Лекция №10

https://github.com/IFMO-Android-2016/lesson10

О чем не рассказывали на прошлых лекциях

Что еще нужно знать об Android?

- Фрагменты
- Content Provider
- Местоположение
- Запрос пермиссий
- Alarm Manager
- Уведомления
- Push (GCM)
- Безопасность
- Мультимедиа
- Камера
- Support библиотеки
- Material дизайн
- Анимация
- Open GL
- Система сборки
- Инструменты
- Тестирование
- Публикация
- Монетизация
- Сбор статистики

Фрагменты

Фрагменты

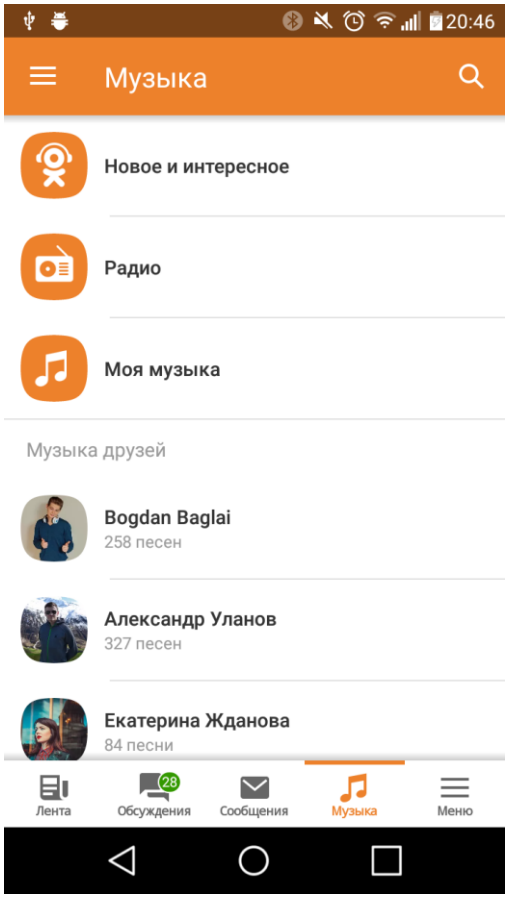
- Фрагмент – модульный компонент со своим жизненным циклом, UI и логикой.
- На практике большая часть UI делается во фрагментах, а в `Activity` остается только общие элементы (меню, App Bar и т.п.)

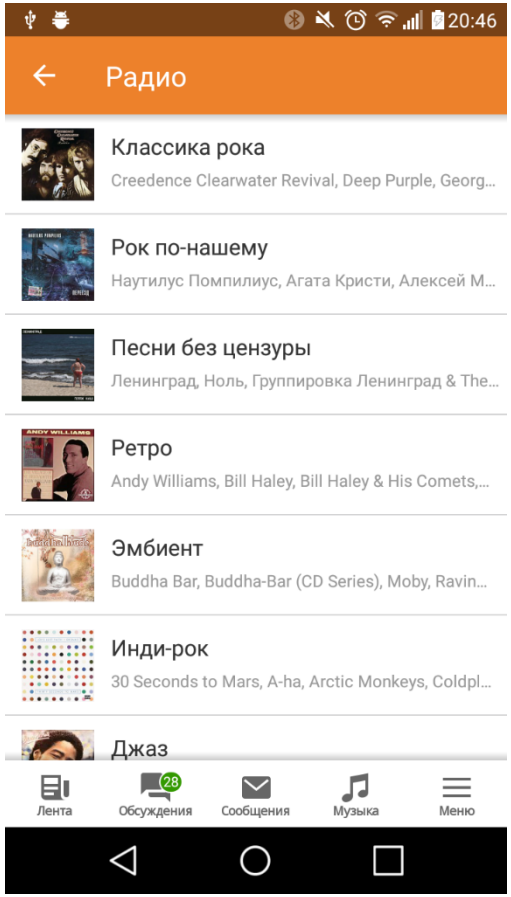
```
public class ArticleFragment extends Fragment {

    public View onCreateView(LayoutInflater inflater,
                             ViewGroup container, Bundle savedInstanceState) {
        View view = inflater.inflate(R.layout.article_view,
                                     container, false);
        // Инициализация UI
        // ...
        return view;
    }
```

Фрагменты

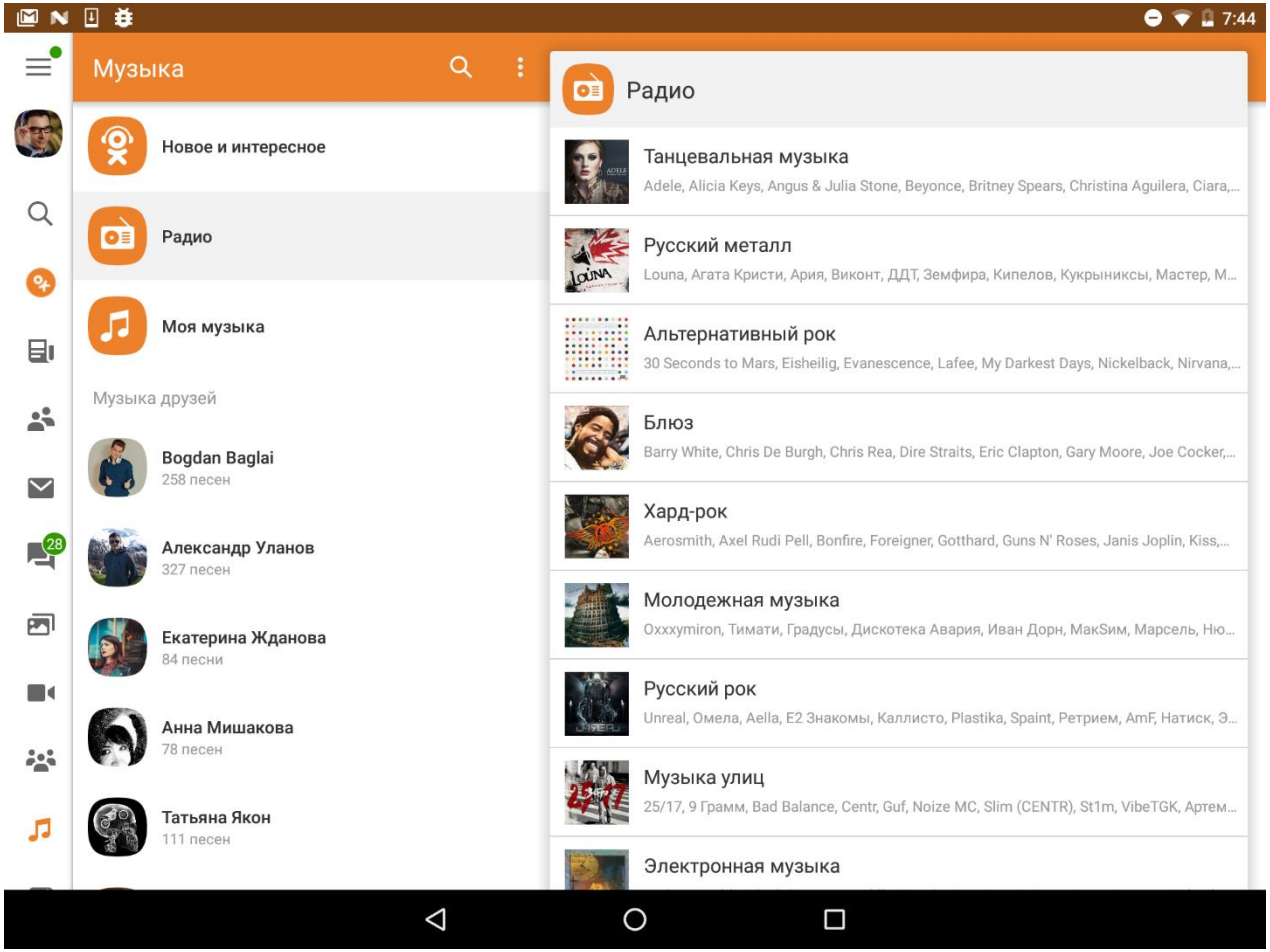
Фрагменты особенно полезны при создании интерфейсов, адаптированных к планшетам.





Фрагменты

Два фрагмента внутри одной активности:



Фрагменты

- Класс `android.app.Fragment` появился в Android 3.0
- Но использовать надо `android.support.v4.app.Fragment`

https://developer.android.com/training/basics/fragments/index.html

Broadcast Receiver

Content Provider

Activity

Service

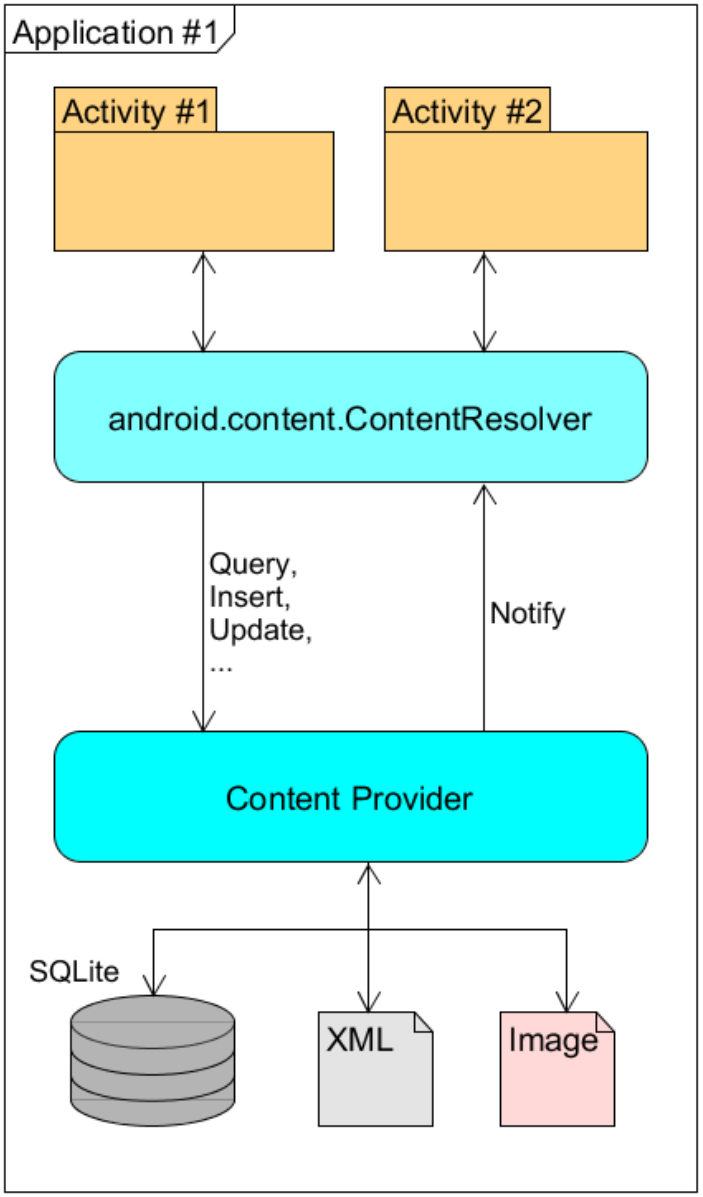
Content Provider

- Content Provider управляет доступом к структурированному набору данных (обычно к БД на основе SQLite, но не только) и **обеспечивает безопасность** данных.

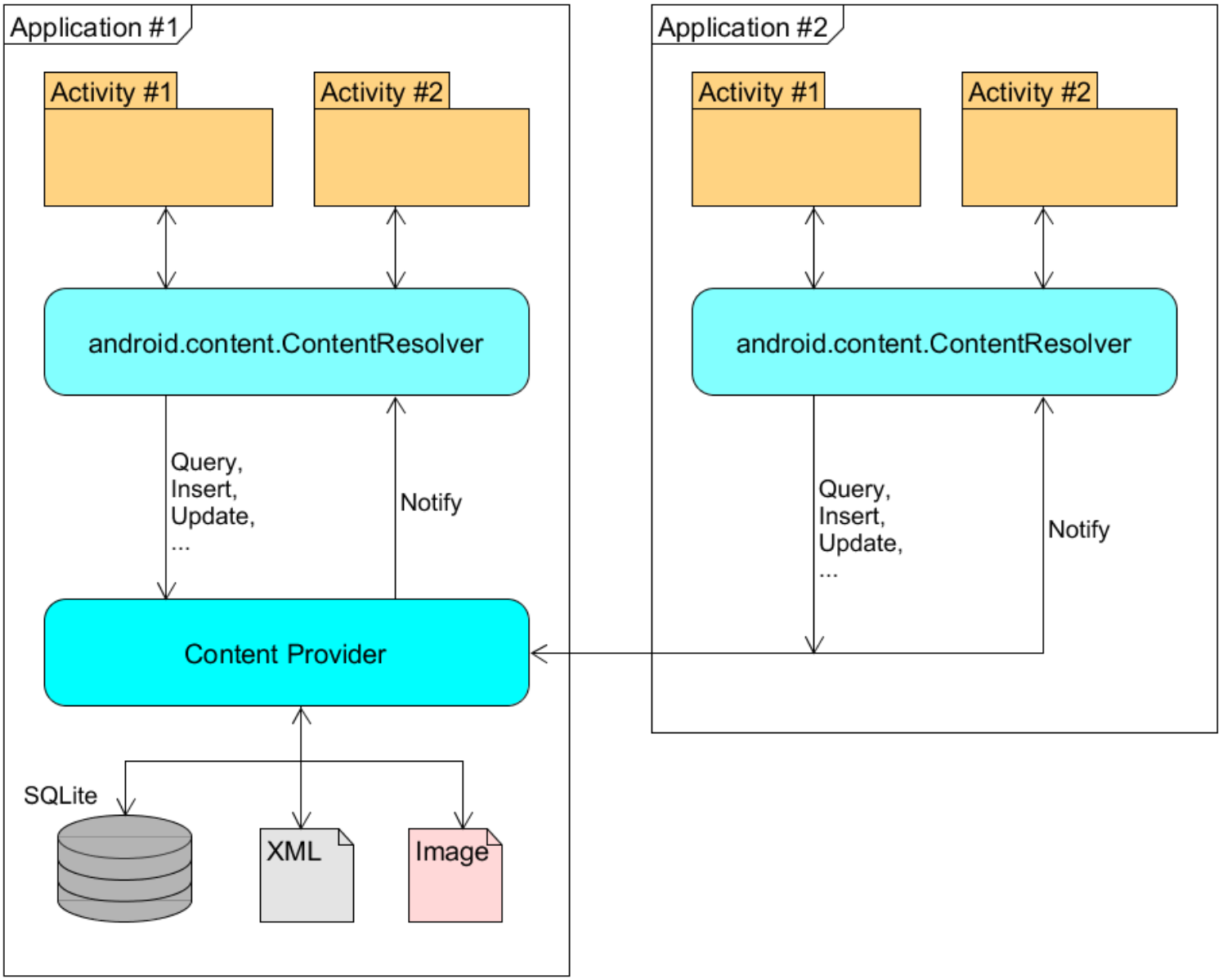
- Позволяет реализовать модель, в которой код, предоставляющий данные выполняется в одном процессе, а код, использующий данные – в другом процессе (в т.ч. **в другом приложении**).

https://developer.android.com/guide/topics/providers/content-providers.html

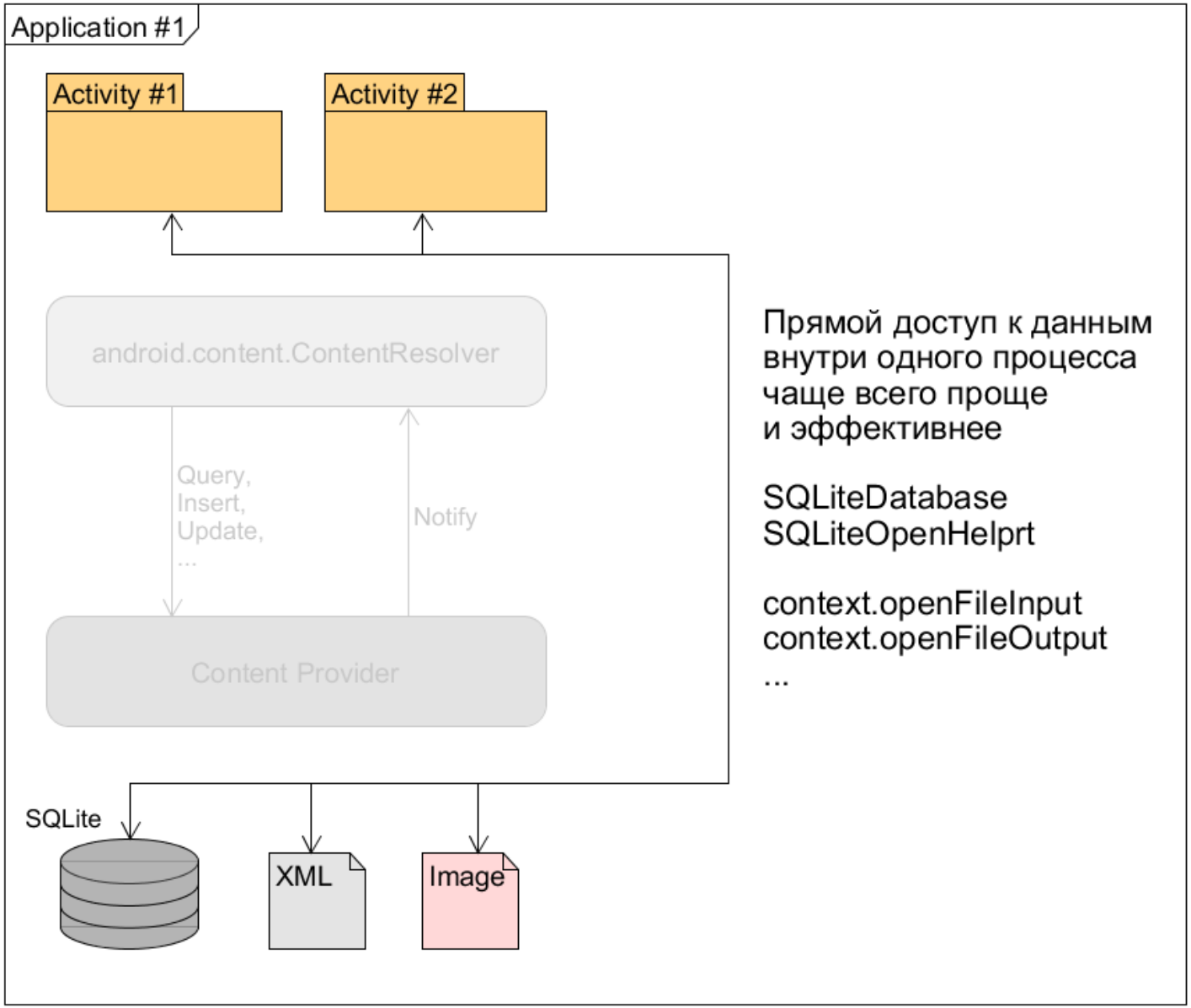
Content Provider



Content Provider



Content Provider



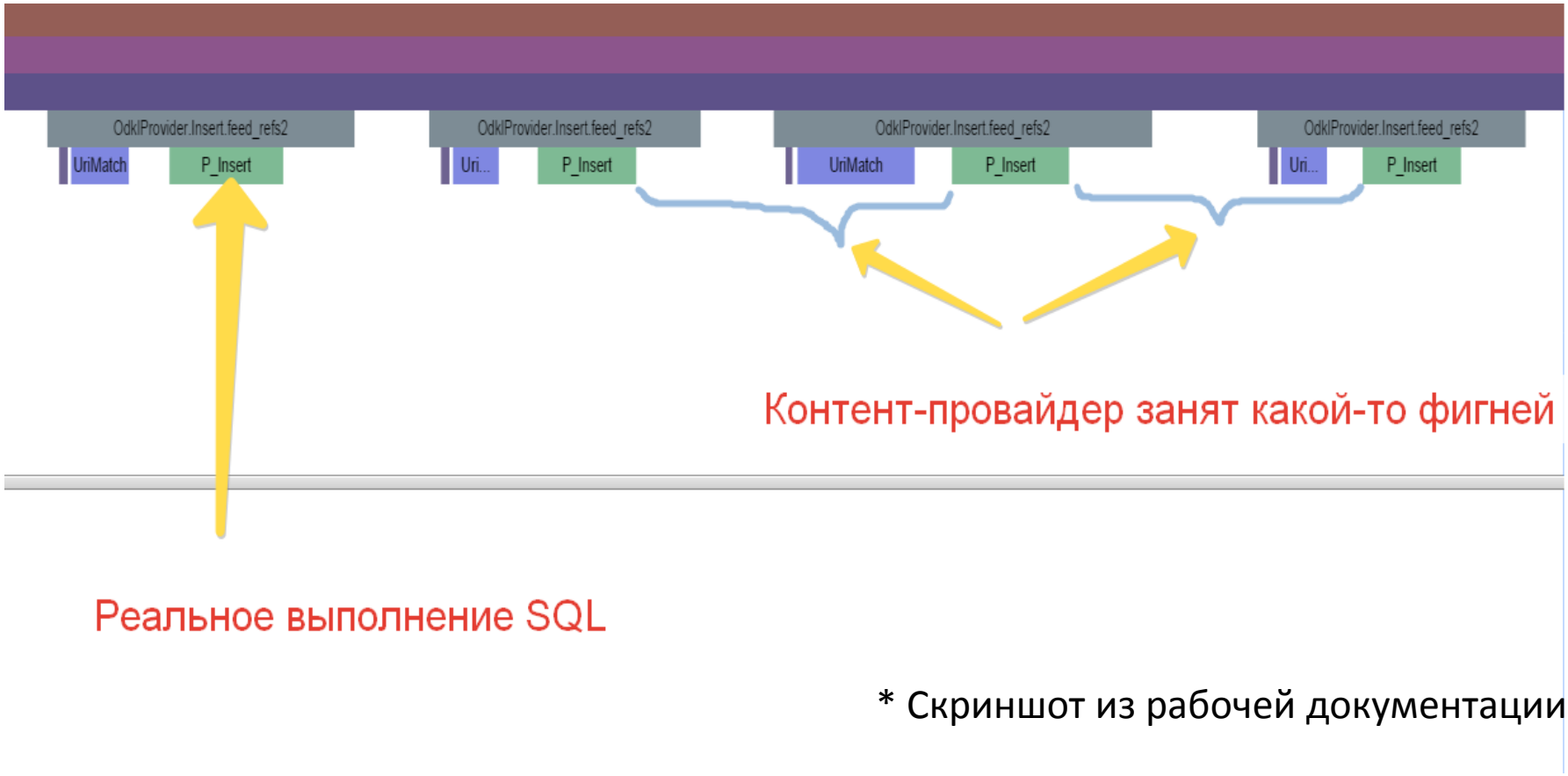
Прямой доступ к данным
внутри одного процесса
чаще всего проще
и эффективнее

SQLiteDatabase
SQLiteOpenHelprt

context.openFileInput
context.openFileOutput
...

Content Provider

Использование Content Provider-а связано с накладными расходами



* Скриншот из рабочей документации

Доступ к данным пользователя

- Приложение может получить доступ к некоторым пользовательским данным при помощи стандартных системных Content Provider-ов.
- Структура данных и способы доступа описаны в контрактах (`android.provider.*`)
- Приложение выступает в роли клиента Content Provider-а, может **получать и записывать** данные при помощи `ContentResolver`

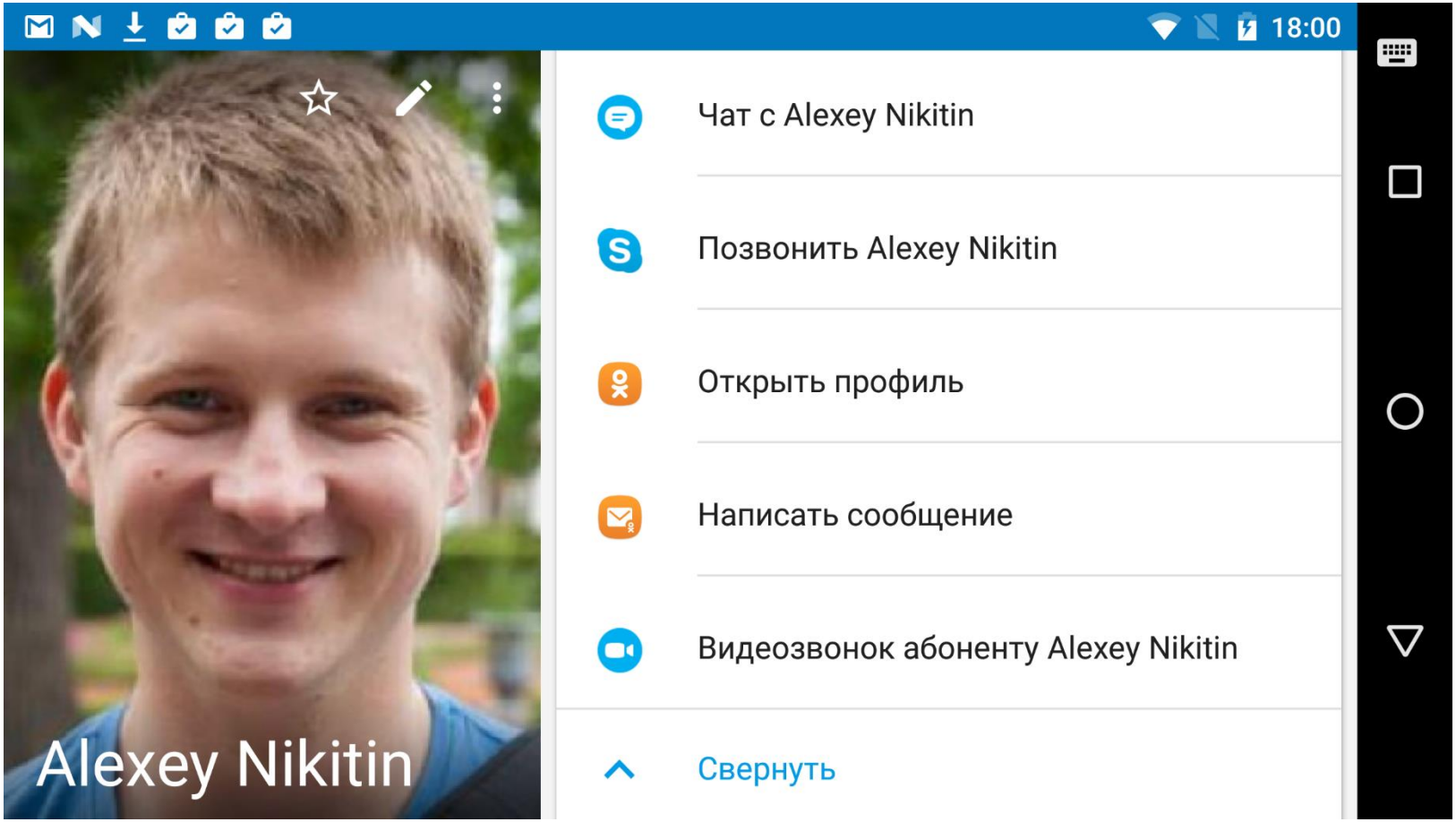
Доступ к данным пользователя

- ContactsContract – записная книжка телефона:
  - Список контактов
  - Номера телефонов (поиск по номеру)
  - E-mail адреса
  - Дни рождения
  - Привязка контактов к социальным сетям и мессенджерам,
  - Синхронизация контактов с «облаком»

https://developer.android.com/guide/topics/providers/contacts-provider.html

Синхронизация контактов

Приложения могут встраиваться в записную книжку телефона при помощи системного провайдера контактов



Доступ к данным пользователя

- CalendarContract
  - Календарь: события, встречи, напоминания
  - Доступ к данным Google Calendar, Exchange

https://developer.android.com/guide/topics/providers/calendar-provider.html

- CallLog – история звонков
- Telephony – SMS (можно только посмотреть, не отправить)

Доступ к данным пользователя

- MediaStore – коллекции медиа данных
  - Изображения
  - Видео
  - Музыка

Автоматически сканирует External Storage, находит новые медиа файлы и добавляет в коллекцию.

*См. лекцию №8 о доступе к медиа файлам

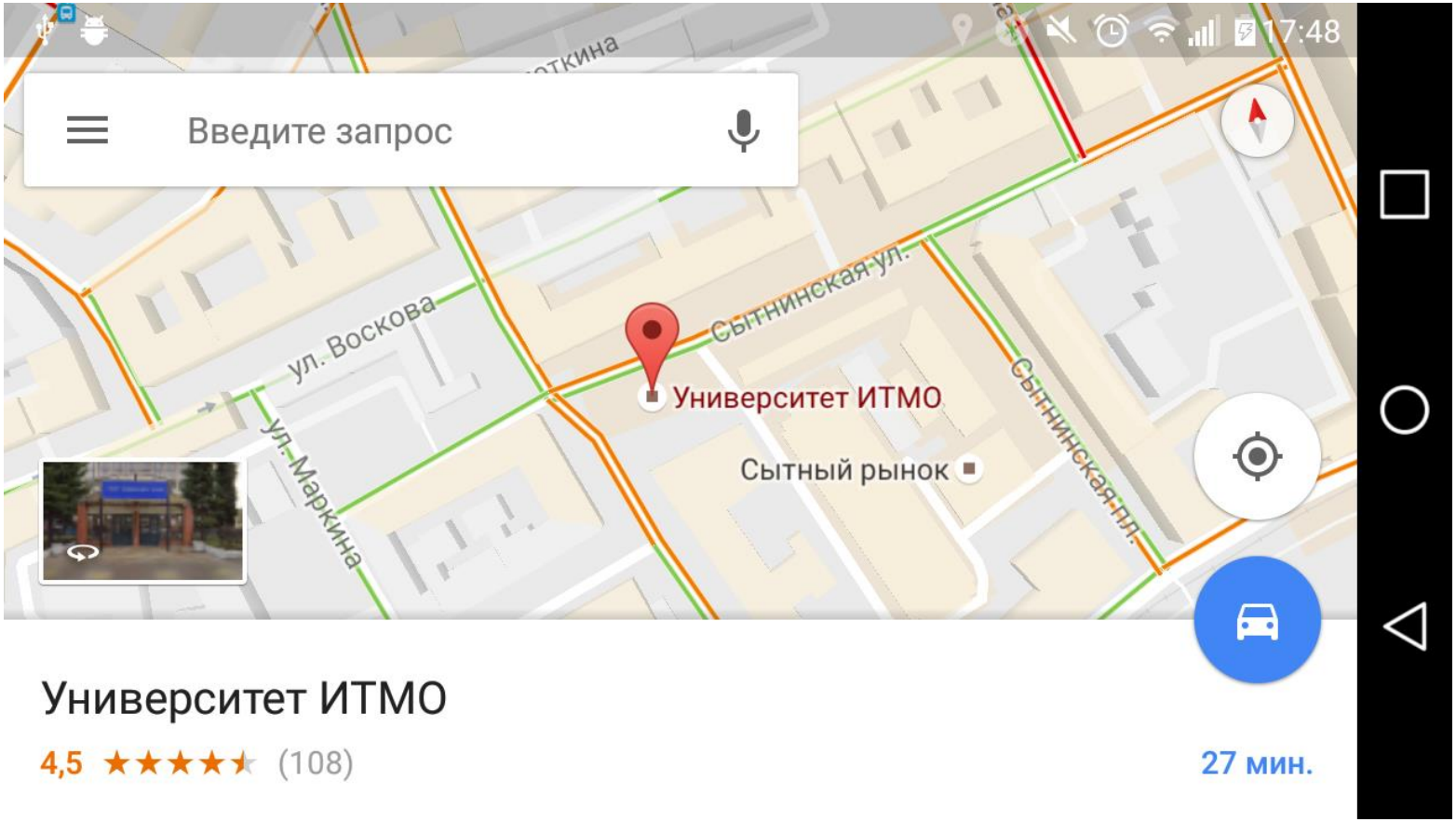
External Storage

Общие файлы:

```
/sdcard/
     Music/
     Pictures/
     Ringtones/
     Download/
     …

Context.getExternalFilesDir(
          Environment.DIRECTORY_XXX)
```

Местоположение



Местоположение

- Навигационные спутники: GPS
  - А также ГЛОНАСС, 北斗卫星导航系统 (BeiDu, BDS), Galileo – **прозрачно** при наличии поддержки в чипсете

- Мобильные сети – используя информацию о ближайших «вышках» и мощности сигнала, можно грубо определить местоположение

- Wi-Fi – «глобальная» база данных о положении постоянных точек доступа.

Местоположение

Способы определения местоположения отличаются по следующим параметрам:

- Точность
- Скорость
- **Энергопотребление**
- Стоимость

Неправильное использование местоположения может привести к чрезмерному расходованию батареи

Местоположение

Android Framework Location API

(android.location.*)

- Низкоуровневый системный framework для определения местоположения

- Источники данных о местоположении:
  - `GPS_PROVIDER` – спутники
  - `NETWORK_PROVIDER` – мобильные сети и Wi-Fi
  - `PASSIVE_PROVIDER` – пассивный (рекомендуется)

Местоположение

Google Play Services Location API

(FusedLocationProviderApi)

- Простой высокоуровневый framework, в котором реализованы оптимальные стратегии определения местоположения

Support Library

Support Library

`android.support.*`

Официальный набор библиотек от Google в дополнение к Android SDK

- Поддержка новых фич Android SDK на старых версиях Android
- Стандартные UI элементы и стили, широко используемые в приложениях
- Просто полезные утилиты

Support Library

Поддержка старых версий Android

Обычный код:

```
if (Build.VERSION.SDK_INT >=
            Build.VERSION_CODES.LOLLIPOP) {
    view.setElevation(5.0f);
}
```

Код с Support Library:

```
import android.support.v4.view.ViewCompat;

ViewCompat.setElevation(view, 5.0f);
```

Support Library

Замена стандартных классов из Android SDK на более новую реализацию без старых багов:

android.app.Loader → android.support.v4.app.Loader

android.app.Fragment → android.support.v4.app.Fragment

Всегда лучше использовать аналогичный класс из Support Library – он будет поддерживаться и постоянно обновляться вместе с приложением.

Support Annotations

Аннотации для Android кода:

com.android.support:annotations

- Делают код более читаемым
- Уменьшают количество ошибок
- Поддерживаются инструментами разработки (Android Studio) и сборки
- НЕ влияют на выполнение кода

Support Annotations

@NonNull

@Nullable

Nullness проверки позволяют предотвратить NullPointerException и уменьшить количество явных проверок в коде.

```
@NonNull
public LoadResult<List<Webcam>> loadInBackground() {
    if (latitude > 90.0) {
        return null;
    }

    return doLoad(latitude, longitude);
}
```

'null' is returned by the method declared as @NotNull more... (Ctrl+F1)

Support Annotations

@IntDef

@StringDef

- Определяет набор значений внутри типов `int` и `String`
- Простая и быстрая замена enum
- Валидация во время сборки

Support Annotations

Документируем возможные возвращаемые значения…

```
public static final int TYPE_PHOTO = 1;
public static final int TYPE_VIDEO = 2;

/**
 * Gets attachment type
 *
 * @return  {@link #TYPE_PHOTO} or {@link #TYPE_VIDEO}
 */
int getAttachmentType() {
    return 0;
}
```

Support Annotations

В Java обычно используют enum, но это добавляет нам один лишний класс и создание всех значений в памяти…

```
enum AttachmentType {
    PHOTO,
    VIDEO
}

@NonNull
AttachmentType getAttachmentType() {
    return AttachmentType.PHOTO;
}
```

Support Annotations

А с @IntDef аннотацией никаких издержек!

```
@IntDef(value = {
        AttachmentType.PHOTO,
        AttachmentType.VIDEO,
})
@interface  AttachmentType {
    int PHOTO = 1;
    int VIDEO = 2;
}

@AttachmentType
int getAtttachmentType() {
    return 0;
}
```

Must be one of: AttachmentType.PHOTO, AttachmentType.VIDEO more... (Ctrl+F1)

Support Annotations

@Keep

@ColorInt

@CheckResult

@IntRange

@AnyThread

@WorkerThread

@ArrayRes

@VisibleForTesting

@ColorRes

@UiThread

@CallSuper

@RequiresPermission

@StringRes

@Px

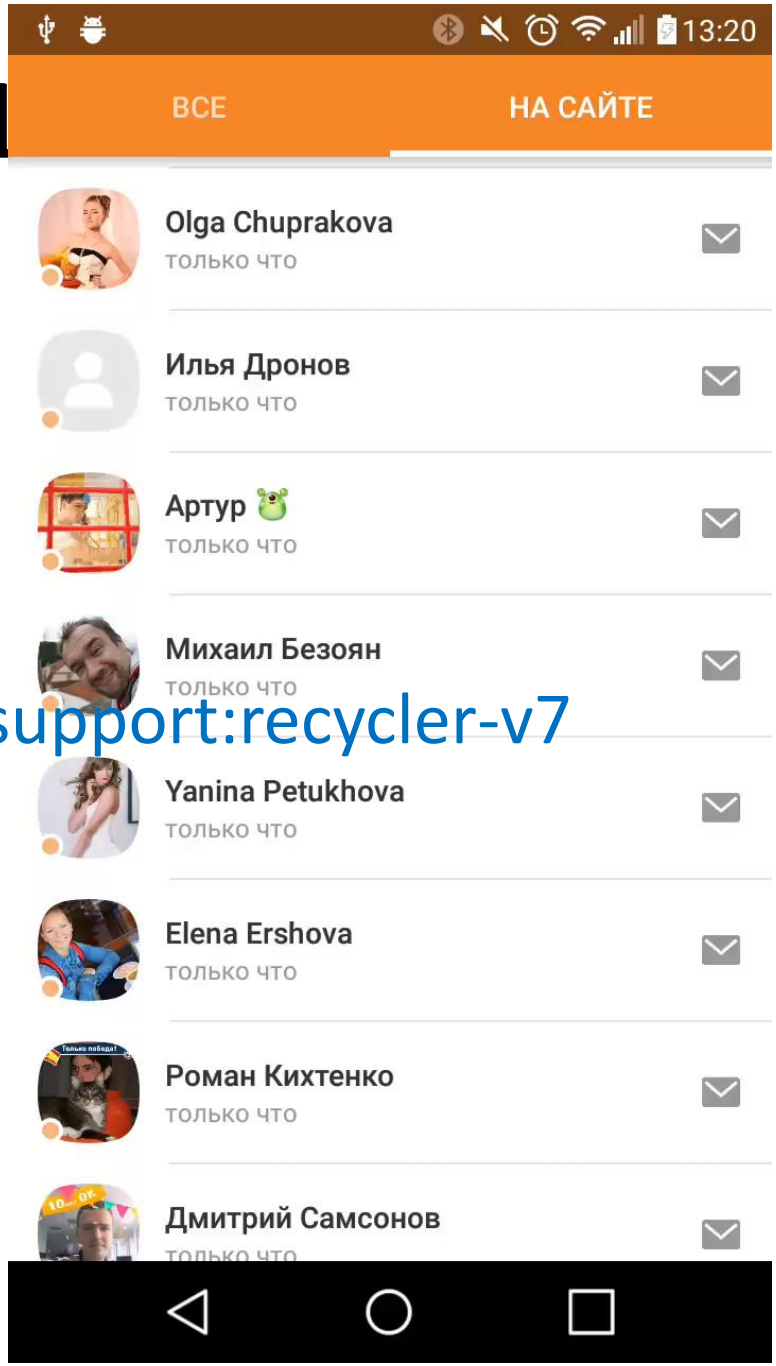
Support L

Класс:

RecyclerView

Зависимость:

com.android.support:recycler-v7



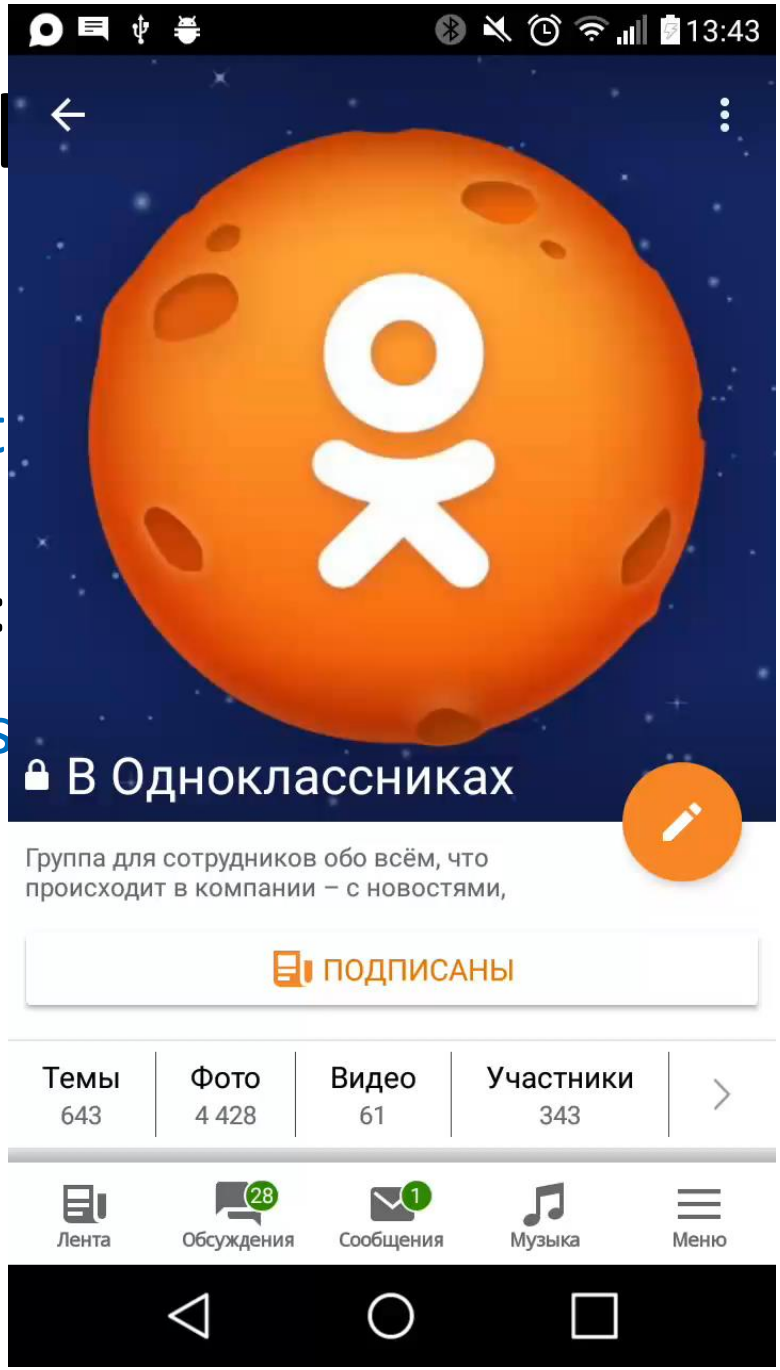
Support L

Класс:

DrawerLayout

Зависимость:

com.android.s



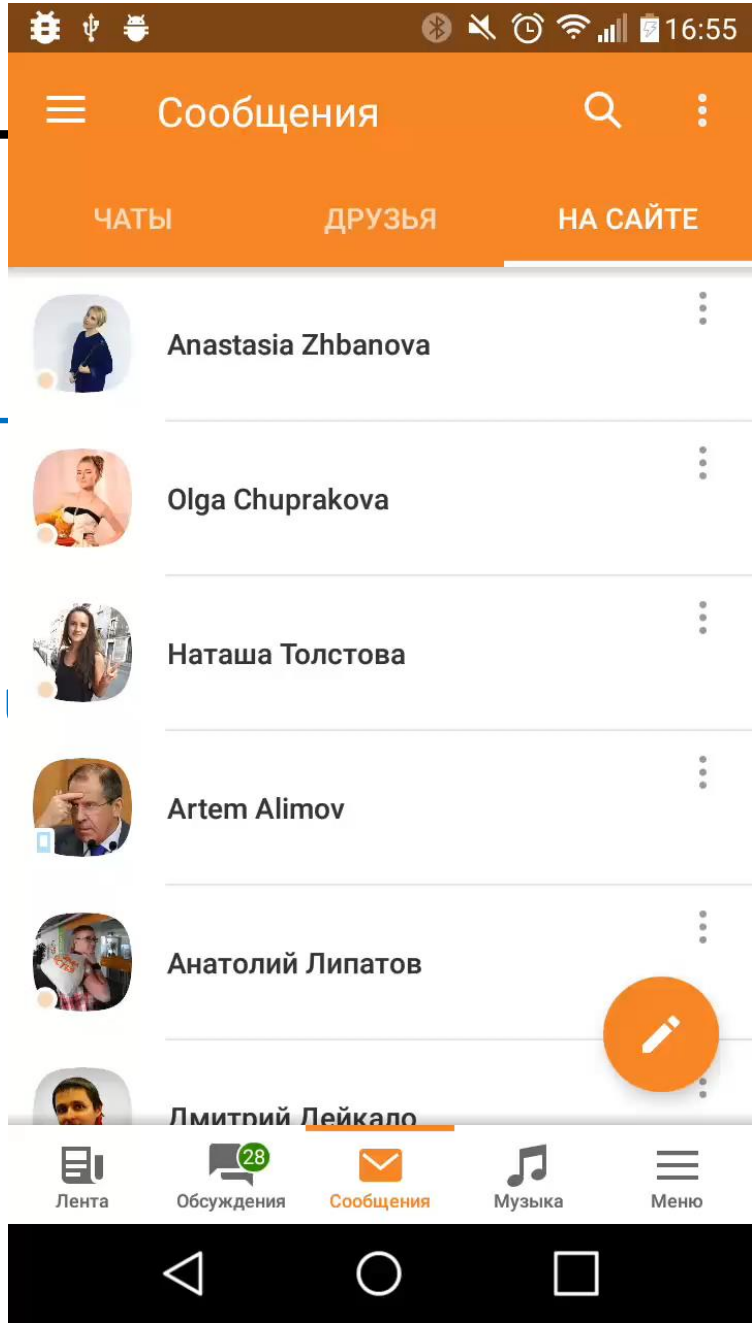
Support L

Класс:

SwipeRefreshL

Зависимость:

com.android.s



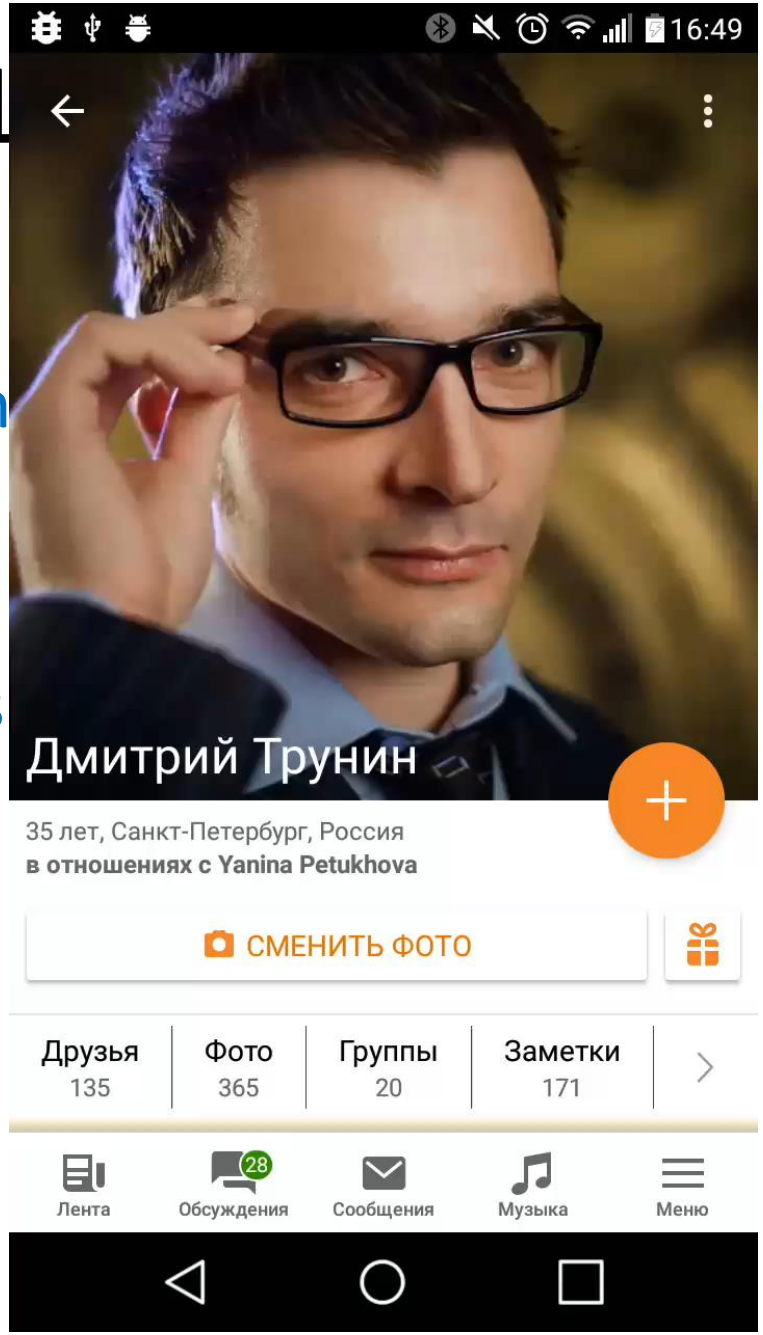
Support L

Класс:

FloatingAction

Зависимость:

com.android.s



Support L

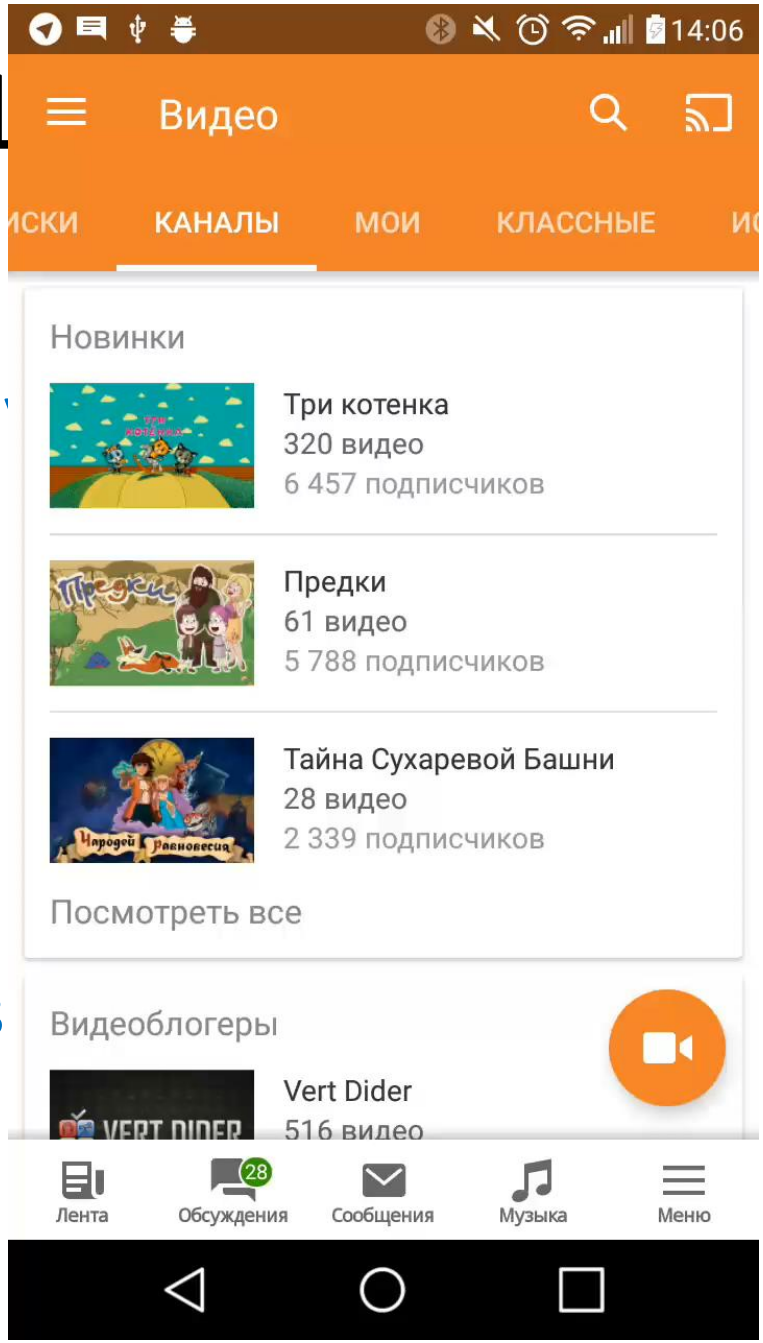
Класс:

CoordinatorLa
AppBarLayout
*.Behavior

и др.

Зависимость:

com.android.s



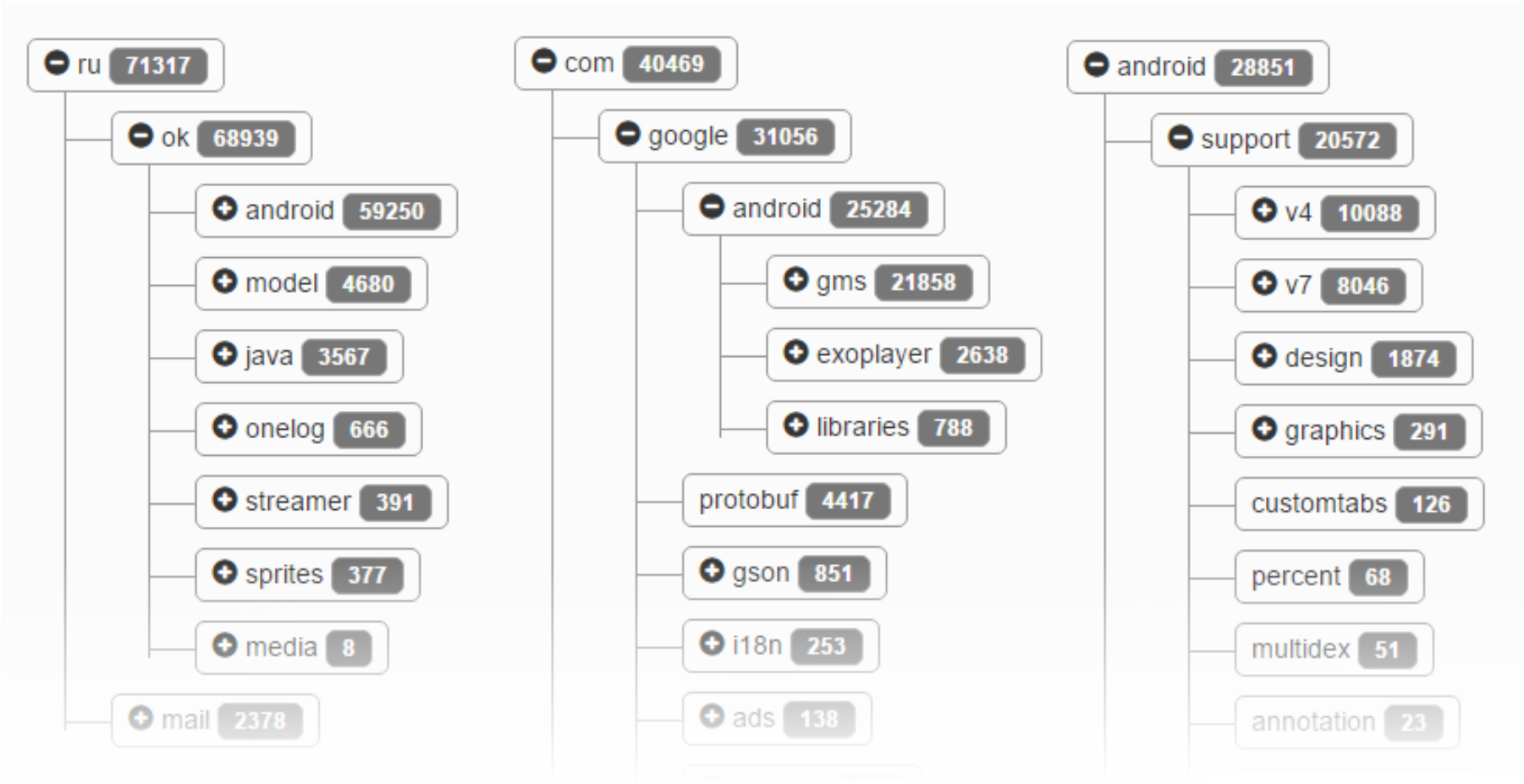
Количество методов

- До Android 5.0 использовалась виртуальная машина Dalvik с ограничением в 65535 методов на один dex-файл.
- Считаются все методы вашего кода и методы **всех библиотек**, которые вы используете
- Начиная с Android 5.0 используется виртуальная машина Art, у которой нет такого ограничения
- 50% пользователей все еще используют Android 4.x

65535 методов должно хватить всем

Считаем методы

Приложение «Одноклассники»: 155к методов



Support Library

MultiDex

Подменяет загрузчик классов приложения для загрузки кода из нескольких dex файлов

Зависимость:

com.android.support:multidex

+ Решает проблему ограничения на 65к методов

– Сильно тормозит при старте приложения

В следующий раз Алексей Никитин расскажет о разных инструментах разработки, помогающих разрабатывать и отлаживать Android приложения

(Возможно, самая полезная лекция)