**Простейшая база данных Часть четвёртая**

Продолжаем работать с базой данных. Будем переделывать адаптер **ArrayAdapter** в **CursorAdapter**.

В предыдущем примере использовался **ArrayAdapter**, работающий с массивом. Для небольшой базы данных это не критично. Но если приложение работает с очень большим объемом информации, её чтение из базы данных займет некоторое время. Кроме того, для хранения массива потребуется много памяти.

Поэтому для извлечения данных используются другие адаптеры, использующие курсоры. Общим предком для таких адаптеров является класс **CursorAdapter**. Главное преимущество курсорных адаптеров - объект **CursorAdapter** получает курсор при создании, но обращается к курсору за данными только тогда, когда потребуется. Например, для списка не нужно выводить все элементы, а только часть из них, которые будут видны на экране. При прокрутке новая порция данных будет снова запрашиваться, а ненужная информация уберётся из памяти.

Первое, что необходимо учитывать при создании курсора для адаптера, — какие столбцы должен содержать курсор. В курсор следует включить все столбцы, которые должны отображаться в списке, вместе со столбцом **\_id**. Столбец **\_id** должен быть включен в курсор, иначе адаптер курсора работать не будет. Это требование Android. При использовании адаптера курсора этот столбец используется для определения строки, на которой щелкнул пользователь.

Если мы хотим выводить только имена котов, то курсор для адаптера будет выглядеть следующим образом.

cursor = db.query("CATS", new String[]{"\_id", "NAME"},

null, null, null, null, null);

Давайте выведем в список имена всех котов через новый адаптер. Старый код взят из прошлого урока.

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

mCatsListView = (ListView) findViewById(R.id.listViewCats);

mDatabaseHelper = new DatabaseHelper(this);

mSqLiteDatabase = mDatabaseHelper.getWritableDatabase();

Cursor cursor = mSqLiteDatabase.query("CATS", new String[]{"\_id", mDatabaseHelper.COLUMN\_CAT\_NAME},

null, null, null, null, null);

CursorAdapter listAdapter = new SimpleCursorAdapter(this,

android.R.layout.simple\_list\_item\_1,

cursor,

new String[]{mDatabaseHelper.COLUMN\_CAT\_NAME},

new int[]{android.R.id.text1},

0);

mCatsListView.setAdapter(listAdapter);

}

При создании адаптера мы указали, как должны выводиться данные, какой курсор следует использовать и какие столбцы должны связываться с теми или иными компонентами.

Так как данные подгружаются динамически, то в этом коде не нужно закрывать курсор. Закрывайте его вместе базой данных в методе **onDestroy()**

@Override

protected void onDestroy() {

super.onDestroy();

mCursor.close(); // поменять видимость объекта

mSqLiteDatabase.close();

}

Соответственно, если мы хотим выводить имена котов и их телефонные номера, то код будет следующим.

mCursor = mSqLiteDatabase.query("CATS", new String[]{"\_id", mDatabaseHelper.COLUMN\_CAT\_NAME,

mDatabaseHelper.COLUMN\_PHONE},

null, null, null, null, null);

CursorAdapter listAdapter = new SimpleCursorAdapter(this,

R.layout.list\_item,

mCursor,

new String[]{mDatabaseHelper.COLUMN\_CAT\_NAME, mDatabaseHelper.COLUMN\_PHONE},

new int[]{R.id.textViewName, R.id.textViewPhone},

0);

mCatsListView.setAdapter(listAdapter);

В адаптере мы увеличили массивы на один элемент и использовали собственную разметку из двух **TextView**.

Также мы можем добавить в массив ещё один элемент, который будет указывать на путь к картинке и связать его с **ImageView**.

Примеры расчитаны на беглое знакомство и не учитывают подводные камни. В других примерах раздела вы можете увидеть более оптимизированный код с учётом потоков, загрузчиков и т.д.