УДК 681.515.8
Здолбіцька Н.В., Мельник Д.С.
Луцький національний технічний університет

СТВОРЕННЯ МОБІЛЬНОГО ДОДАТКУ В IDE ANDROID STUDIO ДЛЯ ДІАГНОСТИКИ СМАРТФОНІВ НА ОСНОВІ ОПЕРАЦІЙНОЇ СИСТЕМИ ANDROID

Здолбіцька Н.В., Мельник Д.С. Створення мобільного додатку в IDE Android Studio для діагностики смартфонів на основі операційної системи Android. Смартфони стали невід'ємною частиною повсякденного життя людей. Даний додаток дасть можливість отримати реальні дані про смартфон при покупці, а також продіагностувати поточний стан вашого смартфону. У даній статті розглядається розробка додатку в IDE Android Studio шляхом отримання системних даних з пристрою.
Ключові слова: смартфон, OS Android, IDE Android Studio, діагностика.

Здолбицкая Н.В., Мельник Д.С. Создание мобильного приложения в IDE Android Studio для диагностики смартфонов на основе операционной системы Android. Смартфоны стали неотъемлемой частью повседневной жизни людей. Данное приложение позволит получить реальные данные о смартфоне при покупке, а также продиагностировать текущее состояние вашего смартфона. В данной статье рассматривается разработка приложения в IDE Android Studio путем получения системных данных с устройства.
Ключевые слова: смартфон, OS Android, IDE Android Studio, диагностика.

Zdolbitska N.V., Melnyk D.S. Research FIR filters using the LabVIEW environment. Create a mobile application in the Android Studio IDE to diagnose smartphones based on the Android OS. Smartphones have become an integral part of everyday life of people. This application will give you an opportunity to get real data about a smartphone when buying, and also to diagnose the current state of your smartphone. This article discusses the development of an application in the Android Studio IDE by obtaining system data from the device.
Keywords: smartphone, OS Android, IDE Android Studio, diagnostics.

Постановка наукової проблеми. Смартфони стали невід'ємною частиною повсякденного життя людей. Завдяки своїй портативності та великому спектру можливостей користувачі надають все більшу перевагу покупці смартфонів ніж комп'ютерів. Набагато простіше і швидше знайти якусь інформацію, відправити повідомлення, подивитись відеоролик на смартфоні аніж на комп'ютері. Тому дана галузь дуже стрімко розвивається, виробники все швидше впроваджують нові технології та покращують свої продукти. Купуючи смартфон, недобросовісний продавець може видати бракований чи гірший по характеристикам товар за завищеною ціною. Даний додаток дозволить зменшити зони ризику для покупця.

Аналіз досліджень. Середовище розробки Android Studio дозволяє швидко за допомогою різноманітних інструментів та можливостей мови програмування Java створювати мобільні додатки на основі операційної системи Android. Оскільки операційна система Android має відкритий код, розробник має доступ до системних даних та можливість використовувати їх. Також слід відзначити, що Android є найбільш популярною ОС на сьогоднішній день, близько на 60% смартфонів встановлена дана ОС. Це сприяє розвитку платформи та подальшому її поширенню, тому і розробка додатків для ОС Android є доволі популярною та прибутковою.

На сайті *developer.android.com/develop* можна знайти інформацію про те як працювати з різними класами, їх методами. Даний сайт містить всю необхідну та найактуальніше документацію.

Аналізуючи Play Market можна зробити висновок, що додатки для діагностики смартфонів мають від 100 тисяч завантажень, що підтверджує необхідність того, що люди все частіше при покупці своїх девайсів, особливо тих, що вже були у використанні використовують дані додатки.

Виклад основного матеріалу та обґрунтування результатів дослідження.

Створюючи проект розробник може обрати найновішу версію Android, використовувати усі найновіші функції. Але у даному випадку у користувачів, які мають нижчі версії Android можуть бути проблеми у використанні додатку, або взагалі неможливість його використання у зв'язку із не підтримкою певних функцій. Даний вибір є не рентабельним з комерційної точки зору, оскільки зазвичай найновіші версії Android встановлено на 1-10% смартфонів. До прикладу Android 7.0 (Nougat), згідно з інформацією наданою в Android Studio встановлено 8.1% девайсів, старішу версію Android 6.0 (Marshmallow) - 39.3%. В той же час обравши нижчу версію ОС розробник не має можливості використовувати всі переваги нових версій, але підтримувати нормальну роботу додатку буде більша кількість смартфонів.

Даний проект створений для Android 5.0, оскільки 71% девайсів підтримуються з цієї версії Android, незважаючи на те, що дана версія не є найновішою, але ще доволі популярна у використанні.

Додаток складається з двох вкладок: Інформація (рис. 1) та Тести (рис. 2). При запуску відкрита вкладка Інформація, на якій зображена інформація про модель смартфону, та кнопки для переходу до інших Activity, на яких відображена інформація відповідно поділена по групах.

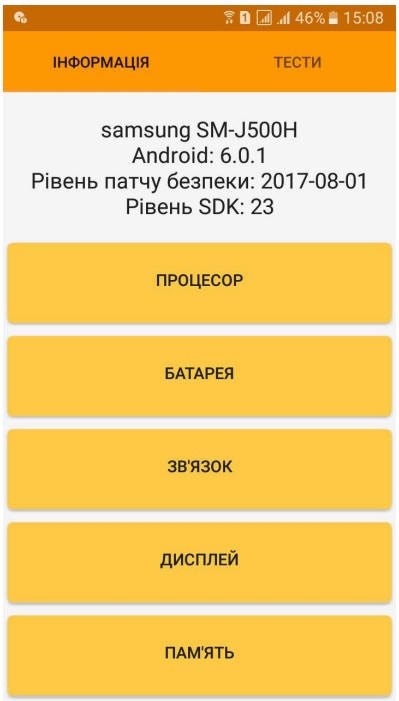

Рис. 1. Головний екран, вкладка Інформація

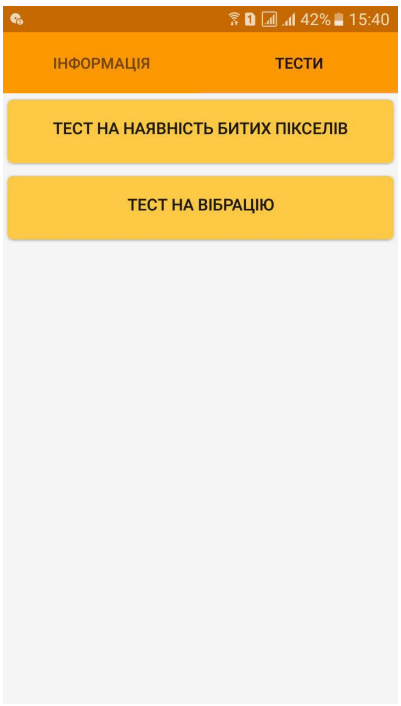

Рис. 2. Головний екран, вкладка Тести

Activity – це компонент додатка, який видає екран, і з яким користувачі можуть взаємодіяти для виконання будь-яких дій, наприклад набрати номер телефону, зробити фото, відправити лист або переглянути карту. Кожній операції присвоюється вікно для промальовування відповідного

призначеного для користувача інтерфейсу. Зазвичай вікно відображається на весь екран, проте його розмір може бути менше, і воно може розміщуватися поверх інших вікон.

Як правило, програма містить кілька операцій, які слабо пов'язані один з одним. Зазвичай одна з операцій в додатку позначається як «основна», пропонована користувачеві при першому запуску програми. У свою чергу, кожна операція може запустити іншу операцію для виконання різних дій. Кожен раз, коли запускається нова операція, попередня операція зупиняється, однак система зберігає її в стек («стек переходів назад»). При запуску нової операції вона поміщається в стек переходів назад і відображається для користувача. Стек переходів назад працює за принципом LIFO (last in – first out) «останнім увійшов – першим вийшов», тому після того як користувач завершив поточну операцію і натиснув кнопку Назад, поточна операція видаляється з стека (і знищується), і відновлюється попередня операція. Саме так відбувається перехід по екранам (Activity), натиснувши кнопку назад по стандарту додаток згорнеться, і можна перейти на екран меню свого смартфону. В розробленому додатку при натисненні кнопки «Назад» на головному екрані, можна побачити сповіщення, яке повідомить: «Натисніть ще раз для виходу» (рис. 3). Реалізовано це наступним чином.

```
private static long back_pressed = 0;

@Override
public void onBackPressed()
{
    if (back_pressed + 2000 > System.currentTimeMillis())
        finish();
    else
        Toast.makeText(getBaseContext(), "Натисніть ще раз для виходу",
                Toast.LENGTH_SHORT).show();
    back_pressed = System.currentTimeMillis();
}
```

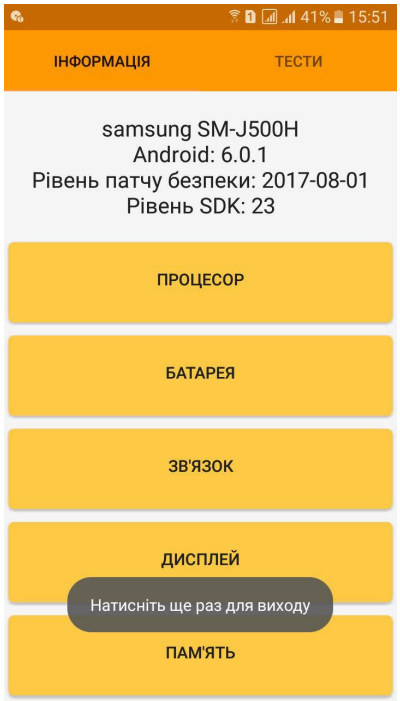

Рис. 3. Повідомлення при натисненні кнопки «Назад»

Деякі дані, такі як частота та використання процесора чи дані про оперативну пам'ять потребують постійного оновлення, для зображення поточних даних. Оновлення даних, на прикладі зображення частоти та використання процесора було реалізовано наступним чином. В методі *onCreate()* запускається метод *start()*, який запускає таймер, в свою чергу таймер кожної секунди запускає метод *get_freq_and_use()*, даний метод зчитує та подає на екран дані про поточну частоту та використання процесора.

```
@Override
protected void onCreate(@Nullable Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.cpu);
    cpuV = (TextView) findViewById(R.id.cpuV);
    freq_and_use = (TextView) findViewById(R.id.freq_and_use);
    header = (TextView) findViewById(R.id.cpu_header);
    header.setText("Процесор");
    get_freq_and_use();
    cpuV.setText("");
    cpuV.append(new Func().getCPUinfo());
    start();
}

protected void start()
{
    CountDownTimer countDownTimer = new CountDownTimer(seconds * 1000, 1000)
{
        @Override
        public void onTick(long l) {
            get_freq_and_use();
            seconds++;
        }
        @Override
        public void onFinish() {

        }
    }.start();
}
```

Для отримання частоти використовувався клас *Frequency*. Об'єкт цього класу має методи *getCPUFrequencyCurrent(), getCPUFrequencyMin()* та *getCPUFrequencyMax()*. Дані про використання можна знайти у системному файлі */proc/stat*, інші дані про процесор отримано з файлу */proc/cpuinfo*.

Дані про батарею можна отримати з допомогою *BatteryManger,* вони теж потребують свого оновлення, але на відмінну від попереднього прикладу, оновлення інформації про батарею не потрібно проводити щосекунди, а лише при зміні даних. Це можна реалізувати за допомогою *BroadcastReceiver*. BroadcatReceiver – це компонент для отримання зовнішніх подій і реакції на них, це об'єкт, який починає виконувати дії при отриманню якого-небудь сигналу (Intent). У даного класу тільки один метод onReceive. У даному випадку сигналом є зміна параметрів батареї *(ACTION_BATTERY_CHANGED).*

```
BroadcastReceiver receiver = new BroadcastReceiver() {
    @Override
    public void onReceive(Context context, Intent intent) {
        battV.setText("");
        battV.append("Максимальне значення: " +
intent.getIntExtra(BatteryManager.EXTRA_SCALE, -1) + "%");
        battV.append("\nПоточне значення: " +
intent.getIntExtra(BatteryManager.EXTRA_LEVEL, -1) + "%");
        battV.append("\nТемпература: " + (float)
intent.getIntExtra(BatteryManager.EXTRA_TEMPERATURE, -1) / 10.0f + "°C");
        battV.append("\nНапруга: " + (float)
intent.getIntExtra(BatteryManager.EXTRA_VOLTAGE, -1) / 1000.0f + "V");
        battV.append("\nТехннологія: " +
intent.getStringExtra(BatteryManager.EXTRA_TECHNOLOGY));
        battV.append("\nСтан: " +
Func.batteryHealth(intent.getIntExtra(BatteryManager.EXTRA_HEALTH, -1)));
        battV.append("\nСтатус: " +
Func.batteryStatus(intent.getIntExtra(BatteryManager.EXTRA_STATUS, -1)));
```

```
        battV.append("\nДжерело: " +
Func.batteryPlugged(intent.getIntExtra(BatteryManager.EXTRA_PLUGGED, -1)));
    }
};
registerReceiver(receiver, new IntentFilter(Intent.ACTION_BATTERY_CHANGED));
```

Інформацію про мобільну мережу можна отримати за допомогою даного об'єкту, використовуючи методи класу *TelephonyManager*:

```
TelephonyManager
tm=(TelephonyManager)getSystemService(Context.TELEPHONY_SERVICE);
```

Інформацію про дисплей можна отримати за допомогою даного об'єкту, використовуючи методи класу *DisplayMetrics*:

```
DisplayMetrics metrics = new DisplayMetrics();
getWindowManager().getDefaultDisplay().getMetrics(metrics);
```

Тест на наявність битих пікселів передбачає зображення кольорів (червоний, зелений, синій, чорний білий) у повний екран. Користувач повинен самостійно оглянути екран і пересвідчитись чи всі пікселі одного кольору. Зміна кольорів відбувається при натиску на екран, при натиску на останній колір буде подано вібро сигнал, користувач побачить повідомлення «Тест завершено» «Для виходу натисніть ще раз».

Тест на вібрацію, вмикає вібрацію на 5 секунд, після чого зображується повідомлення «Тест завершено».

Висновки. IDE Android Studio зручне та просте середовище для створення мобільних додатків. Що спонукає розробників розробляти та розвивати додатки на базі операційної системи Android, що в свою чергу популяризує її і приваблює користувачів. Тому з швидким розвитком ПЗ та смартфонів, а також слідуючи вимогам часу, створення додатків для тестування смартфону є доволі актуальним. А відкритість операційної системи дозволяє отримувати системні дані.

1. Broadcast (Широковещательные сообщения) http://developer.alexanderklimov.ru/android/theory/broadcast.php
2. Блэйк Мик Программирование под Android. – СПб.: Санкт-Петербург, 2012. – 496 с.
3. Здолбіцька Н.В., Здолбіцький А.П., Найдюк Ю.Ю. Програмне забезпечення для керування системою сенсорів на базі ОС Android // Тези Всеукраїнської науково-практичної інтернет-конференції «Сучасні методи, інформаційне та програмне забезпечення систем управління організаційно-технологічними комплексами», 28 квітня 2015 р. – Луцьк, 2015. – С. 41-42.
4. Лучшие операционные системы для смартфонов https://keddr.com/2017/04/luchshie-operatsionnyie-sistemyi-dlya-smartfonov/
5. Разработка, Руководства по API, Операции https://developer.android.com/guide/components/activities.html?hl=ru
6. Сенсори. Прискорення, орієнтація [Електронний ресурс]. – Режим доступу: http://startandroid.ru/ru/uroki/vse-uroki-spiskom/287-urok-137-sensory-uskorenie-orientatsija.html.
7. Шолом П.С., Здолбіцький А.П., Жигаревич О.К., Яручик В.Л. Роботизована система з дистанційним керуванням // Міжвузівський збірник "Комп'терно-інтегровані технології: освіта, наука, виробництво" - 2015. - № 19. - С. 86-90.