## Критерии оценивания и шкала оценки

|  |  |
| --- | --- |
| **Оценка** | **Критерии выставления оценки** |
| «Отлично»  (8-10) | Программа решает поставленную задачу, имеет корректное оформление согласно кодовым соглашениям и содержит не более 2 архитектурных, смысловых или программных ошибок. |
| «Хорошо»  (6-7) | Программа решает поставленную задачу, имеет корректное оформление согласно кодовым соглашениям, имеет 3-4 архитектурных, смысловых или программных ошибки |
| «Удовлетворительно»  (4-5) | Программа решает поставленную задачу, имеет корректное оформление согласно кодовым соглашениям, имеет 5 архитектурных, смысловых или программных ошибок |
| «Неудовлетворительно» (0-3) | Программа не решает поставленную задачу и/или не имеет корректного оформления согласно кодовым соглашениями и/или содержит более 5 архитектурных, смысловых или программных ошибок. |

## Лабораторная работа №9 Разработка простейших приложений для Android

## *Лабораторная работа №9 выдается студентам в одном варианте. Срок выполнения - 2 недели. Форма представления отчета - программа, написанная на языке Java (код и скрины примера работы программы).*

### 2.1. Цель лабораторной работы

* Познакомиться с инструментами разработки Android-приложений.
* На примере простейших программ разобрать структуру типичного Android-приложения.
* Научиться запускать приложение на эмуляторе.
* Научиться тестировать приложение с помощью Dalvik Debug Monitor Server (DDMS).

### 2.2. План лабораторной работы

* Создать эмулятор x86.
* 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

### 2.3. Необходимое программное обеспечение

* Java Development Kit
* Eclipse с установленным плагином Android Development Tools
* Android SDK
* Android NDK
* Intel Hardware Accelerated Execution Manager

### 2.4. Инструкции по выполнению лабораторной работы

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

* **File -> New -> Project -> Android Project from Existing Code**
* Далее указывается директория, в которой находится проект и пример копируется в рабочую среду (Copy project into workspase).

#### 2.4.1. 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 ([Рис 2.1](https://intuit.ru/studies/courses/4462/988/lecture/14989?page=1#image.2.1)).

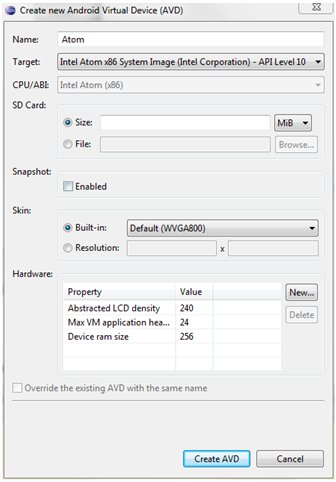


Рисунок 2.1 **–** Настройка эмулятора

* Созданный эмулятор появится в списке виртуальных устройств.

#### 2.4.2. Hello, World!

* Создайте новый проект: **File -> New -> Project -> Android Application Project**
* Осуществите необходимые настройки проекта: задайте название, выберите SDK, по желанию можно нарисовать иконку своего приложения ([Рис 2.2](https://intuit.ru/studies/courses/4462/988/lecture/14989?page=1#image.2.2)).

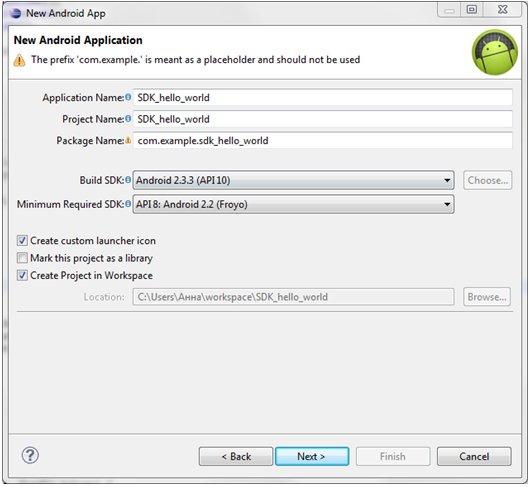


Рисунок 2.2 –Создание нового Android-проекта

* Далее появится окно с предложением создать Activity. Выберите BlankActivity и дайте ему какое-нибудь название.
* Ваше первое приложение готово. Запустите его на эмуляторе.
* 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.

#### 2.4.3. Работа с кнопками

На экране расположены три кнопки (Red, Green, Blue), при нажатии на которые меняется фоновый цвет и появляется соответствующее уведомление ([Рис 2.3](https://intuit.ru/studies/courses/4462/988/lecture/14989?page=1#image.2.3)).

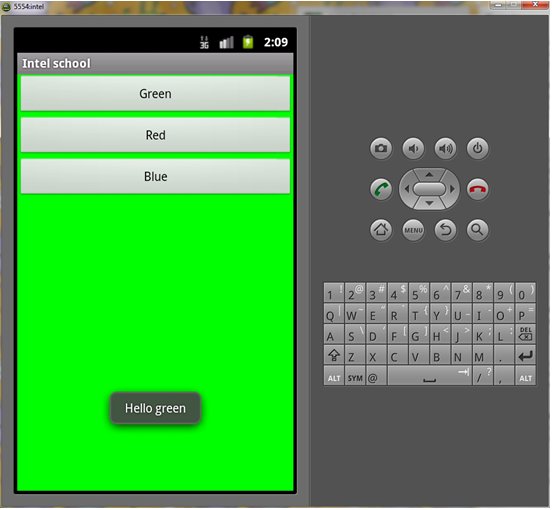


Рисунок 2.3 –Запущенное на эмуляторе приложение Button Example

* Импортируйте в рабочую область проект Button Example.
* Запустите его на эмуляторе и протестируйте.
* Ознакомьтесь с примером создания кнопки (res/layout/activity\_main.xml) и обработки ее нажатия (src/MainActivity.java).
* Дополните приложение еще одной кнопкой, при нажатии на которую в качестве фона устанавливалось бы какое-нибудь изображение (изображение скопируйте в директорию res/drawable).

#### 2.4.4. Работа с анимацией

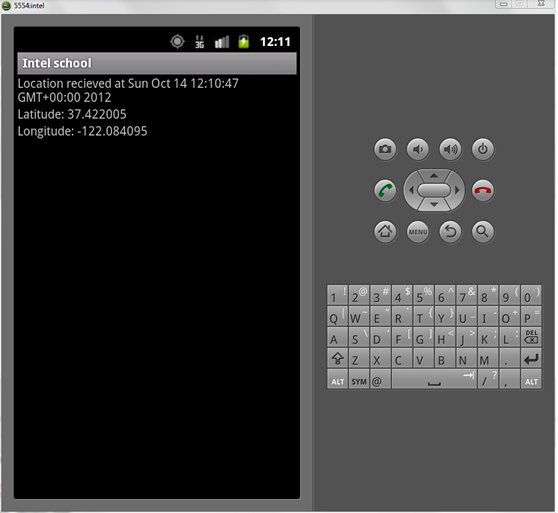
На экране расположены три кнопки (Frame animation, Transform animation, Cancel animation). При нажатии на первую кнопку воспроизводится покадровая анимация, при нажатии на вторую – анимация преобразований, при нажатии на третью анимация прекращается.

* Импортируйте в рабочую область проект Animaton Example.
* Запустите его на эмуляторе и протестируйте.
* Ознакомьтесь с примерами создания покадровой анимации и анимации преобразований (res/anim/frame\_anim.xml и res/anim/transform\_anim.xml) и примерами их применения к объекту (src/MainActivity.java).
* Дополните приложение реализацией своих примеров с анимацией. Описание возможных элементов анимации преобразований представлено в таблице:

|  |  |
| --- | --- |
| **Элемент** | **Атрибуты** |
| <alpha> анимация изменения прозрачности | fromAlpha – начальное значение прозрачности toAlpha – конечное значение прозрачности |
| <scale> анимация изменения размера | fromxScale – начальный масштаб по X  toxScale – конечный масштаб по X  fromYScale – начальный масштаб по Y  toYScale – конечный масштаб по Y  pivotX – Х-координата закрепленного центра  pivotY – Y-координата закрепленного центра |
| <translate> анимация движения (вертикальная/горизонтальная) | fromXDelta – начальное положение по X  toXDelta – конечное положение по X  fromYDelta – начальное положение по Y  toYDelta – конечное положение по Y |
| <rotate> анимация вращения | fromDegrees – начальный угол вращения  toDegrees – конечный угол вращения  pivotX – координата X центра вращения  pivotY – координата Y центра вращения |

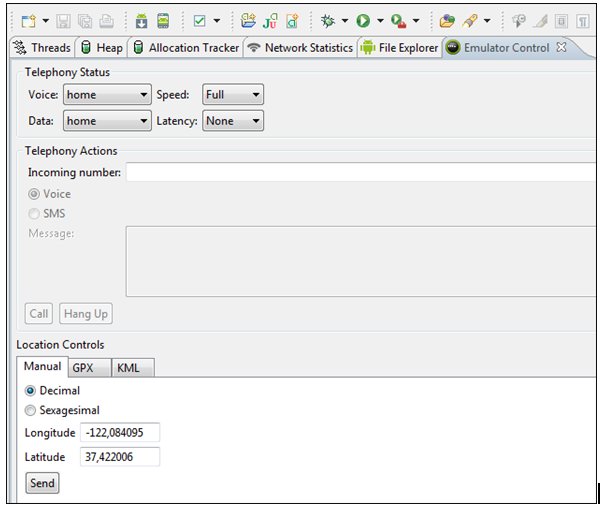
#### 2.4.5. Работа с GPS

На экране представлена следующая информация о местоположении ([Рис 2.4](https://intuit.ru/studies/courses/4462/988/lecture/14989?page=2#image.2.4)):



**Рисунок 2.4 –** Запущенное на эмуляторе приложение Location Example

* Статус (уведомление о том, когда были получены данные)
* Широта
* Долгота
* Импортируйте в рабочую область проект Location Example.
* Запустите его на эмуляторе. На экране появится уведомление "Waiting for location".
* Теперь виртуальному устройству необходимо передать информацию о местоположении. Для этого, не закрывая окно эмулятора, запустите DDMS (**Window -> Open Perspective -> Other -> DDMS**) и через вкладку Emulator Control отправьте устройству необходимые данные ([Рис 2.5](https://intuit.ru/studies/courses/4462/988/lecture/14989?page=2#image.2.5)).



**Рисунок 2.5 –** Значения долготы и широты отправляются на эмулятор с помощью DDMS

* Убедитесь, что программа работает корректно, и эмулятор вывел полученную информацию о местоположении.
* Обратите внимание на тег <uses-permission> в файле AndroidManifest.xml и подумайте, зачем он прописывается в данном приложении.
* Откройте файл src/MainActivity.java и постарайтесь разобраться, как происходит жизненный цикл Activity данного приложения, и какие действия совершаются на каждом этапе.

### ПРИЛОЖЕНИЕ

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();

}

}

1. 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);

}

}

}

1. 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>