ТЕХНОТРЕК

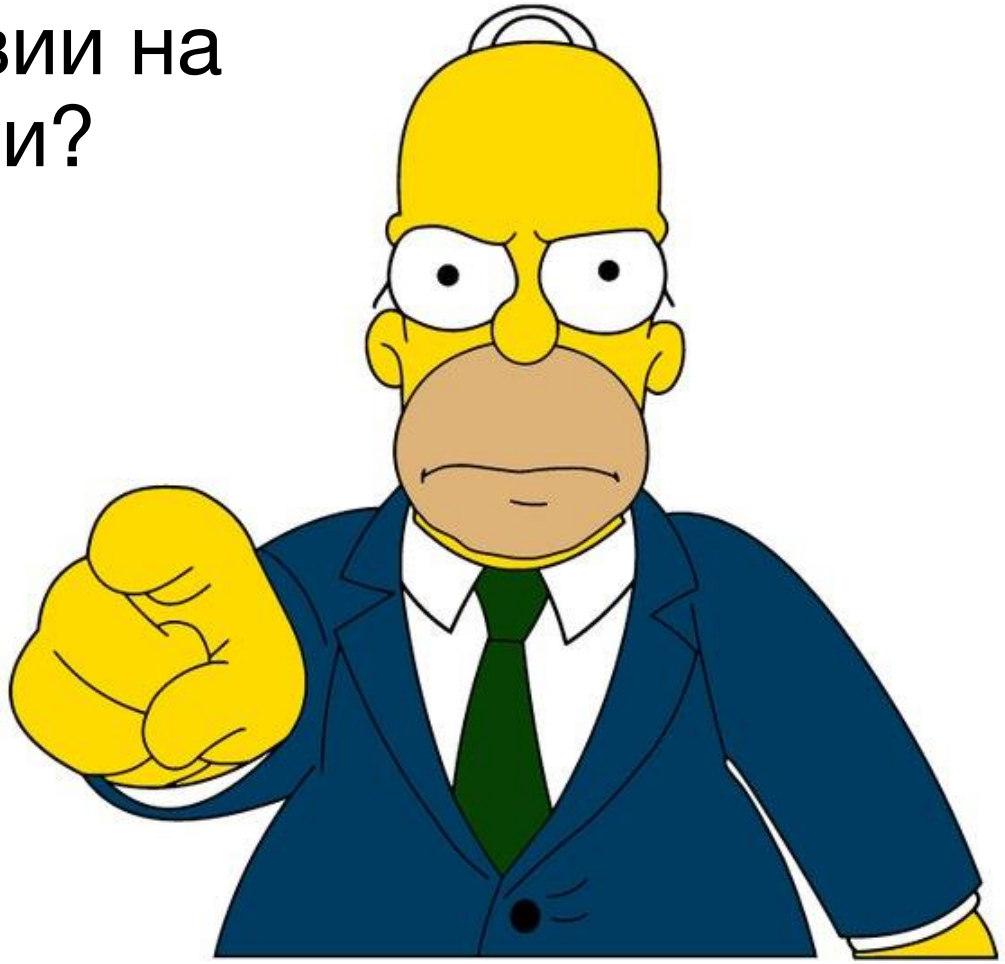
Занятие №10

Разработка приложений на Android

Кирилл Филимонов
Юрий Береза

Напоминание

А ты отметился о присутствии на занятии?

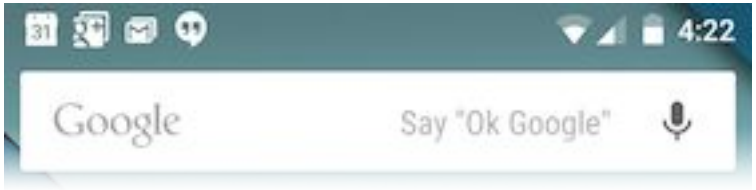


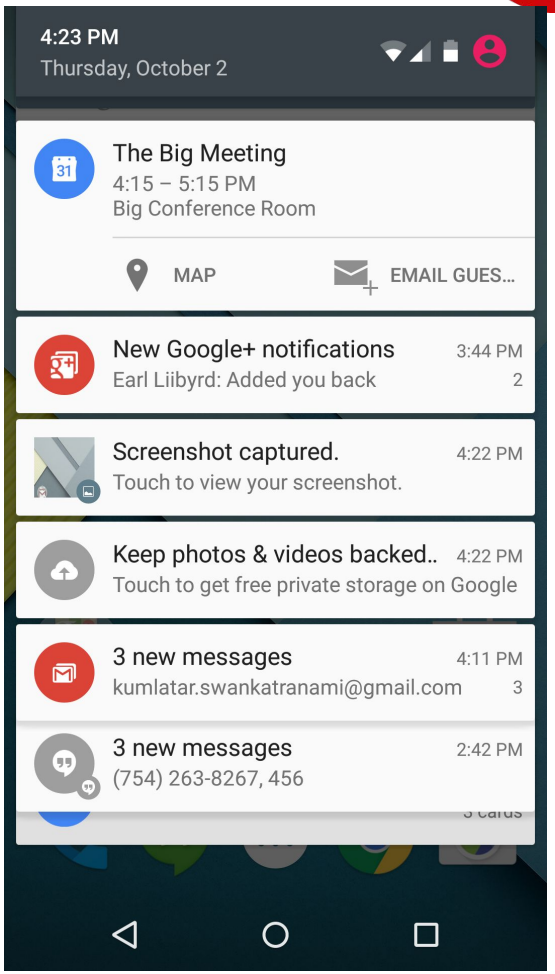
Agenda

- Нотификации
- BroadcastReceivers
- AndroidWidgets
- Подготовка приложения к публикации

Нотификации

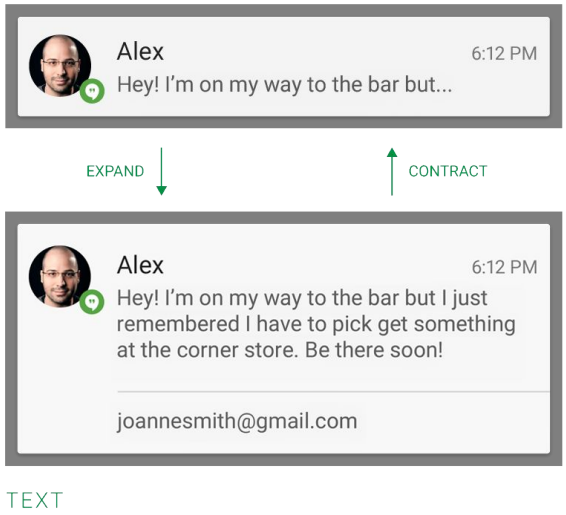
- Информация о событиях без открытия приложения
- События зависящие от времени
- Нотификации должны быть ожидаемы
- Не надо их кидать часто
- Не надо кидать их много



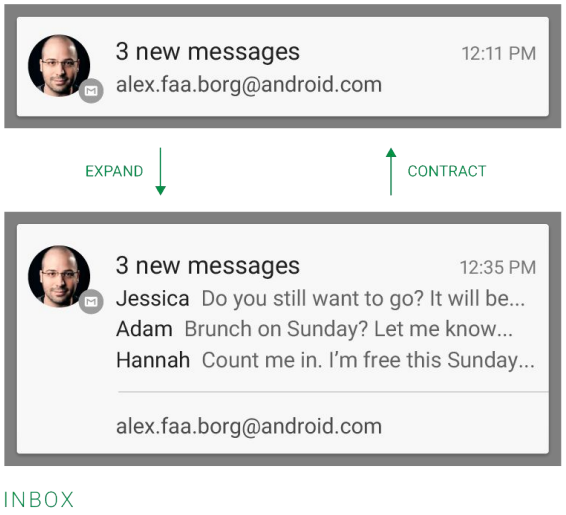


Нотификации

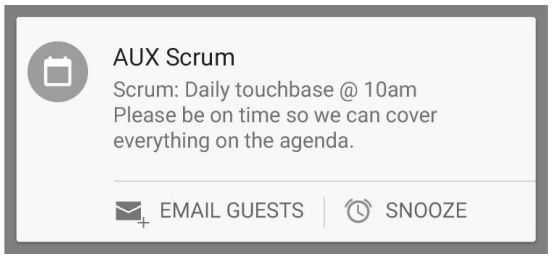
- Можно группировать сообщения
- Можно взаимодействовать с нотификацией
- Можно отображать расширенную информацию

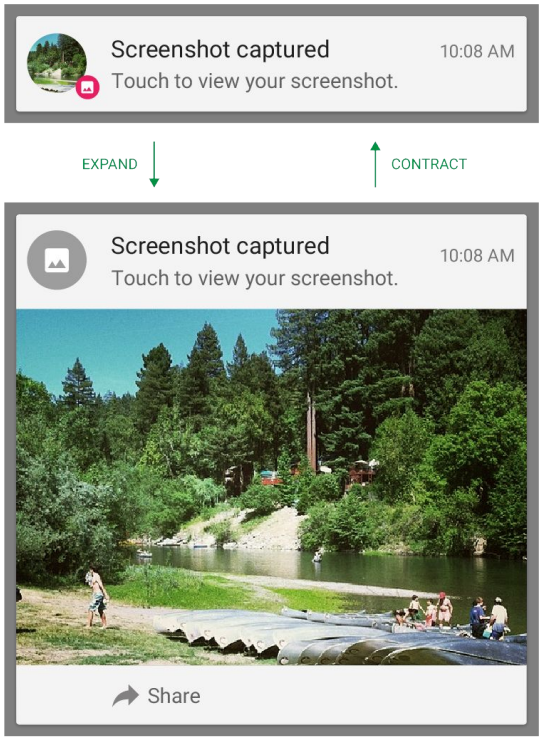


TEXT



INBOX





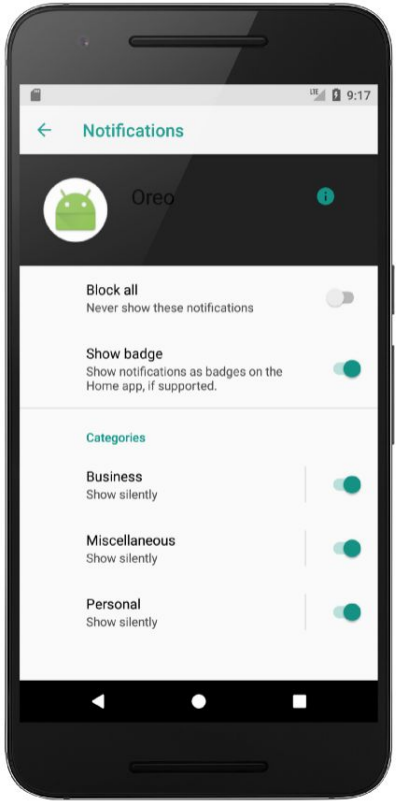
IMAGE

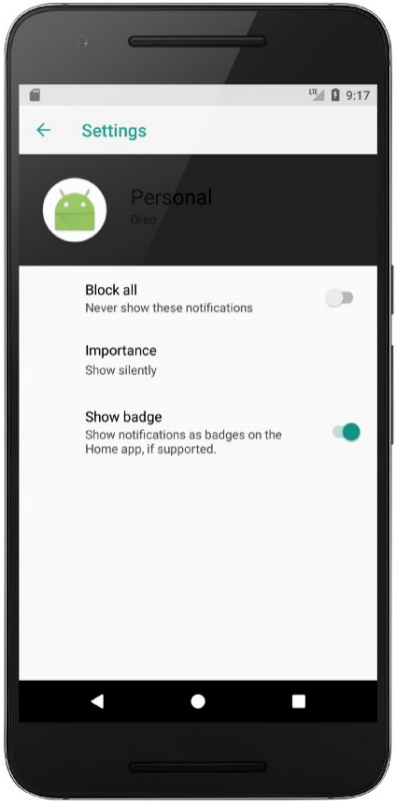
Нотификации

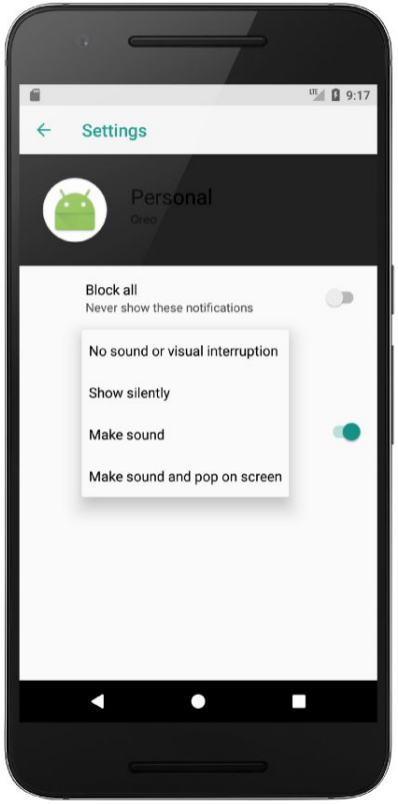
- Бываю перекрывающими (Head-up notification)
- Используются для критических вещей
- Могут отображаться на lock экране
- Могут иметь приватную и публичную часть

Каналы

Требуется targetSdk = 26 (Android O)







Рекомендации и требования

- По клику на уведомление, пользователю должен открываться соответствующий экран приложения. В некоторых случаях достаточно, чтобы по клику уведомление просто убиралось.
- Указание времени события в уведомлении, также является хорошим подходом.
- Рекомендуется схожие события складывать в одно уведомление, а не отображать на каждое событие своё.
- Всегда убирать из статус-бара уведомления, с которыми пользователь уже ознакомился и произвел соответствующие действия.
- Показывать маленькое превью уведомления при его создании в свёрнутом статус-баре
- Позволять пользователю отключать уведомления в настройках приложения.
- Использовать иконки, обозначающие принадлежность уведомления определённому приложению. Иконки делать монохромными
- В случае, если событие требует непосредственной реакции пользователя — вместо уведомлений использовать диалоги.

Простая нотификация

- Создаем билдер
- Заполняем параметры нотификации
- Запускаем ее

```
Notification.Builder builder = new Notification.Builder(context);

builder.setContentIntent(contentIntent)
        .setSmallIcon(R.drawable.ic_launcher_cat)
        // большая картинка
        .setLargeIcon(BitmapFactory.decodeResource(res, R.drawable.hungrycat))
        //.setTicker(res.getString(R.string.warning)) // текст в строке состояния
        .setTicker("Последнее китайское предупреждение!")
        .setWhen(System.currentTimeMillis())
        .setAutoCancel(true)
        //.setContentTitle(res.getString(R.string.notifytitle)) // Заголовок уведомления
        .setContentTitle("Напоминание")
        //.setContentText(res.getString(R.string.notifytext))
        .setContentText("Пора покормить кота"); // Текст уведомленимя

// Notification notification = builder.getNotification(); // до API 16
Notification notification = builder.build();

NotificationManager notificationManager = (NotificationManager) context
        .getSystemService(Context.NOTIFICATION_SERVICE);
notificationManager.notify(NOTIFY_ID, notification);
```

Какие есть возможности

- Группировать
- Звук
- Вибрация
- Управлять светодиодом
- Можно обновляет состояние по ID
- Можно удалять
- Можно создавать свой layout для нотификации
- Можно задать иконку и картинку для отображения
- Можно задавать приоритет
- Можно задавать категорию
- Можно задать персону (контакт) по URI

Broadcast

- В приложении вы можете рассылать и принимать широковещательные сообщения
- Рассылать можно собщения двумя способами
  - нормальные сообщения о намерениях (Normal broadcasts) — посылаемые вызовом метода context.sendBroadcast() и являющиеся полностью асинхронными
  - порядковые сообщения о намерениях (Ordered broadcasts), которые посылаются методом context.sendOrderedBroadcast(). Эти сообщения посылаются одному получателю за один раз.

Broadcast

- Android использует широковещательные сообщения для системных событий
  - таких как уровень зарядки батареи
  - сетевые подключения
  - входящие звонки
  - изменения часового пояса
  - состояние подключения данных
  - входящие сообщения SMS
  - обращения по телефону.
- Другие приложения тоже используют сообщения для уведомления всех о каких-либо событиях

BroadcastReceiver

- Нужен для перехвата широковещательных запросов
- Можно регистрировать через манифест
- Можно регистрировать через код
- Бывают sticky – то есть получают уведомление о событии даже если оно произошло до регистрации ресивера
- Реализуют одну функцию void onReceive(Context context, Intent intent)
- Выполняется в основном потоке

Пример рассылки

```
void    SomeFunc1() {
  static final String TEST_BR = "ru.mail.samle.TEST_BR";
  Intent intent = new Intent(TEST_BR);
  intent.putExtra("data1", var1);
  intent.putExtra("data2", var2);
  sendBroadcast(intent);
 }

void    SomeFunc2() {
  static final String TEST_BR = "ru.mail.samle.TEST_BR";
  String requiredPermission = "ru.mail.samle.MY_BR_PERMISSION”;
  Intent intent = new Intent(TEST_BR);
  intent.putExtra("data1", var1);
  intent.putExtra("data2", var2);
  sendOrderedBroadcast(intent, requiredPermission);
 }
```

Пример ресивера (Манифест)

```
<receiver
    android:name=".MessageReceiver"
    android:enabled="true"
    android:exported="true">
    <intent-filter>
        <action android:name="ru.mail.samle.TEST_BR" />
    </intent-filter>
</receiver>
```

Deprecated начиная с API 26

Пример ресивера

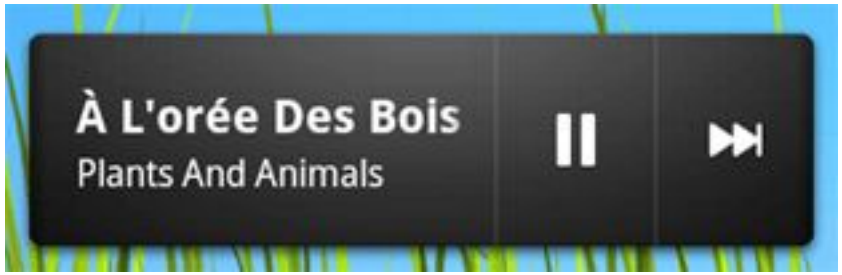
```
public class MessageReceiver extends BroadcastReceiver {
  public MessageReceiver() {
  }

  @Override
  public void onReceive(Context context, Intent intent) {
    Toast.makeText(context, "Обнаружено сообщение: " +
       intent.getStringExtra("ru.mail.samle.TEST_BR"),
            Toast.LENGTH_LONG).show();
  }
}
```

Пример регистрации ресивера через код

```
@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    // создаем BroadcastReceiver
    br = new MessageReceiver();
    // создаем фильтр для BroadcastReceiver
    IntentFilter intFilt = new IntentFilter(TEST_BR);
    // регистрируем (включаем) BroadcastReceiver
    registerReceiver(br, intFilt);
}

@Override
protected void onDestroy() {
    super.onDestroy();
    // дерегистрируем (выключаем) BroadcastReceiver
    unregisterReceiver(br);
}
```

AppWidgets

- Создание виджетов близко по своей сути к созданию Activity.
- С помощью Layout вы можете отобразить что-то на экране рабочего стола.
- Созданные вами виджеты на самом деле отображаются менеджером рабочего стола
- Основными компонентами виджета являются
  - BroadcastReceiver
  - RemoteViews
  - иногда Service.



RemoteViews

- Компонент который будет содержать ваш Layout с набором View внутри и будет передан менеджеру виджетов.
- RemoteViews не имеет отношения к обычным View в Android, и набор Layout и View внутри RemoteViews довольно ограничен.

RemoteViews

- В качестве Layout вы можете использовать FrameLayout, LinearLayout и RelativeLayout.
- В качестве View внутри Layout вы можете использовать Button, ImageButton, ImageView, ProgressBar, TextView и такие редкие как AnalogClock и Chronometer.
- Начиная с API 11 вы так же можете использовать ListView, GridView, ViewFlipper (так называемые AdapterBased Widgets), а начиная с API 16 GridLayout.
- Остальные View использовать нельзя. В том числе нельзя использовать CustomView

BroadcastReceiver

- BroadcastReceiver получает событие об обновлении виджета.
- Код будет выполнен в основном потоке, поэтому выполняться он должен за короткий промежуток времени.
- После завершения выполнения метода onReceive, BroadcastReceiver будет уничтожен вместе с контекстом.
- Для запуска сервисов используйте ApplicationContext, в противном случаем вы можете увидеть в логе ошибку Context has been leaked.
- Если вам необходимо выполнить длительную операцию создайте для этой цели Service.

Создание виджетов AndroidManifest.xml

```
<uses-feature android:name="android.software.app_widgets"
android:required="false"/>
<application …>
  <receiver
       android:name=".AppWidget"
       android:icon="@drawable/cw"
       android:label="@string/app_name">
    <intent-filter>
      <action
android:name="android.appwidget.action.APPWIDGET_UPDATE"/>
    </intent-filter>
    <meta-data
         android:name="android.appwidget.provider"
         android:resource="@xml/widget_provider"/>
  </receiver>
</application>
```

Intent-filter

- android.appwidget.action.APPWIDGET_UPDATE – Обновление виджета
- android.appwidget.action.ACTION_APPWIDGET_CONFIGURE - На самом деле это событие не будет отправлено в BroadcastReceiver. ОС лишь поймет, что необходимо запустить Activity для конфигурации вашего виджета.
- android.appwidget.action.ACTION_APPWIDGET_DELETED - Будет отправлено событие, что один из виджетов удален (на экране может храниться несколько копий виджета)
- android.appwidget.action.ACTION_APPWIDGET_DISABLED - Будет отправлено, когда последний виджет на экране будет удален
- android.appwidget.action.ACTION_APPWIDGET_ENABLED - Будет отправлено, когда виджет в первый раз добавлен на экран (первая копия)
- android.appwidget.action.APPWIDGET_OPTIONS_CHANGED - Будет отправлено, когда изменились размеры виджета (min API 16)

Meta-data

Секция <meta-data> указывает, что этот BroadcastReceiver является поставщиком виджета (android.appwidget.provider)

```
<appwidget-provider
xmlns:android="http://schemas.android.com/apk/res/android"
  android:minWidth="144dip"
  android:minHeight="72dip"
  android:initialLayout="@layout/widget_initial_layout"
  android:updatePeriodMillis="1800000"
  android:resizeMode="horizontal|vertical"
  android:configure="ru.mail.my.sample.widget.SettingsActivity"
/>
```

Meta-data

- minWidth и minHeight нужны для того чтобы менеджеры рабочего стола поняли какое минимальное количество ячеек может занять виджет.
- initialLayout используется в том случае, когда уже системе необходимо отобразить ваш виджет а событие APPWIDGET_UPDATE еще не пришло.
- updatePeriodMills указывает как часто система будет отправлять событие APPWIDGET_UPDATE. Минимальное время - 1 раз в 30 минут, или 1 800 000 милисекунд. 0 – не вызывать update
- resizeMode был добавлен лишь в API 12, и указывает как ваш виджет может изменять свой размер

AppWidgetProvider

- Существует BroadcastReceiver который используется для получения событий от менеджера виджетов - android.appwidget.AppWidgetProvider
- Создан для упрощения работы в получаемыми событиями
- В нем уже определен метод onReceive, и разработчику лишь необходимо переопределить один из методов
  - onUpdate
  - onDeleted
  - onDisabled
  - onEnabled

Код провайдера

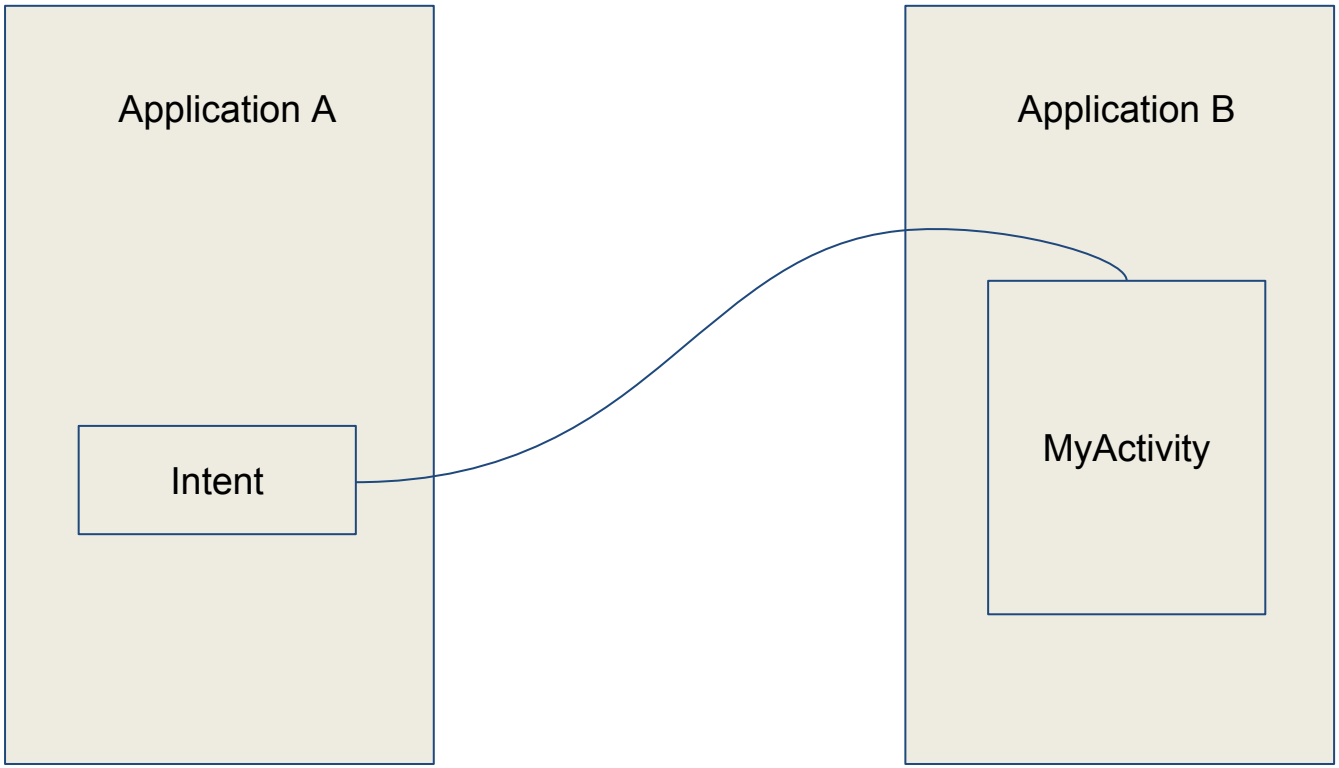
```
public class AppsWidgetProvider extends AppWidgetProvider {
    @Override
    public void onUpdate(Context context, AppWidgetManager appWidgetManager, int[]
appWidgetIds) {
        super.onUpdate(context, appWidgetManager, appWidgetIds);
        Context ctx = context.getApplicationContext();
        ctx.startService(new Intent(ctx,
WidgetUpdateService.class).setAction("ACTION_UPDATE"));
    }

    @Override
    public void onAppWidgetOptionsChanged(Context context, AppWidgetManager
appWidgetManager, int appWidgetId, Bundle newOptions) {
        super.onAppWidgetOptionsChanged(context, appWidgetManager, appWidgetId,
newOptions);
        performUpdate(context, new int[]{appWidgetId});
    }

    public static synchronized void updateWidgets(Context context) {
        AppWidgetManager appWidgetManager = AppWidgetManager.getInstance(context);
        ComponentName thisWidget = new ComponentName(context, AppsWidgetProvider.class);
        int[] appWidgetIds = appWidgetManager.getAppWidgetIds(thisWidget);
        performUpdate(context, appWidgetIds);
    }
}
```

Код провайдера 2

```
public class AppsWidgetProvider extends AppWidgetProvider {
  public static synchronized void performUpdate(Context context, int[] appWidgetIds) {
        AppWidgetManager appWidgetManager = AppWidgetManager.getInstance(context);
        for (int widgetId : appWidgetIds) {
            Bundle appWidgetOptions = appWidgetManager.getAppWidgetOptions(widgetId);
            RemoteViews remoteViews = createWidget(context, appWidgetOptions);
            if (remoteViews != null) {
                appWidgetManager.updateAppWidget(widgetId, remoteViews);
            }
        }
    }
    public static RemoteViews createWidget(Context ctx, Bundle appWidgetOptions) {
        RemoteViews views = new RemoteViews(ctx.getPackageName(),
R.layout.small_widget_layout);
        views.setTextViewText(R.id.app1_name, "Title");
        views.setImageViewResource(R.id.app1_icon, R.drawable.camera);
        views.setOnClickPendingIntent(R.id.app1, createLauncherIntent(ctx, "camera"));
    }
    static PendingIntent createLauncherIntent(Context ctx, String appname) {
        Intent i = new Intent();
        ComponentName cn = new ComponentName("ru.mail.my.sample.widget",
"ru.mail.my.sample.widget.WidgetUpdateService");
        i.setComponent(cn);
        i.setAction(WidgetUpdateService.ACTION_LAUNCH_APP);
        i.putExtra(WidgetUpdateService.EXTRA_PKG_NAME, appname);
        PendingIntent pendingIntent = PendingIntent.getService(ctx, 1, i,
PendingIntent.FLAG_UPDATE_CURRENT);
        return pendingIntent;
    }
}
```

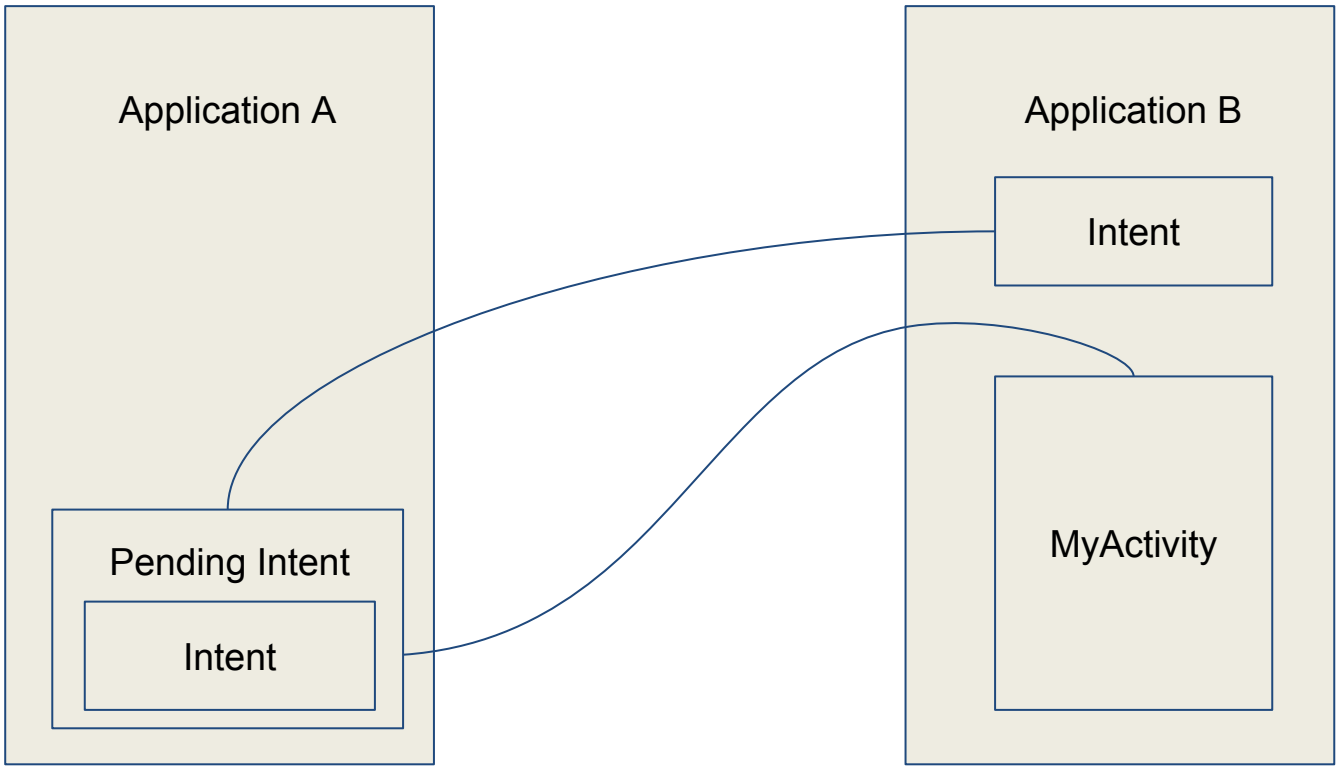
Intent

```
Intent i = new Intent("com.my", "com.my.MyActivity");
context.startActivity(i);
```



Pending Intent

```
Intent i = new Intent("com.my", "com.my.MyActivity");
PendingIntent.getActivity(context, 123, i,
                          PendingIntent.FLAG_CANCEL_CURRENT);
```



Pending Intent

Помните! Если вы хотите совершить два действия, вам необходимо создать два Intent и два PendingIntent с ДВУМЯ РАЗНЫМИ ID

```
Intent i = new Intent("com.my", "com.my.MyActivity");
Intent i2 = new Intent("com.my", "com.my.MyActivity2");

PendingIntent.getActivity(context, 123, i,
                    PendingIntent.FLAG_CANCEL_CURRENT);

PendingIntent.getActivity(context, 321, i2,
                    PendingIntent.FLAG_CANCEL_CURRENT);
```

- FLAG_NO_CREATE
- FLAG_CANCEL_CURRENT
- FLAG_UPDATE_CURRENT
- FLAG_IMMUTABLE

Разберем по порядку

- onUpdate будет вызвана, когда прийдет событие android.appwidget.action.APPWIDGET_UPDATE, получает в качестве параметров context, AppWidgetManager - класс, которому необходимо будет передать созданный RemoteViews, и appWidgetIds - список ID виджетов, которые необходимо обновить
- onAppWidgetOptionsChanged будет вызвана, если размеры виджета изменились. Параметры функции схожи с onUpdate, однако ID виджета тут один, так как за один раз может измениться размер только одного виджета
- updateWidgets объявлена как static synchronized, так как эта ф-ция будет вызвана сервисом после обновления данных и в сервисе для обновления данных используется фоновый поток. Отличительная особенность RemoteViews от обычных View - вы можете обновлять их из другого потока.
- updateWidgets отвечает за создание виджетов. Создается объект RemoteViews в ф-ции crateWidget, с указанием layout, который необходимо использовать.

Разберем по порядку

- с помощью ф-ций setTextViewText, setImageViewResource задаются ресурсы для View внутри RemoteViews.
- С помощью ф-ции views.setOnClickPendingIntent можно задать Intent, который будет вызван при нажатии на View внутрни виджета
- Создается этот PendingIntent внутри ф-ции createLauncherIntent и с помощью него, при нажатии на виджет будет запущен сервис.

AdapterBased Widgets

- Виджеты, поставщиком данных в которых выступает адаптер. Их создание немного отличается от обычных виджетов.
- Виджеты на самом деле отображаются в другом приложении, т.е. в другом процессе и нет возможности передать реальный объект через RemoteViews. Для решения этой проблемы в API 11 появились два класса RemoteViewsService и RemoteViewsFactory.
- Об этом будет в дополнительных материалах

Выводы

- Кажется что сложно
- Но на практике это все достаточно просто
- Не всем приложениям нужны виджеты
- Надо понимать что виджет виден всегда

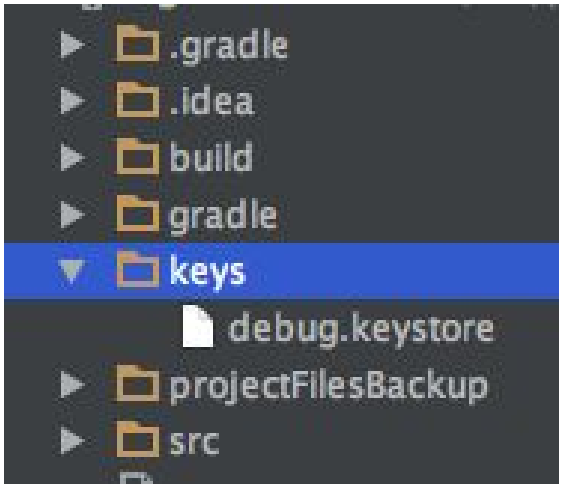
Подпись приложения

One debug key to rule them all

```
$ keytool -genkey -v -keystore debug.keystore -storepass
android -alias androiddebugkey -keypass android -keyalg RSA
-keysize 2048 -validity 10000
```

Подпись приложения

One debug key to rule them all



build.gradle

```
android {
signingConfigs {
    debug {
        storeFile file("keys/debug.keystore")
    }

}
```

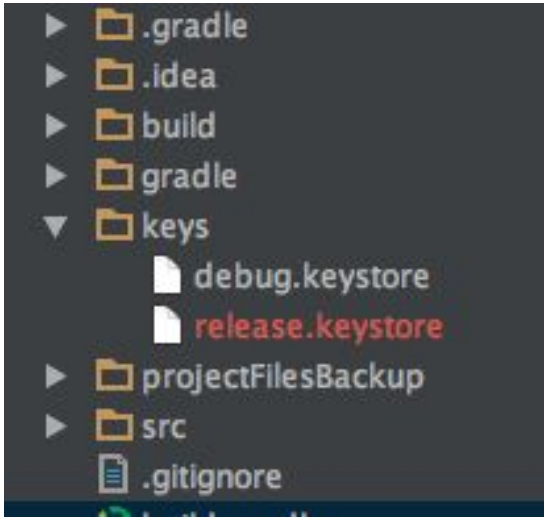
Подпись приложения

Release key

```
$ keytool -genkey -v -keystore release.keystore -storepass
mypassword -alias myapp -keypass supersecrete -keyalg RSA
-keysize 2048 -validity 10000
```

Подпись приложения

Release key



build.gradle

```
def keystorePropertiesFile =
rootProject.file("keystore.properties");
def keystoreProperties = new Properties()
keystoreProperties.load(new
FileInputStream(keystorePropertiesFile))

android {
signingConfigs {
    release {
        storeFile file("keys/release.keystore")
        storePassword keystoreProperties['STORE_PASSWORD']
        keyAlias keystoreProperties['KEY_ALIAS']
        keyPassword keystoreProperties['KEY_PASSWORD']
    }
}
```

Подпись приложения

keystore.properties

```
STORE_PASSWORD = mypassword
KEY_ALIAS = myapp
KEY_PASSWORD = supersecrete
```

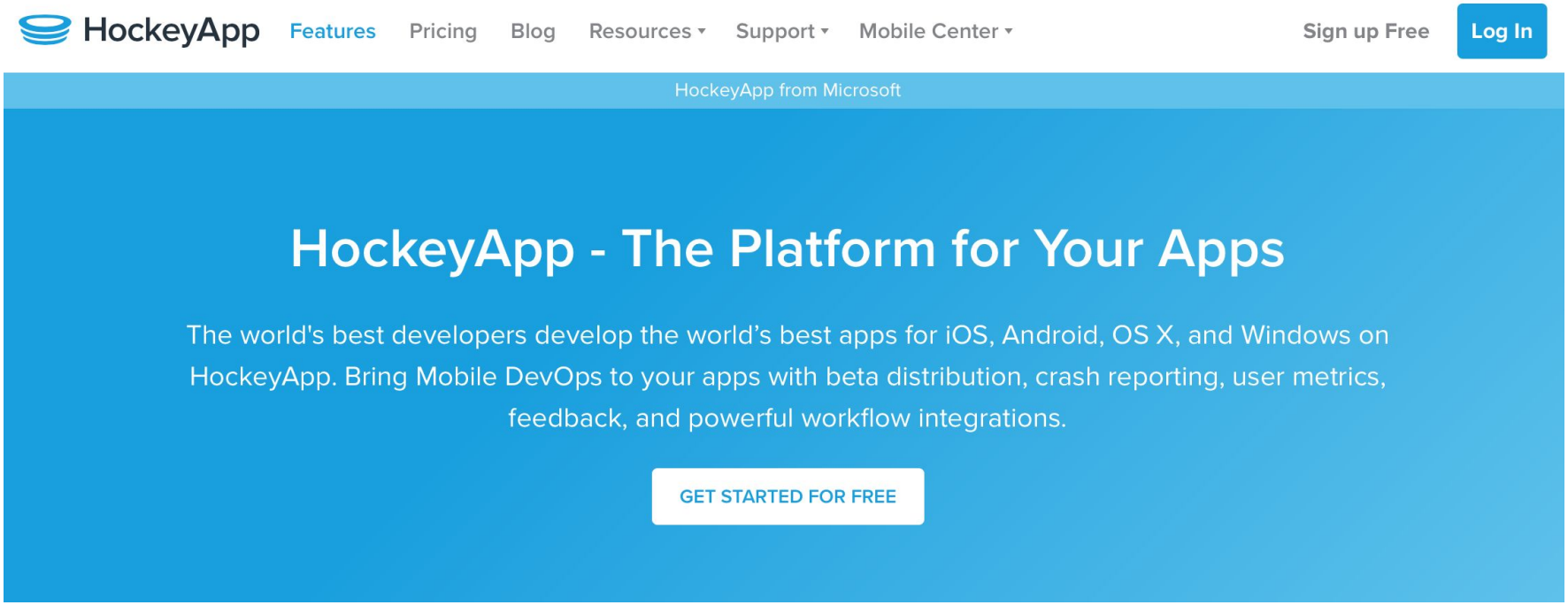
ProGuard

build.gradle

```
buildTypes {
    release {
        zipAlignEnabled true
        minifyEnabled true
        proguardFiles
            getDefaultProguardFile('proguard-android.txt'),
                'proguard-rules.pro'
    }
}
```

HockeyApp

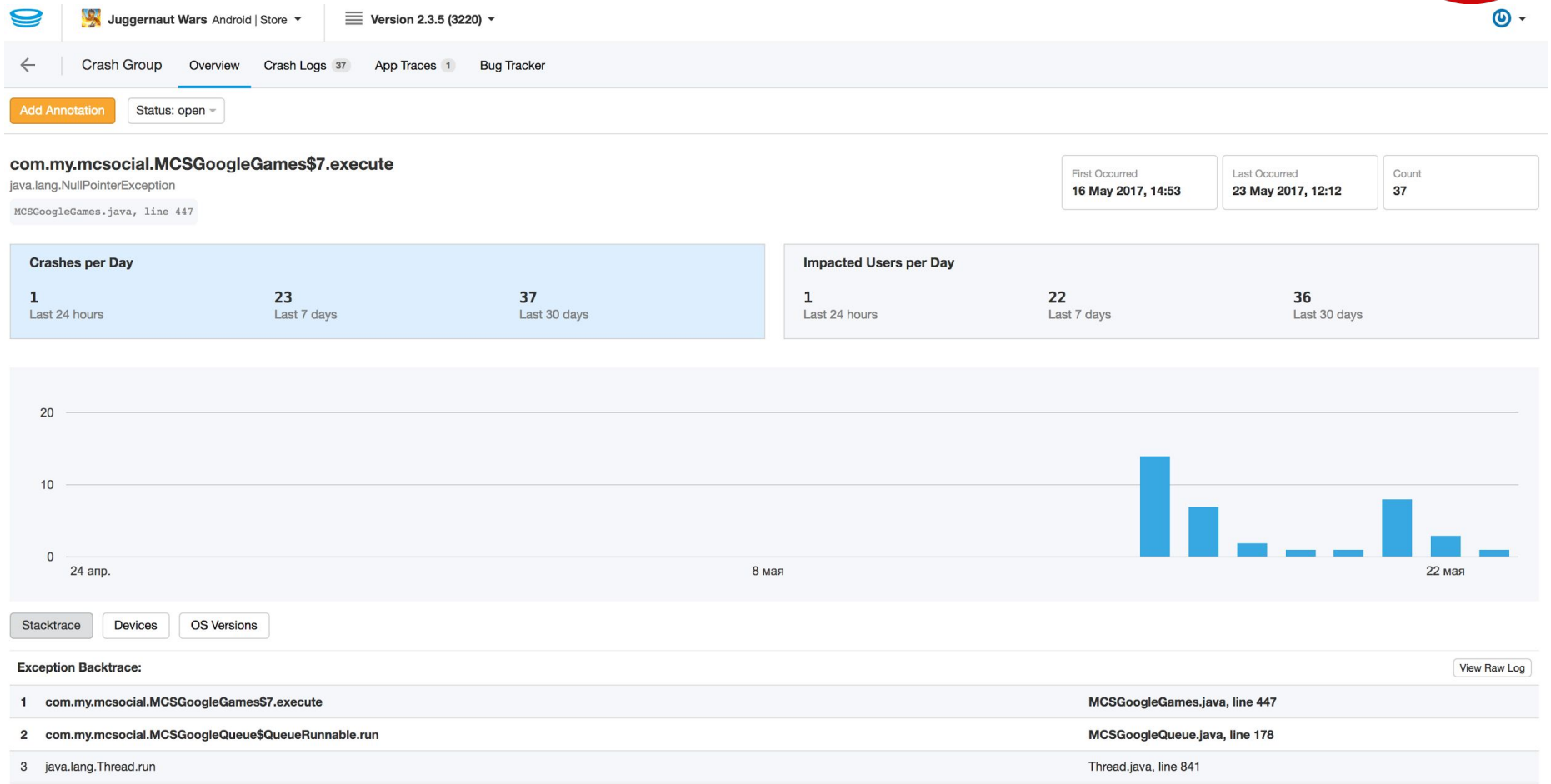
https://hockeyapp.net



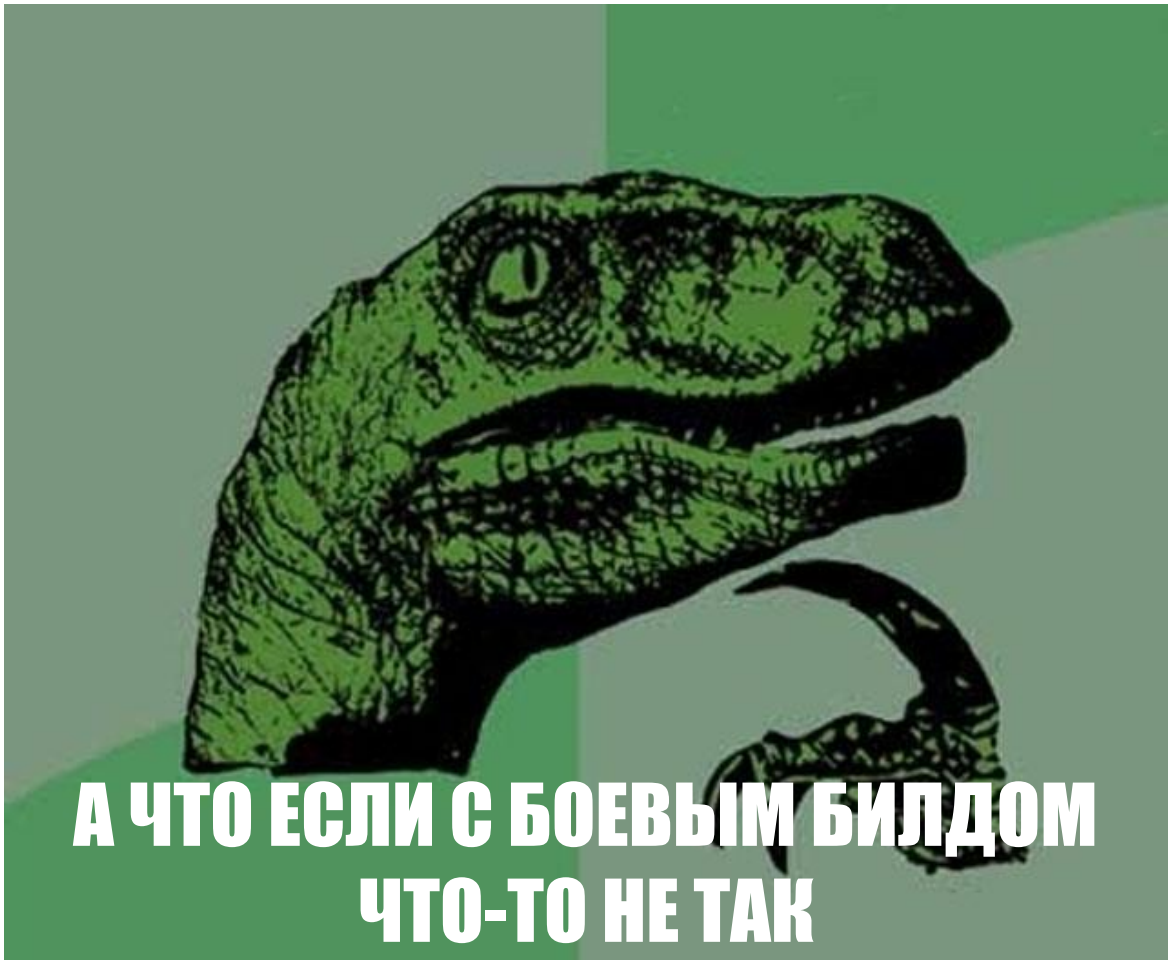
HockeyApp

Juggernaut Champions Release	Android	Enterprise	03 Feb 2017, 11:06	
Juggernaut Champions Release	iOS	Enterprise	28 Feb 2017, 12:55	
Juggernaut Champions Store	Android	Enterprise	09 Nov 2016, 15:29	
Juggernaut Champions Store	iOS	Enterprise	17 May 2017, 15:54	
Juggernaut Champions Test	Android	Beta	06 Dec 2016, 13:27	
Juggernaut Champions Test	iOS	Beta	28 Feb 2017, 12:55	
Juggernaut Wars	Android	Enterprise	18 May 2017, 13:37	
Juggernaut Wars	Android	Store	12 May 2017, 18:27	
Juggernaut Wars	iOS	Enterprise	18 May 2017, 12:46	
Juggernaut Wars	iOS	Store	05 May 2017, 19:43	
JW Test	Android	Beta	19 May 2017, 16:32	
JW Test	iOS	Beta	22 May 2017, 19:20	
Lucky Fields	Android	Beta	19 Mar 2014, 11:19	
Lucky Fields	iOS	Beta	19 Mar 2014, 11:39	
MRGService Example	Android	Enterprise	23 May 2017, 15:11	
MRGservice Example	iOS	Enterprise	23 May 2017, 14:07	
Vikings	Android	Enterprise	19 Jun 2015, 17:53	
Vikings	iOS	Enterprise	19 Jun 2015, 17:43	
Игры@Mail.Ru	Android	Enterprise	29 May 2015, 14:09	
Игры@Mail.Ru	Android	Store	06 Feb 2015, 14:52	
Игры@Mail.Ru	iPhone	Enterprise	02 Jun 2015, 10:34	
Игры@Mail.Ru	iPhone	Store	02 Jun 2015, 10:48	
Эволюция: Битва за Утопию	iOS	Store	15 Apr 2015, 16:28	

HockeyApp


Juggernaut Wars Android | Store
Version 2.3.5 (3220)
Crash Group
Overview
Crash Logs 37
App Traces 1
Bug Tracker
Add Annotation
Status: open
com.my.mcsocial.MCSGoogleGames$7.execute
java.lang.NullPointerException
MCSGoogleGames.java, line 447
First Occurred
16 May 2017, 14:53
Last Occurred
23 May 2017, 12:12
Count
37
Crashes per Day
1
Last 24 hours
23
Last 7 days
37
Last 30 days
Impacted Users per Day
1
Last 24 hours
22
Last 7 days
36
Last 30 days
20
10
0
24 апр.
8 мая
22 мая
Stacktrace
Devices
OS Versions
Exception Backtrace:
View Raw Log
1 com.my.mcsocial.MCSGoogleGames$7.execute MCSGoogleGames.java, line 447
2 com.my.mcsocial.MCSGoogleQueue$QueueRunnable.run MCSGoogleQueue.java, line 178
3 java.lang.Thread.run Thread.java, line 841

После релиза



Просмотр логов боевой версии

```
public class MyLoger {
    public static void v(String TAG, String message){
        if (Log.isLoggable(TAG, Log.VERBOSE)) {
            Log.v(TAG, message);
        }
    }
    public static void d(String TAG, String message){
        if (Log.isLoggable(TAG, Log.DEBUG)) {
            Log.d(TAG, message);
        }
    }

}
```

```
$ adb shell setprop log.tag.<YourTag> VERBOSE
```

Просмотр содержимого файлов

Только если это эмулятор или root-ованное android-устройство

```
$ adb root
$ adb remount
$ adb shell
```

Сборка debug версии как release

build.gradle

```
android {
signingConfigs {
    release {
        storeFile file("keys/debug.keystore")
    }

}
```

```
$ adb install -r yourapp-debug.apk

$ adb shell

$ run-as com.your.package.name

$ cd /data/data/com.your.package/name
```

APKTool разбираем приложение

https://github.com/iBotPeaches/Apktool

https://ibotpeaches.github.io/Apktool/install/

```
$ apktool d yourapp.apk
```

Модифицируем AndroidManifest.xml

```
<application android:debuggable=”true” … />
```

APKTool собираем обратно

```
$ apktool b yourapk/ -o yourapp.apk

$ zipalign -v -p 4 yourapp.apk yourapp-aligned.apk

$ jarsigner -verbose -sigalg SHA1withRSA -digestalg SHA1\
  -keystore release.keystore yourapp-aligned.apk \
  myapp -signedjar yourapp-aligned-signed.apk

$ adb install -r yourapp-debug.apk

$ adb shell

$ run-as com.your.package.name

$ cd /data/data/com.your.package/name
```

ТЕХНОТРЕК

Спасибо за внимание!

Кирилл Филимонов – Kirill.Filimonov@gmail.com

Юрий Береза – ybereza@gmail.com