09

Базы данных

Nikolay Moskvin
moskvin@sibext.com

14 ноября 2013 г.

Содержание

Холст и Карандаш

- **Canvas** – это холст, на нем можно рисовать
- **Paint** – это карандаш им можно рисовать
- **View.onDraw** – метод, где на готовый холст можно наносить рисунки
- **View.invalidate** – метод, который заставит view заново вызвать onDraw

Холст

Основные методы рисования (У всех последним параметром необходимо указывать ссылку на **Paint**)

- **drawCircle** – круг по координатам центра и радиусу
- **drawLine** – линия по двум точкам
- **drawRect** – четырехугольник по двум точкам
- **drawText** – произвольный текст в указанную координату
- **drawPath** – используя термины moveTo и lineTo можно нарисовать траекторию
- **drawBitmap** – рисуем bitmap в указанную координату

Где можно использовать?

- Доступен в onDraw у любой View.
- Возможность создавать свой из Bitmap.

```
Bitmap b = Bitmap.createBitmap(100, 100, Bitmap.Config.ARGB_8888);
Canvas c = new Canvas(b);
```

Карандаш

Основные методы карандаша

- **setAntiAlias** – включаем/выключаем сглаженность (по умолчанию выкл)
- **setColor** – устанавливаем цвет
- **setStyle** – указываем нужно ли заливать фигуру или нет
- **setAlpha** – устанавливаем прозрачность
- **setStrokeWidth** – устанавливаем толщину карандаша
- **measureText** – можно узнать сколько указанный текст займет в ширину
- **ascent** – можно узнать сколько буква из текста займет места в высоту

Внимание

Необходимо помнить, что **onDraw** метод должен отрабатывать как можно быстрее, по этому в этом методе не допустимо создавать новые экземпляры классов. Экземпляр класса **Paint** необходимо готовить заранее в конструкторе View!

Карандаш

Основные методы карандаша

- **setAntiAlias** – включаем/выключаем сглаженность (по умолчанию выкл)
- **setColor** – устанавливаем цвет
- **setStyle** – указываем нужно ли заливать фигуру или нет
- **setAlpha** – устанавливаем прозрачность
- **setStrokeWidth** – устанавливаем толщину карандаша
- **measureText** – можно узнать сколько указанный текст займет в ширину
- **ascent** – можно узнать сколько буква из текста займет места в высоту

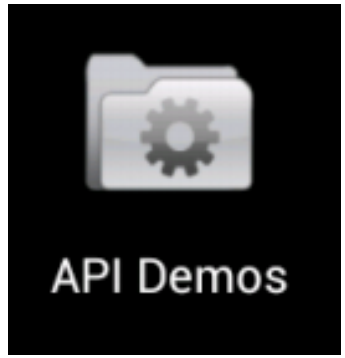
Внимание

Необходимо помнить, что **onDraw** метод должен отрабатывать как можно быстрее, по этому в этом методе не допустимо создавать новые экземпляры классов. Экземпляр класса **Paint** необходимо готовить заранее в конструкторе View!

ApiDemos

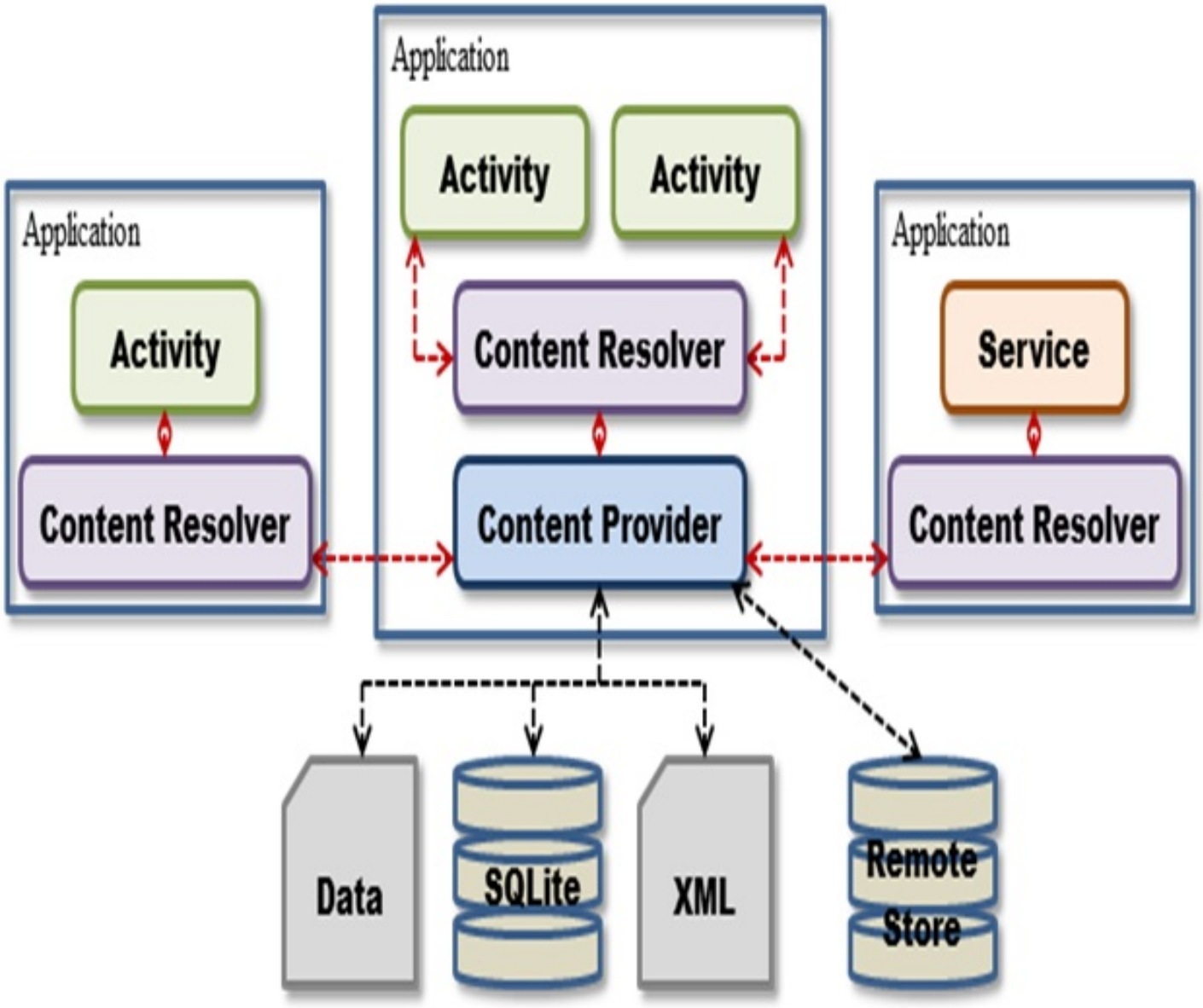
sibext company

Eclipse -> File -> New -> Other -> Android Sample Project -> Android 4.4 -> “**legacy > ApiDemos**”



- Graphics
  - AlphaBitmap, CreateBitmap
  - Arcs
  - Clipping
  - PolyToPoly (com.example.android.apis.graphics.PolyToPoly)
  - ...

Структура приложения



Способы хранения данных

sibext company

- Network connection
- Shared Preferences
- Internal Storage
- External Storage
- SQLite databases

Способы хранения данных

sibext company

- Network connection
- Shared Preferences
- Internal Storage
- External Storage
- SQLite databases

Способы хранения данных

sibext company

- Network connection
- Shared Preferences
- Internal Storage
- External Storage
- SQLite databases

Способы хранения данных

sibext company

- Network connection
- Shared Preferences
- Internal Storage
- External Storage
- SQLite databases

Способы хранения данных

sibext company

- Network connection
- Shared Preferences
- Internal Storage
- External Storage
- **SQLite databases**

Причины необходимости БД

sibext company

- Хранить данные необходимо во внешней памяти
- Данные – сложная структура данных
- Объем данных недетерминирован
- Слияния и объединения данных
- Сортировка данных
- Выборка данных по параметрам
- . . .

SQLite

- Открытый исходный код на C
- Встраивается в приложение
- Легковесная БД с основным функционалом
- `http://sqlite.org`

ContentValues

sibext company

- Набор значений для **ContentResolver** или **SQLiteOpenHelper**
- Доступ по ключ-значению

```
ContentValues cvFirst = new ContentValues();
cvFirst.put("name", "Dmitry");
ContentValues cv = new ContentValues();
cv.put("person_id", 1);
cv.putAll(cvFirst);
cv.containtsKey("person_id");
cv.get("person_id");
cv.remove("person_id");
```

Курсор

http://developer.android.com/reference/android/database/sqlite/SQLiteCursor.html

- Форма для предоставления результатов данных из БД
- Чтение текущего значения и переход к следующей строке выборки

```
c = db.query(TABLE_TASK,
      new String[]{TASK_ID, COLOR, BODY},
      null, null, null, null, null);

 if (c.moveToFirst()) {
     do {
        task = Task.populateByCursor(c);
     } while (operationCursor.moveToNext());
 }
class Task {
...
  public static Task populateByCursor(Cursor cursor) {
    Task task = new Task();
    task.taskId = cursor.getInt(cursor.getColumnIndex(TASK_ID));
    task.color = cursor.getString(cursor.getColumnIndex(COLOR));
    task.dueDate = cursor.getString(cursor.getColumnIndex(DUE_DATE));
    task.title = cursor.getString(cursor.getColumnIndex(TITLE));
    task.body = cursor.getString(cursor.getColumnIndex(BODY));
    task.mainWorker = cursor.getString(cursor.getColumnIndex(MAIN_WORKER));
    return task;
  }
}
```

Помощник

http://developer.android.com/reference/android/database/sqlite/SQLiteOpenHelper.html

- Позволяет создать и следить за версией базы данных

```
public class DatabaseHelper extends SQLiteOpenHelper implements BaseColumns {
  public DatabaseHelper(Context context) {
            super(context, DB_NAME, null, DB_VERSION);
            this.context = context;
  }
  @Override
  public void onCreate(SQLiteDatabase db) {
...
    db.execSQL("CREATE TABLE " + TableStatistic.TABLE_NAME + " ( "
            + TableStatistic.TABLE_FIELD_ID + " INTEGER, "
            + TableStatistic.TABLE_FIELD_TASK_TYPE + " INTEGER, "
            + TableStatistic.TABLE_FIELD_TASK_COMPLEX + " INTEGER, "
            + TableStatistic.TABLE_FIELD_COUNT_OF_CORRECT_ANSWERS
            + " INTEGER, " + TableStatistic.TABLE_FIELD_ATTEMPTS_COUNT
            + " INTEGER, FOREIGN KEY (" + TableStatistic.TABLE_FIELD_ID
            + ") REFERENCES " + TableUsers.TABLE_NAME + "("
            + TableUsers.TABLE_FIELD_ID + "));");

...
  }

  @Override
  public void onUpgrade(SQLiteDatabase db, int old, int newVersion) {
...
    db.execSQL("DROP TABLE IF EXISTS " + TableStatistic.TABLE_NAME);
...
    onCreate(db);
  }
}
```

Провайдер

http://developer.android.com/reference/android/content/ContentProvider.html

- Предоставляем доступ через ContentResolver

```
public class DatabaseProvider extends ContentProvider {
 static {
  URI_MATCHER = new UriMatcher(UriMatcher.NO_MATCH);
  URI_MATCHER.addURI(Base.URI, TableStatistic.TABLE_NAME, STATISTIC);
 }
 public static final Uri CONTENT_URI = Uri.parse("content://" + URI);
 private SQLiteDatabase db;
  @Override
  public synchronized Cursor query(Uri uri,
      String[] projection, String selection,
      String[] selectionArgs, String sortOrder) {
    db = (new DatabaseHelper(getContext())).getReadableDatabase();
    switch (URI_MATCHER.match(uri)) {
    case STATISTIC: {
      Cursor c = db.rawQuery("SELECT "
              + TableStatistic.TABLE_FIELD_TASK_TYPE + ", "
              + TableStatistic.TABLE_FIELD_TASK_COMPLEX + " from "
              + TableStatistic.TABLE_NAME + " where "
              + selection, null);
      c.setNotificationUri(getContext().getContentResolver(), uri);
      return c;
    }}
    db.close();
    return null;
  }}
```

Использование без провайдера

- используем помощника из компонента

```
databasehelper helper = new databasehelper(this);
sqlitedatabase db = helper.getreadabledatabase();
...
db.close();
...
db = helper.getwritabledatabase();
...
db.close();
```

- помощник позаботиться об открытии именно вашей бд, указываем на чтение мы хотим открыть или на запись

Основные операции

- **insert** – добавляем новые стоки в таблицу
- **update** – обновление стоки в таблице
- **delete** – удаление стоки из таблице
- **query/rawQuery** – выборка данных из таблиц
- **execSQL** – выполнение произвольного запроса
- `http://developer.android.com/reference/android/database/sqlite/SQLiteDatabase.html`

Транзакции

Необходимость в том, чтобы выполнялись все изменения в операции, если хоть одно из них не выполняется, то операция не применяется.

- Обычная транзакция

```
db.beginTransaction();
try {
  ...
  db.setTransactionSuccessful();
} finally { db.endTransaction(); }
```

- Транзакция со слушателем

```
db.beginTransactionWithListener(new SQLiteTransactionListener() {
public void onBegin() {}
bublic void onCommit() {}
public void onRollback() {}
});
try {
  ...
  db.setTransactionSuccessful();
} finally { db.endTransaction();
```

http://developer.android.com/reference/android/database/sqlite/SQLiteTransactionListener.html

SQLite клиент

- **sqlite3** – консольный инструмент для формирования Raw запросов к БД
- **Разные версии SQlite** – приводит к тому, что одни возможности поддерживаются в одном API, но не поддерживаются на другом!
- **Создание БД** – создается во внутренней памяти устройства
- **Права на БД** – имеет только то приложение, которое его создало.

Shared Preferences

sibext company

- Простое хранилище примитивных типов
- Доступ к данным через ключ-значение
- Получить доступ можно двумя способами:
  - **getSharedPreferences()** – используйте этот метод если вам нужно несколько файлов для хранения.
  - **getPreferences()** – используйте этот метод если вам нужен только один файл для одной Activity
- Если нужно сохранять Пользовательские настройки, то можно воспользоваться – **PreferenceActivity**

Редактор для preferences

Для редактирования значений необходимо сделать следующее:

- Вызвать метод **edit()** и получить экземпляр класса редактора – **SharedPreferences.Editor**
- Вызвать **put**Boolean(), **put**String() или **put**Int() . . . или **remove**
- Вызвать **commit()** или **apply()**

Для получения необходимо:

- Вызвать **get**Boolean(), **get**String() или **get**Int() . . .

Пример для SharedPreferences

```
public class Calc extends Activity {
    public static final String PREFS_NAME = "MyPrefsFile";

    @Override
    protected void onCreate(Bundle state){
       super.onCreate(state);
       . . .

       // Restore preferences
       SharedPreferences settings = getSharedPreferences(PREFS_NAME, 0);
       boolean silent = settings.getBoolean("silentMode", false);
       setSilent(silent);
    }

    @Override
    protected void onStop(){
       super.onStop();

      // We need an Editor object to make preference changes.
      // All objects are from android.context.Context
      SharedPreferences settings = getSharedPreferences(PREFS_NAME, 0);
      SharedPreferences.Editor editor = settings.edit();
      editor.putBoolean("silentMode", mSilentMode);

      // Commit the edits!
      editor.commit();
    }
}
```