

### Лабораторная работа №11. Работа с уведомлениями

Задание: создать приложение, помещающее по нажатию на кнопку какое-то сообщение со звуком в панель уведомлений/статус-панель на эмуляторе.

Создайте стандартный проект с Empty Views Activity и поместите кнопку в интерфейс приложения. Уведомление будет приходить после нажатия на данную кнопку. Перейдите в файл MainActivity.java (для Котлина код будет позже) и в методе onCreate создайте ссылку на кнопку, а затем создайте для неё listener и метод onClick. Далее в методе onClick настройте уведомление (данный код актуален только для версий Android ниже Oreo, современный код см. после скриншотов):

Java:

```
Context context = MainActivity.this;
Notification notification = new Notification.Builder(context)
    .setContentTitle("Hi from Alex")
    .setContentText("Hey, listen!")
    .setTicker("new notification!")
    .setSmallIcon(android.R.drawable.ic_dialog_alert)
    .setSound(Uri.parse("android.resource://" + getPackageName() + "/" + R.raw.al_heylisten))
    .build();
NotificationManager notificationManager = (NotificationManager)
    getSystemService(Context.NOTIFICATION_SERVICE);
notificationManager.notify(0, notification);
```

Здесь мы устанавливаем заголовок (setContentTitle) и текст (setContentText) для уведомления, а также небольшой текст, выводящийся в небольшую панель уведомлений ("шторку") рядом с иконкой (setTicker, см. рисунок 47), иконку и звуковой файл, который будет воспроизводиться при получении уведомления. Звуковой файл нужно поместить в папку res/raw, однако для начала её стоит создать. Нажмите правой кнопкой мыши по папке res и создайте Android resource directory. Resource type выберите raw и нажмите ОК. Обратите внимание, что имя звукового файла не может начинаться с цифры. Не забудьте добавить в приложение опциональное меню с пунктом о программе, который открывает диалоговое окно с вашими ФИО, см. предыдущие 2 задания. Теперь всё готово, можно запускать приложение и нажимать на кнопку, после чего в панели уведомлений появится уведомление от приложения, как показано на рисунках 47-48.

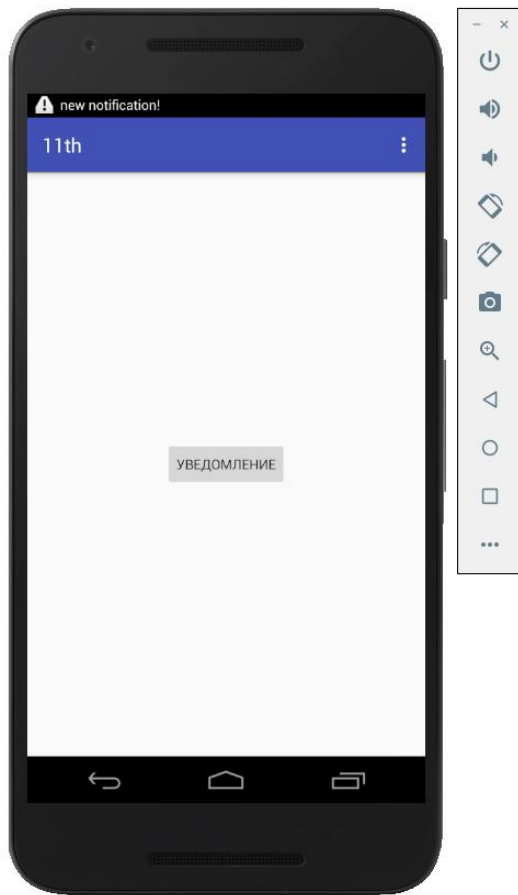


Рисунок 47 – Отображение краткого уведомления в панели уведомлений (API 16)

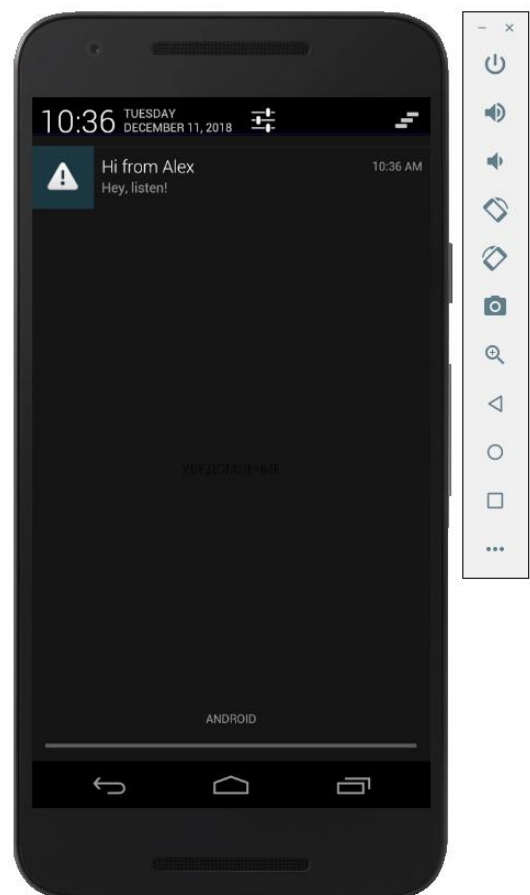


Рисунок 48 – Отображение уведомления в панели уведомлений (API 16)

Для версий Android, начиная с Oreo (Android 8, или API 26) нужен другой код. Дело в том, что начиная с Oreo появились так называемые каналы, которые пользователь приложения может настроить самостоятельно, перейдя в настройки приложения и открыв пункт уведомлений (Notifications). Поэтому, если у Вас приложение только для Oreo и выше, в коде нужно создавать и настраивать канал уведомлений `NotificationChannel`. Если в `gradle`-файле приложения заданы `minSdkVersion` и `targetSdkVersion` больше либо равные 26 (Oreo), предусматривать проверку текущей версии Android на устройстве не надо – просто надо взять новый код с поддержкой каналов. Однако часто бывает, что приложение должно поддерживать и старые версии, и, например, `minSdkVersion=16` при `targetSdkVersion=28`. Тогда в коде нужно делать проверку версии Android на устройстве, и для версии, начиная с Oreo (O), работать с каналами, как показано в коде ниже:

Java:

```
Context context = MainActivity.this;
NotificationChannel newnotchan= null;
if (android.os.Build.VERSION.SDK_INT >= android.os.Build.VERSION_CODES.O) {
    newnotchan= new
NotificationChannel("mychannel1", "mychannel", NotificationManager.IMPORTANCE_HIGH);
    AudioAttributes audioAttributes = new AudioAttributes.Builder()
        .setContentType(AudioAttributes.CONTENT_TYPE_MUSIC)
        .setUsage(AudioAttributes.USAGE_MEDIA)
        .build();
```

```

newnotchan.setSound(Uri.parse(ContentResolver.SCHEME_ANDROID_RESOURCE+"://"+getPackageName()+"/
"+R.raw.al_heylisten), audioAttributes);
}
NotificationManager notificationManager = (NotificationManager)
getApplicationContext().getSystemService(Context.NOTIFICATION_SERVICE);
if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O) {
    notificationManager.createNotificationChannel(newnotchan);
}
Notification notification = null;

if (android.os.Build.VERSION.SDK_INT >= android.os.Build.VERSION_CODES.O) {
    notification = new Notification.Builder(context, "mychannel1")
        .setContentTitle("Hi from Alex")
        .setContentText("Hey, listen!")
        .setTicker("new notification!")
        .setChannelId("mychannel1")
        .setSmallIcon(android.R.drawable.ic_dialog_alert)
        .setOngoing(true)
        .build();
}

notificationManager.notify(0, notification);

```

Kotlin:

```

val context: Context = this@MainActivity
var newnotchan: NotificationChannel? = null
if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O) {
    newnotchan = NotificationChannel(
        "mychannel1",
        "mychannel",
        NotificationManager.IMPORTANCE_HIGH
    )
    val audioAttributes = AudioAttributes.Builder()
        .setContentType(AudioAttributes.CONTENT_TYPE_MUSIC)
        .setUsage(AudioAttributes.USAGE_MEDIA)
        .build()
    newnotchan.setSound(
        Uri.parse(ContentResolver.SCHEME_ANDROID_RESOURCE + "://" + packageName +
"/" + R.raw.al_heylisten),
        audioAttributes
    )
}
val notificationManager =
    applicationContext.getSystemService(NOTIFICATION_SERVICE) as
NotificationManager
if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O) {
    notificationManager.createNotificationChannel(newnotchan!!)
}
var notification: Notification? = null

if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O) {
    notification = Notification.Builder(context, "mychannel1")
        .setContentTitle("Hi from Alex")
        .setContentText("Hey, listen!")
        .setTicker("new notification!")

```

```

        .setChannelId("mychannel1")
        .setSmallIcon(android.R.drawable.ic_dialog_alert)
        .setOngoing(true)
        .build()
    }

    notificationManager.notify(0, notification)

```

Обратите внимание – для того, чтобы слышать именно тот звук, который расположен в папке raw, нужно сконфигурировать для канала так называемые AudioAttributes, а метод setSound, в котором нужны путь к файлу и эти аудиоатрибуты, затем использовать для канала, а не для уведомления. В приведённом выше коде отсутствует часть для версий Android ниже Oreo в конструкторе else, но она есть в начале описания этой лабораторной работы, и её в этот конструктор не составляет труда добавить, таким образом, сделав это приложение универсальным.

В последней строке (.notify) вы, наверняка, заметили ошибку – в последних версиях Android нужно дополнительно добавить разрешение для приложения размещать уведомления в файл манифеста:

```
<uses-permission android:name="android.permission.POST_NOTIFICATIONS" />
```

Кроме этого, после установки приложения на телефон/смартфон нужно **дать разрешение приложению на отправку уведомлений**. Это делается в настройках системы – приложения – ваше приложение – уведомления – поставить переключатель в положение ON (Settings/Apps/YourNotificationAppName/Notifications/Toggle on). После этого, если вы уже безрезультатно запускали приложение, можно увидеть название канала, который используется для отправки уведомлений приложением (рисунок 52). Результат работы приложения показан на скриншотах ниже, при этом проигрывается нужный звук уведомления.

Если вы хотите, чтобы разрешения для приложения выдавались системой автоматически, и чтобы не надо было вручную давать разрешение приложению на отправку уведомлений (как на рисунке 52), как написано в предыдущем абзаце, можно предусмотреть разрешение уведомления прямо в коде приложения с помощью динамических разрешений (рисунок 49). Для этого надо добавить в код следующее: метод requestMultiplePermissions() в любое место описания класса MainActivity, и затем метод onRequestPermissionsResult():

Java:

```

public void requestMultiplePermissions() {
    ActivityCompat.requestPermissions(this,
        new String[] {
            POST_NOTIFICATIONS
        },
        PERMISSION_REQUEST_CODE);
}

public void onRequestPermissionsResult(int requestCode, @NonNull String[] permissions, @NonNull int[] grantResults) {
    if (requestCode == PERMISSION_REQUEST_CODE) {

```

```

    if (grantResults[0] == PackageManager.PERMISSION_GRANTED) {
        return;
    } else finish();
}
super.onRequestPermissionsResult(requestCode, permissions, grantResults);
}

```

Kotlin:

```

fun requestMultiplePermissions() {
    ActivityCompat.requestPermissions(
        this,
        arrayOf<String>(
            POST_NOTIFICATIONS
        ),
        PERMISSION_REQUEST_CODE
    )
}

override fun onRequestPermissionsResult(
    requestCode: Int,
    vararg permissions: String,
    grantResults: IntArray
) {
    if (requestCode == PERMISSION_REQUEST_CODE) {
        if (grantResults[0] == PackageManager.PERMISSION_GRANTED) {
            return
        } else finish()
    }
    super.onRequestPermissionsResult(requestCode, permissions, grantResults)
}

```

Соответственно, в код нажатия кнопки метода onClick() (в самое начало) нужно добавить строку:

Java:

```
requestMultiplePermissions();
```

Kotlin:

```
requestMultiplePermissions()
```

А в начало класса можно добавить нужную переменную, например:

Kotlin:

```
val PERMISSION_REQUEST_CODE = 1
```

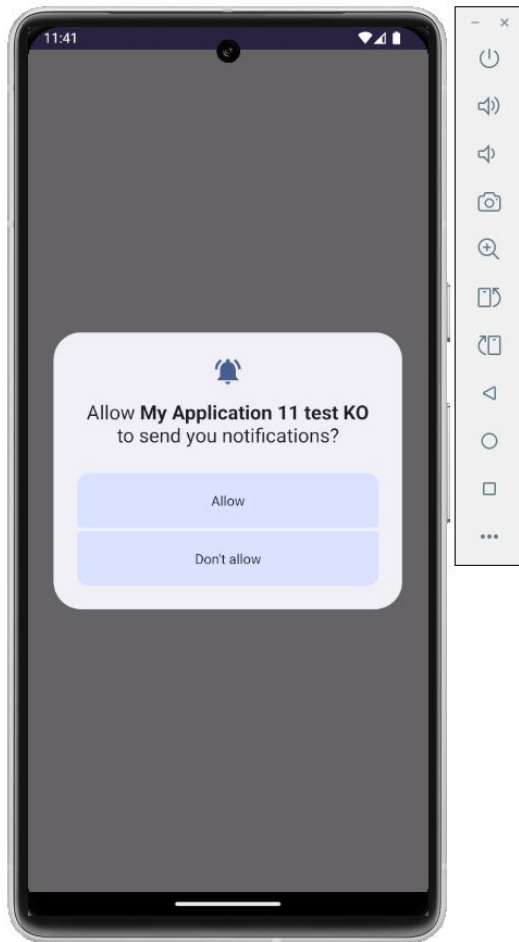


Рисунок 49 – Runtime permissions

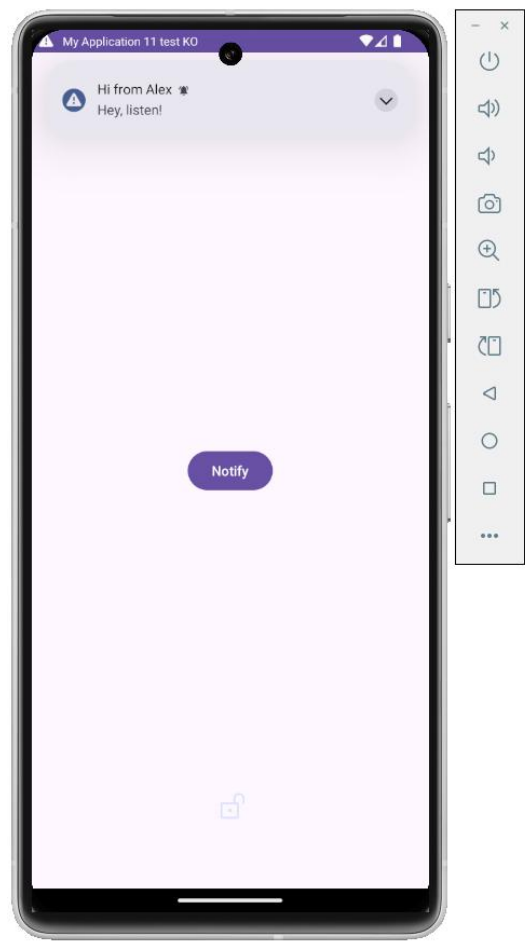


Рисунок 50 – Отображение уведомления на экране (API 35)

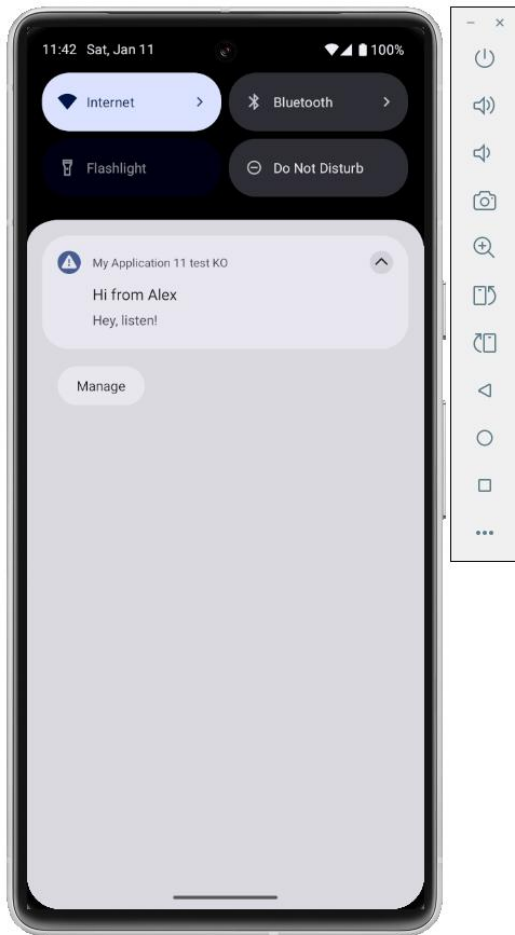


Рисунок 51 – Отображение уведомления в панели уведомлений (API 35)

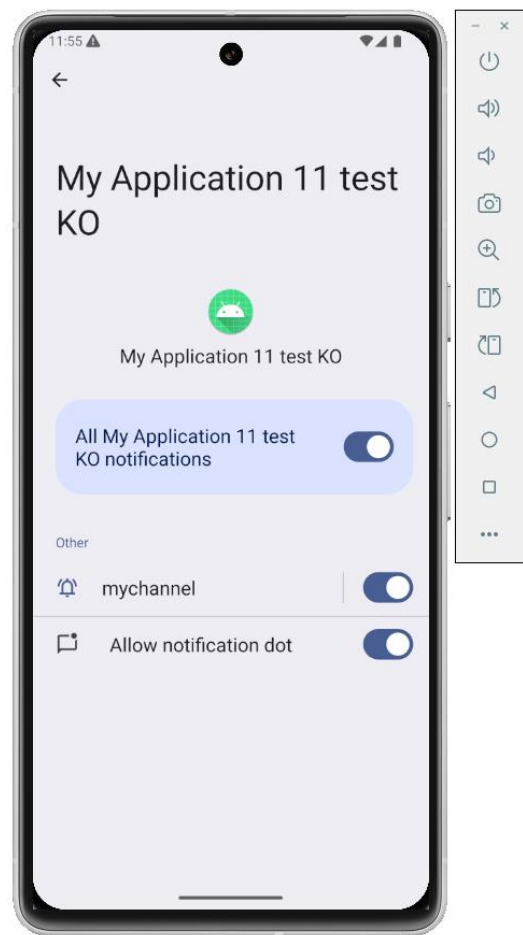


Рисунок 52 – Разрешение на показ уведомлений для приложения (API 35)