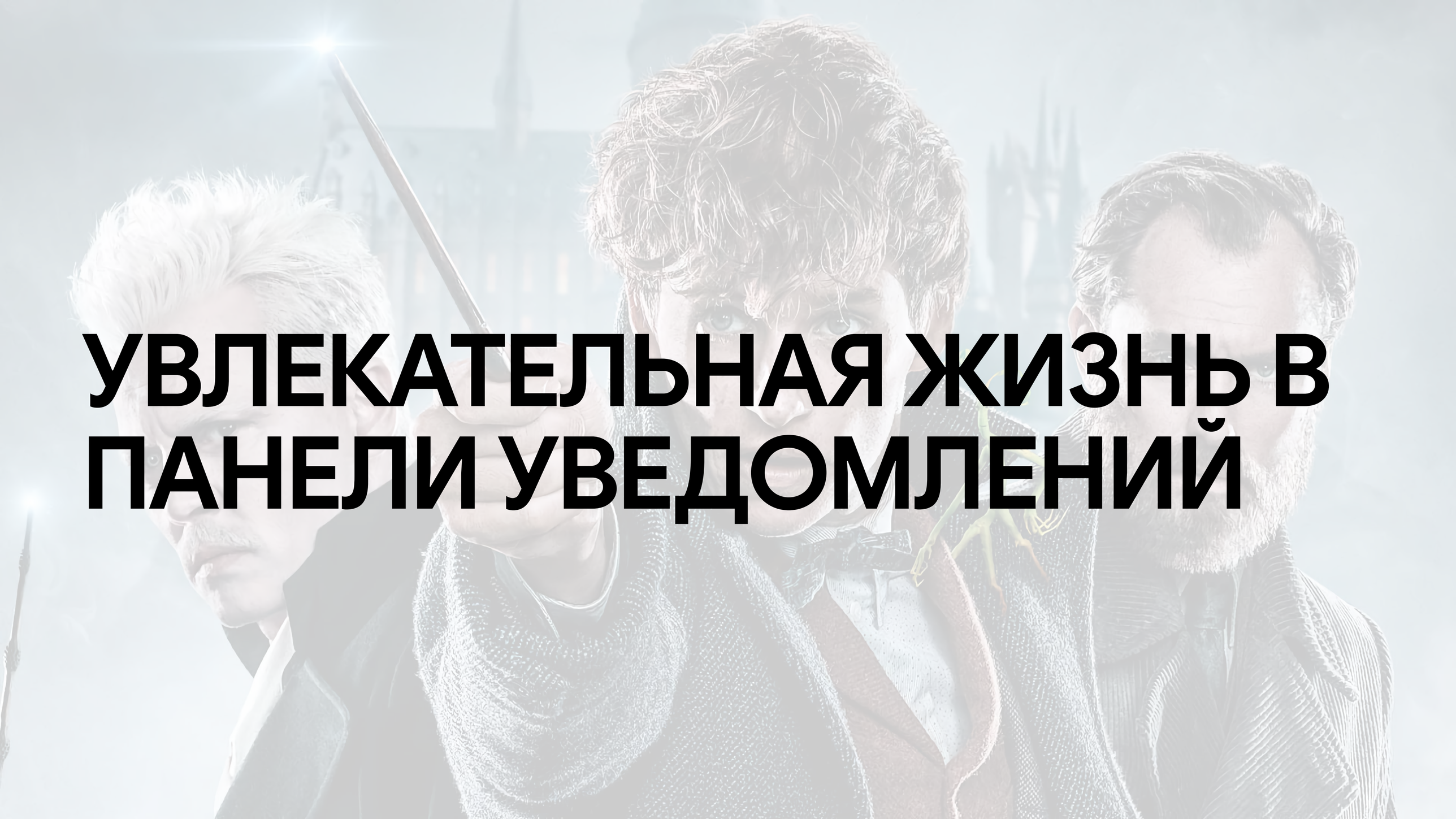
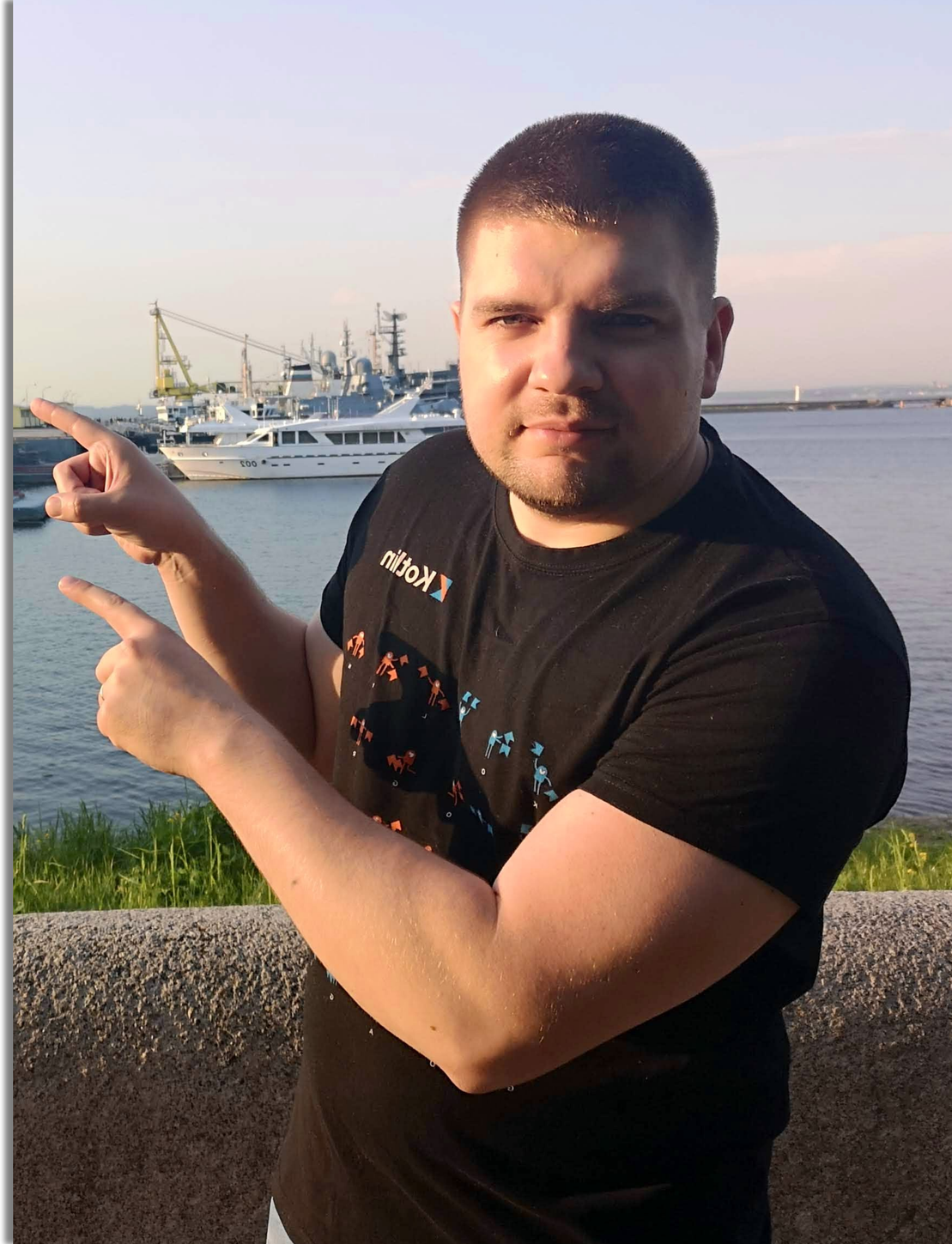


УВЛЕКАТЕЛЬНАЯ ЖИЗНЬ В ПАНЕЛИ УВЕДОМЛЕНИЙ

КТО Я

КИРИЛЛ РОЗОВ

- Опыт в разработке под Android - 8+ лет
- Фанат Kotlin
- Mobile Lead в Replika.ai
- Автор проекта “Android Broadcast”



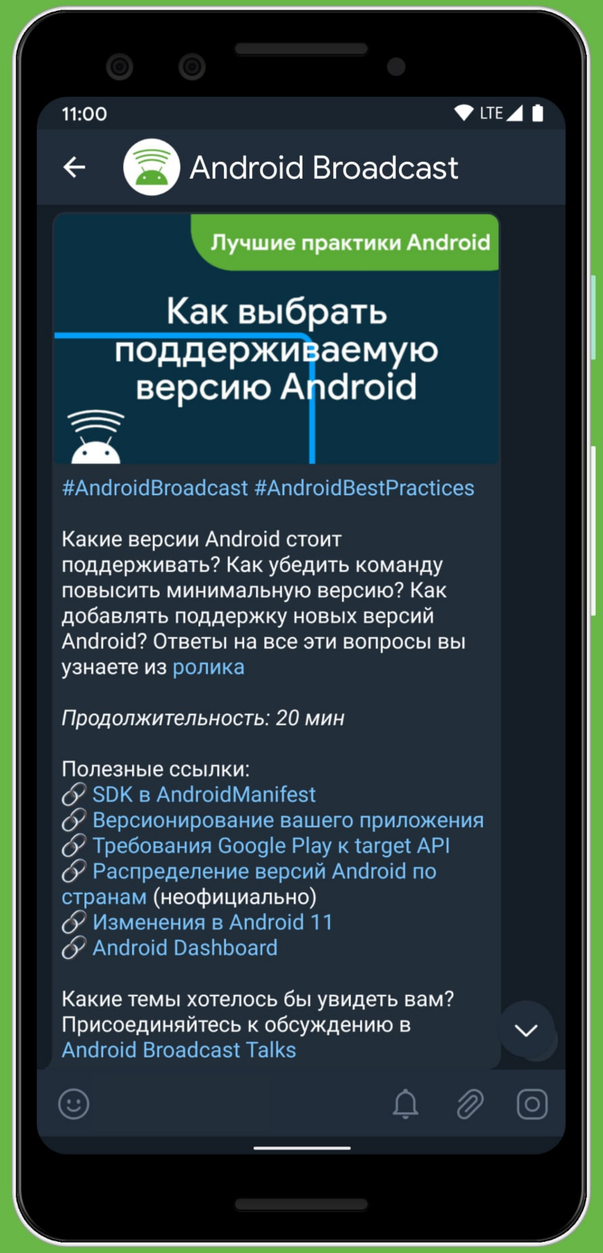
kirill_rozov

krlrozov

Android Broadcast

Подборка лучших новостей из мира Android разработки

@android_broadcast



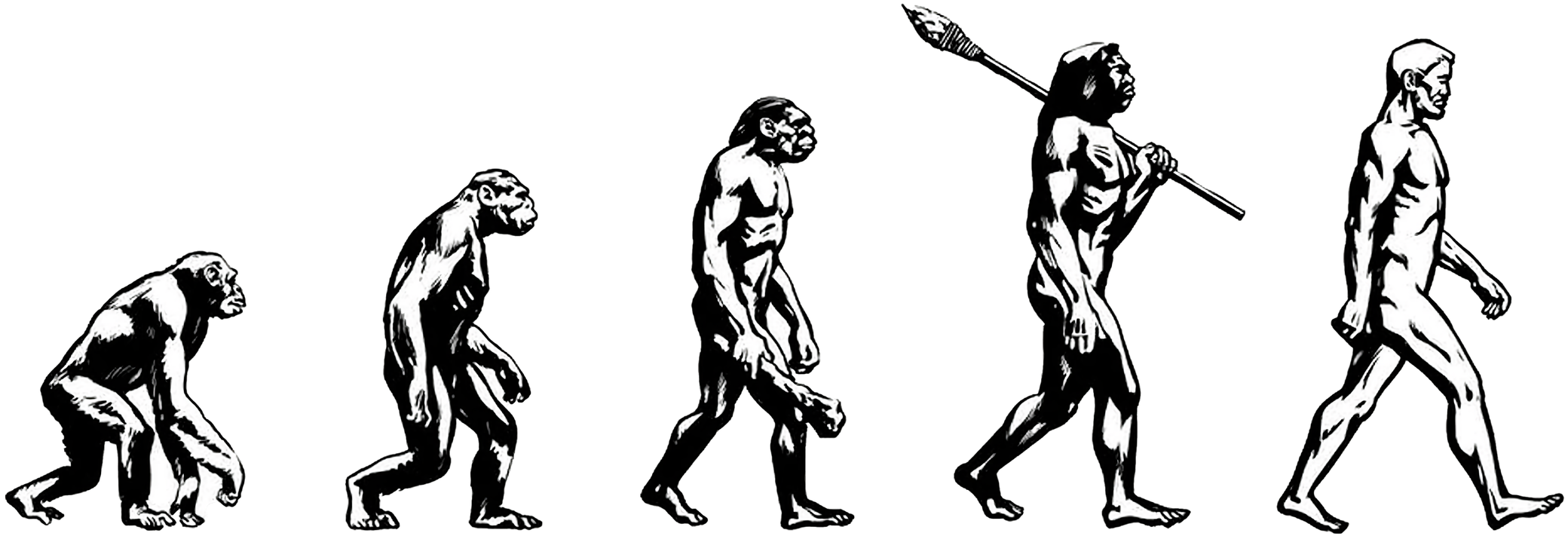
О ЧЕМ ПОГОВОРИМ

1. Эволюция уведомлений
2. Проблемы уведомлений в Android
3. AndroidX NotificationCompat
4. Библиотека Android Notification DSL
5. Firebase Cloud Messaging

ПОЧЕМУ УВЕДОМЛЕНИЯ ВАЖНЫ?

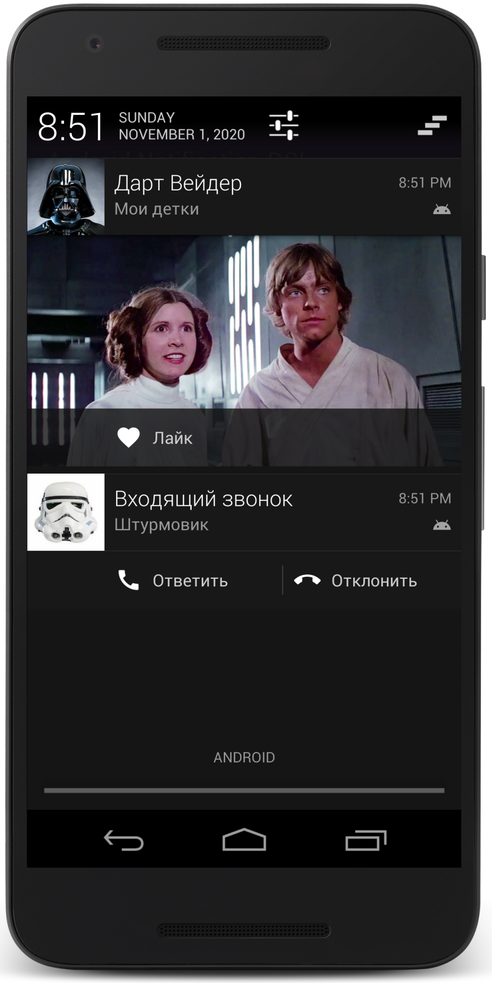
- Быстрый формат для получения информации от приложения без входа в него
- Возможность напомнить что стоит открыть приложение
- В “одном свайпе” от любого экрана
- Огромные возможности для мессенджеров
- Уникальная возможность нативных приложений, без Runtime Permission
- Ограничения на работу Service в фоне и запрет на запуск Activity из фона требуют работу с уведомлениями

1. ЭВОЛЮЦИЯ УВЕДОМЛЕНИЙ



ANDROID 4.1

- Действия в уведомлениях
- Увеличен максимальный размер (с 64 dp до 256 dp)
- *BigPictureStyle*, *BigTextStyle* и *InboxStyle*
- Приоритеты уведомлений



2012

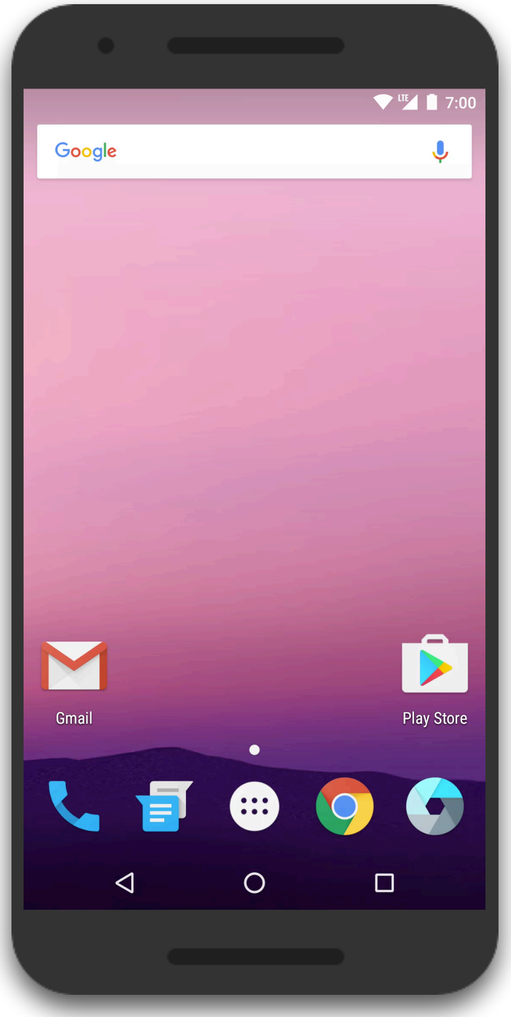
ANDROID 5.0

- Уведомления на экране блокировки
- Категории уведомлений
- Привязка контакта к уведомлению
- Heads-Up уведомления (полноэкранные)
- *MediaStyle*

2014

ANDROID 7.0

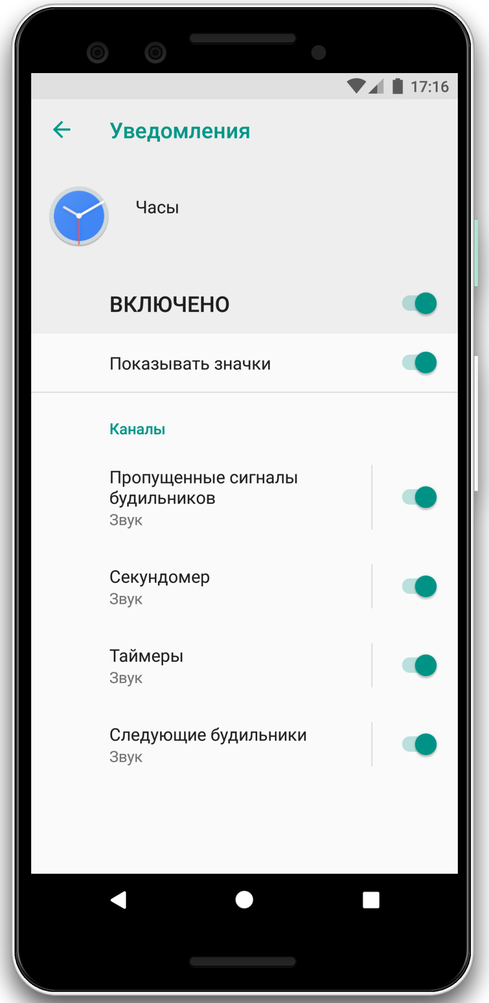
- Новые стили уведомлений:
  MessagingStyle и *DecoratedCustomViewStyle*
- Группировка уведомлений
- Ответ прямо из уведомления



2016

ANDROID 8.0

- Каналы уведомлений (Notification channels)
- Счетчик уведомлений (Notification dots)
- Возможность отложить уведомление
- Таймер показа уведомления
- Изменение цвета фона уведомления



ANDROID 9.0

- Поддержка картинок в *MessagingStyle*
- Сохранение ответов в виде драфта
- Разделение групповых и личных чатов
- Smart Reply (задаются разработчиком)

2018

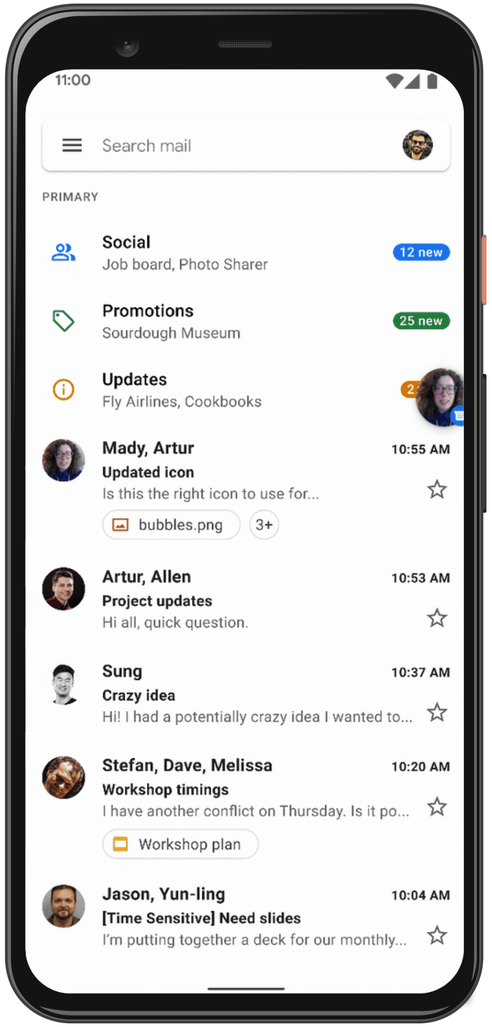
ANDROID 10

- Smart Reply (автогенерация системой)

2019

ANDROID 11

- **Conversation Bubbles**
  // Привет Facebook Messenger



2020

2. ПРОБЛЕМЫ УВЕДОМЛЕНИЙ

Notification.MediaStyle

Notification.Action.WearableExtender

Notification.InboxStyle

Notification.BigPictureStyle

Notification.Action.Builder

Notification.WearableExtender

NOTIFICATION.BUILDER

Notification.BubbleMetadata.Builder

Notification.MessagingStyle

Notification.BigTextStyle

Notification.CarExtender.Builder

ПРОСТОЕ УВЕДОМЛЕНИЕ

```
val builder = NotificationCompat.Builder(context, NOTIFICATION_CHANNEL_DEFAULT)
    .setSmallIcon(R.drawable.ic_android_white_24dp)
    .setContentTitle(“Android Broadcast”)

val editAction = NotificationCompat.Action.Builder(
    R.drawable.ic_baseline_edit_24,
    "Edit",
    context.activityPendingIntent<MainActivity>(ACTION_EDIT_REQUEST_CODE)
).build()
builder.addAction(editAction)

val viewAction = NotificationCompat.Action.Builder(
    R.drawable.ic_baseline_remove_red_eye_24,
    "View",
    context.activityPendingIntent<MainActivity>(ACTION_VIEW_REQUEST_CODE)
).build()
builder.addAction(viewAction)

val notification: Notification = builder.build()
```

ПОЛНЫЙ ОБВЕС

```
NotificationCompat.Builder(context, NOTIFICATION_CHANNEL_DEFAULT)
    .setContentText(message.message)
    .addPerson(message.personUri)
    .setAutoCancel(true)
    .setContentTitle(message.from)
    .setPriority(NotificationCompat.PRIORITY_HIGH)
    .setDefaults(NotificationCompat.DEFAULT_ALL)
    .setOnlyAlertOnce(true)
    .setWhen(message.date.time)
    .setVisibility(NotificationCompat.VISIBILITY_PRIVATE)
    .setLargeIcon(BitmapFactory.decodeFile(message.personIconPath))
    .setCategory(NotificationCompat.CATEGORY_MESSAGE)
    .addAction(
        NotificationCompat.Action.Builder(null, "Mark as Read", markAsReadIntent)
            .setContextual(false)
            .setSemanticAction(NotificationCompat.Action.SEMANTIC_ACTION_MARK_AS_READ)
            .setShowsUserInterface(false)
            .build()
    )
    .setStyle(
        NotificationCompat.BigTextStyle()
            .bigText(message.message)
    )
    .extend(
        NotificationCompat.WearableExtender()
            .setContentIntentAvailableOffline(false)
    )
    .build()
```

ПРОБЛЕМЫ ANDROID NOTIFICATION

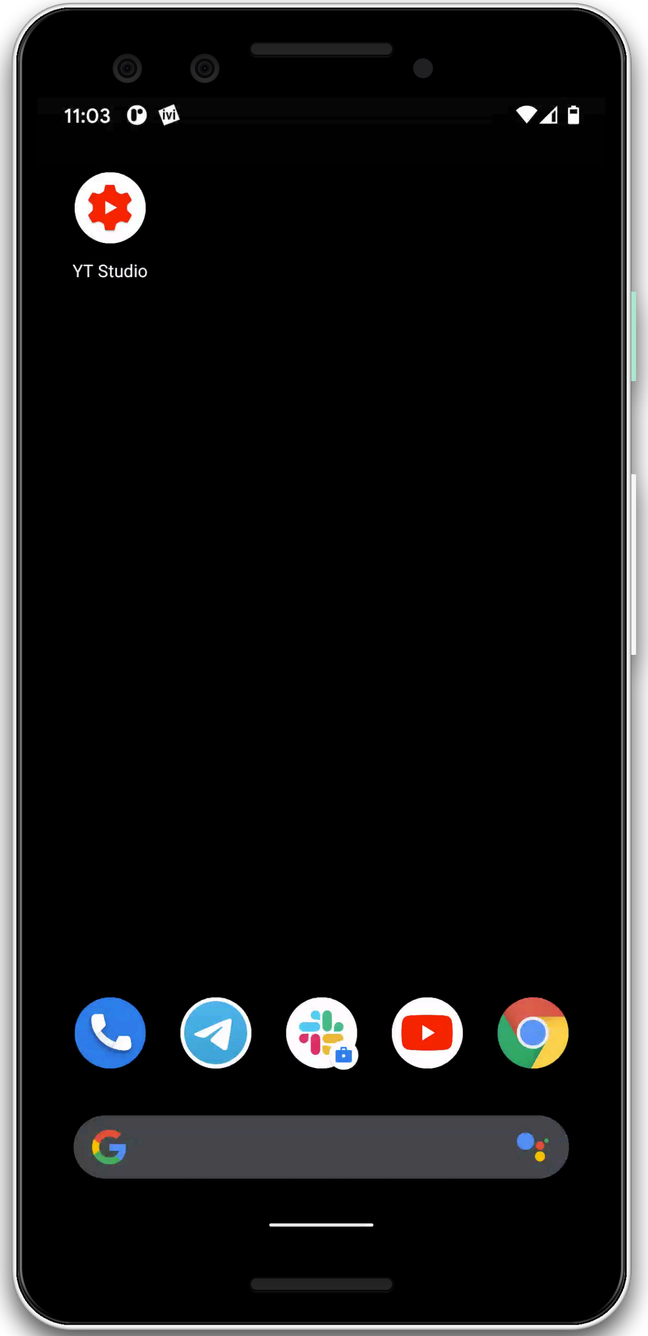
- Какие методы мне надо вызвать чтобы получить хорошее уведомление?
- Вложенность различных Builder
- Невозможность переиспользовать Notification для создания других Notification

```
// Появится в AndroidX Core 1.5.0. Текущая стабильная версия - 1.3.2
```

ОШИБКИ РАЗРАБОТЧИКОВ

CUSTOM VIEW

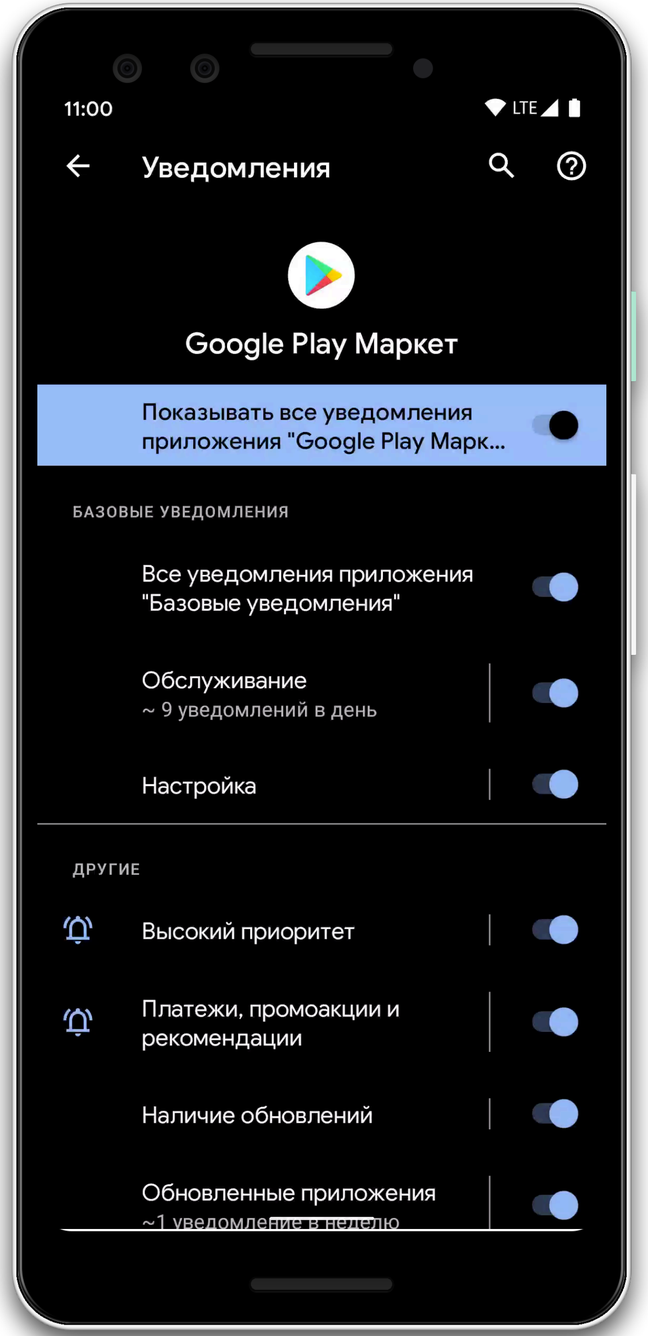
- Неправильный фон
- Выбивается из общего вида
- Каждый производитель имеет особенности внешнего вида уведомлений



ДОСТУПНОСТЬ УВЕДОМЛЕНИЙ

ЧТО МОЖЕТ ОТКЛЮЧИТЬ ПОЛЬЗОВАТЕЛЬ

- Все уведомления
- Группу каналов уведомлений
- Канал уведомлений



ПРОВЕРКА ДОСТУПНОСТИ УВЕДОМЛЕНИЙ

```
// Проверяем что уведомления доступны для приложения
if (!areNotificationsEnabled()) return false
// Проверяем каналы уведомлений на Android 8.0+
if (Build.VERSION.SDK_INT ⩾ Build.VERSION_CODES.O) {
    // Проверяем что канал уведомлений включен
    val channel = getNotificationChannel(channelId) ?: return true
    if (channel.importance == NotificationManager.IMPORTANCE_NONE) return false
    // Проверяем что группа уведомлений не заблокирована
    if (Build.VERSION.SDK_INT ⩾ Build.VERSION_CODES.P) {
        val channelGroup = channel.group?.let(::getNotificationChannelGroup)
        if (channelGroup ≠ null && channelGroup.isBlocked) return false
    }
}
return true
```

3. ANDROIDX NOTIFICATION COMPAT

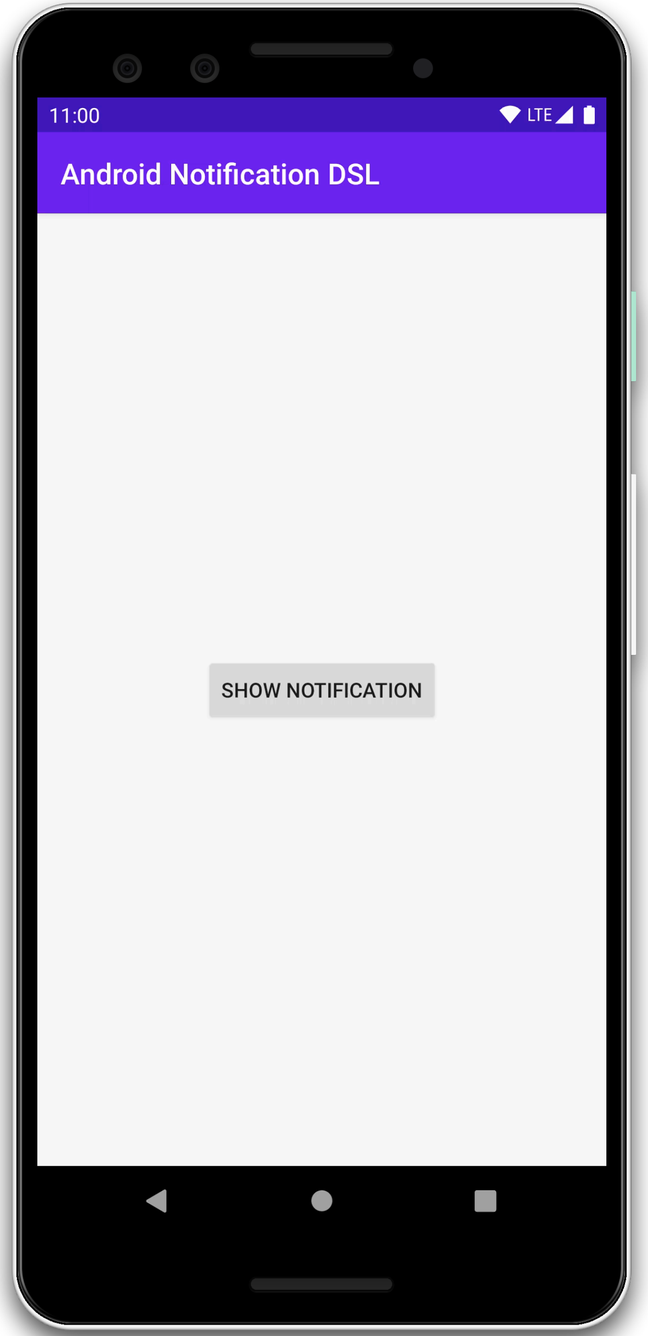
ФРАГМЕНТИРОВАННОСТЬ УВЕДОМЛЕНИЙ

```
val summary = NotificationCompat.Builder(context, CHANNEL_HIGH)
    .setGroupSummary(true)
    .setGroup(GROUP_KEY)
    .setContentTitle(context.getText(R.string.notification_summary_title))
    .setContentText(context.getText(R.string.notification_summary_text))
    .setSmallIcon(R.drawable.ic_android_white_24dp)
    .build()

val notification1 = NotificationCompat.Builder(context, CHANNEL_LOW)
    .setContentTitle(context.getText(R.string.notification_1_title))
    .setSmallIcon(R.drawable.ic_android_white_24dp)
    .setGroup(GROUP_KEY)
    .build()

val notification2 = NotificationCompat.Builder(context, CHANNEL_LOW)
    .setContentTitle(context.getText(R.string.notification_2_title))
    .setSmallIcon(R.drawable.ic_android_white_24dp)
    .setGroup(GROUP_KEY)
    .build()
```

ФРАГМЕНТИРОВАННОСТЬ УВЕДОМЛЕНИЙ

ANDROID 11

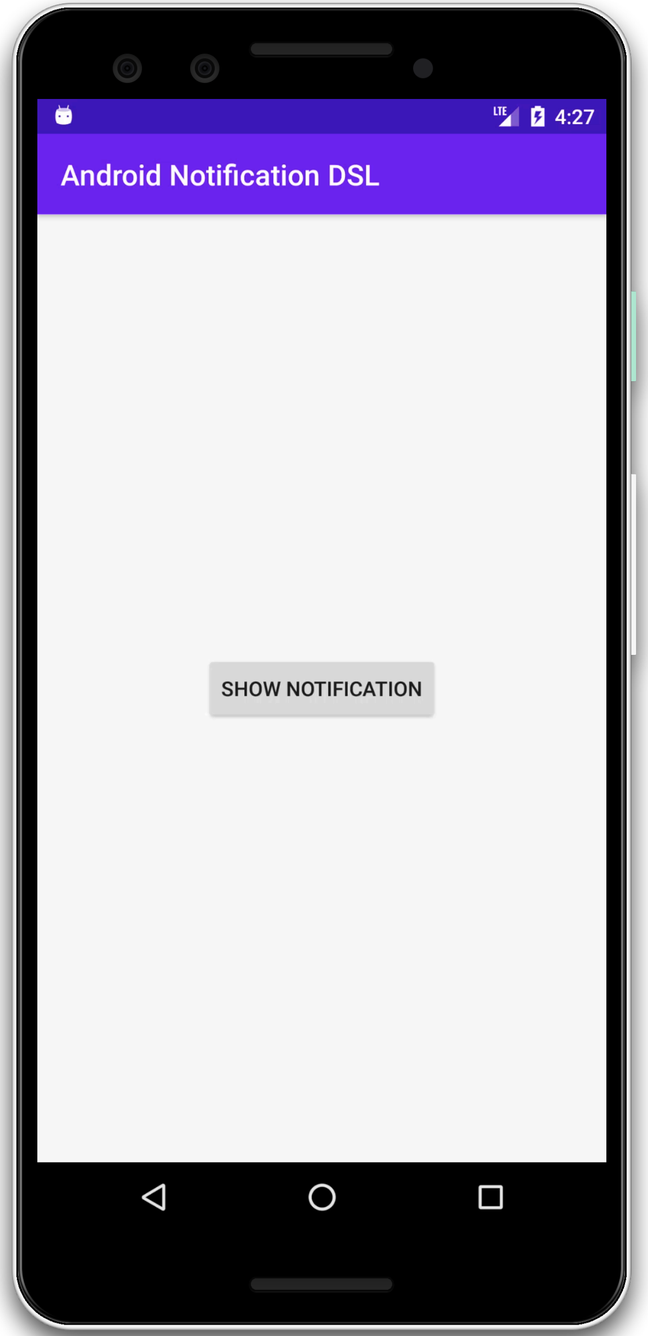
- Поддержка групп
- Поддержка каналов уведомлений



ФРАГМЕНТИРОВАННОСТЬ УВЕДОМЛЕНИЙ

ANDROID 6

- Нет поддержки групп
- Нет поддержки каналов уведомлений



ФРАГМЕНТИРОВАННОСТЬ УВЕДОМЛЕНИЙ

```
val summary = NotificationCompat.Builder(context, CHANNEL_HIGH)
    .setGroupSummary(true)
    .setGroup(GROUP_KEY)
    .setContentTitle(context.getText(R.string.notification_summary_title))
    .setContentText(context.getText(R.string.notification_summary_text))
    .setSmallIcon(R.drawable.ic_android_white_24dp)
    .build()

val notification1 = NotificationCompat.Builder(context, CHANNEL_LOW)
    .setContentTitle(context.getText(R.string.notification_1_title))
    .setSmallIcon(R.drawable.ic_android_white_24dp)
    .setGroup(GROUP_KEY)
    .build()

val notification2 = NotificationCompat.Builder(context, CHANNEL_LOW)
    .setContentTitle(context.getText(R.string.notification_2_title))
    .setSmallIcon(R.drawable.ic_android_white_24dp)
    .setGroup(GROUP_KEY)
    .build()
```

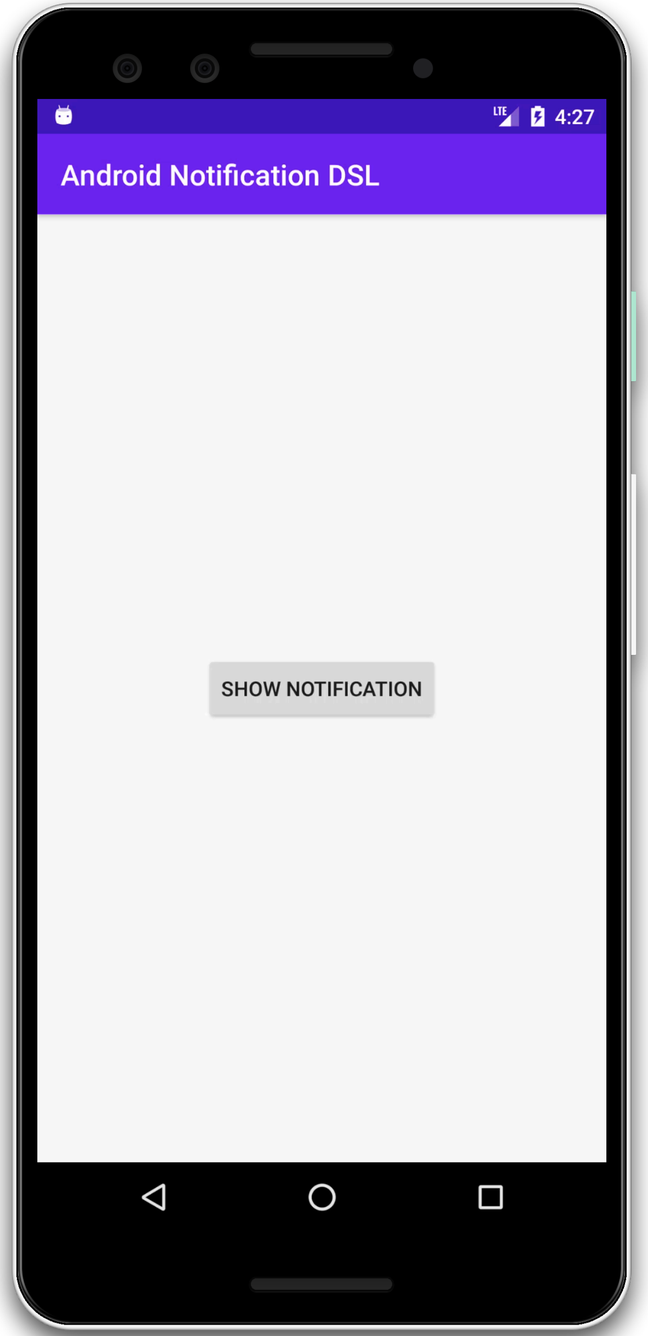
ФРАГМЕНТИРОВАННОСТЬ УВЕДОМЛЕНИЙ

```
val summary = NotificationCompat.Builder(context, CHANNEL_HIGH)
    .setGroupSummary(true)
    .setGroup(GROUP_KEY)
    .setContentTitle(context.getText(R.string.notification_summary_title))
    .setContentText(context.getText(R.string.notification_summary_text))
    .setSmallIcon(R.drawable.ic_android_white_24dp)
    .setPriority(NotificationCompat.PRIORITY_HIGH)
    .build()

val notification1 = NotificationCompat.Builder(context, CHANNEL_LOW)
    .setContentTitle(context.getText(R.string.notification_1_title))
    .setSmallIcon(R.drawable.ic_android_white_24dp)
    .setGroup(GROUP_KEY)
    .setPriority(NotificationCompat.PRIORITY_LOW)
    .build()

val notification2 = NotificationCompat.Builder(context, CHANNEL_LOW)
    .setContentTitle(context.getText(R.string.notification_2_title))
    .setSmallIcon(R.drawable.ic_android_white_24dp)
    .setPriority(NotificationCompat.PRIORITY_LOW)
    .setGroup(GROUP_KEY)
    .build()
```

ФРАГМЕНТИРОВАННОСТЬ УВЕДОМЛЕНИЙ

ANDROID 6 (ПОПЫТКА 2)

- Нет поддержки групп
- Нет поддержки каналов уведомлений
- Приоритет не показывает уведомление сразу



ФРАГМЕНТИРОВАННОСТЬ УВЕДОМЛЕНИЙ

```
val summary = NotificationCompat.Builder(context, CHANNEL_HIGH)
    .setGroupSummary(true)
    .setGroup(GROUP_KEY)
    .setContentTitle(context.getText(R.string.notification_summary_title))
    .setContentText(context.getText(R.string.notification_summary_text))
    .setSmallIcon(R.drawable.ic_android_white_24dp)
    .setPriority(NotificationCompat.PRIORITY_HIGH)
    .build()

val notification1 = NotificationCompat.Builder(context, CHANNEL_LOW)
    .setContentTitle(context.getText(R.string.notification_1_title))
    .setSmallIcon(R.drawable.ic_android_white_24dp)
    .setGroup(GROUP_KEY)
    .setPriority(NotificationCompat.PRIORITY_LOW)
    .build()

val notification2 = NotificationCompat.Builder(context, CHANNEL_LOW)
    .setContentTitle(context.getText(R.string.notification_2_title))
    .setSmallIcon(R.drawable.ic_android_white_24dp)
    .setPriority(NotificationCompat.PRIORITY_LOW)
    .setGroup(GROUP_KEY)
    .build()
```

ФРАГМЕНТИРОВАННОСТЬ УВЕДОМЛЕНИЙ

```
val summary = NotificationCompat.Builder(context, CHANNEL_HIGH)
    .setContentTitle(context.getText(R.string.notification_summary_title))
    .setContentText(context.getText(R.string.notification_summary_text))
    .setSmallIcon(R.drawable.ic_android_white_24dp)
    .setPriority(NotificationCompat.PRIORITY_HIGH)
    .build()

val notification1 = NotificationCompat.Builder(context, CHANNEL_LOW)
    .setContentTitle(context.getText(R.string.notification_1_title))
    .setSmallIcon(R.drawable.ic_android_white_24dp)
    .setPriority(NotificationCompat.PRIORITY_LOW)
    .build()

val notification2 = NotificationCompat.Builder(context, CHANNEL_LOW)
    .setContentTitle(context.getText(R.string.notification_2_title))
    .setSmallIcon(R.drawable.ic_android_white_24dp)
    .setPriority(NotificationCompat.PRIORITY_LOW)
    .build()
```

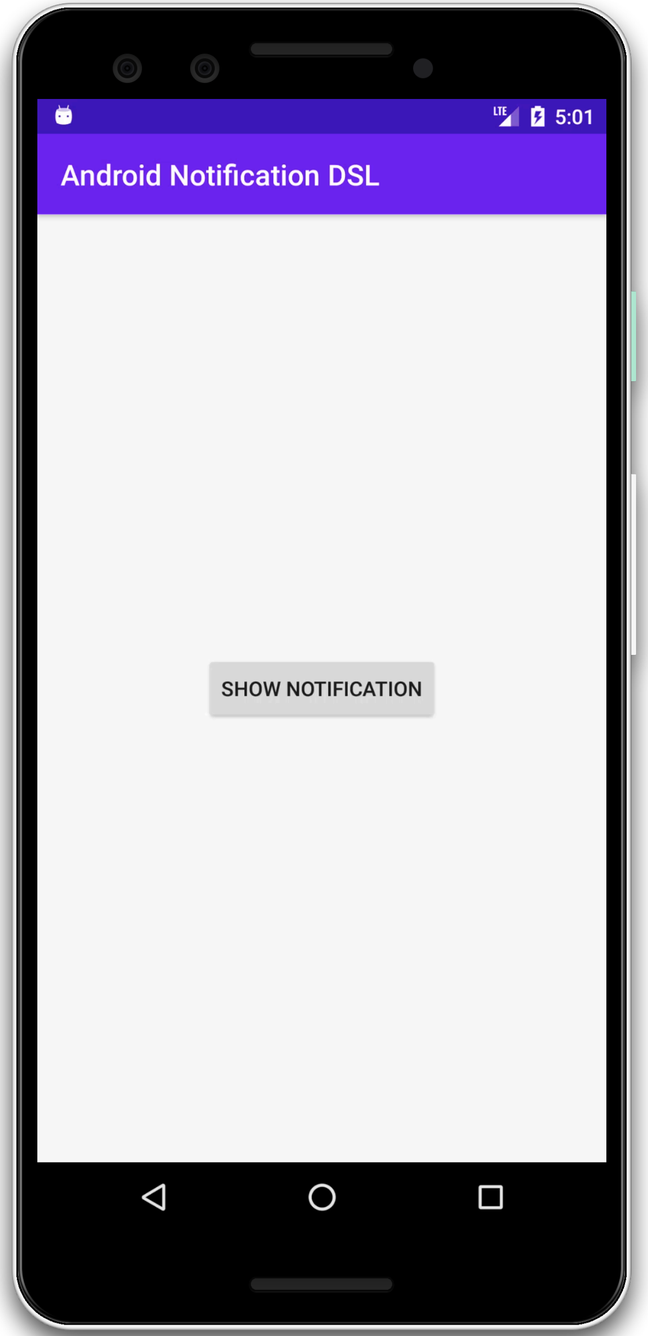
ФРАГМЕНТИРОВАННОСТЬ УВЕДОМЛЕНИЙ

```
val summary = NotificationCompat.Builder(context, CHANNEL_HIGH)
    .setContentTitle(context.getText(R.string.notification_summary_title))
    .setContentText(context.getText(R.string.notification_summary_text))
    .setSmallIcon(R.drawable.ic_android_white_24dp)
    .setPriority(NotificationCompat.PRIORITY_HIGH)
    .setDefaults(NotificationCompat.DEFAULT_ALL)
    .build()

val notification1 = NotificationCompat.Builder(context, CHANNEL_LOW)
    .setContentTitle(context.getText(R.string.notification_1_title))
    .setSmallIcon(R.drawable.ic_android_white_24dp)
    .setPriority(NotificationCompat.PRIORITY_LOW)
    .build()

val notification2 = NotificationCompat.Builder(context, CHANNEL_LOW)
    .setContentTitle(context.getText(R.string.notification_2_title))
    .setSmallIcon(R.drawable.ic_android_white_24dp)
    .setPriority(NotificationCompat.PRIORITY_LOW)
    .build()
```

ФРАГМЕНТИРОВАННОСТЬ УВЕДОМЛЕНИЙ

ANDROID 6 (ПОПЫТКА 3)

- Нет поддержки групп
- Нет поддержки каналов уведомлений
- Высокий приоритет не влияет на показ уведомления сразу
- Задание любого из поддерживаемых *defaults* показывает уведомление сразу



ПО ИТОГУ

```
if (Build.VERSION.SDK_INT ≥ Build.VERSION_CODES.N) {
    val summary = NotificationCompat.Builder(context, NOTIFICATION_CHANNEL_DEFAULT)
        .setContentTitle(context.getText(R.string.notification_summary_title))
        .setContentText(context.getText(R.string.notification_summary_text))
        .setSmallIcon(R.drawable.ic_android_white_24dp)
        .setPriority(NotificationCompat.PRIORITY_MAX)
        .setDefaults(NotificationCompat.DEFAULT_ALL)
        .setGroup(GROUP_KEY)
        .setGroupSummary(true)
        .build()
}

val notification1Builder = NotificationCompat.Builder(context, NOTIFICATION_CHANNEL_LOW)
    .setContentTitle(context.getText(R.string.notification_1_title))
    .setSmallIcon(R.drawable.ic_android_white_24dp)
    .setPriority(NotificationCompat.PRIORITY_LOW)
if (Build.VERSION.SDK_INT ≥ Build.VERSION_CODES.N) {
    notification1Builder.setGroup(GROUP_KEY)
}
val notification1 = notification1Builder.build()

val notification2Builder = NotificationCompat.Builder(context, NOTIFICATION_CHANNEL_LOW)
    .setContentTitle(context.getText(R.string.notification_2_title))
    .setSmallIcon(R.drawable.ic_android_white_24dp)
    .setPriority(NotificationCompat.PRIORITY_LOW)
if (Build.VERSION.SDK_INT ≥ Build.VERSION_CODES.N) {
    notification2Builder.setGroup(GROUP_KEY)
}
val notification2 = notification2Builder.build()
```

NOTIFICATION COMPAT

\+ Безопасный вызов нового API на старых версиях Android

\- Игнорирование отсутствия возможностей в старых версиях ОС

\- Отсутствие Compat API для работы с каналами

// Появится в AndroidX Core 1.5.0. Текущая стабильная версия - 1.3.2

\- Отсутсвие KTX расширений

```
contentText = message.message
autoCancel = true
contentTitle = message.from
priority = NotificationPriority.HIGH
whenTime = message.date.time
largeIcon = BitmapFactory.decodeFile(mes
category = NotificationCategory.MESSAGE
persons += message.personUri
onlyAlertOnce = true
actions { this: Actions
    action( title: "Mark as Read", markA
        contextual = true
        semanticAction = SemanticAct
        showsUserInterface = false
    }
}
bigTextStyle { this: BigTextStyleBuilder
    text = message.message
}
wearable { this: WearableExtender
    contentIntentAvailableOffli
}
```

4. ANDROID NOTIFICATION DSL

tiny.cc/andsl

ANDROID NOTIFICATION DSL

- **Core**
  DSL поверх AndroidX NotificationCompat
- **Extensions**
  Надстройки поверх NotificationManager, специфичные DSL для популярных типов уведомлений
- **Media**
  DSL поверх AndroidX Media

NOTIFICATION DSL

```
notification(context, CHANNEL_DEFAULT, R.drawable.ic_android_white_24dp) {
    contentTitle = "Notification title"
    actions {
        action(
            title = "Edit",
            intent = context.activityPendingIntent(ACTION_EDIT_REQUEST_CODE, MainActivity::class),
            icon = R.drawable.ic_baseline_edit_24
        )
        action(
            title = "View",
            intent = context.activityPendingIntent<MainActivity>(ACTION_VIEW_REQUEST_CODE),
            icon = R.drawable.ic_baseline_remove_red_eye_24
        )
    }
}
```

ПОЛНЫЙ ОБВЕС

```
NotificationCompat.Builder(context, CHANNEL_DEFAULT)
    .setContentText(message.message)
    .addPerson(message.personUri)
    .setAutoCancel(true)
    .setContentTitle(message.from)
    .setPriority(NotificationCompat.PRIORITY_HIGH)
    .setDefaults(NotificationCompat.DEFAULT_ALL)
    .setOnlyAlertOnce(true)
    .setWhen(message.date.time)
    .setVisibility(NotificationCompat.VISIBILITY_PRIVATE)
    .setLargeIcon(BitmapFactory.decodeFile(message.personIconPath))
    .setCategory(NotificationCompat.CATEGORY_MESSAGE)
    .addAction(
        NotificationCompat.Action.Builder(null, "Mark as Read", markAsReadIntent)
            .setContextual(false)
            .setSemanticAction(NotificationCompat.Action.SEMANTIC_ACTION_MARK_AS_READ)
            .setShowsUserInterface(false)
            .build()
    )
    .setStyle(
        NotificationCompat.BigTextStyle()
            .bigText(message.message)
    )
    .extend(
        NotificationCompat.WearableExtender()
            .setContentIntentAvailableOffline(false)
    )
    .build()
```

ПОЛНЫЙ ОБВЕС

```
notification(context, CHANNEL_DEFAULT, R.drawable.ic_android_white_24dp) {
    contentText = message.message
    autoCancel = true
    contentTitle = message.from
    priority = NotificationPriority.HIGH
    whenTime = message.date.time
    largeIcon = BitmapFactory.decodeFile(message.personIconPath)
    category = NotificationCategory.MESSAGE
    persons += message.personUri
    onlyAlertOnce = true
    actions {
        action("Mark as Read", markAsReadIntent) {
            contextual = true
            semanticAction = SemanticAction.MARK_AS_READ
            showsUserInterface = false
        }
    }
    bigTextStyle {
        text = message.message
    }
    wearable {
        contentIntentAvailableOffline = false
    }
}
```

PICTURE NOTIFICATION DSL

```
notification(context, CHANNEL_DEFAULT, R.drawable.ic_android_white_24dp) {
    contentTitle = "Collapsed"
    contentText = "Sample notification"
    largeIcon = R.drawable.sea_collapsed.asBitmap(context.resources)

    bigPictureStyle {
        picture(R.drawable.sea_expanded_big.asBitmap(context.resources))
        largeIcon(null)
        contentTitle(null)
        summaryText("Summary text")
    }
}
```

PICTURE NOTIFICATION DSL

```
bigPictureNotification(context, CHANNEL_DEFAULT, R.drawable.ic_android_white_24dp) {
    title = "Collapsed"
    text = "Sample notification"
    largeIcon = R.drawable.sea_collapsed.asBitmap(context.resources)

    expanded {
        bigPicture = R.drawable.sea_expanded_big.asBitmap(context.resources)
        largeIcon = null
        title = "Expanded"
        text = "Summary text"
    }
}
```

PROGRESS DSL

```
progressNotification(context, CHANNEL_DEFAULT, R.drawable.ic_android_white_24dp) {
    title = "Downloading..."
    progressText = "6 seconds left"
    indeterminated = false

    progress {
        current = 4
        max = 10
    }

    actions {
        action(
            title = "Cancel",
            intent = context.activityPendingIntent(1, MainActivity::class)
        )
    }
}
```

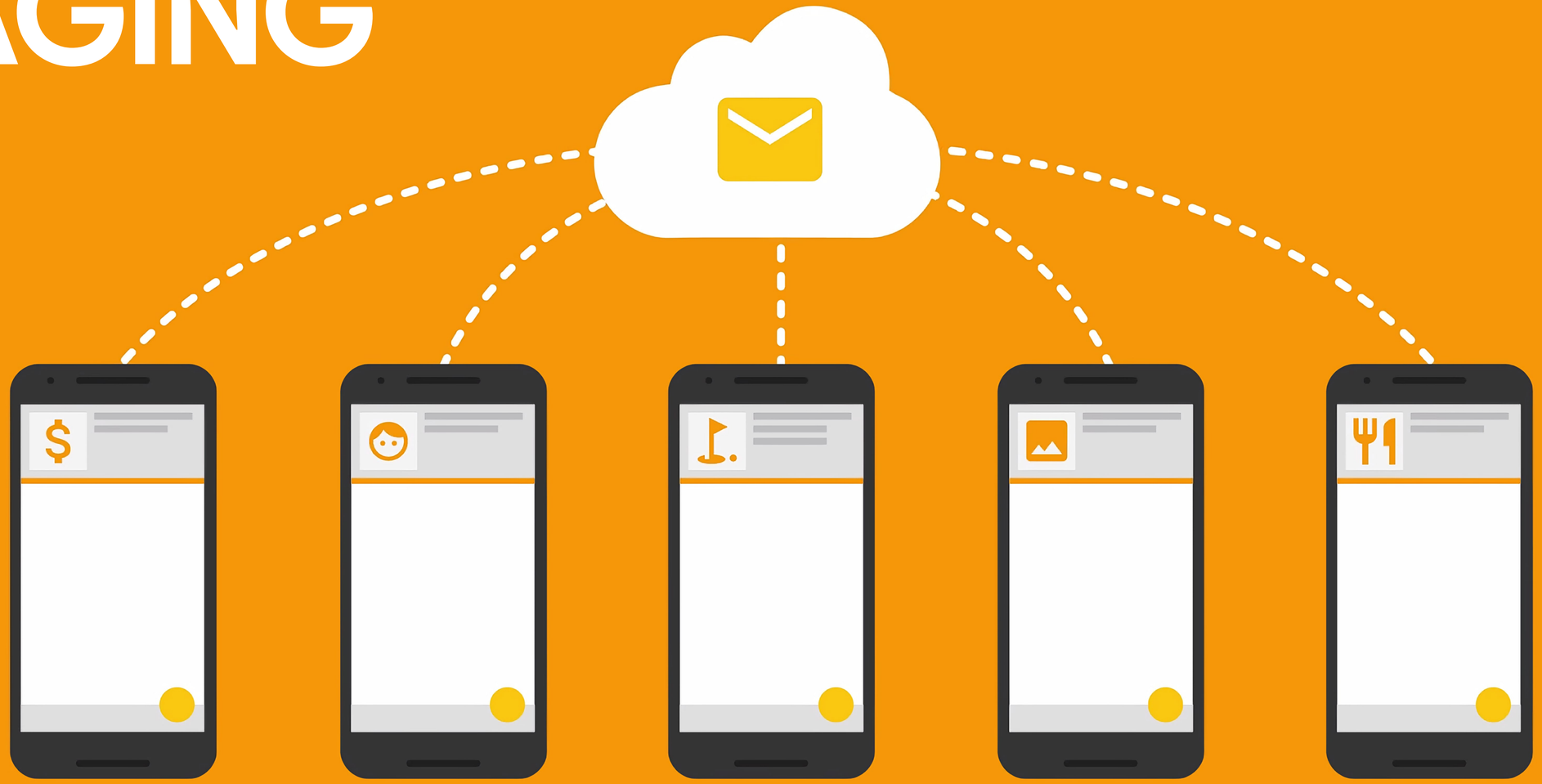
NOTIFICATION GROUP DSL

```
notificationsGroup(context, groupKey = GROUP_KEY, channelId = CHANNEL) {
    summary(SUMMARY_NOTIFICATION_ID, smallIcon = R.drawable.ic_android_white_24dp) {
        contentTitle(R.string.notification_summary_title)
        contentText(R.string.notification_summary_text)
    }

    notification(NOTIFICATION_1_ID, smallIcon = R.drawable.ic_android_white_24dp) {
        contentTitle(R.string.notification_1_title)
    }

    notification(NOTIFICATION_2_ID, smallIcon = R.drawable.ic_android_white_24dp) {
        contentTitle(R.string.notification_2_title)
    }
}
```

NOTIFICATION CHANNELS DSL

```
createNotificationChannels(context) {
    channel(CHANNEL_DEFAULT, "Default", importance = IMPORTANCE_DEFAULT)

    group(CHANNEL_GROUP_1, "Android Broadcast") {
        channel(NOTIFICATION_CHANNEL_1, "Channel 1", importance = IMPORTANCE_HIGH)
        channel(NOTIFICATION_CHANNEL_2, "Channel 2")
    }

    group(CHANNEL_GROUP_2, "Mobius") {
        channel(NOTIFICATION_CHANNEL_3, "Channel 3", importance = IMPORTANCE_LOW)
    }
}
```

5. FIREBASE CLOUD MESSAGING



ТИПЫ СООБЩЕНИЙ FCM

- Notification
- Data

FCM DATA MESSAGE

```
{
  "message": {
    "token": "...",
    "data": {
      "conference": "Mobius 2020 Online",
      "speaker": "Kirill Rozov",
      "company": "Android Broadcast"
    }
  }
}
```

FCM NOTIFICATION

```
{
  "message":{
    "token":"...",
    "notification":{
      "title":"Mobius 2020 Online",
      "body":"Kirill Rozov/Android Broadcast"
    }
  }
}
```

FCM NOTIFICATION

```
{
  "message": {
    "topic": "Announcement",
    "notification": {
      "title":"Mobius 2020 Online",
      "body":"Kirill Rozov/Android Broadcast"
    },
    "android": {
      "notification": {
        "icon": "android_broadcast_logo_small",
        "color": "#6FAA49"
      }
    }
  }
}
```

FCM NOTIFICATION

```
"android" : {
  "collapse_key": string,
  "priority": enum,
  "ttl": string,
  "restricted_package_name": string,
  "data": {
    string: string,
    ...
  },
  "notification": object,
  "fcm_options": object,
  "direct_boot_ok": boolean
}
```

FCM NOTIFICATION

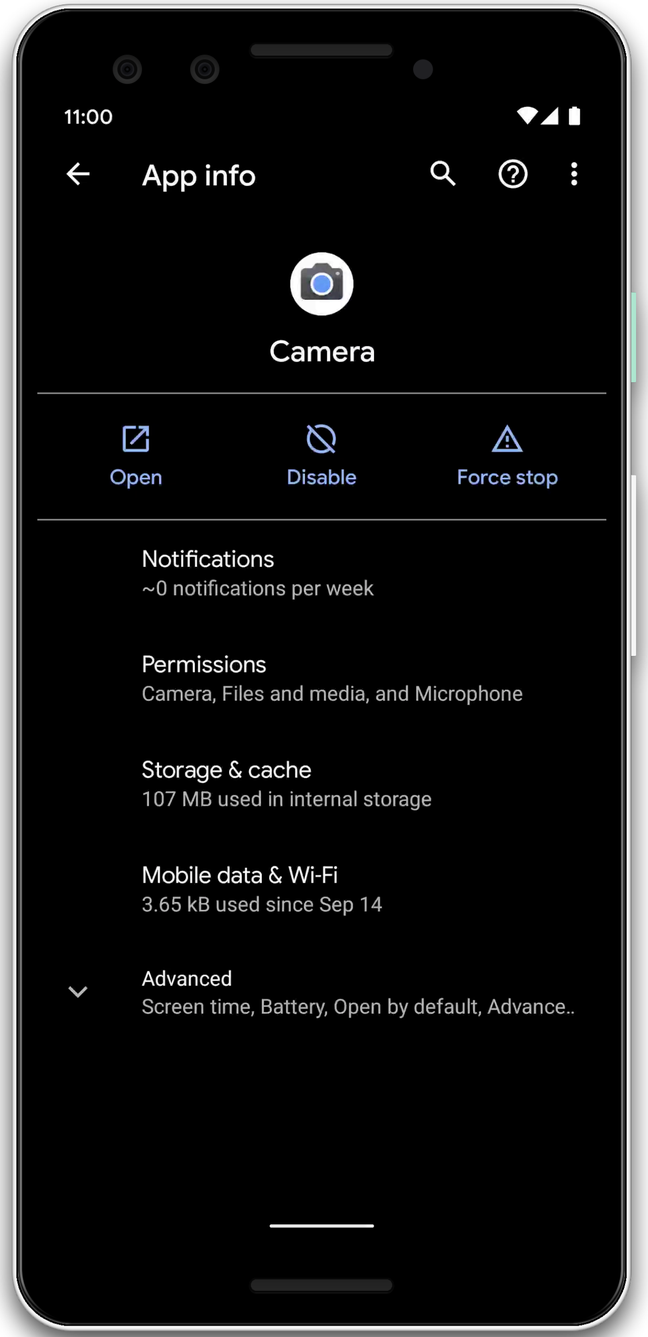
```
"android": {
   "title": string,
   "body": string,
   "icon": string,
   "color": string,
   "sound": string,
   "tag": string,
   "click_action": string,
   "body_loc_key": string,
   "body_loc_args": [string],
   "title_loc_key": string,
   "title_loc_args": [string],
   "channel_id": string,
   "ticker": string,
   “sticky": boolean,
   "event_time": string,
   "local_only": boolean,
   "notification_priority": enum,
   "default_sound": boolean,
   "default_vibrate_timings": boolean,
   "default_light_settings": boolean,
   "vibrate_timings": [string],
   "visibility": enum,
   "notification_count": integer,
   "light_settings": object,
   "image": string

}
```

firebase.google.com/docs/reference/fcm/rest/v1/projects.messages#androidconfig

FORCE STOP

РЫЧАГ ОТКЛЮЧЕНИЯ ПРИЛОЖЕНИЯ

- FCM data больше не придёт
- Фоновая работы выполняться не станет
- На распространяется на уведомления через FCM



ИТОГИ

- NotificationCompat не решает наших проблем
- Возможности уведомлений в Android огромны. Почувствуй Силу!
- Эффективная организация уведомлений позволяют увеличить количество сессий в вашем приложении
- FCM может позволить убрать простейший уведомления и будить ваше приложение
- Android Notification DSL призвана упростить добавление богатых уведомлений

СПАСИБО ЗА ВНИМАНИЕ

kirill_rozov

krlrozov