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 7ИСб-1

Преподаватель В.А. Тихомиров

2021

**Задания**

Управление букашкой с помощью джойстика (Джойстик.exe).

**Содержание**

[Введение 4](#_Toc35433989)

[Ход выполнения работы 5](#_Toc35433991)

[Заключение 12](#_Toc35433992)

[Список использованных источников 13](#_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.example.rgr"**>  
  
 <**application  
 android:allowBackup="true"  
 android:icon="@mipmap/ic\_launcher"  
 android:label="@string/app\_name"  
 android:roundIcon="@mipmap/ic\_launcher\_round"  
 android:supportsRtl="true"  
 android:screenOrientation="landscape"  
 tools:ignore="LockedOrientationActivity"  
 android:theme="@style/Theme.Rgr"**>  
 <**activity android:name=".MainActivity"**>  
 <**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.example.rgr;  
  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.os.Bundle;  
  
**public class** MainActivity **extends** AppCompatActivity {  
 @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.example.rgr;  
  
**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.graphics.Region;  
**import** android.view.GestureDetector;  
**import** android.view.MotionEvent;  
**import** android.view.View;  
  
  
**public class** Board **extends** View **implements** View.OnTouchListener {  
  
 Bitmap **unit**, **controller**;  
 **float startX** = 0, **startY** = 0, **ammountX** = 0, **ammountY** = 0;  
 **boolean controllerPressed** = **false**;  
 **int controllerSpaceRadius** = 220;  
 Integer **unitSpeed** = 4;  
 Region **regionForController**;  
 Integer **unitRotate** = 0;  
 **float unitPosX** = 0, **unitPosY** = 0;  
  
 RectF **rectGame**;  
 Paint **paint**, **paintButton**, **paintText**, **paintBorderGameBoard**;  
  
 **public** Board(Context context) {  
 **super**(context);  
  
 **unit** = BitmapFactory.*decodeResource*(getResources(), R.drawable.***juchok***);  
  
 Bitmap bitmapSource = BitmapFactory.*decodeResource*(getResources(), R.drawable.***circle***);  
 **controller** = Bitmap.*createBitmap*(bitmapSource);  
  
 **paint** = **new** Paint();  
 **paint**.setColor(Color.*rgb*(89, 114, 139));  
 **paint**.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***);  
  
 **paintButton** = **new** Paint();  
 **paintButton**.setColor(Color.***BLACK***);  
 **paintButton**.setStrokeWidth(14);  
 **paintButton**.setStyle(Paint.Style.***STROKE***);  
  
 **paintText** = **new** Paint();  
 **paintText**.setColor(Color.***BLACK***);  
 **paintText**.setAntiAlias(**true**);  
 **paintText**.setTextSize(48);  
 **paintText**.setTextAlign(Paint.Align.***CENTER***);  
  
 **new** Thread(() -> {  
 **while** (**true**) {  
 **if** (**controllerPressed**) {  
 **if** (**ammountX** < 0) {  
 **unitPosX** -= **unitSpeed**;  
 **if** (Math.*abs*(**ammountX**) > Math.*abs*(**ammountY**)) **unitRotate** = 270;  
  
 **if** (**unitPosX** < **rectGame**.**left**)  
 **unitPosX** = **rectGame**.**right**;  
  
 } **else if** (**ammountX** > 0) {  
 **unitPosX** += **unitSpeed**;  
 **if** (Math.*abs*(**ammountX**) > Math.*abs*(**ammountY**)) **unitRotate** = 90;  
  
 **if** (**unitPosX** > **rectGame**.**right**)  
 **unitPosX** = **rectGame**.**left**;  
 }  
  
 **if** (**ammountY** < 0) {  
 **unitPosY** -= **unitSpeed**;  
 **if** (Math.*abs*(**ammountX**) < Math.*abs*(**ammountY**)) **unitRotate** = 0;  
  
 **if** (**unitPosY** < **rectGame**.**top**)  
 **unitPosY** = **rectGame**.**bottom**;  
  
 } **else if** (**ammountY** > 0) {  
 **unitPosY** += **unitSpeed**;  
 **if** (Math.*abs*(**ammountX**) < Math.*abs*(**ammountY**)) **unitRotate** = 180;  
  
 **if** (**unitPosY** > **rectGame**.**bottom**)  
 **unitPosY** = **rectGame**.**top**;  
 }  
 }  
 **try** {  
 invalidate();  
 Thread.*sleep*(10);  
 } **catch** (InterruptedException e) {  
 e.printStackTrace();  
 }  
 }  
 }).start();  
 }  
  
 @Override  
 **protected void** onDraw(Canvas canvas) {  
 *//// DRAW LAYOUT* canvas.drawRGB(117, 144, 172);  
  
 *//// DRAW GAME BOARD* **rectGame** = **new** RectF(getWidth() / 2.5F, getHeight() / 12, getWidth() \* 0.98F, getHeight() \* 0.96F);  
 canvas.drawRect(**rectGame**, **paint**);  
  
 **if** (**unitPosX** == 0 && **unitPosY** == 0) {  
 **unitPosX** = **rectGame**.centerX();  
 **unitPosY** = **rectGame**.centerY();  
 }  
  
 *//// DRAW UNIT* 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(**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 CONTROLLER* canvas.drawCircle(getWidth() / 4, getPivotY(), **controllerSpaceRadius**, **paintButton**);  
 **regionForController** = **new** Region(getWidth() / 4 - **controllerSpaceRadius**, (**int**) getPivotY() - **controllerSpaceRadius**, getWidth() / 4 + **controllerSpaceRadius**, (**int**) getPivotY() + **controllerSpaceRadius**);  
  
 **startX** = getWidth() / 4 - **controller**.getWidth() / 2;  
 **startY** = getPivotY() - **controller**.getHeight() / 2;  
 canvas.drawBitmap(**controller**, **startX** + **ammountX**, **startY** + **ammountY**, **paintButton**);  
  
 invalidate();  
 }  
  
 @Override  
 **public boolean** onTouchEvent(MotionEvent event) {  
  
 **float** x = event.getX();  
 **float** y = event.getY();  
  
 *// Признак попадания в область с джостиком* **if** (**regionForController**.contains((**int**) x, (**int**) y)) {  
 *// Прикосновение к джостику* **if** (event.getAction() == MotionEvent.***ACTION\_DOWN***) {  
 **controllerPressed** = **true**;  
 **return true**;  
 }  
 *// Событие перетаскивания* **else if** (event.getAction() == MotionEvent.***ACTION\_MOVE***) {  
 **controllerPressed** = **true**;  
 **ammountX** = x - **startX** - **controller**.getWidth() / 2;  
 **ammountY** = y - **startY** - **controller**.getHeight() / 2;  
 **return true**;  
 }  
 *// Отпустить джостик* **else if** (event.getAction() == MotionEvent.***ACTION\_UP***) {  
 **ammountX** = 0;  
 **ammountY** = 0;  
 **controllerPressed** = **false**;  
 **return true**;  
 }  
 }  
 **return super**.onTouchEvent(event);  
 }  
  
 @Override  
 **public boolean** dispatchTouchEvent(MotionEvent event) {  
 **return super**.dispatchTouchEvent(event);  
 }  
  
 @Override  
 **public boolean** onTouch(View v, MotionEvent event) {  
 **return true**;  
 }  
}

Результат работы приложения показан на рисунках 1-3.

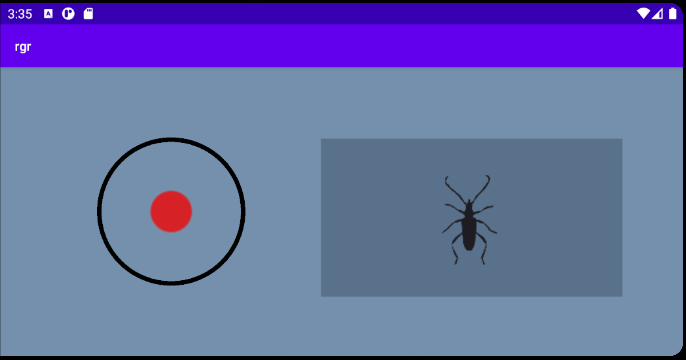


Рисунок 1 – Начальный экран приложения

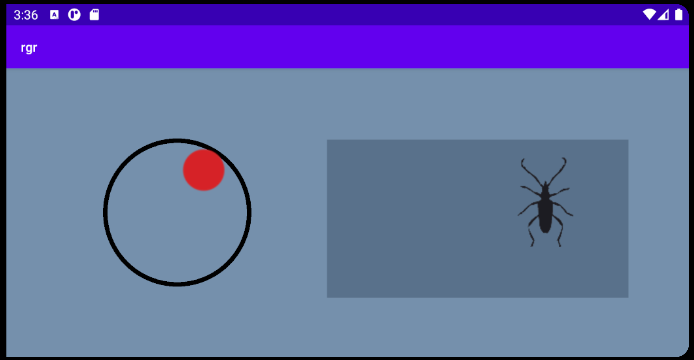


Рисунок 2 – Перемещение букашки (1)

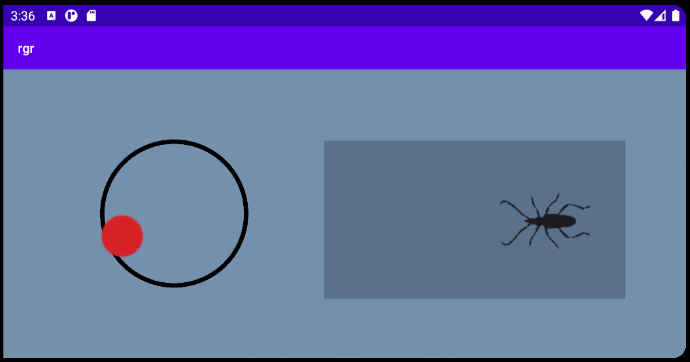


Рисунок 3 – Перемещение букашки (2)

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.