Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное

образовательное учреждение высшего образования

«Комсомольский-на-Амуре государственный университет»

Факультет компьютерных технологий

Кафедра «МОП ЭВМ»

РАСЧЁТНО-ГРАФИЧЕСКАЯ РАБОТА

по дисциплине «Программирование мобильных устройств»

Студент группы 6ИСб-1 Н. Ходжазода

Преподаватель В.А. Тихомиров

2020

**Задания**

Управление букашкой с помощью движков (Слайдер.exe).

**Содержание**

[Введение 4](#_Toc35433989)

[Ход выполнения работы 5](#_Toc35433991)

[Заключение 10](#_Toc35433992)

[Список использованных источников 11](#_Toc35433993)

Введение

Android Studio - это [интегрированная среда разработки](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D0%B3%D1%80%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D1%80%D0%B5%D0%B4%D0%B0_%D1%80%D0%B0%D0%B7%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%BA%D0%B8) (IDE) для работы с платформой [Android](https://ru.wikipedia.org/wiki/Android" \o "Android). [1]

Android Studio, основанная на программном обеспечении [IntelliJ IDEA](https://ru.wikipedia.org/wiki/IntelliJ_IDEA" \o "IntelliJ IDEA) от компании [JetBrains](https://ru.wikipedia.org/wiki/JetBrains" \o "JetBrains), - официальное средство разработки Android приложений. Данная среда разработки доступна для [Windows](https://ru.wikipedia.org/wiki/Windows" \o "), [OS X](https://ru.wikipedia.org/wiki/OS_X) и Linux.

17 мая 2017 на ежегодной конференции Google I/O, [Google](https://ru.wikipedia.org/wiki/Google_(%D0%BA%D0%BE%D0%BC%D0%BF%D0%B0%D0%BD%D0%B8%D1%8F)" \o "Google (компания)) анонсировал поддержку языка [Kotlin](https://ru.wikipedia.org/wiki/Kotlin" \o "Kotlin), используемого в Android Studio, как официального языка программирования для платформы Android в добавление к [Java](https://ru.wikipedia.org/wiki/Java" \o "Java) и [С++](https://ru.wikipedia.org/wiki/C%2B%2B).

Новые функции появляются с каждой новой версией Android Studio. На данный момент доступны следующие функции:

* Расширенный редактор макетов: [WYSIWYG](https://ru.wikipedia.org/wiki/WYSIWYG), способность работать с [UI](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81_%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D1%82%D0%B5%D0%BB%D1%8F) компонентами при помощи [Drag-and-Drop](https://ru.wikipedia.org/wiki/Drag-and-drop" \o "Drag-and-drop), функция предпросмотра макета на нескольких конфигурациях экрана.
* Сборка приложений, основанная на [Gradle](https://ru.wikipedia.org/wiki/Gradle" \o "Gradle).
* Различные виды сборок и генерация нескольких [.apk](https://ru.wikipedia.org/wiki/.APK) файлов
* [Рефакторинг](https://ru.wikipedia.org/wiki/%D0%A0%D0%B5%D1%84%D0%B0%D0%BA%D1%82%D0%BE%D1%80%D0%B8%D0%BD%D0%B3) кода
* [Статический анализатор кода](https://ru.wikipedia.org/w/index.php?title=%D0%A1%D1%82%D0%B0%D1%82%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9_%D0%B0%D0%BD%D0%B0%D0%BB%D0%B8%D0%B7%D0%B0%D1%82%D0%BE%D1%80_%D0%BA%D0%BE%D0%B4%D0%B0&action=edit&redlink=1) (Lint), позволяющий находить проблемы производительности, несовместимости версий и другое.
* Встроенный ProGuard и утилита для подписывания приложений.
* Шаблоны основных макетов и компонентов Android.
* Встроенная поддержка [Google Cloud Platform](https://ru.wikipedia.org/wiki/Google_Cloud_Platform" \o "Google Cloud Platform), которая включает в себя интеграцию с сервисами [Google Cloud Messaging](https://ru.wikipedia.org/wiki/Google_Cloud_Messaging" \o "Google Cloud Messaging) и [App Engine](https://ru.wikipedia.org/w/index.php?title=App_Engine&action=edit&redlink=1" \o "App Engine (страница отсутствует)).
* Новая версия Android Studio 2.1 способна работать с обновленным компилятором Jack, а также получила улучшенную поддержку Java 8 и усовершенствованную функцию Instant Run.
* В Android Studio 3.0 будут по стандарту включены инструменты языка [Kotlin](https://ru.wikipedia.org/wiki/Kotlin" \o "Kotlin) основанные на [JetBrains](https://ru.wikipedia.org/wiki/JetBrains" \o "JetBrains) [IDE](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D0%B3%D1%80%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D1%80%D0%B5%D0%B4%D0%B0_%D1%80%D0%B0%D0%B7%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%BA%D0%B8). [2]

Ход выполнения работы

При разработке приложения были использованы 3 файла в Android Studio.

Содержимое файла AndroidManifest.java приведено в листинге 2.

Листинг 1 - AndroidManifest.java

<?xml version="1.0" encoding="utf-8"?>

<manifest xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:tools="http://schemas.android.com/tools"

package="com.knastu.lab7">

<application android:label="@string/app\_name">

<activity

android:name="MainActivity"

android:label="@string/app\_name"

android:screenOrientation="landscape"

tools:ignore="LockedOrientationActivity">

<intent-filter>

<action android:name="android.intent.action.MAIN" />

<category android:name="android.intent.category.LAUNCHER" />

</intent-filter>

</activity>

</application>

</manifest>

Содержимое файла MainActivity.java приведено в листинге 2.

Листинг 2 - MainActivity.java

package com.knastu.lab7;

import android.app.Activity;

import android.os.Bundle;

public class MainActivity extends Activity {

@Override

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(new Board(this));

}

}

Содержимое файла Board.java приведено в листинге 3.

Листинг 3 - Board.java

**package** com.knastu.lab7;  
  
**import** android.content.Context;  
**import** android.graphics.Bitmap;  
**import** android.graphics.BitmapFactory;  
**import** android.graphics.Canvas;  
**import** android.graphics.Color;  
**import** android.graphics.Paint;  
**import** android.graphics.RectF;  
**import** android.view.MotionEvent;  
**import** android.view.View;  
  
**public class** Board **extends** View {  
  
 Bitmap unit;  
 **float** unitSpeedX = 0;  
 **float** unitSpeedY = 0;  
 **double** unitRotate = 0;  
  
 **float** unitPosX = 0, unitPosY = 0;  
 **float** verticalButtonPosY = 0, horizontalButtonPosX = 0;  
  
 **float** sliderLength = 400, sliderButtonWight = 40, sliderButtonHeight = 60;  
 **boolean** canHorizontalSliderMove = **false**, canVerticalSliderMove = **false**;  
  
 **float** layoutMargin = 30;  
 RectF rectGame, rectVerticalSlider, rectVerticalButton, rectHorizontalSlider, rectHorizontalButton;  
 Paint paintGameBoard, paintSlider, paintBorderGameBoard;  
  
 **public** Board(Context context) {  
 **super**(context);  
  
 unit = BitmapFactory.decodeResource(getResources(), R.drawable.juchok);  
  
 paintGameBoard = **new** Paint();  
 paintGameBoard.setColor(Color.rgb(89, 114, 139));  
 paintGameBoard.setStyle(Paint.Style.FILL\_AND\_STROKE);  
  
 paintBorderGameBoard = **new** Paint();  
 paintBorderGameBoard.setColor(Color.rgb(117, 144, 172));  
 paintBorderGameBoard.setStrokeWidth(unit.getWidth());  
 paintBorderGameBoard.setStyle(Paint.Style.STROKE);  
  
 paintSlider = **new** Paint();  
 paintSlider.setColor(Color.BLACK);  
 paintSlider.setStrokeWidth(14);  
 paintSlider.setStyle(Paint.Style.FILL\_AND\_STROKE);  
 }  
  
 @Override  
 **protected void** onDraw(Canvas canvas) {  
  
 *//// DRAW LAYOUT* canvas.drawRGB(117, 144, 172);  
  
 *//// DRAW GAME BOARD* rectGame = **new** RectF(unit.getWidth(), layoutMargin, getWidth() - layoutMargin, getHeight() - layoutMargin - sliderButtonHeight);  
 canvas.drawRect(rectGame, paintGameBoard);  
  
 **if** (unitPosX == 0 && unitPosY == 0) {  
 unitPosX = rectGame.centerX();  
 unitPosY = rectGame.centerY();  
 }  
  
 *//// DRAW UNIT  
  
 // Rotate unit* unitRotate = (**float**) Math.toDegrees(Math.atan2(unitSpeedY, unitSpeedX));  
  
 *// Move unit* unitPosX += unitSpeedX;  
 unitPosY += unitSpeedY;  
 **if** (unitPosY < rectGame.top)  
 unitPosY = rectGame.bottom;  
 **if** (unitPosY > rectGame.bottom)  
 unitPosY = rectGame.top;  
 **if** (unitPosX < rectGame.left)  
 unitPosX = rectGame.right;  
 **if** (unitPosX > rectGame.right)  
 unitPosX = rectGame.left;  
  
 RectF unit\_rect = **new** RectF(unitPosX - unit.getWidth() / 2, unitPosY - unit.getHeight() / 2, unitPosX + unit.getWidth() / 2, unitPosY + unit.getHeight() / 2);  
  
 canvas.save();  
 canvas.rotate((**float**) unitRotate, unit\_rect.centerX(), unit\_rect.centerY());  
 canvas.drawBitmap(unit, **null**, unit\_rect, **null**);  
 canvas.restore();  
  
 *//// DRAW GAME BOARD BORDER* canvas.drawRect(rectGame, paintBorderGameBoard);  
  
 *//// DRAW VERTICAL SLIDER  
  
 // draw line* rectVerticalSlider = **new** RectF(rectGame.left - layoutMargin - sliderButtonWight, getHeight() / 2 - sliderLength / 2, rectGame.left - layoutMargin, getHeight() / 2 + sliderLength / 2);  
 canvas.drawLine(rectVerticalSlider.centerX(), rectVerticalSlider.top, rectVerticalSlider.centerX(), rectVerticalSlider.bottom, paintSlider);  
  
 *// draw button* rectVerticalButton = **new** RectF(rectVerticalSlider.left, rectVerticalSlider.centerY() - sliderButtonHeight / 2 + verticalButtonPosY, rectVerticalSlider.right, rectVerticalSlider.centerY() + sliderButtonHeight / 2 + verticalButtonPosY);  
 canvas.drawRect(rectVerticalButton, paintSlider);  
  
 *//// DRAW HORIZONTAL SLIDER  
  
 // draw line* rectHorizontalSlider = **new** RectF(rectGame.centerX() - sliderLength / 2, getHeight() - layoutMargin - sliderButtonWight, rectGame.centerX() + sliderLength / 2, getHeight() - layoutMargin);  
 canvas.drawLine(rectHorizontalSlider.centerX() - sliderLength / 2, rectHorizontalSlider.centerY(), rectHorizontalSlider.centerX() + sliderLength / 2, rectHorizontalSlider.centerY(), paintSlider);  
  
 *// draw button* rectHorizontalButton = **new** RectF(rectHorizontalSlider.centerX() - sliderButtonHeight / 2 + horizontalButtonPosX, rectHorizontalSlider.top, rectHorizontalSlider.centerX() + sliderButtonHeight / 2 + horizontalButtonPosX, rectHorizontalSlider.bottom);  
 canvas.drawRect(rectHorizontalButton, paintSlider);  
  
 invalidate();  
 }  
  
 @Override  
 **public boolean** onTouchEvent(MotionEvent event) {  
 **float** positionX = event.getX();  
 **float** positionY = event.getY();  
  
 **switch** (event.getAction()) {  
 **case** MotionEvent.ACTION\_DOWN: {  
  
 **if** (rectVerticalButton.contains(positionX, positionY)) {  
 canVerticalSliderMove = **true**;  
  
 } **else if** (rectHorizontalButton.contains(positionX, positionY)) {  
 canHorizontalSliderMove = **true**;  
 }  
 }  
 **break**;  
  
 **case** MotionEvent.ACTION\_MOVE: {  
 **if** (canVerticalSliderMove) {  
 **if** (rectVerticalSlider.contains(positionX, positionY)) {  
 canVerticalSliderMove = **true**;  
 verticalButtonPosY = positionY - rectVerticalSlider.centerY();  
 unitSpeedY = verticalButtonPosY/10;  
 invalidate();  
 }  
 }  
  
 **if** (canHorizontalSliderMove) {  
 **if** (rectHorizontalSlider.contains(positionX, positionY)) {  
 canHorizontalSliderMove = **true**;  
 horizontalButtonPosX = positionX - rectHorizontalSlider.centerX();  
 unitSpeedX = horizontalButtonPosX/10;  
 invalidate();  
 }  
 }  
 }  
 **break**;  
  
 **case** MotionEvent.ACTION\_UP: {  
 canVerticalSliderMove = **false**;  
 canHorizontalSliderMove = **false**;  
 }  
 **break**;  
 }  
  
 **return true**;  
 }  
}

Результат работы приложения показан на рисунках 1-3.



Рисунок 1 – Начальный экран приложения



Рисунок 2 – Перемещение букашки (1)



Рисунок 3 – Перемещение букашки (2)

Заключение

В ходе выполнения расчётно-графической работы были изучены и рассмотрены возможности [интегрированной среды разработки](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D0%B3%D1%80%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D1%80%D0%B5%D0%B4%D0%B0_%D1%80%D0%B0%D0%B7%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%BA%D0%B8) «Android Studio», в которой было разработано приложение с выпадающим меню.

Полученные навыки и знания будут использоваться в дальнейшем.

Список использованных источников

1. Дейтел, П. Android для разработчиков. 3-е изд. / П. Дейтел, Х. Дейтел, А. Уолд. — СПб.: Питер, 2016. — 512 с.
2. Гриффитс. Д. Head First. Программирование для Android / Дон Гриффитс, Дэвид Гриффитс — СПб.: Питер, 2016. — 704 с.