Некоммерческое акционерное общество
АЛМАТИНСКИЙ УНИВЕРСИТЕТ ЭНЕРГЕТИКИ И СВЯЗИ
Кафедра компьютерных технологий

**УТВЕРЖДАЮ**
Проректор по УМР
_____С. В. Коньшин
“_____”_____ 2016 г.

# ИНТЕРФЕЙСЫ КОМПЬЮТЕРНЫХ И МОБИЛЬНЫХ СИСТЕМ

Методические указания по выполнению расчетно-графических работ для студентов специальности 5В070400 – Вычислительная техника и программное обеспечение

СОГЛАСОВАНО
Начальник УМО
_____М. А. Мустафин
“____” _____2016 г.

Председатель УМС
_____ Б. К. Курпенов
“____”_____2016 г.

Специалист по стандартизации и редактор
_____ Г. А. Акетаева
“___” _____2016 г.

Рассмотрено и одобрено на заседании кафедры КТ,
протокол №11 от «05». 05 . 2016 г.
зав. кафедрой КТ
_____ З. К. Куралбаев

Составители (разработчики):
_____ Г. Т. Мусатаева
_____ А. Т. Конуспаева
_____ Д. О. Байжанова

**Некоммерческое акционерное общество**

**АЛМАТИНСКИЙ УНИВЕРСИТЕТ ЭНЕРГЕТИКИ И СВЯЗИ**

Кафедра компьютерных технологий

# ИНТЕРФЕЙСЫ КОМПЬЮТЕРНЫХ И МОБИЛЬНЫХ СИСТЕМ

Методические указания по выполнению расчетно-графических работ для студентов специальности
5В070400 – Вычислительная техника и программное обеспечение

Алматы 2017

СОСТАВИТЕЛИ: Мусатаева Г.Т., Конуспаева А.Т., Байжанова Д.О. Интерфейсы компьютерных и мобильных систем. Методические указания по выполнению расчетно-графических работ для студентов специальности 5В070400 – Вычислительная техника и программное обеспечение. - Алматы: АУЭС, 2017. - 32 с.

Методические указания предназначены для студентов специальности 5В070400 – Вычислительная техника и программное обеспечение.

Методическая разработка составлена в соответствии с требованиями квалификационной характеристики специалиста, Государственных стандартов, Фирменного стандарта «Работы учебные. Общие требования к построению, изложению, оформлению», Государственного стандарта, а также программы курса. Они составлены с учетом активизации процесса изучения основ курса, закреплению лекционного курса и практических занятий. Методические указания служат основой для выполнения расчетно-графических работ по дисциплине. Материал методических указаний охватывает основные разделы курса.

Иллюстрация – 10, библиография – 14 названий.

Рецензент: к.т.н., доцент кафедры ЭСиЭЭС Курпенов Б.К.

Печатается по плану издания некоммерческого акционерного общества «Алматинский университет энергетики и связи» на 2016 г.

© НАО «Алматинский университет энергетики и связи», 2017 г.

## Введение

При подготовке квалифицированных специалистов используется многообразная и целостная система организационных форм и методов обучения: лекции, семинарские и лабораторно-практические занятия, учебная практика, расчетно-графическая работа и т.д.

Подготовка расчетно-графических работ является одной из важных форм обучения, т.к. она позволяет:

- систематизировать, закреплять и расширять теоретические и практические знания по специальности и применять их при решении конкретных задач;
- развить навыки самостоятельной работы;
- определить уровень подготовленности студентов.

В современных условиях хозяйствования, обусловленных переходом к рыночным отношениям, нестабильным экономическим положением, развитием новых форм хозяйствования, резко возрастают требования к качеству обработки информации. Ломка существующей системы информационного обслуживания привела к своеобразному информационному кризису, реальным выходом из которого является использование современных информационных технологий.

Расчетно-графическая работа по данной дисциплине является действенным элементом учебного процесса, способствующим закреплению, углублению, обобщению и прикладному применению знаний, получаемых студентом при изучении курсов «Информатика», «Программирование на алгоритмических языках», «Технология программирования», «Операционные системы».

## 1 Цель расчетно-графических работ

Цель расчетно-графических работ заключается в формировании у студента навыков практической работы, связанной с проектированием, разработкой и реализацией информационных систем на отдельных участках.

Данная цель может быть достигнута при успешном решении студентами следующего круга задач:

- изучение состава и функций автоматизированных информационных систем;
- приобретение навыков практической разработки проектов;
- освоение работы с основными видами прикладного программного обеспечения.

## 2 Тематика и методические указания по выполнению расчетно-графических работ

### 2.1 Темы расчетно-графических работ

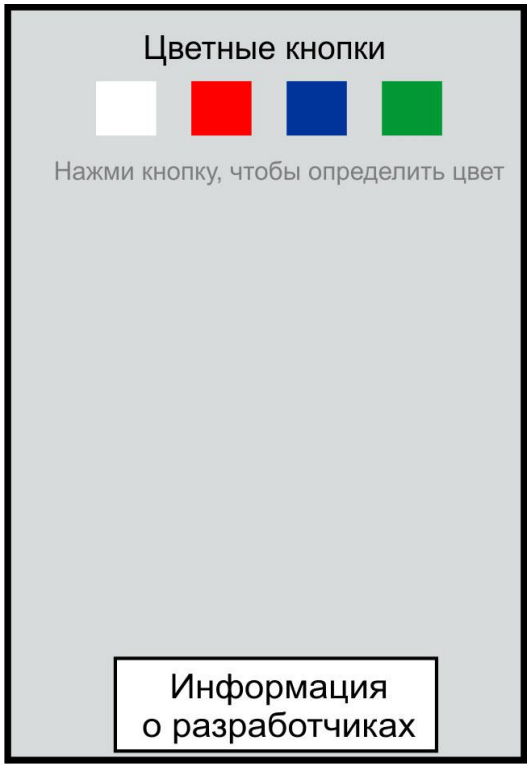
*Примерный перечень тем расчетно-графической работы №1:*

1) Разработка приложения «Угадай число», имеющее следующий вид:

2) Разработка приложения «Информация о программе», имеющее следующий вид:

3) Разработка приложения «Тренажер пальцев» с соблюдением структуры макета и цветовой палитры:

4) Разработка приложения «Определитель цвета» с соблюдением структуры макета и цветовой палитры:




5) Разработка приложения «Детская викторина» с соблюдением структуры макета и цветовой палитры:

6) Разработка приложения «Математическая викторина» с соблюдением структуры макета и цветовой палитры. При нажатии кнопки «Помощь» вывести определение простого числа:

7) Разработка приложения «Сапёр» с соблюдением структуры макета и цветовой палитры:

8) Разработка приложения «Компас» с соблюдением структуры макета и цветовой палитры. При нажатии на кнопки компаса выводить русское название стороны света:

*Примерный перечень тем расчетно-графической работы №2:*

1) Разработка фрагментов пользовательского интерфейса для мобильной системы управления библиотечным фондом.

2) Разработка фрагментов пользовательского интерфейса для мобильной системы управления доской объявлений о купле/продаже произвольных товаров.

3) Разработка фрагментов пользовательского интерфейса для мобильной системы регистрации нарушений правил дорожного движения.

4) Разработка фрагментов пользовательского интерфейса для мобильной системы управления лентой новостей.

5) Разработка фрагментов пользовательского интерфейса для мобильной системы кадрового учета предприятия.

6) Разработка фрагментов пользовательского интерфейса для мобильной системы управления конструированием ювелирных украшений.

7) Разработка фрагментов пользовательского интерфейса для мобильной системы информационной поддержки преподавателя.

8) Разработка фрагментов пользовательского интерфейса для мобильной системы рассылки рекламных новостей и уведомлений.

*Примерный перечень тем расчетно-графической работы №3:*

1) Проектирование и разработка программного приложения для домашней аудиотеки.

2) Проектирование и разработка программного приложения для тренера спортивной команды.

3) Проектирование и разработка программного приложения для агентства по аренде квартир.

4) Проектирование и разработка программного приложения для модельного агентства.

5) Проектирование и разработка программного приложения для учета услуг, оказываемых юридической консультационной фирмой.

6) Проектирование и разработка программного приложения для автосервисной фирмы.

7) Проектирование и разработка программного приложения для автозаправочной станции.

8) Проектирование и разработка программного приложения центра по продаже автомобилей.

### 2.2 Методические указания к выполнению расчетно-графических работ

Результатом выполнения раздела по разработке интерфейса должна стать детально спланированная и реализованная структура меню, отражающая весь функционал системы. Спроектированная работа должна быть проверена на контрольном примере.

Расчетно-графические работы могут быть выполнены для любой предметной области. Варианты заданий расчетно-графических работ выбираются по списку преподавателя.

**Расчетно-графическая работа 1. Практика разработки простейших приложений для Android**

Цель расчетно-графической работы:

1) Познакомиться с инструментами разработки Android-приложений.

2) На примере простейших программ разобрать структуру типичного Android-приложения.

3) Научиться запускать приложение на эмуляторе.

4) Научиться тестировать приложение с помощью Dalvik Debug Monitor Server (DDMS).

План расчетно-графической работы:

1) Создать эмулятор x86.

2) Разобрать следующие приложения, запустить их на эмуляторе и выполнить для каждого примера дополнительное задание:

1) Hello, World!

2) Работа с кнопками;

3) Работа с анимацией;

4) Работа с GPS.

Необходимое программное обеспечение

1) Java Development Kit.

2) Eclipse с установленным плагином Android Development Tools.

3) Android SDK.

4) Android NDK.

Intel Hardware Accelerated Execution Manager.

Инструкции по выполнению расчетно-графической работы.

Работа над приложениями 2-4 начинается с импорта подготовленных проектов в рабочую среду (Workspace). Импорт осуществляется следующим образом:

1) File -> New -> Project -> Android Project from Existing Code.

2) Далее указывается директория, в которой находится проект и пример копируется в рабочую среду (Copy project into workspase).

Создание и настройка эмулятора.

Запустите Android Virtual Devices Manager: Window -> AVD Manger.

Создайте новое виртуальное устройство (New).

Настройте эмулятор: задайте название новому устройству, выберите целевой платформой Intel Atom x86 (по желанию можно задать объем SD-карты) и нажмите Create AVD (рисунок 1).

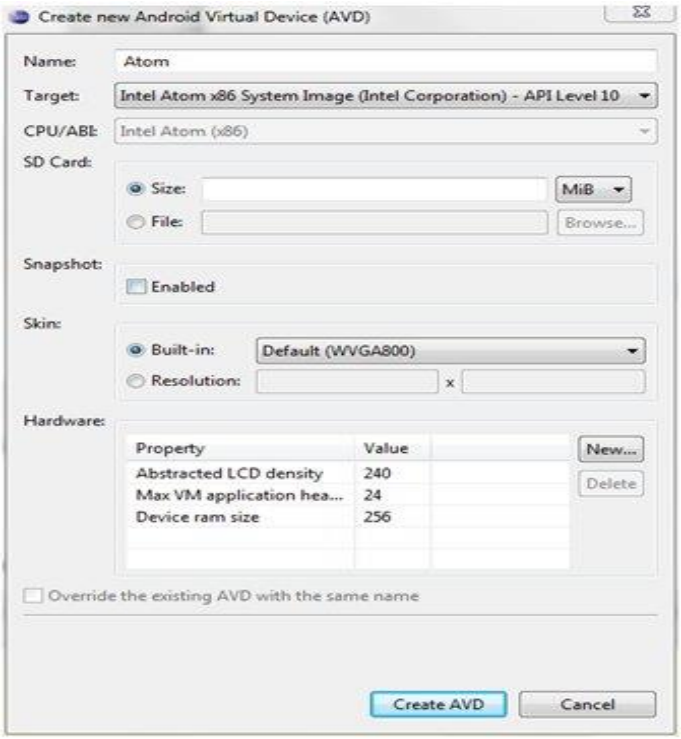



Рисунок 1 - Настройка эмулятора

Созданный эмулятор появится в списке виртуальных устройств.

Создайте новый проект: File -> New -> Project -> Android Application Project.

Осуществите необходимые настройки проекта: задайте название, выберите SDK, по желанию можно нарисовать иконку своего приложения (рисунок 2).

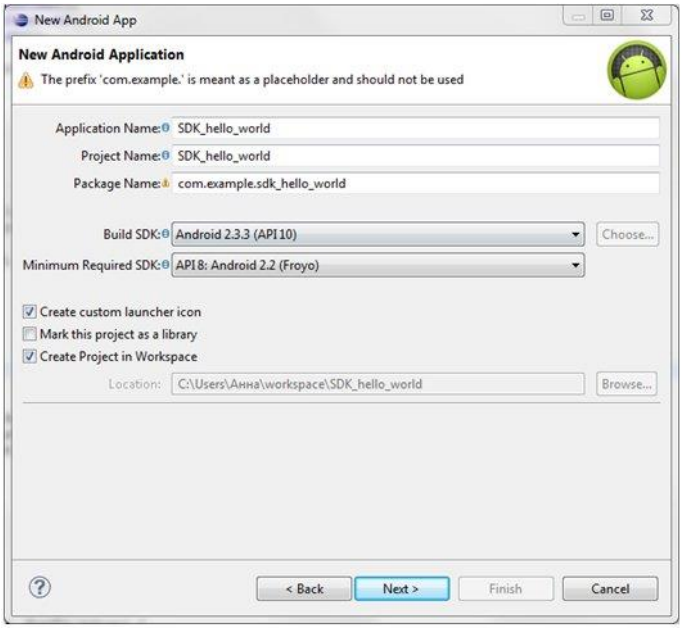



Рисунок 2 - Создание нового Android-проекта

1. Далее появится окно с предложением создать Activity. Выберите BlankActivity и дайте ему какое-нибудь название.

2. Ваше первое приложение готово. Запустите его на эмуляторе.

3. Откройте файлы res/layout/activity_main.xml, src/MainActivity.java и AndroidManifest.xml. Посмотрите, что в них содержится. Зная, как выглядит готовое приложение, постарайтесь разобраться, за что отвечает каждая составляющая этих файлов.

Приложение к расчетно-графической работе 1.

Данное приложение содержит тексты программ, рассматриваемых в примерах.

1. Button Example

res/layout/activity_main.xml

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
package="com.example.application"
android:versionCode="1"
android:versionName="1.0" >

<uses-sdk
 android:minSdkVersion="8"
 android:targetSdkVersion="15" />

 <application
android:icon="@drawable/ic_launcher"
android:label="@string/app_name">
<activity
android:name=".MainActivity"
android:label="@string/title_activity_main" >
<intent-filter>
<action android:name="android.intent.action.MAIN" />

<category android:name="android.intent.category.LAUNCHER" />
 </intent-filter>
 </activity>
 </application>

</manifest>
```

src/MainActivity.java

```
package com.example.application;

import android.app.Activity;
import android.graphics.Color;
import android.os.Bundle;
```

```
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.LinearLayout;
import android.widget.Toast;

public class MainActivity extends Activity implements OnClickListener {

private Button switchToGreen;
private Button switchToRed;
private Button switchToBlue;
private LinearLayout screenLayout;
private Toast informationToast;

@Override
public void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);

//init buttons
switchToBlue = (Button) findViewById(R.id.switchBlue);
switchToGreen = (Button) findViewById(R.id.switchGreen);
switchToRed = (Button) findViewById(R.id.switchRed);
screenLayout = (LinearLayout) findViewById(R.id.screenLayout);

//setup listeners
switchToBlue.setOnClickListener(this);
switchToRed.setOnClickListener(this);
switchToGreen.setOnClickListener(this);

informationToast = Toast.makeText(this, "", Toast.LENGTH_SHORT);
}

public void onClick(View view) {
if (switchToBlue.equals(view)) {
screenLayout.setBackgroundColor(Color.BLUE);
showToast("Hello blue");
} else if (switchToRed.equals(view)) {
screenLayout.setBackgroundColor(Color.RED);
showToast("Hello red");
} else if (switchToGreen.equals(view)) {
screenLayout.setBackgroundColor(Color.GREEN);
showToast("Hello green");
}
```

```
}

private void showToast(String text) {
informationToast.cancel();
informationToast.setText(text);
informationToast.show();
}
}
```

Animation Example

res/anim/frame_anim.xml

```
<?xml version="1.0" encoding="utf-8"?>
<animation-list xmlns:android="http://schemas.android.com/apk/res/android"
android:oneshot="false" >

<item
android:drawable="@drawable/ic_launcher"
android:duration="200"/>
<item
android:drawable="@drawable/ic_launcher1"
android:duration="200"/>
<item
android:drawable="@drawable/ic_launcher2"
android:duration="200"/>
<item
android:drawable="@drawable/ic_launcher3"
android:duration="200"/>

</animation-list>
```

res/anim/transform_anim.xml

```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android"
android:shareInterpolator="false">

<scale
android:duration="700"
android:fillAfter="false"
android:fromXScale="1.0"
android:fromYScale="1.0"
android:interpolator="@android:anim/accelerate_decelerate_interpolator"
android:pivotX="50%"
android:pivotY="50%"
android:toXScale="1.4"
android:toYScale="0.6" />
```

```
<set android:interpolator="@android:anim/decelerate_interpolator" >
<scale
android:duration="400"
android:fillBefore="false"
android:fromXScale="1.4"
android:fromYScale="0.6"
android:pivotX="50%"
android:pivotY="50%"
android:startOffset="700"
android:toXScale="0.0"
android:toYScale="0.0" />

<rotate
android:duration="400"
android:fromDegrees="0"
android:pivotX="50%"
android:pivotY="50%"
android:startOffset="700"
android:toDegrees="-45"
android:toYScale="0.0" />
</set>

</set>
```

src/MainActivity.java

```
package com.example.application;

import android.app.Activity;
import android.graphics.Color;
import android.graphics.drawable.AnimationDrawable;
import android.os.Bundle;
import android.view.View;
import android.view.View.OnClickListener;
import android.view.animation.Animation;
import android.view.animation.AnimationUtils;
import android.widget.Button;
import android.widget.ImageView;

public class MainActivity extends Activity implements OnClickListener {

private Button startFrameAnim;
private Button startTransformAnim;
private Button cancelAnim;
```

```
private ImageView animationView;

@Override
public void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);

startFrameAnim = (Button) findViewById(R.id.frameAnimationStart);
startTransformAnim=(Button) findViewById(R.id.transformAnimationStart);
cancelAnim = (Button) findViewById(R.id.cancelAnimation);
animationView = (ImageView) findViewById(R.id.animationView);

startFrameAnim.setOnClickListener(this);
startTransformAnim.setOnClickListener(this);
cancelAnim.setOnClickListener(this);
}

public void onClick(View v) {
if (startFrameAnim.equals(v)) {
animationView.setBackgroundResource(R.anim.frame_anim);
AnimationDrawable animation =
(AnimationDrawable) animationView.getBackground();
animation.start();
} else if (startTransformAnim.equals(v)) {
animationView.setBackgroundResource(R.drawable.ic_launcher);
Animation transformAnimation =
AnimationUtils.loadAnimation(this, R.anim.transform_anim);
animationView.startAnimation(transformAnimation);
} else if (cancelAnim.equals(v)) {
animationView.setBackgroundColor(Color.BLACK);
 }
 }
}
```

Location Example

src/MainActivity.java

```
package com.example.application;

import java.util.Date;

import android.app.Activity;
import android.location.Criteria;
import android.location.Location;
import android.location.LocationListener;
```

```
import android.location.LocationManager;
import android.os.Bundle;
import android.widget.TextView;

public class MainActivity extends Activity implements LocationListener {
private TextView latitudeLabel;
private TextView longitudeLabel;
private TextView statusLabel;
private LocationManager locationManager;

@Override
public void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);

latitudeLabel = (TextView) findViewById(R.id.latitudeLabel);
longitudeLabel = (TextView) findViewById(R.id.longitudeLabel);
statusLabel = (TextView) findViewById(R.id.statusLabel);

locationManager=(LocationManager)
getSystemService(Activity.LOCATION_SERVICE);
}

@Override
protected void onResume() {
super.onResume();
// construct a criteria with best accuracy
Criteria criteria = new Criteria();
criteria.setAccuracy(Criteria.ACCURACY_FINE);
// get best ENABLED provider that meets the criteria
String provider = locationManager.getBestProvider(criteria, true);
//request the updates
locationManager.requestLocationUpdates(provider, 0, 0, this);
}

@Override
protected void onPause() {
super.onPause();
locationManager.removeUpdates(this);
}

public void onLocationChanged(Location location) {
statusLabel.setText("Location recieved at " + new Date());
latitudeLabel.setText("Latitude: " + location.getLatitude());
```

```
longitudeLabel.setText("Longitude: " + location.getLongitude());
}

public void onProviderDisabled(String provider) {
}

public void onProviderEnabled(String provider) {
}

public void onStatusChanged(String provider, int status, Bundle extras) {
}
}
```

AndroidManifest.xml

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
package="com.example.application"
android:versionCode="1"
android:versionName="1.0" >

<uses-sdk
android:minSdkVersion="8"
android:targetSdkVersion="15" />

<uses-permission
android:name="android.permission.ACCESS_COARSE_LOCATION"/>
<uses-permission
android:name="android.permission.ACCESS_FINE_LOCATION"/>

<application
android:icon="@drawable/ic_launcher"
android:label="@string/app_name">
<activity
android:name=".MainActivity"
android:label="@string/title_activity_main" >
<intent-filter>
<action android:name="android.intent.action.MAIN" />

<category android:name="android.intent.category.LAUNCHER" />
</intent-filter>
</activity>
</application>

</manifest>
```

**Расчетно-графическая работа 2. Основы тестирования и отладки приложений на смартфоне**

Задачи расчетно-графической работы - рассмотреть создание тестирующего приложения на основе JUnit.

Инструкции по выполнению расчетно-графической работы.

Создание тестирующего приложения.

1. В качестве приложения для тестирования будем использовать уже готовое приложение, входящее в набор созданных разработчиками Android SDK примеров. Приложение называется Spinner и содержится в наборе примеров (Samples). Мы взяли приложение из набора примеров для API 8.

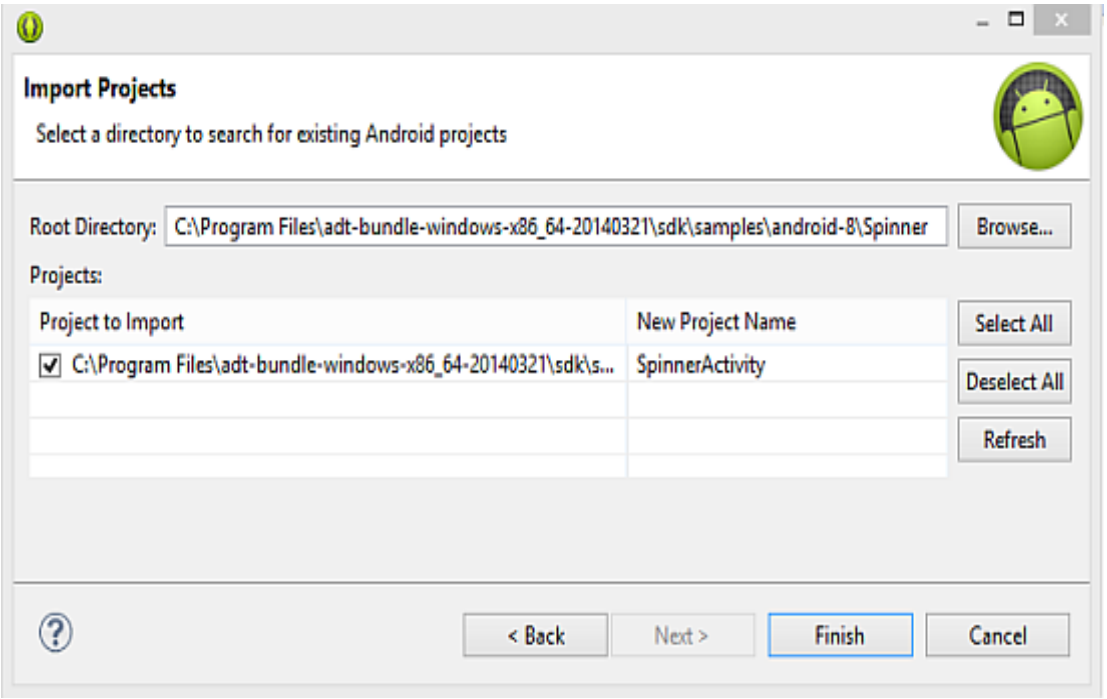



Рисунок 3 - Импорт существующего Android-приложения

Запустите Eclipse, создайте новое Android приложение, используя уже существующее:

New->Project...->Android->Android Project from Existing Code

2. Создадим тестирующее приложение.

В Eclipse New->Project...->Android->Android Test Project

Задайте имя проекта, обычно имя тестирующего проекта собирается следующим образом: имя тестируемого проекта + слово Test, в нашем случае получится SpinnerActivityTest. Далее необходимо указать тестируемый проект.

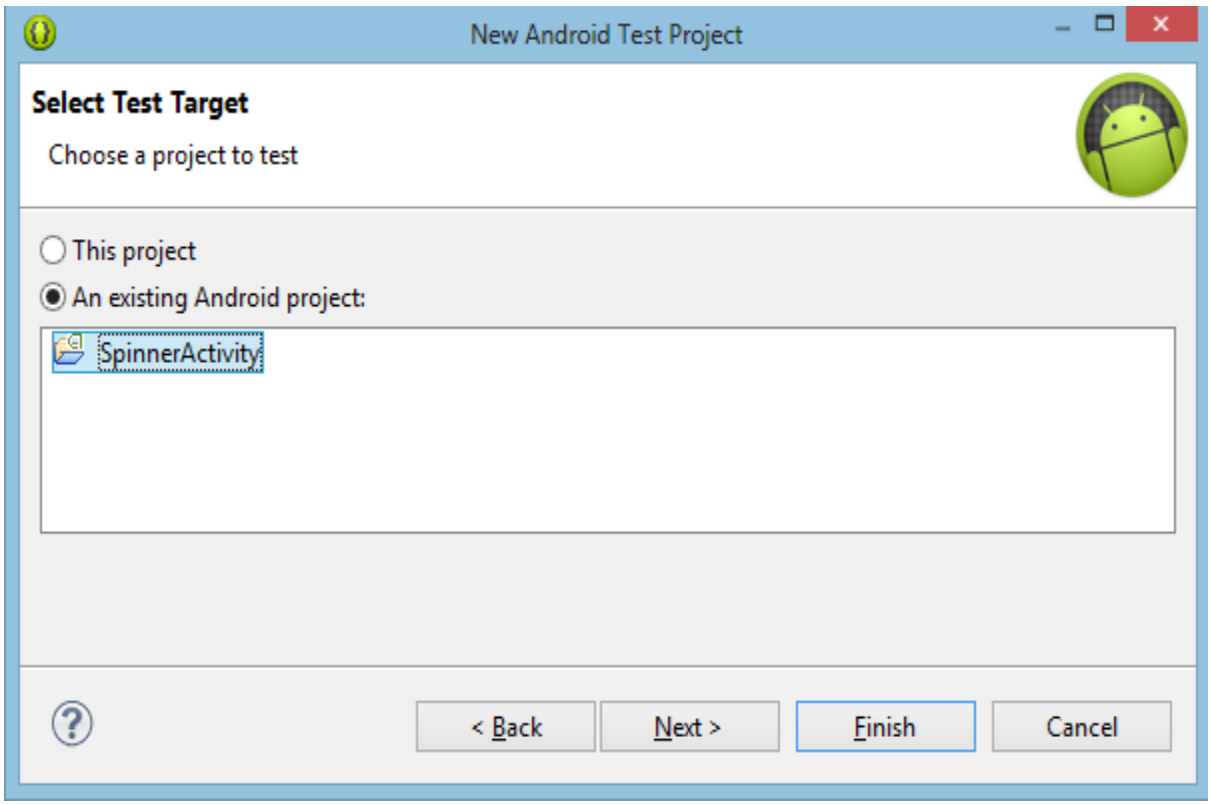



Рисунок 4 - Выбор тестируемого проекта

Теперь у нас есть простейшее Android приложение (SpinnerActivity) и пустое тестирующее приложение (SpinnerActivityTest).

Файл AndroidManifest.xml тестирующего приложения выглядит, как изображено на рисунке 5. В разделе <instrumentation> указано, где располагается тестируемый проект.

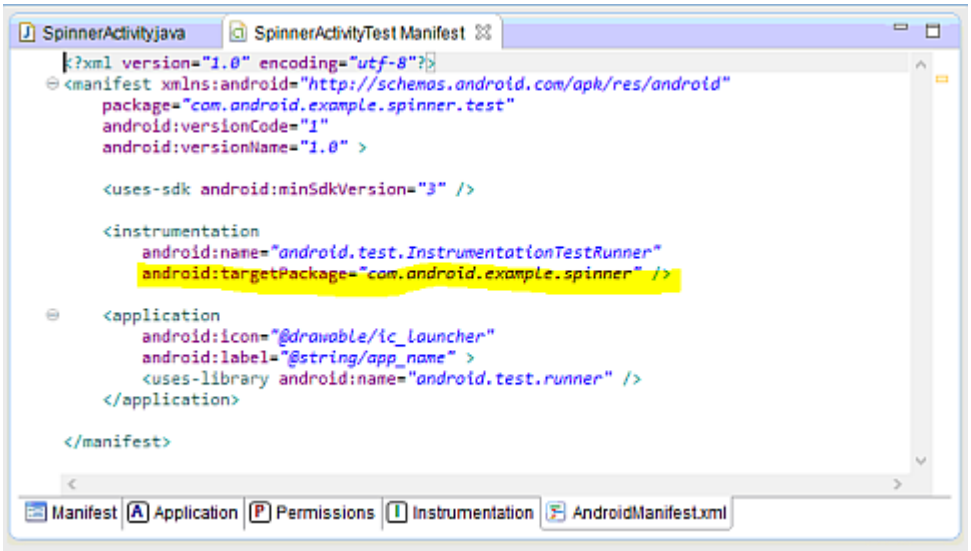



Рисунок 5 - Файл манифеста приложения SpinnerActivityTest

3. Создадим класс тестов.

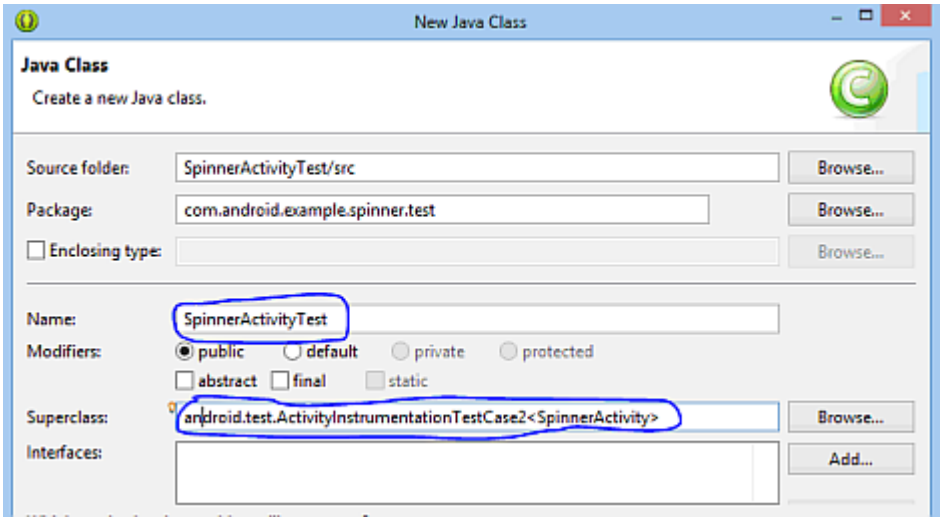



Рисунок 6 - Создание класса тестовых сценариев

Запуск теста и просмотр результатов.

Перед запуском теста необходимо удостовериться, что запущен эмулятор или подключено устройство, на котором будет производиться тестирование.

Для запуска теста в Package Explorer щелкните правой кнопкой на названии проекта тестов (SpinnerActivityTest), а затем выберите Run As→Android JUnit Test.

После запуска теста появится новая вкладка JUnit рядом с вкладкой Package Explorer.

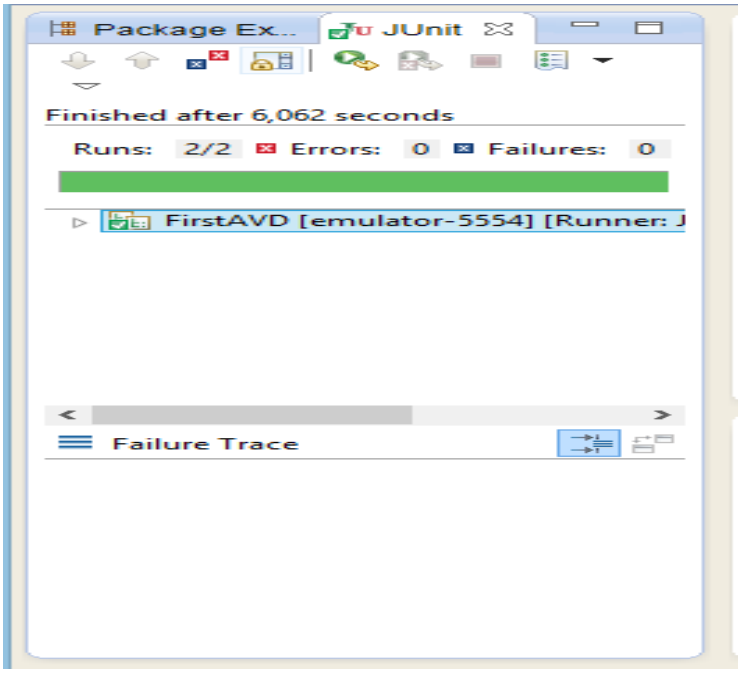



Рисунок 7 - Вкладка JUnit

**Расчетно-графическая работа 3. Использование мобильной связи в приложениях для смартфона**

Задачи расчетно-графической работы:

1) Разработать приложение, которое может звонить по телефонным номерам.

2) Проанализировать работу этого приложения на смартфоне.

Инструкции по выполнению расчетно-графической работы.

Приложение, описанное ниже, позволяет выполнять телефонные звонки через встроенное приложение для звонков, используя при этом собственный список контактов. Конечно, то же самое можно сделать вручную или добавить эти номера в адресную книгу телефона. Однако описанный способ может пригодится в некоторых ситуациях, например, в качестве части внутреннего приложения для хранения заранее определенного списка номеров сотрудников компании. В этом случае пользователь, используя корпоративное приложение, может вызывать нужные номера со своего личного смартфона, не добавляя контакты в адресную книгу. В некоторых случаях таких номеров может быть очень много, и специальное приложение позволит использовать рабочий справочник, не смешивая служебную информацию с личной.

Мы создадим приложение, содержащее несколько кнопок. Нажатие на кнопку вызывает конкретный телефонный номер, проассоциированный с ней. Для простоты понимания в программе не используются базы данных, все нужные сведения «защиты» в коде приложения. При желании несложно доработать программу и вынести эту информацию в отдельный файл.

Создайте проект SimpleDialer. Рекомендуемые настройки на рисунке 8.

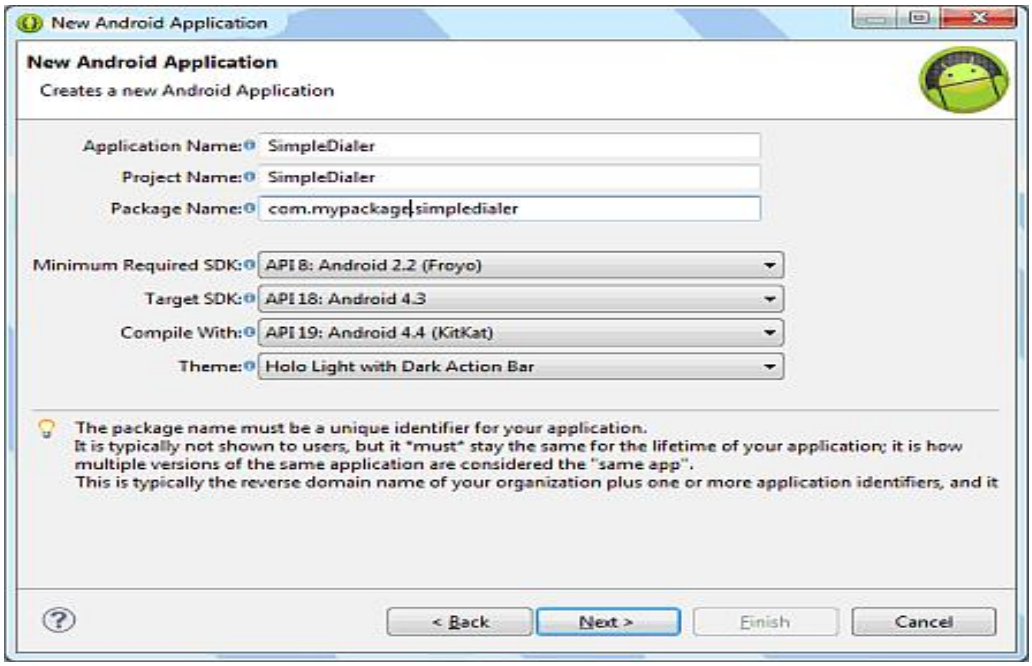



Рисунок 8 - Создание нового приложения

Добавьте следующую строчку в файл strings.xml:

```
<string name="main_label">My Friends</string>
```

В файле layout.xml создайте объекты TextView и шесть кнопок:

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
android:layout_width="fill_parent"
android:layout_height="fill_parent"
android:orientation="vertical"
android:padding="15dip" >
<TextView
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_gravity="center"
android:layout_marginBottom="25dip"
android:text="@string/main_label"
android:textSize="22sp" />
<Button
android:id="@+id/button1"
android:layout_width="fill_parent"
android:layout_height="wrap_content"
android:textSize="18sp" />
…//здесь добавить описания остальных кнопок
</LinearLayout>
```

Перейдите в файл MainActivity.java. Создайте переменную для хранения числа записей и два массива строк:

```
private int entries = 6;
private String phoneNum[];
private String buttonLabels[];
```

Так как кнопки будут реагировать на нажатие, добавьте в заголовок класса implements OnClickListener (нужно также добавить import android.view.View.OnClickListener;).

В методе onCreate() пропишите:

```
phoneNum = new String[entries];
buttonLabels = new String[entries];
```

и создайте шесть кнопок:

```
Button button1 = (Button)findViewById(R.id.button1);
button1.setText(buttonLabels[0]);
button1.setOnClickListener(this);

Button button2 = (Button)findViewById(R.id.button2);
button2.setText(buttonLabels[1]);
button2.setOnClickListener(this)
```

Создайте новый метод lunchDialer():

```
public void launchDialer(String number){
String numberToDial = "tel:"+number;
startActivity(new Intent(Intent.ACTION_DIAL, Uri.parse(numberToDial)));
}
```

Данный метод отвечает за вызов встроенного телефонного приложения, которое позволяет совершать звонки. Он получает номер телефона в качестве параметра и создает специально отформатированную строку numberToDial, которая является универсальным идентификатором (Uniform Resource Identifier). Далее создается новый интент со стандартным действием ACTION_DIAL (звонок), который используется для старта активности, отвечающей за звонки.

Заполните массивы через метод populateArrays()(не забудьте вызвать его в начале):

```
public void populateArrays(){
phoneNum[0] = "123-456-78-90";
phoneNum[1] = "234-567-89-01";
phoneNum[2] = "345-678-90-12";
phoneNum[3] = "456-789-01-23";
phoneNum[4] = "567-890-12-34";
phoneNum[5] = "678-901-23-45";
buttonLabels[0] = "Иванов Ваня";
buttonLabels[1] = "Петров Петя";
buttonLabels[2] = "Семеныч Сеня";
buttonLabels[3] = "Кузнецова Катя";
buttonLabels[4] = "Смирнова Саша";
buttonLabels[5] = "Попова Полина";
}
```

Далее опишите метод onClick():

```
@Override
public void onClick(View v) {
switch (v.getId()) {
case R.id.button1:
launchDialer(phoneNum[0]);
break;
case R.id.button2:
launchDialer(phoneNum[1]);
break;
case R.id.button3:
launchDialer(phoneNum[2]);
break;
```

```
case R.id.button4:
launchDialer(phoneNum[3]);
break;
case R.id.button5:
launchDialer(phoneNum[4]);
break;
case R.id.button6:
launchDialer(phoneNum[5]);
break;
}
```

Приложение можно запускать.

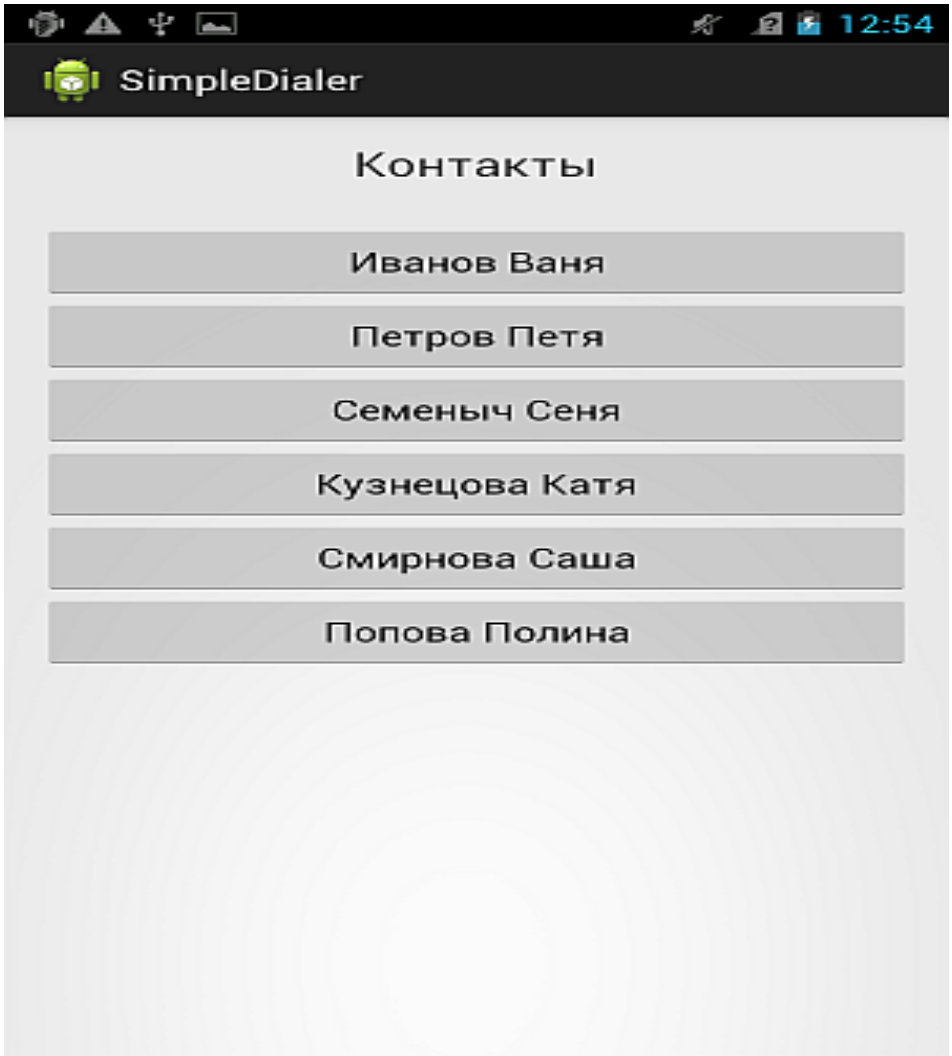



Рисунок 9 - Приложение, запущенное на устройстве

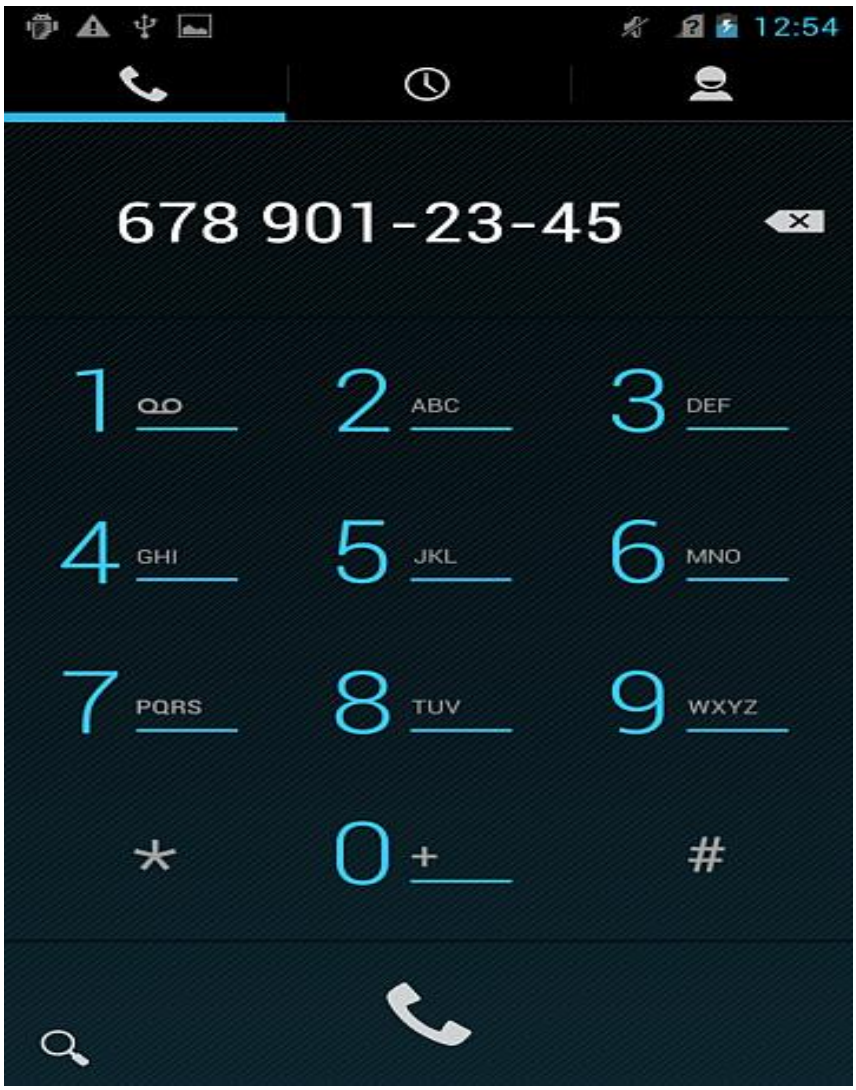



Рисунок 10 - С помощью приложения можно совершать звонки

Листинг 1. Файл strings.xml

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
android:layout_width="fill_parent"
android:layout_height="fill_parent"
android:orientation="vertical"
android:padding="15dip" >

<TextView
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_gravity="center"
android:layout_marginBottom="25dip"
android:text="@string/main_label"
android:textSize="22sp" />

<Button
```

```
android:id="@+id/button1"
android:layout_width="fill_parent"
android:layout_height="wrap_content"
android:textSize="18sp" />

<Button
android:id="@+id/button2"
android:layout_width="fill_parent"
android:layout_height="wrap_content"
android:textSize="18sp" />

<Button
android:id="@+id/button3"
android:layout_width="fill_parent"
android:layout_height="wrap_content"
android:textSize="18sp" />

<Button
android:id="@+id/button4"
android:layout_width="fill_parent"
android:layout_height="wrap_content"
android:textSize="18sp" />

<Button
android:id="@+id/button5"
android:layout_width="fill_parent"
android:layout_height="wrap_content"
android:textSize="18sp" />

<Button
android:id="@+id/button6"
android:layout_width="fill_parent"
android:layout_height="wrap_content"
android:textSize="18sp" />

</LinearLayout>
```

Листинг 2. Файл activity_main.xml

```
package mypackage.simpledialer;
import android.os.Bundle;
import android.app.Activity;
import android.view.Menu;
import android.content.Intent;
import android.net.Uri;
```

```
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;

public class MainActivity extends Activity implements OnClickListener {

private int entries = 6;
private String phoneNum[];
private String buttonLabels[];

@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);

phoneNum = new String[entries];
buttonLabels = new String[entries];

populateArrays();

Button button1 = (Button)findViewById(R.id.button1);
button1.setText(buttonLabels[0]);
button1.setOnClickListener(this);

Button button2 = (Button)findViewById(R.id.button2);
button2.setText(buttonLabels[1]);
utton2.setOnClickListener(this);

Button button3 = (Button)findViewById(R.id.button3);
button3.setText(buttonLabels[2]);
button3.setOnClickListener(this);

Button button4 = (Button)findViewById(R.id.button4);
button4.setText(buttonLabels[3]);
button4.setOnClickListener(this);

Button button5 = (Button)findViewById(R.id.button5);
button5.setText(buttonLabels[4]);
button5.setOnClickListener(this);

Button button6 = (Button)findViewById(R.id.button6);
button6.setText(buttonLabels[5]);
button6.setOnClickListener(this);
}
```

```
@Override
public boolean onCreateOptionsMenu(Menu menu) {
getMenuInflater().inflate(R.menu.main, menu);
return true;
}

public void launchDialer(String number){
String numberToDial = "tel:"+number;
startActivity(new Intent(Intent.ACTION_DIAL, Uri.parse(numberToDial)));
}
phoneNum[0] = "123-456-78-90";
phoneNum[1] = "234-567-89-01";
phoneNum[2] = "345-678-90-12";
phoneNum[3] = "456-789-01-23";
phoneNum[4] = "567-890-12-34";
phoneNum[5] = "678-901-23-45";
buttonLabels[0] = "Иванов Ваня";
buttonLabels[1] = "Петров Петя";
buttonLabels[2] = "Семеныч Сеня";
buttonLabels[3] = "Кузнецова Катя";
buttonLabels[4] = "Смирнова Саша";
buttonLabels[5] = "Попова Полина"; }

@Override
public void onClick(View v) {
switch (v.getId()) {

case R.id.button1:
launchDialer(phoneNum[0]);
break;

case R.id.button2:
launchDialer(phoneNum[1]);
break;

case R.id.button3:
launchDialer(phoneNum[2]);
break;

case R.id.button4:
launchDialer(phoneNum[3]);
break;
```

```
case R.id.button5:
launchDialer(phoneNum[4]);
break;

case R.id.button6:
launchDialer(phoneNum[5]);
break;
}
}
}
```

## Список литературы

1 Торрес Р.Дж. Практическое руководство по проектированию и разработке пользовательского интерфейса. - Пер. с англ. – М.: Вильямс, 2015.

2 Гасов В.М., Меньков А.В, Соломонов Л.А., Шигин А.В. Системное проектирование взаимодействия человека с техническими системами. Практическое пособие//Под ред.Четверикова В.Н. – М.: Высшая школа, 2016.

3 Жумагалиев Б.И. Средства взаимодействия в автоматизированных системах. Учебное пособие. – Алматы: КазНТУ, 2010.

4 Джеф Раскин, Интерфейс: новые направления в проектировании компьютерных систем. - Пер. с англ. - СПб.: Символ-Плюс, 2016.

5 Коутс Р., Влеймник И, Интерфейс «человек-машина». - М.: Мир, 2010.

6 Гасов В.М., Соломонов Л. А. Инженерно-психологическое проектирование взаимодействия человека с техническими средствами. Практическое пособие//Под ред,Четверикова В.Н. - М.: Высшая школа, 2010.

7 Соломонов Л.А,, Филипович Ю.Н., Шульгин В.А. Персональные автоматизированные информационные системы. Практическое пособие//Под ред. Четверикова В.Н. - М.: Высшая школа, 2009.

8 Алиев Т.М., Вигдоров Д.И., Кривошеев В.П. Системы отображения информации. - М.: Высшая школа, 2008.

9 Гасов В.М., Коротаев А.И., Сенькин С.И. Отображение информации. Практиеское пособие//Под ред. Четверикова В.Н. - М.: Высшая школа, 2011.

10 Айден К., Колесниченко О., Крамер М., Фибельман Х., Шишигин И. Аппаратные средства PC. - СПб.: BHV, 2008.

11 Борзенко А. IBM PC: устройство, ремонт, модернизация. – М.: Высшая школа, 2005.

12 Венда В.Ф., Инженерная психология и синтез систем отображения информации. – М.: Высшая школа, 2005.

13 Смоляров А.М. Системы отображения информации и инженерная психология. – М.: Высшая школа, 2012.

14 Основы инженерной психологии//Под ред. В.Ф.Ломова. - М.: Высшая школа, 2006.

## Содержание

Введение…….. 3
1 Цель расчетно-графических работ……. 3
2 Тематика и методические указания по выполнению расчетно-графических работ……... 4
2.1 Темы расчетно-графических работ…….. 4
2.2 Методические указания к выполнению расчетно-графических работ 8
Расчетно-графическая работа 1. Практика разработки простейших приложений для Android……. 9
Расчетно-графическая работа 2. Основы тестирования и отладки приложений на смартфоне…….. 18
Расчетно-графическая работа 3. Использование мобильной связи в приложениях для смартфона……... 21
Список литературы…….. 30

Сводный план 2016 г. поз. 152

Гульсим Толегеновна Мусатаева
Айгуль Толегеновна Конуспаева
Дина Ондасыновна Байжанова

ИНТЕРФЕЙСЫ КОМПЬЮТЕРНЫХ И МОБИЛЬНЫХ СИСТЕМ

Методические указания по выполнению расчетно-графических работ для студентов специальности
5В070400 – Вычислительная техника и программное обеспечение

Редактор Л.Т.Сластихина
Специалист по стандартизации Н.К.Молдабекова

Подписано в печать ___.___._____.
Тираж 20 экз.
Объем 2,0 уч.- изд. л.

Формат 60x84 1/16
Бумага типографская №1
Заказ _____ Цена 1000 тн.

Копировально-множительное бюро
некоммерческого акционерного общества
«Алматинский университет энергетики и связи»
050013 Алматы, Байтурсынулы, 126