREAM\_SANDWICH\_MR1

&& (Build.FINGERPRINT.startsWith("generic")

|| Build.FINGERPRINT.startsWith("unknown")

|| Build.MODEL.contains("google\_sdk")

|| Build.MODEL.contains("Emulator")

|| Build.MODEL.contains("Android SDK built for x86"));

}

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

ActivityManager activityManager

= (ActivityManager) getSystemService(Context.ACTIVITY\_SERVICE);

ConfigurationInfo configurationInfo = activityManager.getDeviceConfigurationInfo();

final boolean supportsEs2 =

configurationInfo.reqGlEsVersion >= 0x20000 || isProbablyEmulator();

if (supportsEs2) {

glSurfaceView = new GLSurfaceView(this);

if (isProbablyEmulator()) {

// Avoids crashes on startup with some emulator images.

glSurfaceView.setEGLConfigChooser(8, 8, 8, 8, 16, 0);

}

glSurfaceView.setRenderer(new RendererWrapper(this));

rendererSet = true;

setContentView(glSurfaceView);

} else {

// Should never be seen in production, since the manifest filters

// unsupported devices.

Toast.makeText(this, "This device does not support OpenGL ES 2.0.",

Toast.LENGTH\_LONG).show();

return;

}

}

@Override

protected void onPause() {

super.onPause();

if (rendererSet) {

glSurfaceView.onPause();

}

}

@Override

protected void onResume() {

super.onResume();

if (rendererSet) {

glSurfaceView.onResume();

}

}

}

**RendererWrapper.java**

package com.example.user.lab5;

import android.content.Context;

import android.graphics.Bitmap;

import android.graphics.BitmapFactory;

import android.opengl.GLSurfaceView;

import android.opengl.GLUtils;

import java.io.InputStream;

import javax.microedition.khronos.egl.EGLConfig;

import javax.microedition.khronos.opengles.GL10;

class RendererWrapper implements GLSurfaceView.Renderer {

static public int[] texture\_name = {

R.drawable.paint

};

Context c;

public RendererWrapper(Context context) {

c = context;

}

static public int[] textures = new int [texture\_name.length];

private void loadGLTexture(GL10 gl) {

gl.glGenTextures(1, textures, 0);

for (int i = 0; i < texture\_name.length; ++i) {

gl.glBindTexture(GL10.GL\_TEXTURE\_2D, textures[i]);

gl.glTexParameterf(GL10.GL\_TEXTURE\_2D, GL10.GL\_TEXTURE\_MIN\_FILTER, GL10.GL\_LINEAR);

InputStream is = c.getResources().openRawResource(texture\_name[i]);

Bitmap bitmap = BitmapFactory.decodeStream(is);

GLUtils.texImage2D(GL10.GL\_TEXTURE\_2D, 0, bitmap, 0);

bitmap.recycle();

}

}

@Override

public void onSurfaceCreated(GL10 gl, EGLConfig config) {

loadGLTexture(gl);

JNIWrapper.onsurfacecreated();

}

@Override

public void onSurfaceChanged(GL10 gl, int width, int height) {

JNIWrapper.onsurfacechanged(width, height);

}

@Override

public void onDrawFrame(GL10 gl) {

JNIWrapper.ondrawframe();

}

}

**JNIWrapper.java**

package com.example.user.lab5;

public class JNIWrapper {

static {

System.loadLibrary("native-lib");

}

public static native void onsurfacecreated();

public static native void onsurfacechanged(int width, int height);

public static native void ondrawframe();

}

**native-lib.cpp**

#include <jni.h>

#include <string>

#include <GLES2/gl2.h>

#include <GLES/gl.h>

extern "C"

JNIEXPORT void JNICALL

Java\_com\_example\_user\_lab5\_JNIWrapper\_onsurfacecreated(JNIEnv \*env, jclass cls) {

glClearColor(1.0f, 1.0f, 1.0f, 0.0f);

glMatrixMode(GL\_PROJECTION);

glLoadIdentity();

glOrthof(-8, 8, -8, 8, -8, 8);

glEnable(GL\_DEPTH\_TEST);

glClearDepthf(1);

glMatrixMode(GL\_MODELVIEW);

glLoadIdentity();

}

extern "C"

JNIEXPORT void JNICALL

Java\_com\_example\_user\_lab5\_JNIWrapper\_onsurfacechanged(JNIEnv \*env, jclass cls, jint width,

jint height) {

}

GLfloat a[12] = {

-1, 1, 0,

-1, -1, 0,

1, -1, 0,

1, 1, 0

};

GLfloat texCoords[8] = { // Texture coords for the above face (NEW)

0.0f, 1.0f, // A. left-bottom (NEW)

1.0f, 1.0f, // B. right-bottom (NEW)

0.0f, 0.0f, // C. left-top (NEW)

1.0f, 0.0f // D. right-top (NEW)

};

int angle = 0;

extern "C"

JNIEXPORT void JNICALL Java\_com\_example\_user\_lab5\_JNIWrapper\_ondrawframe(JNIEnv \*env, jclass cls) {

glClear(GL\_COLOR\_BUFFER\_BIT | GL\_DEPTH\_BUFFER\_BIT);

glLoadIdentity();

glScalef(2, 2, 2);

glEnableClientState(GL\_VERTEX\_ARRAY);

glEnableClientState(GL\_TEXTURE\_COORD\_ARRAY);

glEnable(GL\_TEXTURE\_2D);

angle = (angle == 360) ? 0 : angle + 2;

glRotatef(angle, 1, 1, 0);

//лицевая грань

glPushMatrix();

glVertexPointer(3, GL\_FLOAT, 0, a);

glTexCoordPointer(2, GL\_FLOAT, 0, texCoords);

glDrawArrays(GL\_TRIANGLE\_FAN, 0, 4);

glPopMatrix();

//задняя

glPushMatrix();

glTranslatef(0, 0, -2);

glVertexPointer(3, GL\_FLOAT, 0, a);

glTexCoordPointer(2, GL\_FLOAT, 0, texCoords);

glDrawArrays(GL\_TRIANGLE\_FAN, 0, 4);

glPopMatrix();

//верхняя

glPushMatrix();

glTranslatef(0, 1, -1);

glRotatef(90, 1, 0, 0);

glVertexPointer(3, GL\_FLOAT, 0, a);

glTexCoordPointer(2, GL\_FLOAT, 0, texCoords);

glDrawArrays(GL\_TRIANGLE\_FAN, 0, 4);

glPopMatrix();

//нижняя

glPushMatrix();

glRotatef(90, 1, 0, 0);

glTranslatef(0, -1, 1);

glVertexPointer(3, GL\_FLOAT, 0, a);

glTexCoordPointer(2, GL\_FLOAT, 0, texCoords);

glDrawArrays(GL\_TRIANGLE\_FAN, 0, 4);

glPopMatrix();

//левая

glPushMatrix();

glRotatef(90, 0, 1, 0);

glTranslatef(1, 0, -1);

glVertexPointer(3, GL\_FLOAT, 0, a);

glTexCoordPointer(2, GL\_FLOAT, 0, texCoords);

glDrawArrays(GL\_TRIANGLE\_FAN, 0, 4);

glPopMatrix();

//правая

glPushMatrix();

glRotatef(90, 0, 1, 0);

glTranslatef(1, 0, 1);

glVertexPointer(3, GL\_FLOAT, 0, a);

glTexCoordPointer(2, GL\_FLOAT, 0, texCoords);

glDrawArrays(GL\_TRIANGLE\_FAN, 0, 4);

glPopMatrix();

glDisable(GL\_TEXTURE\_2D);

glDisableClientState(GL\_TEXTURE\_COORD\_ARRAY);

glDisableClientState(GL\_VERTEX\_ARRAY);

}