**Список дел с использованием CursorLoader**

Переделаем приложение [«Список дел»](http://developer.alexanderklimov.ru/android/sqlite/todo.php), избавившись от устаревших методов. В новых версиях Android появился специальный класс **CursorLoader**, который имеет ряд преимуществ по сравнению со старыми способами. Самый главный плюс - работа с курсорами происходит в асинхронном режиме и вам не нужно размещать работу с базой данных в AsyncTask, который я не использовал в старом примере, так как база была небольшой и для учебного урока можно было обойтись малой кровью. По умолчанию данный класс расчитан на работу с контент-провайдерами. Поэтому мы напишем свой класс, наследующий от **CursorLoader**, который может работать с базой данных напрямую.

Разметка для активностей и элементов списка осталась прежней. Можете взять код из прошлого примера и скопировать в новый проект.

Класс **ToDoDatabase** также оставляем без изменений. Там стандартные операции CRUD.

Изменения коснутся в классах активностей. В активности **MainActivity** нужно реализовать интерфейс **implements LoaderCallbacks<Cursor>**. А также создать новый класс-наследник**MyCursorLoader** и вызывать соответствующие методы. На данный момент я не вникал глубоко в работу нового класса **CursorLoader** и поэтому не готов объяснить, как его надо вообще использовать. Но, используя различные источники в сети, я написал свой код, который вполне работоспособен и не использует устаревшие методы. Не знаю, насколько правильно написан пример. Может позже я вернусь к нему, набравшись опыта.

package ru.alexanderklimov.todo;

import android.os.Bundle;

import android.app.ListActivity;

import android.app.LoaderManager.LoaderCallbacks;

import android.content.Context;

import android.content.CursorLoader;

import android.content.Intent;

import android.content.Loader;

import android.database.Cursor;

import android.view.ContextMenu;

import android.view.ContextMenu.ContextMenuInfo;

import android.view.Menu;

import android.view.MenuItem;

import android.view.View;

import android.widget.AdapterView.AdapterContextMenuInfo;

import android.widget.ListView;

import android.widget.SimpleCursorAdapter;

public class MainActivity extends ListActivity implements

LoaderCallbacks<Cursor> {

private ToDoDatabase 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 SimpleCursorAdapter adapter;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

this.getListView().setDividerHeight(2);

registerForContextMenu(getListView());

dbHelper = new ToDoDatabase(this);

fillData();

}

@Override

protected void onResume() {

// TODO Auto-generated method stub

super.onResume();

// Первоначальный способ. Тоже работает

// Cursor cursor = dbHelper.getAllTodos();

// adapter.changeCursor(cursor);

getLoaderManager().getLoader(0).forceLoad();

}

@Override

public boolean onCreateOptionsMenu(Menu menu) {

// Inflate the menu; this adds items to the action bar if it is present.

getMenuInflater().inflate(R.menu.main, menu);

return true;

}

// Реакция на выбор меню

@Override

public boolean onMenuItemSelected(int featureId, MenuItem item) {

switch (item.getItemId()) {

case R.id.insert:

createNewTask();

return true;

}

return super.onMenuItemSelected(featureId, item);

}

@Override

public boolean onOptionsItemSelected(MenuItem item) {

switch (item.getItemId()) {

case R.id.insert:

createNewTask();

return true;

}

return super.onOptionsItemSelected(item);

}

@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

public boolean onContextItemSelected(MenuItem item) {

switch (item.getItemId()) {

case DELETE\_ID:

// получаем из пункта контекстного меню данные по пункту списка

AdapterContextMenuInfo info = (AdapterContextMenuInfo) item

.getMenuInfo();

// извлекаем id записи и удаляем соответствующую запись в БД

dbHelper.deleteTodo(info.id);

// получаем новый курсор с данными

getLoaderManager().getLoader(0).forceLoad();

// fillData();

return true;

}

return super.onContextItemSelected(item);

}

private void createNewTask() {

Intent intent = new Intent(this, EditActivity.class);

startActivityForResult(intent, ACTIVITY\_CREATE);

}

private void fillData() {

String[] from = new String[] { ToDoDatabase.COLUMN\_SUMMARY };

int[] to = new int[] { R.id.label };

adapter = new SimpleCursorAdapter(this, R.layout.list\_row, null, from,

to, 0);

setListAdapter(adapter);

// создаем лоадер для чтения данных

getLoaderManager().initLoader(0, null, this);

// adapter.notifyDataSetInvalidated();

}

@Override

protected void onListItemClick(ListView l, View v, int position, long id) {

super.onListItemClick(l, v, position, id);

Intent intent = new Intent(this, EditActivity.class);

intent.putExtra(ToDoDatabase.COLUMN\_ID, id);

// активность вернет результат если будет вызван с помощью этого метода

startActivityForResult(intent, ACTIVITY\_EDIT);

}

@Override

protected void onActivityResult(int requestCode, int resultCode,

Intent intent) {

super.onActivityResult(requestCode, resultCode, intent);

if (resultCode == RESULT\_OK) {

fillData();

}

}

@Override

protected void onDestroy() {

super.onDestroy();

dbHelper.close();

}

@Override

public Loader<Cursor> onCreateLoader(int id, Bundle args) {

// TODO Auto-generated method stub

return new MyCursorLoader(this, dbHelper);

}

@Override

public void onLoadFinished(Loader<Cursor> loader, Cursor cursor) {

// TODO Auto-generated method stub

adapter.swapCursor(cursor);

}

@Override

public void onLoaderReset(Loader<Cursor> loader) {

// TODO Auto-generated method stub

adapter.swapCursor(null);

}

static class MyCursorLoader extends CursorLoader {

ToDoDatabase db;

public MyCursorLoader(Context context, ToDoDatabase db) {

super(context);

this.db = db;

}

@Override

public Cursor loadInBackground() {

Cursor cursor = db.getAllTodos();

return cursor;

}

}

}

В классе **EditActivity** вроде изменений не делал, но на всякий случай скопирую сюда.

package ru.alexanderklimov.todo;

import android.os.Bundle;

import android.app.Activity;

import android.database.Cursor;

import android.text.TextUtils;

import android.util.Log;

import android.view.View;

import android.widget.Button;

import android.widget.EditText;

import android.widget.Spinner;

import android.widget.Toast;

public class EditActivity extends Activity {

private EditText mTitleText;

private EditText mBodyText;

private Long mRowId;

private ToDoDatabase mDbHelper;

private Spinner mCategory;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

mDbHelper = new ToDoDatabase(this);

setContentView(R.layout.activity\_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 = (savedInstanceState == null) ? null

: (Long) savedInstanceState

.getSerializable(ToDoDatabase.COLUMN\_ID);

if (extras != null) {

mRowId = extras.getLong(ToDoDatabase.COLUMN\_ID);

}

populateFields();

confirmButton.setOnClickListener(new View.OnClickListener() {

public void onClick(View view) {

if (TextUtils.isEmpty(mTitleText.getText().toString())) {

Toast.makeText(EditActivity.this, "Данные не введены",

Toast.LENGTH\_LONG).show();

} else {

saveState();

setResult(RESULT\_OK);

finish();

}

}

});

}

private void populateFields() {

if (mRowId != null) {

Cursor todo = mDbHelper.getTodo(mRowId);

String category = todo.getString(todo

.getColumnIndexOrThrow(ToDoDatabase.COLUMN\_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(ToDoDatabase.COLUMN\_SUMMARY)));

mBodyText.setText(todo.getString(todo

.getColumnIndexOrThrow(ToDoDatabase.COLUMN\_DESCRIPTION)));

todo.close();

}

}

@Override

protected void onSaveInstanceState(Bundle outState) {

super.onSaveInstanceState(outState);

// saveState();

// outState.putSerializable(ToDoDatabase.COLUMN\_ID, 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 (description.length() == 0 && summary.length() == 0) {

return;

}

if (mRowId == null) {

long id = mDbHelper.createNewTodo(category, summary, description);

if (id > 0) {

mRowId = id;

}

} else {

mDbHelper.updateTodo(mRowId, category, summary, description);

}

}

}

Чисто визуально ничего не изменилось.