[Основы работы с SQLite в Андроид](http://vlad8.com/tech/android-sqlite-database/" \o "Permanent Link to Основы работы с SQLite в Андроид)

SQLite — это база данных с открытыми исходными кодами, включаемая по умолчанию в состав Андроид. SQLite поддерживает стандартные возможности реляционных баз данных – синтаксис, транзакции и *prepared statements*. Кроме SQLite требует очень небольшого количества памяти для работы (примерно 250 кб).

Использование SQLite в Андроид не требует установки БД или администрирования. Вы указываете SQL-запрос для работы с БД и необходимые операции администрирования выполняются автоматически. Работа с базами данных в Андроид может быть медленной из-за операций ввода/вывода, поэтому все необходимые операции рекомендуется выполнять с помощью класса AsyncTask (т.е. в фоне).

SQLite поддерживает типы данных **TEXT** (схожий с String в Java), **INTEGER** (схожий с long в Java) и **REAL** (схожий с double в Java). Все остальные типы данных должны быть сконвертированы в один из этих перед сохранением в базу данных. SQLite сам по себе не проверяет, соответствуют ли записанные данные типу данных соответствующего столбца, вы можете записать целое число в столбец с типом Integer.

Если ваше приложение создает базу данных, она сохраняется в папке «**DATA/data/APP\_NAME/databases/FILENAME**». DATA  — это путь, возвращаемый методом Environment.getDataDirectory(), APP\_NAME — имя вашего приложения и FILENAME — это имя, которое вы даете базе данных при создании. Environment.getDataDirectory() обычно возвращает SD-карту в качестве места.

База данных SQLite доступна только приложению, которое создает ее. Если вы хотите дать доступ к данным другим приложениям, вы можете использовать ContentProvider.

**SQLiteOpenHelper**

Для работы с БД в Андроид-приложении обычно создается класс-потомок SQLightOpenHelper. В этом классе следует переопределить методы onCreate() для создания БД и onUpgrade() для обновления БД в случае изменений в схеме базы данных. Оба метода работают с объектом SQLiteDatabase.

SQLiteOpenHelper предоставляет методы *getReadableDatabase*() и *getWriteableDatabase*() для доступа к объекту SQLiteDatabase, который позволяет читать и писать в БД.

Для первичного ключа БД всегда необходимо использовать идентификатор «\_id», т.к. некоторые функции Android следуют этому стандарту.

**SQLiteDatabase и курсор**

Класс SQLiteDatabase предоставляет методы *insert*(), *update*(), *delete*() и *execSQL*() (который дает возможность напрямую выполнять SQL-запросы). Объект ContentValues позволяет определить ключи/значения для вставки и обновления данных.

Запросы могут быть созданы либо через метод *rawQuery*(), принимающий SQL, либо метод *query*(), предоставляющий интерфейс для указания динамических данных SQLiteQueryBuilder. SQLiteQueryBuilder схож с интерфейсом провайдера данных (сontent provider), поэтому обычно он используется для работы с провайдерами данных.

Запрос к БД всегда возвращает «курсор», представляющий собой результат запроса. Чтобы получить число элементов в запросе, мы можем воспользоваться методом *getCount().*Для перемещения между записями в результате используются методы *moveToFirst() и moveToNext().*Метод*isAfterLast()*позволяет проверить, есть ли еще данные в результате.

Метод выполнения запроса *query()*имеет параметры:

|  |  |
| --- | --- |
| 1 | String dbName, int[] columnNames, String whereClause, String[] valuesForWhereClause, String[] groupBy, String[] having, String[] orderBy |

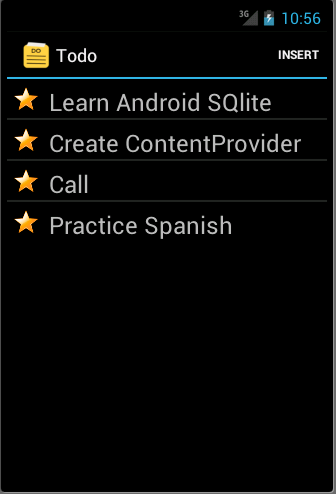
Как видите, они соответствуют синтаксису SQL-запросов. Если вам нужно выбрать все данные, используйте **null**в условии where (null можно указать вместо любого из параметров запроса, например groupBy, orderBy и т.п.). Вообще условие where указывается без самого слова where, например: «\_id = 19 and summary = ?»

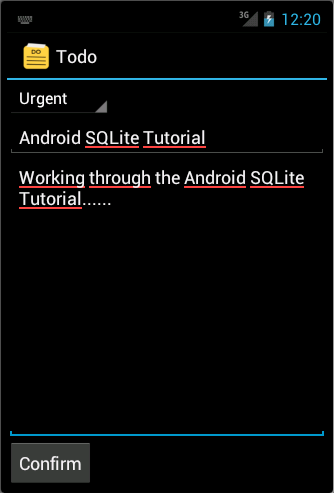
Возвращаемый курсор может быть использован напрямую с помощью SimpleCursorAdapter в ListViews.

**Делаем приложение «Список дел»**

Мы создадим приложение «Список дел», которое позволяет пользователю вести список дел, которые нужно выполнить. Эти дела будут сохраняться в БД SQLite. Приложение будет состоять из двух активити, одно для просмотра всего списка дел и второе для создания конкретного дела. Связь между активити будет поддерживаться с помощью интентов (см. туториал про [интенты](http://vlad8.com/tech/2011/07/android-intents-tutorial/)).

Результат наших усилий будет выглядеть вот так:





**Создаем проект**

Создайте проект «de.vogella.android.todos» с активити «TodosOverview» и добавьте еще одно активити«TodoDetails». Затем создайте пакет «de.vogella.android.todos.database», в котором будут содержаться классы для работы с базой данных.

**Работа с БД**

Начнем с создания следующего класса «TodoDatabaseHelper»

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37 | package de.vogella.android.todos.database;    import android.content.Context;  import android.database.sqlite.SQLiteDatabase;  import android.database.sqlite.SQLiteOpenHelper;  import android.util.Log;    public class TodoDatabaseHelper extends SQLiteOpenHelper {      private static final String DATABASE\_NAME = "applicationdata";        private static final int DATABASE\_VERSION = 1;        // запрос на создание базы данных      private static final String DATABASE\_CREATE = "create table todo (\_id integer primary key autoincrement, "              + "category text not null, summary text not null, description text not null);";        public TodoDatabaseHelper(Context context) {          super(context, DATABASE\_NAME, null, DATABASE\_VERSION);      }        // метод вызывается при создании базы данных      @Override      public void onCreate(SQLiteDatabase database) {          database.execSQL(DATABASE\_CREATE);      }        // метод вызывается при обновлении базы данных, например, когда вы увеличиваете номер версии базы данных      @Override      public void onUpgrade(SQLiteDatabase database, int oldVersion,              int newVersion) {          Log.w(TodoDatabaseHelper.class.getName(),                  "Upgrading database from version " + oldVersion + " to "                          + newVersion + ", which will destroy all old data");          database.execSQL("DROP TABLE IF EXISTS todo");          onCreate(database);      }  } |

На основе этого вспомогательного класса мы можем написать класс «TodoDbAdapter», с помощью которого уже выполнять запросы, создавать и обновлять данные элементов списка дел. Метод *open()*подключит базу данных с помощью вспомогательного класса. Для обновления и создания значений мы будем использовать класс «android.content.ContentValues». Он позволяет хранить данные в виде «ключ/значение». Вы используете названия столбцов как ключи в ContentValues и передаете объект для вставки или обновления значений в БД.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97 | package de.vogella.android.todos.database;    import android.content.ContentValues;  import android.content.Context;  import android.database.Cursor;  import android.database.SQLException;  import android.database.sqlite.SQLiteDatabase;    public class TodoDbAdapter {        // поля базы данных      public static final String KEY\_ROWID = "\_id";      public static final String KEY\_CATEGORY = "category";      public static final String KEY\_SUMMARY = "summary";      public static final String KEY\_DESCRIPTION = "description";      private static final String DATABASE\_TABLE = "todo";      private Context context;      private SQLiteDatabase database;      private TodoDatabaseHelper dbHelper;        public TodoDbAdapter(Context context) {          this.context = context;      }        public TodoDbAdapter open() throws SQLException {          dbHelper = new TodoDatabaseHelper(context);          database = dbHelper.getWritableDatabase();          return this;      }        public void close() {          dbHelper.close();      }        /\*\*       \* создать новый элемент списка дел. если создан успешно - возвращается номер строки rowId       \* иначе -1       \*/      public long createTodo(String category, String summary, String description) {          ContentValues initialValues = createContentValues(category, summary,                  description);            return database.insert(DATABASE\_TABLE, null, initialValues);      }        /\*\*       \* обновить список       \*/      public boolean updateTodo(long rowId, String category, String summary,              String description) {          ContentValues updateValues = createContentValues(category, summary,                  description);            return database.update(DATABASE\_TABLE, updateValues, KEY\_ROWID + "="                  + rowId, null) > 0;      }        /\*\*       \* удаляет элемент списка       \*/      public boolean deleteTodo(long rowId) {          return database.delete(DATABASE\_TABLE, KEY\_ROWID + "=" + rowId, null) > 0;      }        /\*\*       \* возвращает курсор со всеми элементами списка дел       \*       \* @return курсор с результатами всех записей       \*/      public Cursor fetchAllTodos() {          return database.query(DATABASE\_TABLE, new String[] { KEY\_ROWID,                  KEY\_CATEGORY, KEY\_SUMMARY, KEY\_DESCRIPTION }, null, null, null,                  null, null);      }        /\*\*       \* возвращает курсор, спозиционированный на указанной записи       \*/      public Cursor fetchTodo(long rowId) throws SQLException {          Cursor mCursor = database.query(true, DATABASE\_TABLE, new String[] {                  KEY\_ROWID, KEY\_CATEGORY, KEY\_SUMMARY, KEY\_DESCRIPTION },                  KEY\_ROWID + "=" + rowId, null, null, null, null, null);          if (mCursor != null) {              mCursor.moveToFirst();          }          return mCursor;      }        private ContentValues createContentValues(String category, String summary,              String description) {          ContentValues values = new ContentValues();          values.put(KEY\_CATEGORY, category);          values.put(KEY\_SUMMARY, summary);          values.put(KEY\_DESCRIPTION, description);          return values;      }  } |

**Ресурсы**

Добавим несколько ресурсов, которые мы будет использовать позже, в том числе listmenu.xml (в папке menu), который будет использоваться позже для создания новых дел в списке.

|  |  |
| --- | --- |
| 1  2  3  4  5 | <?xml version="1.0" encoding="utf-8"?>  <menu    xmlns:android="http://schemas.android.com/apk/res/android">  <item android:id="@+id/insert" android:title="Insert"></item>  </menu> |

Для списка приоритетов создадим массив строк. Добавьте такой файл priority.xml в папке /res/values.

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | <?xml version="1.0" encoding="utf-8"?>  <resources>  <string-array name="priorities"><item>Urgent</item>  <item>Reminder</item>  </string-array>  </resources> |

Мы также создадим еще один ресурс для строк в нашем приложении – файл strings.xml в /res/values.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 | <?xml version="1.0" encoding="utf-8"?>  <resources>      <string name="hello">Hello World, Todo!</string>      <string name="app\_name">Todo</string>      <string name="no\_todos">Currently there are no Todo items maintained</string>      <string name="menu\_insert">Add Item</string>      <string name="menu\_delete">Delete Todo</string>      <string name="todo\_summary">Summary</string>      <string name="todo\_description">Delete Todo</string>      <string name="todo\_edit\_summary">Summary</string>      <string name="todo\_edit\_description">Description</string>      <string name="todo\_edit\_confirm">Confirm</string>      <color name="listcolor">#FFE87C</color>      <color name="black">#000000</color>  </resources> |

**Активити**

Мы определили три лэйаута — один для списка, один для строк списка и один для редактирования отдельного дела в списке. При этом лэйаут списка указывает на иконку. Вы можете найти иконки, например, здесь: <http://iconlet.com/>

Создайте файл лэйаута todo\_list.xml, он будет определять вид списка дел.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10 | <?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" android:background="@color/listcolor">      <ListView android:id="@android:id/list" android:layout\_width="wrap\_content"          android:layout\_height="wrap\_content" ></ListView>      <TextView android:id="@android:id/empty" android:layout\_width="wrap\_content"          android:layout\_height="wrap\_content" android:text="@string/no\_todos" />    </LinearLayout> |

Вставьте иконку “reminder” в папку /res/layout/, которая будет использоваться в следущем лэйауте (или же вы можете просто убрать упоминание об иконке из этого лэйаута), после этого создайте файл todo\_row.xml, который будет использоваться для отображения отдельной строки.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11 | <?xml version="1.0" encoding="utf-8"?>  <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"      android:layout\_height="wrap\_content" android:layout\_width="fill\_parent">      <ImageView android:id="@+id/icon" android:src="@drawable/reminder"          android:layout\_marginLeft="4px" android:layout\_marginRight="8px"          android:layout\_height="40px" android:layout\_marginTop="8px"          android:layout\_width="30px">      </ImageView>      <TextView android:text="@+id/TextView01" android:layout\_height="wrap\_content" android:id="@+id/label"          android:textSize="40px" android:layout\_marginTop="6px" android:layout\_width="fill\_parent" android:textColor="@color/black"></TextView>  </LinearLayout> |

Создатим файл лэйаута «todo\_edit». Этот лэйаут будет использован дальше для отображения и изменения отдельного элемента списка дел.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23 | <?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" android:background="@color/listcolor">        <Spinner android:id="@+id/category" android:layout\_width="wrap\_content"          android:layout\_height="wrap\_content" android:entries="@array/priorities"></Spinner>      <LinearLayout android:id="@+id/LinearLayout01"          android:layout\_height="wrap\_content" android:layout\_width="fill\_parent">          <EditText android:layout\_height="wrap\_content" android:id="@+id/todo\_edit\_summary"              android:layout\_weight="1" android:layout\_width="wrap\_content"              android:hint="Summary"></EditText>        </LinearLayout>        <EditText android:layout\_width="fill\_parent"          android:layout\_height="fill\_parent" android:layout\_weight="1"          android:id="@+id/todo\_edit\_description" android:hint="Description" android:gravity="top"></EditText>      <Button android:layout\_width="wrap\_content"          android:layout\_height="wrap\_content" android:id="@+id/todo\_edit\_button"          android:text="@string/todo\_edit\_confirm"></Button>    </LinearLayout> |

Наконец, поменяем код наших активити на следующий ниже. Сперва «TodoOverview.java»

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99  100  101  102  103  104  105  106  107  108  109  110  111  112  113  114  115  116  117  118  119  120  121  122  123  124  125  126  127  128 | package de.vogella.android.todos;    import android.app.ListActivity;  import android.content.Intent;  import android.database.Cursor;  import android.os.Bundle;  import android.view.ContextMenu;  import android.view.ContextMenu.ContextMenuInfo;  import android.view.Menu;  import android.view.MenuInflater;  import android.view.MenuItem;  import android.view.View;  import android.widget.AdapterView.AdapterContextMenuInfo;  import android.widget.ListView;  import android.widget.SimpleCursorAdapter;  import de.vogella.android.todos.database.TodoDbAdapter;    public class TodosOverview extends ListActivity {      private TodoDbAdapter dbHelper;      private static final int ACTIVITY\_CREATE = 0;      private static final int ACTIVITY\_EDIT = 1;      private static final int DELETE\_ID = Menu.FIRST + 1;      private Cursor cursor;        /\*\* Called when the activity is first created. \*/      @Override      public void onCreate(Bundle savedInstanceState) {          super.onCreate(savedInstanceState);          setContentView(R.layout.todo\_list);          this.getListView().setDividerHeight(2);          dbHelper = new TodoDbAdapter(this);          dbHelper.open();          fillData();          registerForContextMenu(getListView());      }        // Создаем меню, основанное на XML-файле      @Override      public boolean onCreateOptionsMenu(Menu menu) {          MenuInflater inflater = getMenuInflater();          inflater.inflate(R.menu.listmenu, menu);          return true;      }        // Реакция на выбор меню      @Override      public boolean onMenuItemSelected(int featureId, MenuItem item) {          switch (item.getItemId()) {          case R.id.insert:              createTodo();              return true;          }          return super.onMenuItemSelected(featureId, item);      }        @Override      public boolean onOptionsItemSelected(MenuItem item) {          switch (item.getItemId()) {          case R.id.insert:              createTodo();              return true;          }          return super.onOptionsItemSelected(item);      }        @Override      public boolean onContextItemSelected(MenuItem item) {          switch (item.getItemId()) {          case DELETE\_ID:              AdapterContextMenuInfo info = (AdapterContextMenuInfo) item                      .getMenuInfo();              dbHelper.deleteTodo(info.id);              fillData();              return true;          }          return super.onContextItemSelected(item);      }        private void createTodo() {          Intent i = new Intent(this, TodoDetails.class);          startActivityForResult(i, ACTIVITY\_CREATE);      }        @Override      protected void onListItemClick(ListView l, View v, int position, long id) {          super.onListItemClick(l, v, position, id);          Intent i = new Intent(this, TodoDetails.class);          i.putExtra(TodoDbAdapter.KEY\_ROWID, id);          // активити вернет результат если будет вызвано с помощью этого метода          startActivityForResult(i, ACTIVITY\_EDIT);      }        @Override      protected void onActivityResult(int requestCode, int resultCode,              Intent intent) {          super.onActivityResult(requestCode, resultCode, intent);          fillData();        }        private void fillData() {          cursor = dbHelper.fetchAllTodos();          startManagingCursor(cursor);            String[] from = new String[] { TodoDbAdapter.KEY\_SUMMARY };          int[] to = new int[] { R.id.label };            // Теперь создадим адаптер массива и установим его для отображения наших данных          SimpleCursorAdapter notes = new SimpleCursorAdapter(this,                  R.layout.todo\_row, cursor, from, to);          setListAdapter(notes);      }        @Override      public void onCreateContextMenu(ContextMenu menu, View v,              ContextMenuInfo menuInfo) {          super.onCreateContextMenu(menu, v, menuInfo);          menu.add(0, DELETE\_ID, 0, R.string.menu\_delete);      }        @Override      protected void onDestroy() {          super.onDestroy();          if (dbHelper != null) {              dbHelper.close();          }      }  } |

И затем «TodoDetails.java»

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99  100  101  102  103 | package de.vogella.android.todos;    import android.app.Activity;  import android.database.Cursor;  import android.os.Bundle;  import android.util.Log;  import android.view.View;  import android.widget.Button;  import android.widget.EditText;  import android.widget.Spinner;  import de.vogella.android.todos.database.TodoDbAdapter;    public class TodoDetails extends Activity {      private EditText mTitleText;      private EditText mBodyText;      private Long mRowId;      private TodoDbAdapter mDbHelper;      private Spinner mCategory;        @Override      protected void onCreate(Bundle bundle) {          super.onCreate(bundle);          mDbHelper = new TodoDbAdapter(this);          mDbHelper.open();          setContentView(R.layout.todo\_edit);          mCategory = (Spinner) findViewById(R.id.category);          mTitleText = (EditText) findViewById(R.id.todo\_edit\_summary);          mBodyText = (EditText) findViewById(R.id.todo\_edit\_description);            Button confirmButton = (Button) findViewById(R.id.todo\_edit\_button);          mRowId = null;          Bundle extras = getIntent().getExtras();          mRowId = (bundle == null) ? null : (Long) bundle                  .getSerializable(TodoDbAdapter.KEY\_ROWID);          if (extras != null) {              mRowId = extras.getLong(TodoDbAdapter.KEY\_ROWID);          }          populateFields();          confirmButton.setOnClickListener(new View.OnClickListener() {              public void onClick(View view) {                  setResult(RESULT\_OK);                  finish();              }            });      }        private void populateFields() {          if (mRowId != null) {              Cursor todo = mDbHelper.fetchTodo(mRowId);              startManagingCursor(todo);              String category = todo.getString(todo                      .getColumnIndexOrThrow(TodoDbAdapter.KEY\_CATEGORY));                for (int i=0; i<mCategory.getCount();i++){                    String s = (String) mCategory.getItemAtPosition(i);                  Log.e(null, s +" " + category);                  if (s.equalsIgnoreCase(category)){                      mCategory.setSelection(i);                  }              }                mTitleText.setText(todo.getString(todo                      .getColumnIndexOrThrow(TodoDbAdapter.KEY\_SUMMARY)));              mBodyText.setText(todo.getString(todo                      .getColumnIndexOrThrow(TodoDbAdapter.KEY\_DESCRIPTION)));          }      }        protected void onSaveInstanceState(Bundle outState) {          super.onSaveInstanceState(outState);          saveState();          outState.putSerializable(TodoDbAdapter.KEY\_ROWID, mRowId);      }        @Override      protected void onPause() {          super.onPause();          saveState();      }        @Override      protected void onResume() {          super.onResume();          populateFields();      }        private void saveState() {          String category = (String) mCategory.getSelectedItem();          String summary = mTitleText.getText().toString();          String description = mBodyText.getText().toString();            if (mRowId == null) {              long id = mDbHelper.createTodo(category, summary, description);              if (id > 0) {                  mRowId = id;              }          } else {              mDbHelper.updateTodo(mRowId, category, summary, description);          }      }  } |

Окончательный AndroidManifest.xml выглядит вот так:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21 | <?xml version="1.0" encoding="utf-8"?>  <manifest xmlns:android="http://schemas.android.com/apk/res/android"      package="de.vogella.android.todos" android:versionCode="1"      android:versionName="1.0">      <application android:icon="@drawable/todo" android:label="@string/app\_name">          <activity android:name=".TodosOverview" android:label="@string/app\_name">              <intent-filter>                  <action android:name="android.intent.action.MAIN" />                  <category android:name="android.intent.category.LAUNCHER" />              </intent-filter>          </activity>            <activity android:name=".TodoDetails"              android:windowSoftInputMode="stateVisible|adjustResize">              >        </activity>      </application>      <uses-sdk android:minSdkVersion="9" />    </manifest> |

**Запускаем приложение**

Запустите ваше приложение: вы должны увидеть список дел и иметь возможность создавать новые дела через меню. Существующий элемент списка также можно удалить, нажав на него.