МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
Университет ИТМО

А.В. Куркин

ПРОГРАММИРОВАНИЕ ПОД ПЛАТФОРМУ ANDROID

Учебное пособие

УНИВЕРСИТЕТ ИТМО

Санкт-Петербург
2015

УДК 004.655, 004.451, 621.396.21(074.8)

Куркин А.В, Программирование под платформу Andriod. Учебное пособие. – СПб.: Университет ИТМО, 2015. – 35с.

В пособии излагаются методические рекомендации к выполнению лабораторных работ по дисциплине «Программирование под платформу Andriod». Предназначено для студентов, обучающихся по всем профилям подготовки бакалавров направления: 210700 Инфокоммуникационные технологии и системы связи.

Рекомендовано к печати решением совета Естественнонаучного факультета (Протокол № 1 от 29 января 2015 года).

УНИВЕРСИТЕТ ИТМО

Университет ИТМО – ведущий вуз России в области информационных и фотонных технологий, один из немногих российских вузов, получивших в 2009 году статус национального исследовательского университета. С 2013 года Университет ИТМО – участник программы повышения конкурентоспособности российских университетов среди ведущих мировых научно-образовательных центров, известной как проект «5 в 100». Цель Университета ИТМО – становление исследовательского университета мирового уровня, предпринимательского по типу, ориентированного на интернационализацию всех направлений деятельности.

Оглавление

Введение

Курс лабораторных работ рассчитан на приобретение студентами практических навыков по программированию Android приложений и состоит из 6 лабораторных работ, охватывающих широкий спектр разработки. Каждая лабораторная работа содержит краткие теоретические сведения по теме и практические задания, которые студент должен выполнить индивидуально.

Целью дисциплины является подготовка студентов, квалификационного уровня бакалавр, которые имеют начальный уровень подготовки в области объектно-ориентированного программирования, базис специальных знаний в области информационных технологий, владеют основами проектирования, имеют начальные навыки программирования Java приложений.

Задачами дисциплины являются:

- Введение в программирование приложений под операционную систему Android. Знакомство с особенностями среды программирования, с основными элементами пользовательского интерфейса.
- Изучение инструментов и среды программирования, основ проектирования мобильных приложений.
- Изучение структуры Android-приложений, использовании при программировании внешних ресурсов.
- Приобретение навыков работы с файлами, базами данных, пользовательскими настройками, разделяемыми данными и использования межпрограммного взаимодействия.
- Исследование программных интерфейсов, обеспечивающих функции телефонии, SMS, Wi-Fi,
- Изучение способов создания фоновых служб, сигнализации и подключения механизма уведомлений.

В результате выполнения лабораторных работ студенты будут

знать:

- элементы объектно-ориентированного анализа и дизайна в Android приложениях;
- принципы работы с файловой системой;
- проектирование и создание баз данных.

уметь:

- программировать приложения на языке Java;
- использовать SDK Android;
- разрабатывать пользовательские интерфейсы.

Лабораторная работа №1. Создание activity и передача параметров между ними

Цель работы:

Знакомство с интерфейсом среды программирования. Изучение структуры проекта

Теоретические сведения.

Рассмотрите основные понятия Android проекта.

Структура проекта

- src—«исходный код» приложения (java-классы)
- assets —пустая директория. Может использоваться для сохранения raw-файлов.
- gen—хранилище генерируемых системных файлов. В частности, здесь располагается файл R.java, в котором хранятся идентификаторы всех ресурсов, создаваемых в проекте (строковые ресурсы и т.п.).
- libs—различные библиотеки, используемые приложением
- res—ресурсы проекта.
- AndroidManifest.xml—файл описания проекта (поддерживаемые версии SDK, версия приложения и т.п.)
- project.properties—файл, включающий настройки проекта, такие как build target.

Ресурсы проекта

- anim/ Содержит XML файлы, компилируемые в анимационные объекты.
- color/ Содержит XML файлы описывающие цвета.
- drawable/ Содержит растровые файлы (PNG, JPEG, orGIF), 9-Patch файлы, и XML файлы, описывающие Drawableshapes или Drawableobjects включающие множественные состояния(нормальное, нажатое, состояние фокуса).
- layout/ Содержит XML файлы описывающие макеты экрана
- menu/ Содержит XML файлы, определяющие меню приложения.
- raw/ Для хранения произвольных файлов.
- values/ Содержит XML файлы, компилируемые во множество видов ресурсов (strings.xmlи т.д).
- xml/ СодержитXML файлы конфигурирующие компоненты приложения. Например, XML файл определяющий экран настроек.

Пример простейшего файла AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
package="com.example.untitled"
android:versionCode="1"
```

```
android:versionName="1.0">
<uses-sdkandroid:minSdkVersion="19"/>
<application android:label="@string/app_name"
android:icon="@drawable/ic_launcher">
<activity android:name="MyActivity"
android:label="@string/app_name">
<intent-filter>
<action android:name="android.intent.action.MAIN"/>
<category android:name="android.intent.category.LAUNCHER"/>
</intent-filter>
</activity>
</application>
</manifest>
```

Для начала работы требуется:

- Java Development Kit
- Android Software Development Kit

Android Virtual Device Manager

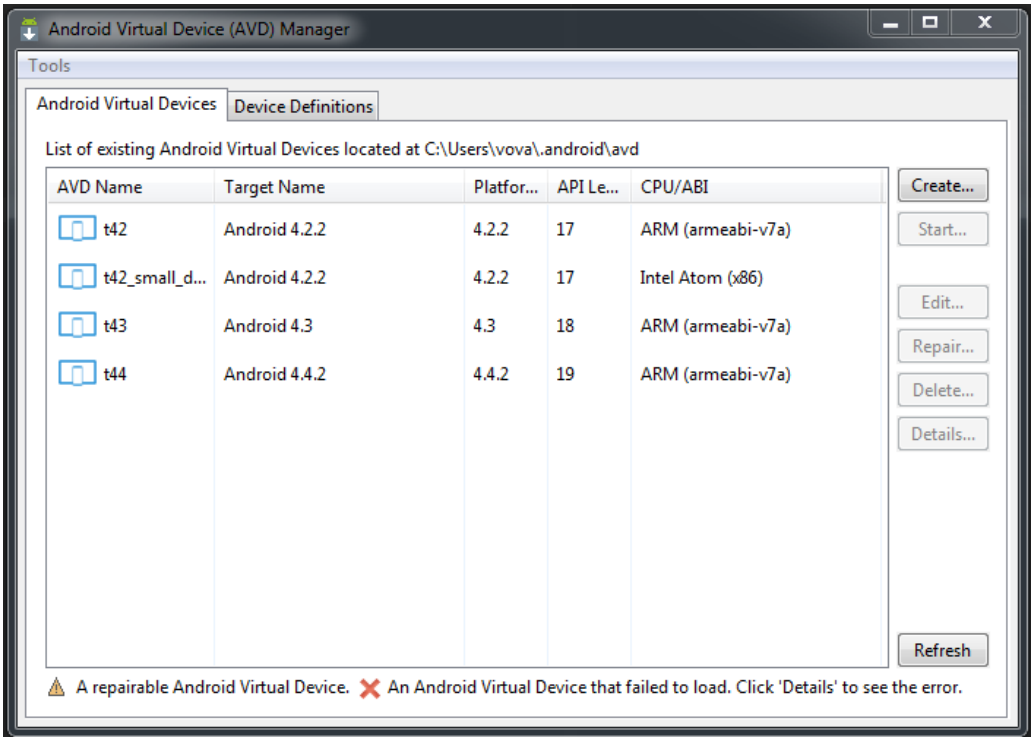

Рис.1.1. Интерфейс виртуального менеджера

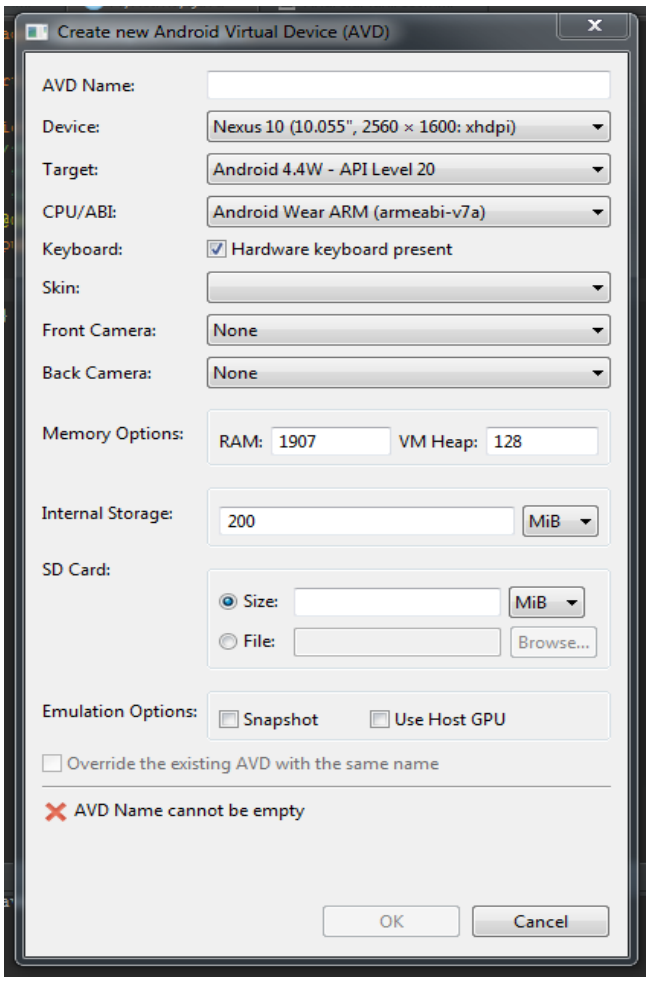

Рис. 1.2. Создание нового виртуального устройства

Задания лабораторной работы

Задание 1. Создание проекта приложения

Запустите среду программирования в IDE InelliJ Idea

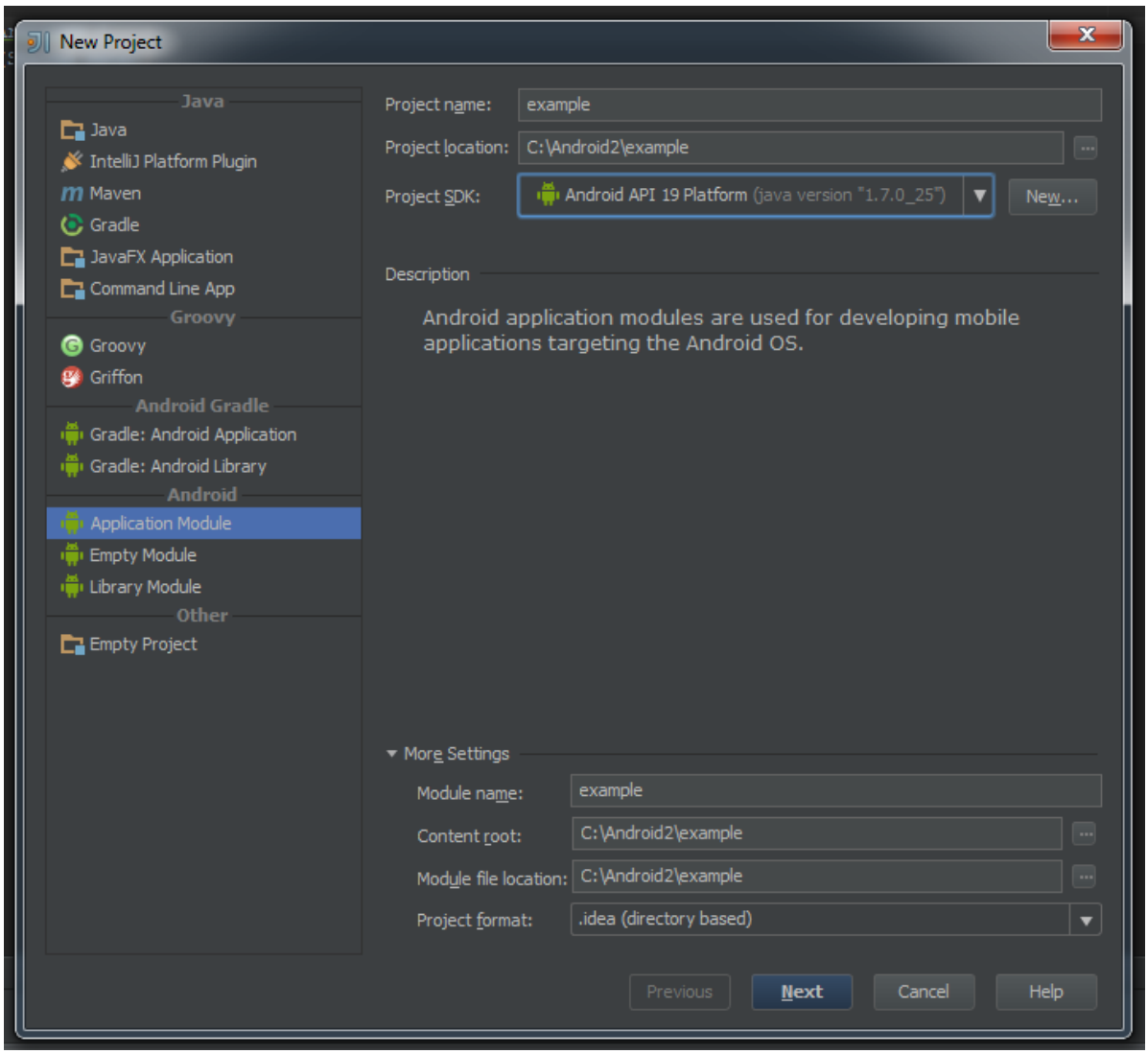

Рис. 1.3. Создание нового проекта в среде IntellyJ Idea

Введите данные проекта. Сохраните проект.

Задание 2. Создание приложений с одним экраном (Activity)

Необходимо создать два **activity** и организовать переход между ними.

Содержимое Activity 1

- кнопка с именем btn1

Содержимое Activity 2:

- TextView с текстом «Переданный параметр: *значение_параметра*».

значение_параметра – принятый параметр из Activity 1.

При запуске приложения пользователь должен попадать на экран с **Activity 1**. После нажатия на кнопку btn1 необходимо осуществить переход к **Activity 2** и передавать параметр из **Activity 1**. В качестве значения передаваемого параметра использовать свою фамилию.

Ключевые классы: Activity, Intent, Button, TextView.

Лабораторная работа №2. Основы верстки

Цель работы:

Изучить основы верстки. Научиться управлять интерфейсом мобильного устройства при разработке программного приложения.

Теоретические сведения

Просмотрите основные сведения о классах, которые понадобятся при разработке приложения.

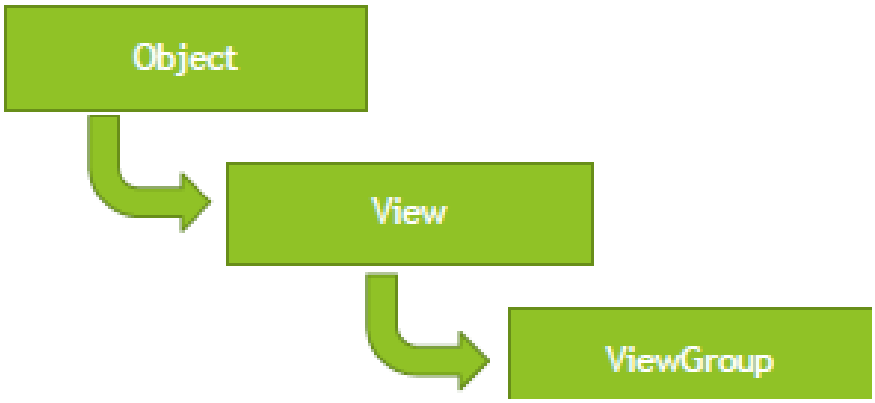

Рис.2.1. Иерархия классов View и ViewGroup

Дерево представлений для Activity представлено на рисунке 2.2.

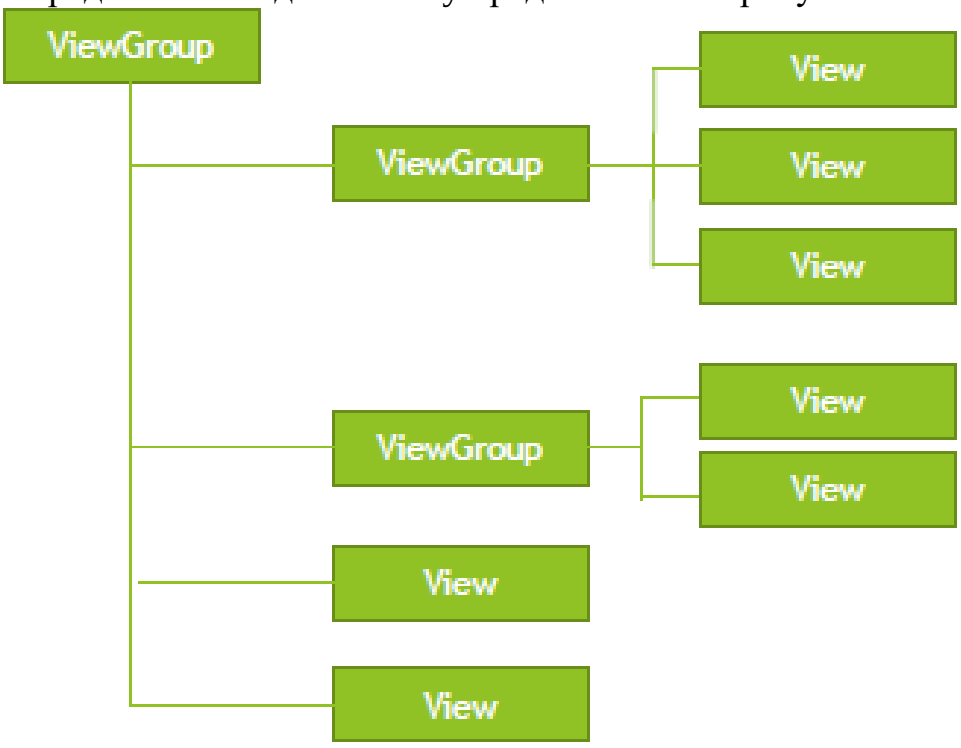

Рис.2.2. Дерево представлений для Activity

Файл разметки имеет следующую структуру

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
android:orientation="vertical"
android:layout_width="fill_parent"
android:layout_height="fill_parent"
```

```
>
<LinearLayout
android:layout_width="match_parent"
android:layout_height="0dp"
android:layout_weight="0.25"
android:padding="5dp">
<Button
android:layout_width="0dp"
android:layout_weight="0.33"
android:layout_height="wrap_content"
android:text="Button1"
android:id="@+id/button3" android:layout_gravity="right"/>
<Button
android:layout_width="0dp"
android:layout_weight="0.33"
android:layout_height="wrap_content"
android:text="Button2"
android:id="@+id/button"/>
<Button
android:layout_width="0dp"
android:layout_weight="0.33"
android:layout_height="wrap_content"
android:text="Button3"
android:id="@+id/button2" android:layout_gravity="center_horizontal"/>
</LinearLayout>
<LinearLayout
android:layout_width="match_parent"
android:layout_height="0dp"
android:paddingLeft="40dp"
android:paddingRight="40dp"
android:layout_weight="0.5"
android:gravity="center_vertical">
<Button
android:layout_width="0dp"
android:layout_weight="0.33"
android:layout_height="wrap_content"
android:text="Button4"
android:id="@+id/button3"/>
<Button
android:layout_width="0dp"
android:layout_weight="0.33"
```

```
android:layout_height="wrap_content"
android:text="Button5"
android:id="@+id/button"/>
</LinearLayout>
<LinearLayout
android:layout_width="match_parent"
android:layout_height="0dp"
android:layout_weight="0.25"
android:padding="5dp"
android:gravity="bottom">
…
</LinearLayout>
</LinearLayout>
```

Распространенные виды макетов

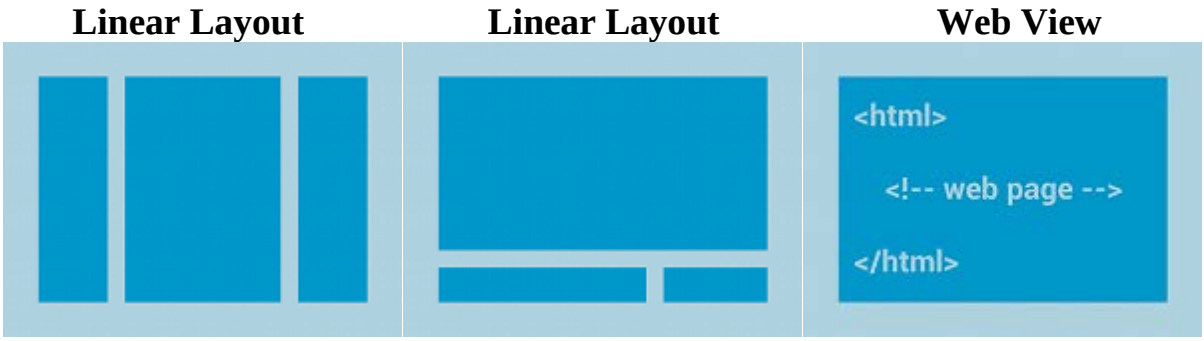

Рис. 2.3. Распространенные виды макетов

Макеты с адаптером

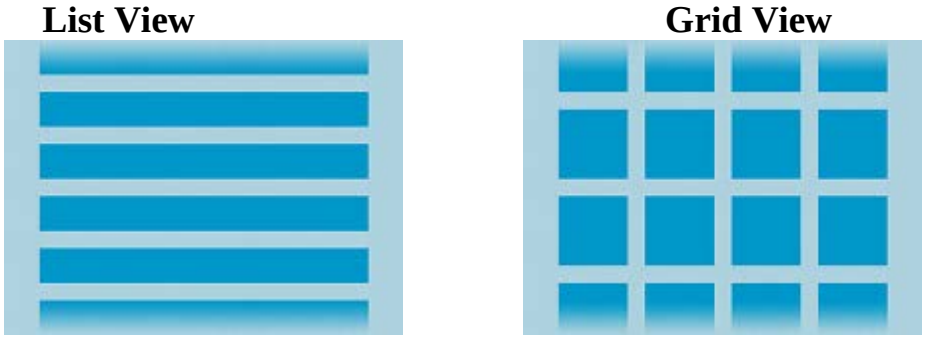

Рис.2.4. Интерфейс макетов с адаптером

Атрибуты LinearLayout

Attribute Name	*Related Method*	*Description*
android:baselineAligned	setBaselineAligned(boolean)	When set to false, prevents the layout from aligning its children's baselines.
android:baselineAlignedChildIndex	setBaselineAlignedChildIndex(int)	When a linear layout is part of another layout that is baseline aligned, it can specify which of its children to baseline align to (that is, which child TextView).
android:divider	setDividerDrawable(Drawable)	Drawable to use as a vertical divider between buttons.
android:gravity	setGravity(int)	Specifies how an object should position its content, on both the X and Y axes, within its own bounds.
android:measureWithLargestChild	setMeasureWithLargestChildEnabled(boolean)	When set to true, all children with a weight will be considered having the minimum size of the largest child.
android:orientation	setOrientation(int)	Should the layout be a column or a row? Use "horizontal" for a row, "vertical" for a column.
android:weightSum		Defines the maximum weight sum.

Задания лабораторной работы

Задание 1. Разработать мобильное приложение, состоящее из четырех activity.

После запуска приложения пользователь должен попадать на экран с activity1. На данном экране должно быть представлено меню, состоящее из четырех кнопок. Высота кнопок должна составлять 20% от высоты экрана. Расстояние между кнопками – 2%. Первая и последняя кнопка должны быть на равном расстоянии от краев экрана. Ширина кнопок 75%, выравнивание посередине.

После нажатия на первую кнопку пользователь должен переходить к activity2, его внешний вид представлен на рисунке 1. Верстка должна осуществляться с использованием LinearLayout, ширина кнопок должна задаваться в процентах от ширины экрана.

Рис. 2.5. Внешний вид экрана для первого задания

После нажатия на вторую кнопку в activity1 пользователь должен переходить к activity3, его внешний вид представлен на рисунке 2. Верстка должна осуществляться с использованием RelativeLayout (не использовать LinearLayout).

Рис. 2.6. Результат первого этапа выполнения задания

Третья кнопка в activity1 должна создавать activity3. Внешний вид activity3 представлен на рис. 2.7.

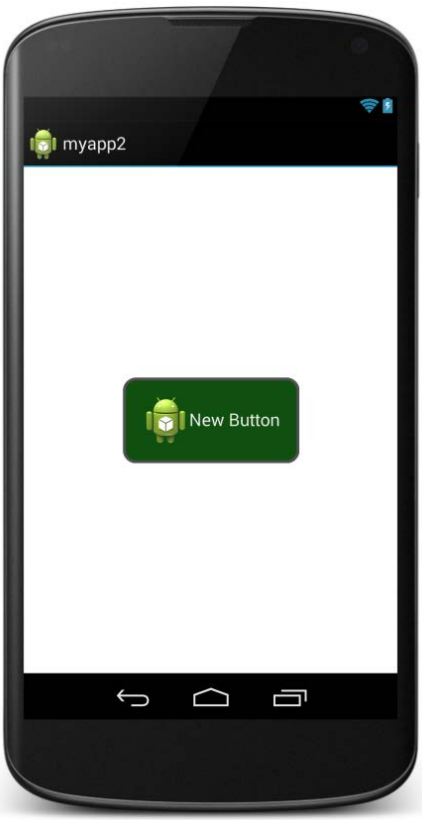

Рис.2.3. Интерфейс приложения на этапе activity3.

Кнопка должна быть выровнена по центру экрана. Цвет обводки кнопки #505050. Толщина обводки в соответствии с месяцем вашего рождения (от 1 до 12). Радиус скругления 24dp. Цвет фона экрана #FFFFFF. При нажатии на кнопку ее цвет должен изменяться на светло-зеленый.

Нажатие на четвертую кнопку в activity1 должно приводить к закрытию приложения.

Лабораторная работа №3. Работа с базой данных

Цель работы

Изучить работу Android приложения с базой данных.

Теоретические сведения.

SQLite

- SQLite—компактная встраиваемая реляционная база данных с открытым исходным кодом. Поддерживает динамическое типизированиеданных.
- Возможные типы полей: INTEGER, REAL, TEXT, BLOB.

SQLiteOpenHelper

Абстрактные методы

- onCreate( ) -вызывается при первом создании базы данных;
- onUpgrade( ) -вызывается при модификации базы данных

Реализация метода onCreate

```
public void onCreate( SQLiteDatabasedb) {
db.execSQL( "CREATE TABLE"+ TABLE_NAME
+ "(_id INTEGER PRIMARY KEYAUTOINCREMENT ,"
+ COL NAME + " TEXT ,"+ COL PHONE + " TEXT ) ; " ) ;
```

Реализация метода onUpgrade

```
@ Override
publicvoid onUpgrade( SQLiteDatabasedb, intoldVersion, intnewVersion)
{
db.execSQL( "DROP TABLE IF EXISTS"+ TABLE_NAME ) ;
onCreate( db) ;
}
```

Чтение из базы данных

```
Cursor query( String table, String [ ] columns,
String selection, String [ ] selectionArgs,
String groupBy, String having , String sortOrder)
```

Позиционирование курсора

- moveToFirst( ) -перемещает курсор в первую запись в выборке;
- moveToLast( ) -перемешает курсор в последнюю запись в выборке;
- moveToNext( ) -перемешает курсор в следующую запись и одновременноопределяет, существует ли эта запись. Метод moveToNext( ) возвращаеttrue, если курсор указывает на другую строку после перемещения, и false,если текущая запись была последней в выборке;
- moveToPrevious( ) -перемешает курсор в предыдущую запись;
- moveToPosition( ) -перемешает курсор в указанную позицию;
- getPosition( ) -возвращает текущий индекс позиции курсора.

- isFirst( ) ;
- isLast( ) ;
- isBeforeFirst( ) ;
- isAfterLast( ) .

Обновление и удаление записей

- intupdate(String table, ContentValuesvalues, String whereClause, String [ ] whereArgs)
- intdelete (String table, String whereClause, String [ ] whereArgs)

Задания на лабораторную работу

Задание 1. Необходимо создать приложение, взаимодействующее с базой данных. Первое активити должно содержать три кнопки. При нажатии на первую кнопку должно открываться новое активити, выводящее информацию из таблицы «Одногруппники» в удобном для восприятия формате.

При запуске приложения необходимо:

1. Создать БД, если ее не существует.
2. Создать таблицу «Одногруппники», содержащую поля:
    - ID;
    - ФИО;
    - Время добавления записи.
3. Удалять все записи из БД, а затем вносить 5 записей об одногруппниках.

При нажатии на вторую кнопку необходимо внести еще одну запись в таблицу.

При нажатии на третью кнопку необходимо заменить ФИО в последней внесенной записи на Иванов Иван Иванович.

Задание 2. Создать новое отдельное приложение на основе приложения, созданного в части 1.

Переопределить функцию onUpgrade. При изменении версии БД необходимо удалить таблицу «Одногруппники», создать таблицу «Одногруппники» содержащую следующие поля:

- ID;
- Фамилия;
- Имя;
- Отчество;
- Время добавления записи.

Изменить версию базы данных.

Лабораторная работа №4. Взаимодействие с сервером.

Цель работы:

Изучить работу с потоками. Научиться работать с мультимедиа файлами. Изучить работу с классом AsyncTask

Теоретические сведения

Главный и обычный потоки

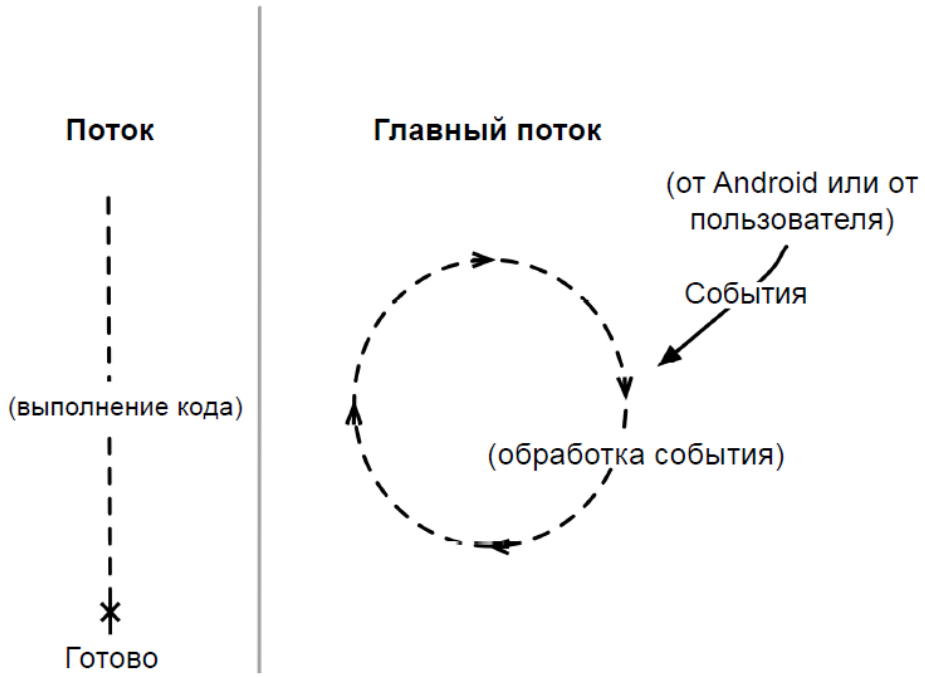

Рис.3.1. Графическое представление потоков

Асинхронные потоки в Android

AsyncTask-это специальный абстрактный класс, предоставляющий набор методов для реализации:

- onPreExecute-для размещения инициализирующего кода (UI поток)
- doInBackground-для размещения тяжелого кода, который будет выполняться в другом потоке
- onProgressUpdate-для информирования о прогрессе (UI поток)
- onPostExecute-для обработки результата, возвращенного doInBackground(UI поток) и вспомогательных методов
- isCancelled-для получения информации об отмене задачи
- publishProgress-для перевода сообщения о прогрессе в UI поток с последующим вызовом onProgressUpdate

Последовательность выполнения методов AsyncTask

onPreExecute

doInBackground

onPostExecute

publishProgress

onProgressUpdate

Правила использования AsyncTask:

Объект AsyncTaskдолжен быть создан в UI-потоке

- Метод execute должен быть вызван в UI-потоке
- Метод executeможет быть запущен только один раз
- Не вызывайте методы onPreExecute, doInBackground, onPostExecuteи onProgressUpdate

Передача данных в AsyncTask

Объявляем класс

```
Class MyAsyncTaskextends AsyncTask<String, Integer, Long>{
…
}
```

- Первый параметр используется методом doInBackground protectedLong doInBackground(String... urls)
- Второй параметр используется методом onProgressUpdate protectedvoidonProgressUpdate(Integer… progress)
- Третий параметр используется методом onPostExecute protectedvoidonPostExecute(Longresult)

Промежуточные данные

Последовательность действий для передачи промежуточных данных в основной поток программы:

- В методе doInBackgroundвызываем метод publishProgress
- В методе onProgressUpdate обрабатываем переданный в publishProgress параметр и выводим прогресс

Метод get

- Возвращает результат выполнения метода doInBackground
- Вызывается из UI потока

```
MyAsyncTaskat = newMyAsyncTask();
…
result= at.get();
```

Задание на лабораторную работу

Задание 1. Рассмотрите пример передачи данных.

```
MyAsyncTaskat = newMyAsyncTask();
at.execute(“url1”, “url2”);

doInBackground(String... urls)
```

Задание 2. Рассмотрите пример вывода промежуточных данных.

```
@Override
Protected void doInBackground(String... urls) {
        try{
                int cnt= 0;
                for(Stringurl: urls) {
                        // обрабатываем первый параметр
                                …
                        // выводим промежуточные результаты
```

```
                    cnt++;
                    publishProgress(cnt);
            }
            TimeUnit.SECONDS.sleep(1);
    } catch(InterruptedExceptione) {
            e.printStackTrace();
            }
            return null;
    }
    @Override
            protected void onProgressUpdate(Integer... values) {
                    super.onProgressUpdate(values);
                    tv.setText("обработано " + values[0] + " параметров");
    }
```

Задание 3. Проверьте пример создания простой асинхронной задачи

```
public class MainActivity extends Activity{
MyAsyncTask at;
TextView tv;

public void onCreate(Bundle savedInstance State) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        tv= (TextView) findViewById(R.id.tv);
        MyAsyncTask at = new MyAsyncTask();
        at.execute();
}
}
```

Задание 4. На основании изученных примеров разработать приложение сохраняющее статистику проигрываемых песен на радио Мегабайт. Для сохранения песни и названия необходимо создать базу данных, содержащую таблицу со следующими полями:

1) ID
2) Исполнитель
3) Название трека
4) Время внесения записи

При включении приложения необходимо производить проверку подключения к Интернету. В случае если подключение отсутствует – выводить всплывающее сообщение (Toast) с предупреждением о запуске в автономном режиме (доступен только просмотр внесенных ранее записей).

После включения приложение должно производить асинхронный опрос сервера с интервалом 20 секунд. Если название трека не совпадает с последней записью в таблице необходимо произвести запись в БД.

URL адрес, по которому можно получить информацию о текущем треке и исполнителе:

http://media.ifmo.ru/api_get_current_song.php

Формат возвращаемых данных – JSON.

В случае успешного выполнения запроса результат будет иметь вид:

{“result”: “success”, “info” : “*Исполнитель – Название трека*” }

В случае ошибки API вернет следующую строку:

{“result”: “error”, “info” : “*Информация об ошибке*” }

Для успешного взаимодействия с API при обращении к странице необходимо передавать логин (login) и пароль(password) как POST-параметры.

login**: 4707login**

password**: 4707pass**

Приложение должно содержать активити, позволяющее просматривать внесенные в базу данных записи.

Лабораторная работа №5. Хранение данных. Настройки и внешние файлы.

Цель работы

Изучить инструменты хранения данных, а также работу с внешними файлами.

Теоретические сведения

Варианты взаимодействия с сервером

1. Сокеты

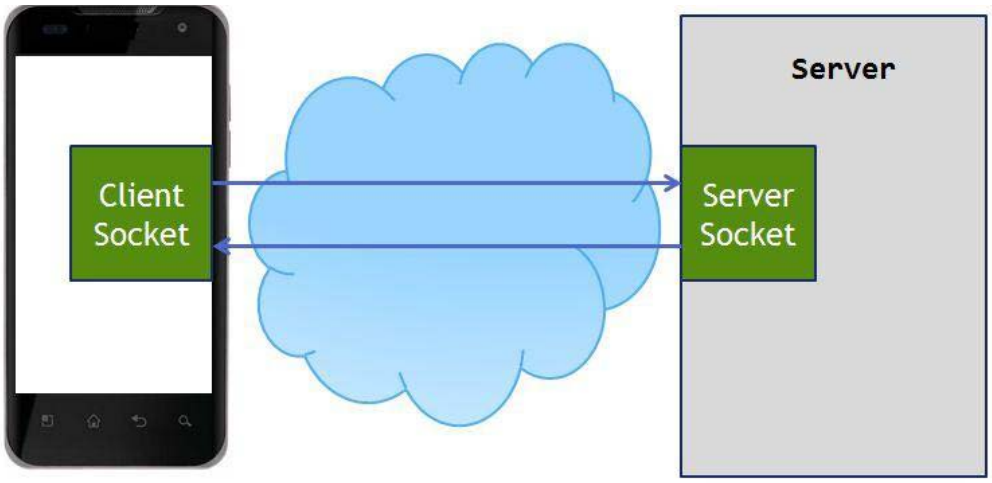

Рис. 5.1. Схема взаимодействия клиентского сокета с серверным

Используется в приложениях, где важна скорость доставки сообщения, важен порядок доставки сообщений и необходимо держать стабильное соединение с сервером. Такой способ зачастую реализуется в мессенджерах и играх.

2. Частые опросы

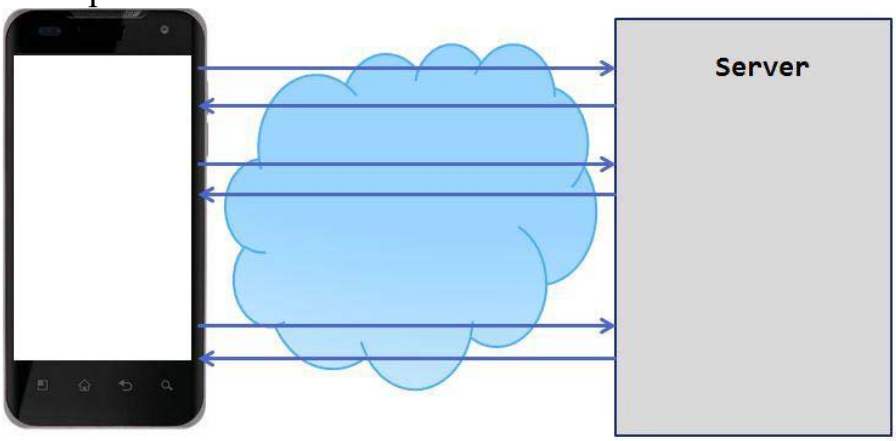

Рис. 5.2. Схема реализации частых опросов

Клиент посылает запрос на сервер и запрашивает новые данные. Сервер отвечает на запрос клиента и отдает все, что у него накопилось к этому моменту.

3. Длинные опросы

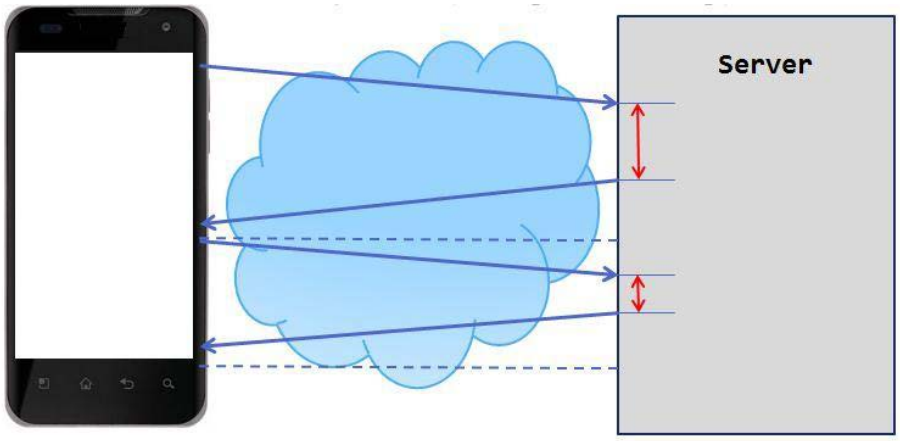

Рис. 5.3. Схема реализации длинных опросов

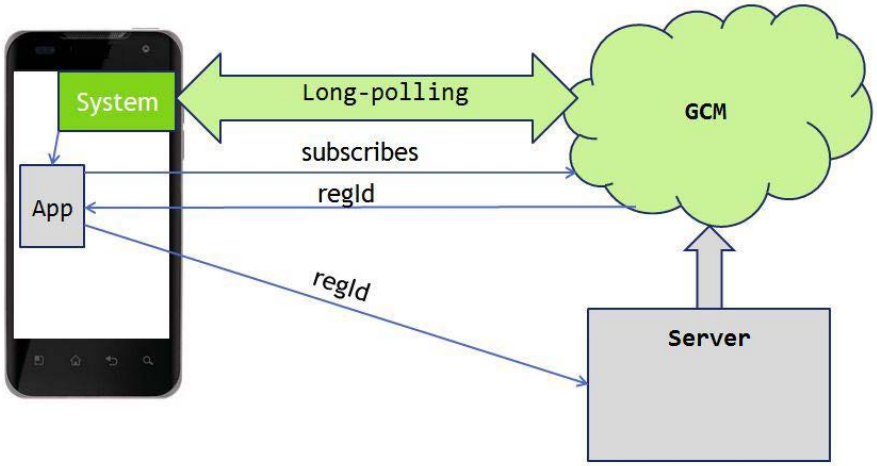

Рис. 5.4. Механизм реализации длинных опросов

Метод длинных опросов

- Реализация этого подхода достаточно сложна на мобильном клиенте в первую очередь из-за нестабильности мобильного соединения.
- При данном подходе расходуется меньше трафика, чем при обычном polling'е, т.к. сокращается количество установок соединений с сервером.
- Механизм long polling, или пуш-уведомлений (push notifications), реализован в самой платформе Android. (ваше приложение подписывается у сервиса Google Cloud Messaging (GCM) на получение пуш-уведомлений)

Библиотеки для работы с сетью

java.net	Содержит классы, связанные с сетевыми функциями, в том числе сокеты потоков и датаграмм, протокол IP, а также общие средства для работы с HTTP. Это многоцелевой ресурс для работы с сетями.
java.io	Пакет не относится непосредственно к сетям. Его классы используются сокетами и соединениями, содержащимися в других пакетах Java. Они используются также для обмена с локальными файлами (что часто происходит при взаимодействии с сетью).

java.nio	Содержит классы, которые служат буфером для определенных типов данных. Удобен для организации сетевой связи между двумя конечными точками средствами Java.
org.apache.*	Набор пакетов, которые обеспечивают точный контроль и функции для HTTP-коммуникаций на основе Apache - популярного веб-сервера с открытым исходным кодом.
android.net	Содержит дополнительные сокеты доступа к сети в дополнение к основным классам java.net.*. Этот пакет включает в себя класс URI, который часто используется в разработке приложений Android, не связанных с сетью.
android.net.http	Содержит классы для работы с сертификатами SSL.
android.net.wifi	Содержит классы для реализации всех аспектов WiFi (802.11 Wireless Ethernet) на платформе Android.

Разрешения

- android.permission.INTERNET
- android.permission.ACCESS_NETWORK_STATE

Задание на лабораторную работу

Задание 1. Изучите пример подключения к сети.

```
public void myClickHandler(View view) {
...
ConnectivityManager connMgr = (ConnectivityManager)
        getSystemService(Context.CONNECTIVITY_SERVICE);
 NetworkInfo networkInfo = connMgr.getActiveNetworkInfo();
if (networkInfo != null && networkInfo.isConnected()) {
        // fetch data
} else {
        // display error
        }
        ...
}
```

Задание 2. Изучите код приложений

HttpURLConnection

```
private String downloadUrl(String myurl) throws IOException {
        InputStream is = null;
        // Only display the first 500 characters of the retrieved
        // web page content.
        int len = 500;
        try {
                URL url = new URL(myurl);
                HttpURLConnection conn = (HttpURLConnection)
        url.openConnection();
```

```
            conn.setReadTimeout(10000 /* milliseconds */);
            conn.setConnectTimeout(15000 /* milliseconds */);
            conn.setRequestMethod("GET"); conn.setDoInput(true);
            // Starts the query conn.connect();
            int response = conn.getResponseCode();
            Log.d(DEBUG_TAG, "The response is: " + response); is =
            conn.getInputStream();
            // Convert the InputStream into a string
            String contentAsString = readIt(is, len);
            return contentAsString;
        // Makes sure that the InputStream is closed after the app is
        // finished using it.
        } finally {
            if (is != null) {
            is.close();
            }
        }
}
```

Преобразование полученной информации к типу Srting

```
public String readIt (InputStream stream, int len) throws IOException,
UnsupportedEncodingException {
        Reader reader = null;
        reader = new InputStreamReader(stream, "UTF-8");
        char[] buffer = new char[len];
        reader.read(buffer);
        return new String(buffer);
        }
```

Http GET запрос

- Создаем HttpClient
  `HttpClient client = new DefaultHttpClient();`
- Создаем объект HttpGet
  `HttpGet request = new HttpGet("http://www.example.com");`
- Выполняем HTTP запрос
  `HttpResponse response;`

```
try {
        response = client.execute(request);
        Log.d("Response of GET request", response.toString());
        } catch (ClientProtocolException e) {
            // TODO Auto-generated catch block
        e.printStackTrace();
        } catch (IOException e) {
```

```
            // TODO Auto-generated catch block
        e.printStackTrace();
        }
```

Взаимодействие с сервером через сокеты

```
public class Requester extends Thread {
        Socket requestSocket;
        String message;
        StringBuilder returnStringBuffer = new StringBuilder();
        Message lmsg;
        int ch;
        @Override public void run() {
                try {
                this.requestSocket = new Socket("remote.servername.com",
        13);
                InputStreamReader isr = new
        InputStreamReader(this.requestSocket. getInputStream(), "ISO-8859-
        1");
                while ((this.ch = isr.read()) != -1) {
                this.returnStringBuffer.append((char) this.ch);
                        }
                this.message = this.returnStringBuffer.toString();
                this.lmsg = new Message();
                this.lmsg.obj = this.message;
                this.lmsg.what = 0;
                h.sendMessage(this.lmsg);
                this.requestSocket.close();
        }
        catch (Exception ee) {
                Log.d("sample application", "failed to read data" +
        ee.getMessage());
                }
        }
}
```

Задание 3. Работа с внешними файлами.

Разработать мобильное приложение, позволяющее пользователю асинхронно скачивать файлы журнала Научно-технический вестник. Файлы хранятся на сервере в формате PDF и расположены по адресу:

http://ntv.ifmo.ru/file/journal/идентификатор_журнала.pdf

Не для всех ID имеются журналы, поэтому необходимо предусмотреть сообщение об отсутствии файла. В случае если файл не найден, ответ от сервера будет содержать главную страницу сайта.

Определить существует ли файл можно по возвращаемому сервером заголовку (параметр content-type).

Примеры ссылок:

http://ntv.ifmo.ru/file/journal/1.pdf – возвращен PDF файл

http://ntv.ifmo.ru/file/journal/2.pdf – файл не найден, возвращена главная страница сайта

Файлы должны храниться на устройстве в папке, создаваемой при первом запуске приложения (путь до папки и ее название определите самостоятельно).

После окончания загрузки файла должна становиться доступной кнопка «Смотреть» и кнопка «Удалить».

При нажатии на кнопку «Смотреть» должно происходить открытие сохраненного на устройстве файла. Предусмотреть ошибку, если на устройстве не установлено приложение, открывающее PDF файлы.

При нажатии на кнопку «Удалить» загруженный файл должен удаляться с устройства.

Рис.5.5. Интерфейс приложения

Задание 4. Хранение и чтение настроек.

При запуске приложения пользователю должно выводиться всплывающее полупрозрачное уведомление (popupWindow), с краткой инструкцией по использованию приложения (можете написать случайный текст), чекбоксом «Больше не показывать» и кнопкой «ОК».

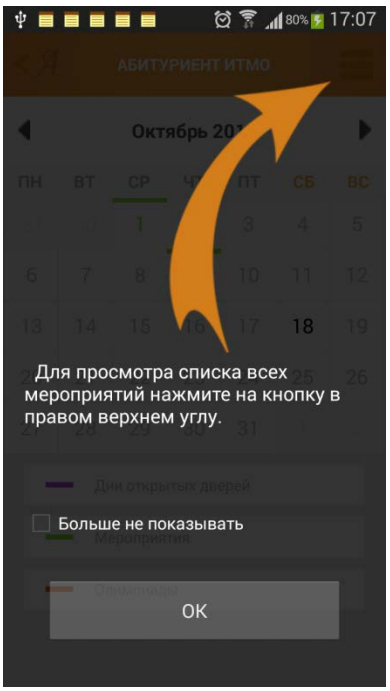

Рис.5.6. Результат работы приложения

Если чекбокс был отмечен и нажата кнопка ОК, необходимо произвести сохранение данного параметра используя SharedPreferences. При следующем запуске приложения производить проверку параметра, и не выводить всплывающее сообщение, если чекбокс был отмечен.

Лабораторная работа №6. Уведомления.

Цель работы

Изучить инструменты хранения данных, а также работу с внешними файлами.

Теоретические сведения

Варианты хранения данных на устройстве

- Preferences (аналог INI-файлам в ОС Windows)
- SQLite - база данных, таблицы
- обычные файлы - внутренние и внешние (на SD карте)

Preferences

Значения сохраняются в виде пары: имя, значение

- Для хранения используется файл формата XML
- Путь к сохраняемым настройкам /data/data/YOUR_PACKAGE_NAME/shared_prefs/YOUR_PREFS_NAME.xml

Отличие методов getPreferences и getSharedPreferences

```
public SharedPreferences getPreferences(int mode) {
        return getSharedPreferences(getLocalClassName(), mode);
 }
```

Метод getSharedPreferences

public abstract SharedPreferences getSharedPreferences (String name, int mode)

- Name – имя файла с настройками
- Mode – режим
    - MODE_PRIVATE – режим по умолчанию. Файл доступен из только из текущего приложения.
    - MODE_WORLD_READABLE (deprecated in API level 17) – разрешено чтение всеми приложениями.
    - MODE_WORLD_WRITEABLE - (deprecated in API level 17) – разрешена запись всеми приложениями.
    - MODE_MULTI_PROCESS – режим для взаимодействия с файлом из нескольких процессов

Создание экрана настроек

pref.xml

```
<?xml version="1.0" encoding="utf-8"?>
<PreferenceScreen
xmlns:android="http://schemas.android.com/apk/res/android">
        <CheckBoxPreference
                android:key="notif"
                android:summary="Enable notifications"
                android:title="Notifications">
        </CheckBoxPreference>
```

```
        <EditTextPreference
                android:key="address"
                android:summary="Address for notifications"
                android:title="Address">
        </EditTextPreference>
</PreferenceScreen>
```

Создание экрана настроек

PrefActivity.java

```
public class PrefActivity extends PreferenceActivity {
@Override
protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        addPreferencesFromResource(R.xml.pref);
        }
}
```

Создание экрана настроек

main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
         xmlns:android=http://schemas.android.com/apk/res/android
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:orientation="vertical">
        <TextView
                android:id="@+id/tvInfo"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="">
        </TextView>
</LinearLayout>
```

Создание экрана настроек

MainActivity.java

```
public class MainActivity extends Activity {
        TextView tvInfo;
        SharedPreferences sp;
        /** Called when the activity is first created. */
        public void onCreate(Bundle savedInstanceState) {
                super.onCreate(savedInstanceState);
                setContentView(R.layout.main);
                tvInfo = (TextView) findViewById(R.id.tvInfo);
                sp = PreferenceManager.getDefaultSharedPreferences(this);
```

```
        // полная очистка настроек
        // sp.edit().clear().commit();
    }
    protected void onResume() {
        Boolean notif = sp.getBoolean("notif", false);
        String address = sp.getString("address", "");
        String text = "Notifications are " + ((notif) ? "enabled, address = " +
address : "disabled");
        tvInfo.setText(text);
        super.onResume();
        }
        public boolean onCreateOptionsMenu(Menu menu) {
            MenuItem mi = menu.add(0, 1, 0, "Preferences");
        mi.setIntent(new Intent(this, PrefActivity.class));
            return super.onCreateOptionsMenu(menu);
        }
}
```

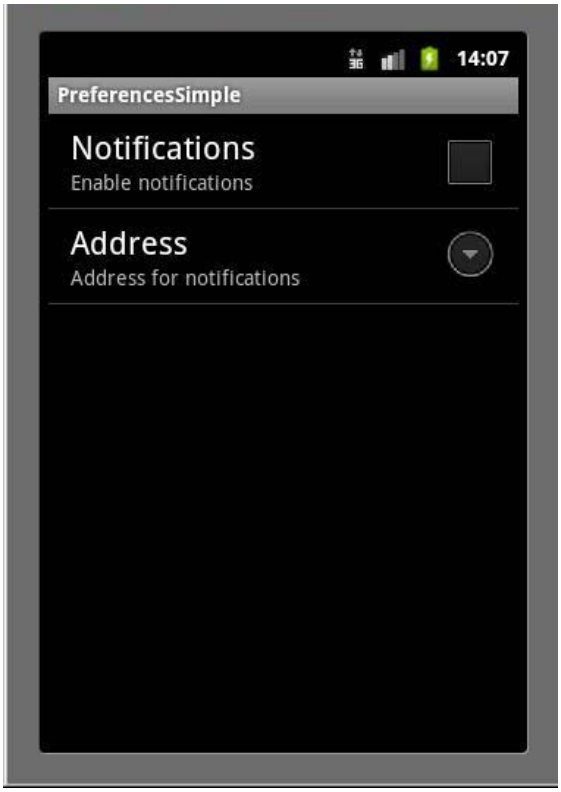

Рис.6.1. Интерфейс приложения

Работа с файлами

Основные используемые классы

- BufferedReader, BufferedWriter
- InputStreamReader, OutputStreamWriter
- openFileInput, openFileOutput\
- FileReader, FileWriter

- File

Пример

```
public class MainActivity extends Activity {
	final String LOG_TAG = "myLogs";
	final String FILENAME = "file";
	final String DIR_SD = "MyFiles";
	final String FILENAME_SD = "fileSD";
	/** Called when the activity is first created. */
	@Override
	public void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);
		setContentView(R.layout.main);
	}
	public void onclick(View v) {
		switch (v.getId()) {
		case R.id.btnWrite:
			writeFile();
			break;
		case R.id.btnRead:
			readFile();
			break;
		case R.id.btnWriteSD:
			writeFileSD();
			break;
		case R.id.btnReadSD:
			readFileSD();
			break;
	}
}
```

Запись файла в память устройства

```
void writeFile() {
try {
// отрываем поток для записи
BufferedWriter bw = new BufferedWriter(new OutputStreamWriter(
	openFileOutput(FILENAME, MODE_PRIVATE)));
	// пишем данные
	bw.write("Содержимое файла");
	// закрываем поток
	bw.close();
	Log.d(LOG_TAG, "Файл записан");
	} catch (FileNotFoundException e) {
	e.printStackTrace(); }
```

```
        catch (IOException e) {
        e.printStackTrace();
        }
}
```

Чтение файла из памяти устройства

```
void readFile() {
try {
// открываем поток для чтения BufferedReader
br = new BufferedReader(new InputStreamReader(
openFileInput(FILENAME)));
String str = "";
// читаем содержимое
while ((str = br.readLine()) != null) {
        Log.d(LOG_TAG, str);
}
} catch (FileNotFoundException e) {
        e.printStackTrace();
}
catch (IOException e) {
                e.printStackTrace();
        }
}
```

Запись на SD карту

```
void writeFileSD() {
        // проверяем доступность SD
        if (!Environment.getExternalStorageState().equals(
        Environment.MEDIA_MOUNTED)) {
        Log.d(LOG_TAG, "SD-карта не доступна: " +
Environment.getExternalStorageState());
        return;
        }
// получаем путь к SD
File sdPath = Environment.getExternalStorageDirectory();
// добавляем свой каталог к пути
sdPath = new File(sdPath.getAbsolutePath() + "/" + DIR_SD);
// создаем каталог sdPath.mkdirs();
// формируем объект File, который содержит путь к файлу
File sdFile = new File(sdPath, FILENAME_SD);
try {
        // открываем поток для записи
        BufferedWriter bw = new BufferedWriter(new FileWriter(sdFile));
```

```
            // пишем данные
            bw.write("Содержимое файла на SD");
            // закрываем поток
            bw.close();
            Log.d(LOG_TAG, "Файл записан на SD: " + sdFile.getAbsolutePath());
        }
            catch (IOException e) {
            e.printStackTrace();
        }
        }
```

Чтение с SD карты

```
        void readFileSD() {
        // проверяем доступность SD
        if (!Environment.getExternalStorageState().equals(
        Environment.MEDIA_MOUNTED)) {
        Log.d(LOG_TAG, "SD-карта не доступна: " +
Environment.getExternalStorageState());
        return;
        }
        // получаем путь к SD
        File sdPath = Environment.getExternalStorageDirectory();
        // добавляем свой каталог к пути
        sdPath = new File(sdPath.getAbsolutePath() + "/" + DIR_SD);
        // формируем объект File, который содержит путь к файлу
        File sdFile = new File(sdPath, FILENAME_SD); try {
        // открываем поток для чтения
        BufferedReader br = new BufferedReader(new FileReader(sdFile));
        String str = "";
        // читаем содержимое
        while ((str = br.readLine()) != null) {
        Log.d(LOG_TAG, str);
        }
        } catch (FileNotFoundException e) {
        e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        }
        }
```

Разрешение на работу с файлами на SD карте

- android.permission.READ_EXTERNAL_STORAGE
- android.permission.WRITE_EXTERNAL_STORAGE

Задание на лабораторную работу

Задание 1. Разработать мобильное приложение, позволяющее устанавливать напоминания.

Требования к приложению

1) Создание напоминаний и сохранение их в базу данных. В БД должны храниться следующие значения (Заголовок, Текст уведомления, дата уведомления)
2) Просмотр установленных уведомлений
3) Удаление уведомлений
4) Дата напоминания должна устанавливаться с помощью TimePickerDialog и DatePickerDialog
5) Стилизовать напоминание в Notification Center и Status bar (установить собственный лого)
6) При нажатии на уведомление в Notification Center переходить в активити приложения с полным текстом уведомления.

Классы для создания приложения: Notification, NotificationManager, PendingIntent, BroadcastReceiver, AlarmManager

Рекомендуемая литература:

1. Голощапов А. Google Android. Программирование для мобильных устройств (+ CD-ROM); БХВ-Петербург - Москва, 2011. - 448 с.
2. Голощапов Алексей Google Android. Программирование для мобильных устройств; БХВ-Петербург - Москва, 2012. - 448 с.
3. Дэрси Лорен , Кондер Шейн Android за 24 часа. Программирование приложений под операционную систему Google; Рид Групп - Москва, 2011. - 464 с.
4. Колисниченко Денис Программирование для Android. Самоучитель; БХВ-Петербург - Москва, 2012. - 272 с.
5. Майер Рето Android 2. Программирование приложений для планшетных компьютеров и смартфонов; Эксмо - Москва, 2011. - 672 с.
6. Майер Рето Android 4. Программирование приложений для планшетных компьютеров и смартфонов; Эксмо - Москва, 2013. - 816 с.
7. Медникс Зигард , Дорнин Лайрд , Мик Блэйк , Накамура Масуми Программирование под Android; Питер - Москва, 2013. - 560 с.
8. Немцова Т. И., Голова С. Ю., Абрамова И. В. Программирование на языке высокого уровня. Программирование на языке Object Pascal (+ CD-ROM); Форум, Инфра-М - Москва, 2009. - 496 с.
9. Ретабоуил Сильвен Android NDK. Разработка приложений под Android на C/C++; ДМК Пресс - Москва, 2012. - 496 с.
10. Цехнер Марио Программирование игр под Android; Питер - Москва, 2012. - 688 с