УДК 004

Особенности создания приложения будильник для последних версий Android

Звайгзне Алексей Юрьевич
Приамурский государственный университет имени Шолом-Алейхема
Студент

Аннотация
В данной статье рассматривается процесс разработки Android приложения– будильник с помощью Android Studio для новых версий Android. Исследуются особенности создания приложения на последней версии платформы Android (API старше 21). Проверяется работоспособность скомпилированного готового приложения на предмет ошибок совместимости версий.
Ключевые слова: Android, Android Studio, AndroidManifest, SimpleDateFormat, MaterialTimePicker

Features of creating an alarm clock application for the latest Android versions

Zvaigzne Alexey Yurievich
Sholom-Aleichem Priamursky State University
Student

Abstract
This article discusses the process of developing an Android alarm clock application using Android Studio for new versions of Android. The features of creating an application on the latest version of the Android platform (API older than 21) are investigated. The operability of the compiled ready-made application is checked for version compatibility errors.
Keywords: Android, Android Studio, AndroidManifest, SimpleDateFormat, Material Timing Tool

1 Введение

1.1 Актуальность

В связи с постоянным обновлением платформы Android, есть необходимость постоянного мониторинга обновлений изменений в самой платформе, не только для обычных пользователей, но и разработчикам, разрабатывающим новые приложения и следящих за обновлениями все ещё актуальных. Так как небольшие и не значительные нововведения на первый взгляд не сильно меняют структуру приложения, но имеют большую значимость для реализации на более новых платформах.

Разработчики Android учитывают это, поэтому в большинстве случаев, изменения моментально адаптируют под большинство библиотек без необходимости пересоздания приложений с нуля.

1.2 Обзор исследований

М. Рокатанской в статье об обновлениях библиотеки Pending Intent указывал необходимые нововведения, которые необходимо учитывать при разработке приложения [1]. В разделе на официальном описывается работа с унифицированным индикатором ресурсов [2]. М. Вольфсон в своей книге описывает инструменты необходимые разработчику Android приложений, SDK [3]. Т. Хагос в своей работе объясняет принципы эффективной разработки Android приложений используя инструменты отладки [4]. В книге Н. Верма рассматривается процесс создания кроссплатформенного приложения для сервисов такси [5].

1.3 Цель исследования

Разработка приложения Будильник с учетом нововведений в платформе Android и проверка работоспособности приложения.

2 Материалы и методы

Приложение разрабатывается на последней версии Android Studio, а также используются последние версии библиотек. Проверка приложения происходит на виртуальном устройстве, интегрированное в саму среду разработки.

3 Результаты и обсуждения

Для создания нового проекта, надо выбрать пустую активность и присвоить ей имя, язык написания программы и минимальную версию Android платформы для работы (рис. 1).

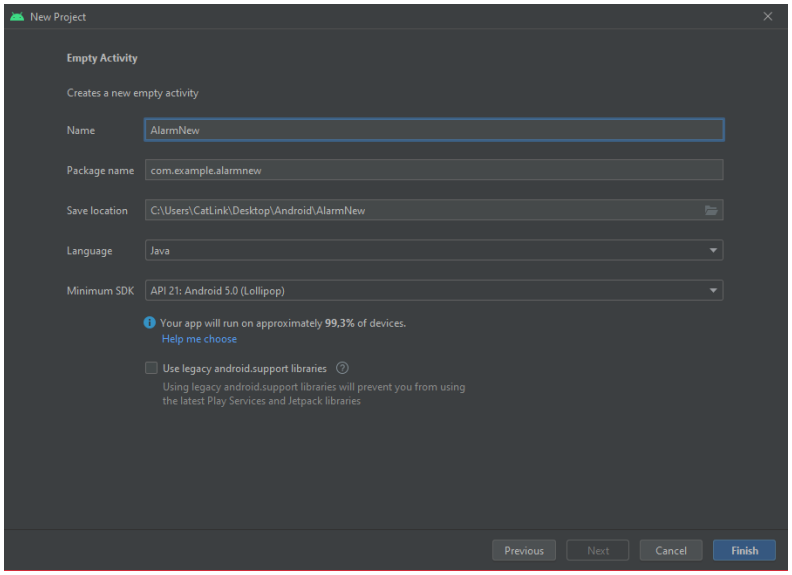

Рисунок 1. Окно создания проекта

После создания проекта следует отредактировать файл AndroidManifest.xml, добавив туда необходимые требования, которые указаны в последних версия Android начиная с 11 версии (рис. 2).

Рисунок 2. Правки в файле AndroidManifest.xml

Первая строчка кода отвечает за поведение устройства в выключенном состоянии. В новых версиях платформы, используются алгоритмы приоритета выполнения задач. Пока устройство находится в спящем режиме алгоритмы оптимизации батареи с определенной периодичностью замораживает исполнение команд проверяя время, стоит ли активировать сейчас или оставить на потом. Если до выполнения определенной задачи остается небольшой промежуток времени, система оставляет активной эту задачу до её выполнения, не давая системе уйти в режим сна, условно от пяти минут до получаса, если же времени до необходимости применения активности ещё много, то система замораживает её исполнение до следующей проверки.

Вторая строчка отвечает за показ активности поверх остальных окон, необходимость внедрения этого требования появилась, начиная с 6 версии платформы Android. Это было сделано из соображений безопасности, пользователь должен в приложении самостоятельно выдать приложению права на эту функцию при первом запуске приложения.

Для реализации будильника на последней версии Android Studio нет необходимости импортировать библиотеки визуализации компонентов Material Design, они по умолчанию уже есть в проекте.

Можно приступать к визуальной части разработки приложения. Первое что необходимо это настроить слой, добавить кнопку или задать слой вручную через редактор кода (рис. 3-4).

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <Button
        android:id="@+id/alarm_button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Установить будильник"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

Рисунок 3. Вид кода главной активности

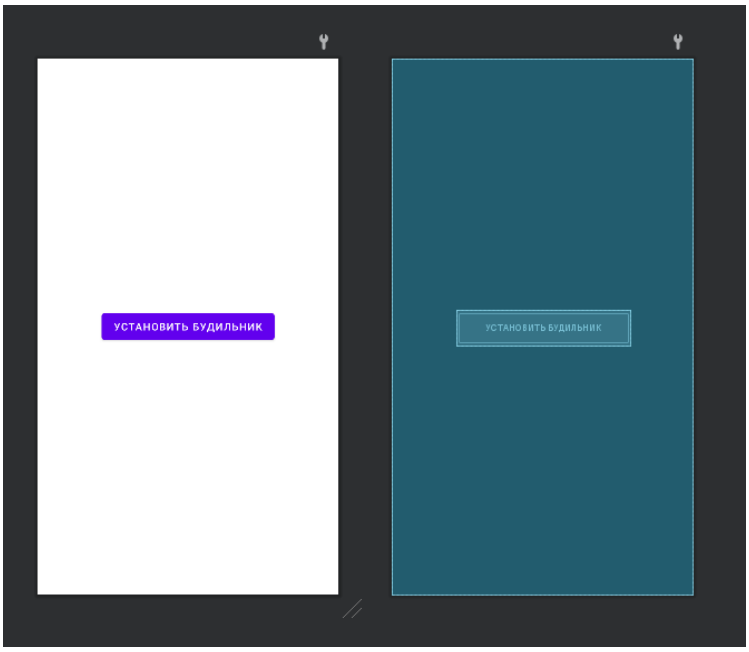

Рисунок 4. Вид визуальной части главной активности

Так как само по себе нажатие кнопки не будет воздействовать никак на активность, нет необходимости генерировать метод Onclick (действие по нажатию). Все действия активности будут прописаны напрямую в файле

класса активности. После создания класса в него добавляются все нужные библиотеки для работы (рис. 5).

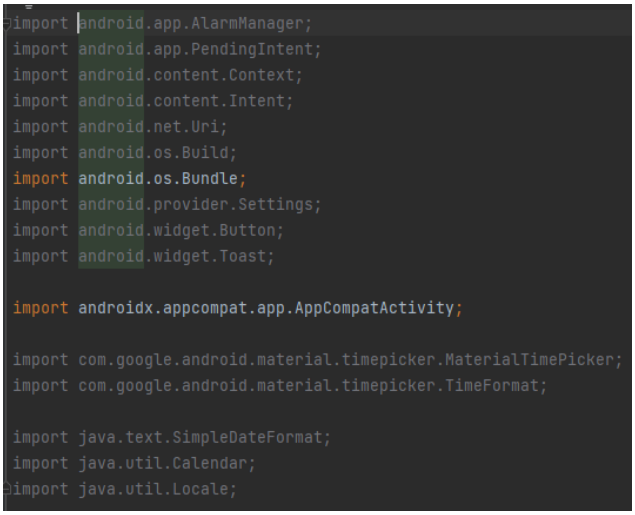

Рисунок 5. Добавление библиотек в главную активность

Для взаимодействия с кнопкой добавляется ссылка на неё в главной активности (рис. 6).

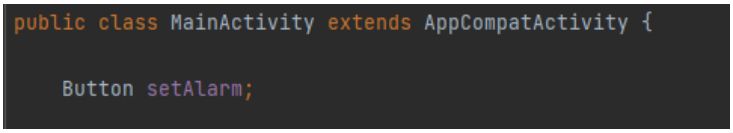

Рисунок 6. Добавление ссылки на кнопку в главную активность

Первым делом в активности обозначается формат хранения данных будильника с помощью команды SimpleDateFormat, присвоить новую переменную sdf и обозначить шаблон формата «HH:mm» – «часы:минуты». Указать ссылку на кнопку в активности через команду findViewById и установить шаблон времени используя элемент MaterialTimePicker и выдав ему параметры по умолчанию при открытии тикера, устанвливается двадцатичетырёхчасовой формат времени, время на двенадцать часов, ноль минут и указав название тикера «Выберете время будильника» (рис. 7).

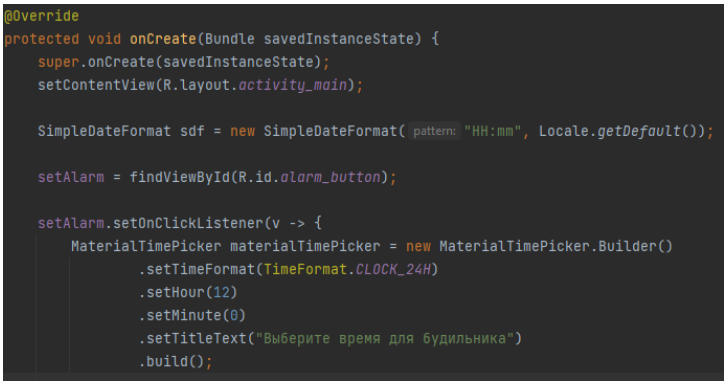

Рисунок 7. Вид кода присеста для визуального отображения выбора времени для будильника

Назначение будильника в системе происходит путем присвоения ему точности времени, отбрасывая погрешности в секунды и миллисекунды, получение данных, часов и минут из тикера и передача системному сервису будильника, выбранное значение. Для подтверждения указанного времени, так же можно вывести всплывающее уведомление Toast с указанным временем (рис. 8).

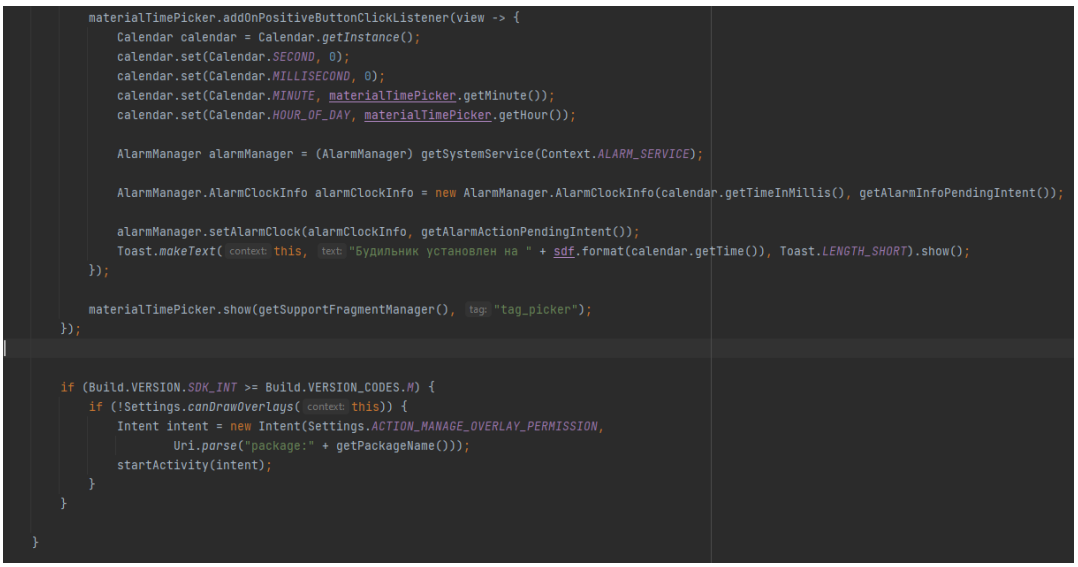

Рисунок 8. Код установки будильника

Нижняя часть кода проверяет, если система, выше заданного значения, открывается диалоговое окно запрашивающая необходимые системные требования, в данном случае, показ поверх остальных окон и задания будильника.

Так как выполнения будильника это отложенная активность необходимо задать его через набор отложенных команд PendingIntent, необходимо чтобы каждый раз при проверке задач, активность, связанная с будильником, попадала на верхнюю строчку, для этого у активности надо задать соответствующий флаг и если это новая версия платформы, указать флаг запрещающий другим программам вносить в него правки используя FLAG_IMMUTABLE и FLAG_UPDATE_CURRENT для того чтобы только созданное приложение могло изменять собственное значение активности (рис. 9).

Рисунок 9. Фиксирование задачи будильника в системе

Осталось создать ещё пустую одну активность, отвечающую за вывод окна с будильником и текущей мелодией звонка будильника выбранной в системе, можно указать ей имя AlarmActivity, название слоя, например, activity_alarm.

Сначала можно заполнить слой активности добавить текст выводящий будильник (рис. 10).

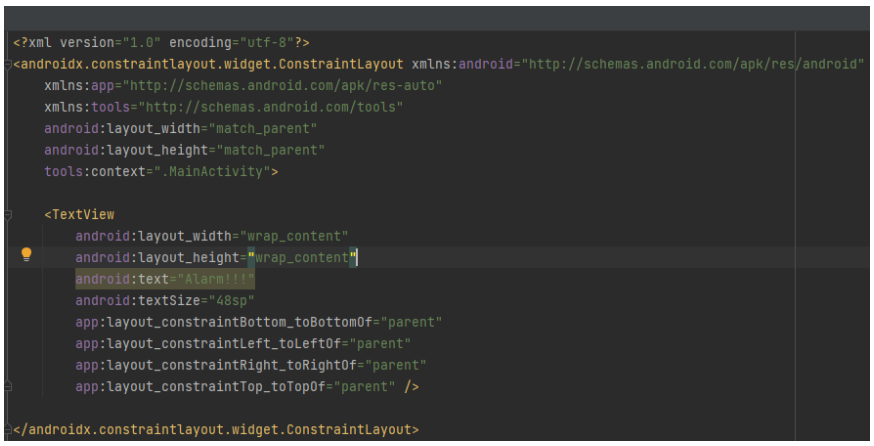

Рисунок 10. Код слоя активности окна будильника

Саму же активность по порядку можно начать с импорта необходимых библиотек. Отличительной библиотекой можно выделить URI, отвечающую за унифицированную систему ссылок, подуровень системы для понимания к какой именно среде системы требуется обратиться и что именно оттуда необходимо получить или дать. В данном случае, будет выполнять поиск значений будильника и рингтона в системных сервисах (рис. 11).

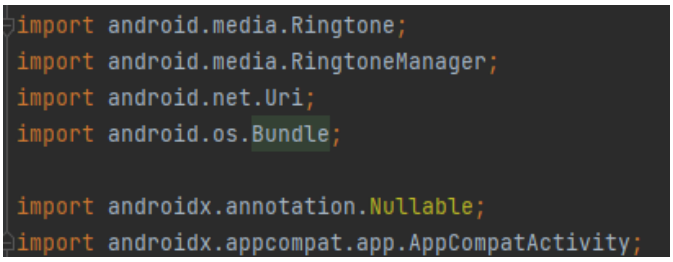

Рисунок 11. Импортирование необходимых библиотек в активность будильника

Первым дело в активности обозначается текущая мелодия будильника, выбирается из списка по флагу TYPE_ALARM. Если рингтон будильника обозначен, активность будет выбирать именно его, в противном случае. Будильник выдаст ошибку и не будет обрабатывать активность будильника закрывшись с ошибкой (рис. 12).

```
public class AlarmActivity extends AppCompatActivity {

    Ringtone ringtone;

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

        setContentView(R.layout.activity_alarm);

        Uri notificationUri = RingtoneManager.getDefaultUri(RingtoneManager.TYPE_ALARM);
        ringtone = RingtoneManager.getRingtone( context: this, notificationUri);
        if (ringtone == null) {
            notificationUri = RingtoneManager.getDefaultUri(RingtoneManager.TYPE_RINGTONE);
            ringtone = RingtoneManager.getRingtone( context: this, notificationUri);
        }
        if (ringtone != null) {
            ringtone.play();
        }
    }
```

Рисунок 12. Код активности будильника

Чтобы остановить мелодию надо добавить процедуру, отвечающую за закрытие активности и остановки проигрывания мелодии в случае закрытия окна активности будильника.

```
    @Override
    protected void onDestroy() {
        if (ringtone != null && ringtone.isPlaying()) {
            ringtone.stop();
        }
        super.onDestroy();
    }
}
```

Рисунок 13. Код отключения будильника

Код реализован следующим образом, если мелодия будильника обозначена и проигрывается в данный момент, то её необходимо выключить. А процедуру будильника уничтожить. Если этого не сделать, пока не закончится минута, будильник будет пытаться каждый момент времени её вызвать повторно.

Теперь осталось скомпилировать и проверить полученный результат (рис. 14-15).

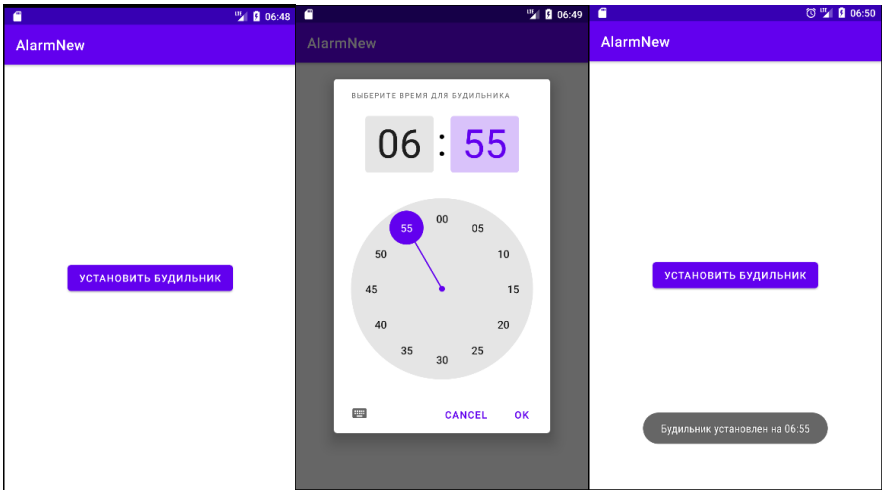

Рисунок 14. Работа приложения первая активность установка будильника

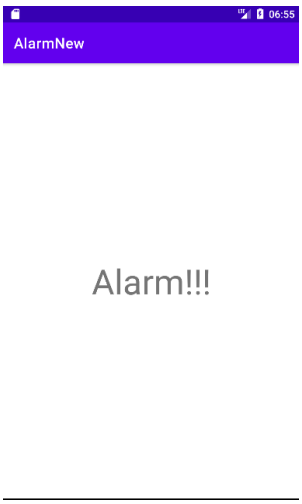

Рисунок 15. Вывод активности в момент срабатывания будильника

Выводы

Созданное приложение отлично работает на последних версиях платформы, за одним исключением, необходимости выдачи разрешений для установки системного будильника и отображения поверх остальных окон, в случае если этого не сделать приложение выдаст ошибку и не выведет активность, отвечающую за будильник, хотя появится всплывающее сообщение об успешной установке будильника.

Реализация данного приложения на последней версии платформы может помочь в создании в будущем более сложных приложений с отложенной активностью, отвечающих за повседневные задачи, например, создание записной книжки с заданным интервалом принятия таблеток или выполнения каких-либо задач.

Основной проблемой создания даже простых приложений на современной платформе, это отслеживание нововведений в рабочие библиотеки и обновления платформы Android. Хоть и разработчики платформы стараются свести к минимуму оптимизации кода, не всегда есть возможность отследить изменения библиотек быстро и своевременно, что может привести к полной дееспособности приложения на современных устройствах, это можно проверить путем установки старых, сторонних программ, часть функционала которых будет попросту не рабочей.

Библиографический список

1. Статья на тему важных изменений библиотеки Pending Intent URL: https://habr.com/ru/company/otus/blog/560492/ (дата обращения 20.01.2023).
2. Раздел на официальном сайте для разработчиков Android приложений на тему библиотеки URI URL: https://developer.android.com/reference/java/net/URI (дата обращения 20.01.2023).
3. Wolfson M., Felker D. Android developer tools essentials: Android Studio to Zipalign. O'Reilly Media, Inc., 2013.

4. Hagos T., Hagos T. Android studio //Learn Android Studio 3: Efficient Android App Development. 2018. С. 5-17.
5. Verma N., Kansal S., Malvi H. Development of native mobile application using android studio for cabs and some glimpse of cross platform apps //International Journal of Applied Engineering Research. 2018. Т. 13. №. 16. С. 12527-12530.