

Автоматическое тестирование Android-приложений с использованием инструмента Espresso

Андриенко Иван Сергеевич

Приамурский государственный университет имени Шолом-Алейхема

Студент

Аннотация

В статье рассмотрен процесс автоматического тестирования пользовательского интерфейса Android-приложений с использованием фреймворка Espresso. Приведены основные возможности инструмента, включая проверку взаимодействия с элементами интерфейса, обработку асинхронных задач и написание стабильных тестов. На примере простого приложения продемонстрирована реализация тестов для проверки работы текстовых полей и кнопок. Работа акцентирует внимание на практическом применении Espresso для повышения качества и надежности Android-приложений.

Ключевые слова: Android, автоматическое тестирование, Espresso, UI-тестирование, пользовательский интерфейс, инструменты тестирования, разработка приложений.

Automatic testing of Android applications using the Espresso tool

Andrienko Ivan Sergeevich

Sholom-Aleichem Priamursky State University

Student

Abstract

The article discusses the process of automatic testing of the user interface of Android applications using the Espresso framework. The main features of the tool are given, including checking interaction with interface elements, processing asynchronous tasks, and writing stable tests. Using the example of a simple application, the implementation of tests for checking the operation of text fields and buttons is demonstrated. The work focuses on the practical application of Espresso to improve the quality and reliability of Android applications.

Keywords: Android, automated testing, Espresso, UI testing, user interface, testing tools, application development.

1 Введение

1.1 Актуальность

С ростом количества мобильных приложений увеличивается важность обеспечения их качества и надежности. Тестирование пользовательского интерфейса является ключевым этапом разработки, так как позволяет

проверить корректность взаимодействия пользователей с приложением. Ручное тестирование требует значительных временных затрат и подвержено человеческому фактору, тогда как автоматическое тестирование позволяет оптимизировать этот процесс. Espresso, будучи официальным инструментом от Google, предоставляет мощные возможности для проверки работы интерфейса, взаимодействий с элементами и асинхронных операций. Актуальность темы обусловлена необходимостью интеграции автоматического тестирования в процессы разработки для повышения стабильности приложений и сокращения времени на тестирование.

1.2 Обзор исследований

Автоматическое тестирование Android-приложений является одной из ключевых тем в области обеспечения качества программного обеспечения. С. В. Третьякова и А. В. Свищёв провели сравнительный анализ подходов к тестированию мобильных приложений, выделив преимущества инструментов автоматизации, таких как Espresso, в плане скорости и надежности тестирования. В работе также рассматриваются ограничения некоторых популярных инструментов, включая их сложность в интеграции и настройке [1]. С. А. Гонин и С. Полевщиков акцентируют внимание на теоретических аспектах тестирования Android-приложений, подчеркивая важность выбора подходящих инструментов для различных этапов разработки. Отмечено, что инструменты, подобные Espresso, обладают преимуществами в работе с графическим интерфейсом, что делает их особенно востребованными в современной разработке [2]. В исследовании А. М. Синага и соавторов проведено сравнение инструментов тестирования графических интерфейсов. Среди рассмотренных решений Espresso выделяется своей интеграцией с Android Studio и простотой написания тестов. Сделан вывод, что использование таких инструментов значительно повышает производительность команды разработки [3]. Н. А. Воробьев, Л. Н. Бурмин и Ю. А. Степанов в своей работе анализируют средства тестирования мобильных приложений, уделяя внимание их функциональным возможностям и производительности. Espresso упоминается как инструмент, обеспечивающий стабильные результаты и поддерживающий сложные сценарии взаимодействия с пользовательским интерфейсом [4]. Д. Е. Новичков и С. Д. Яковлева в своем исследовании описывают практики и методы обеспечения качества мобильных приложений, включая автоматическое тестирование. Особое внимание уделяется инструментам, упрощающим проверку пользовательского интерфейса, что подтверждает актуальность использования Espresso для тестирования Android-приложений [5].

1.3 Цель исследования

Целью данной работы является анализ возможностей и практического применения инструмента Espresso для автоматического тестирования пользовательского интерфейса Android-приложений. Исследование

направлено на демонстрацию эффективности использования Espresso для повышения качества мобильных приложений, а также разработку рекомендаций по интеграции автоматического тестирования в процесс разработки.

2 Материалы и методы

В исследовании использовался инструмент автоматического тестирования Espresso, интегрированный в Android Studio. Были изучены возможности Espresso для работы с пользовательским интерфейсом, включая проверку взаимодействия с текстовыми полями, кнопками и асинхронными процессами.

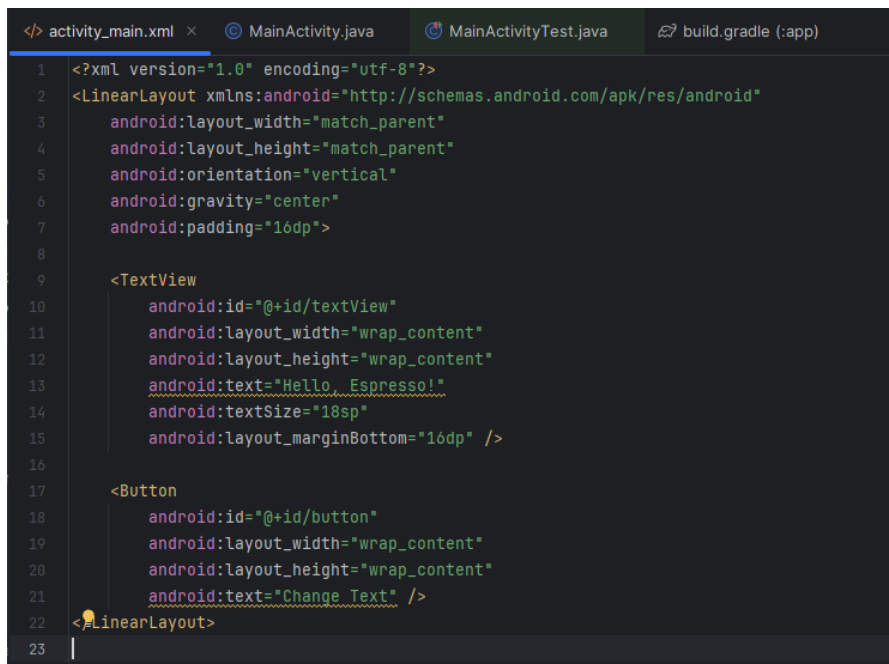
3 Результаты и обсуждения

Espresso – это инструмент автоматического тестирования пользовательского интерфейса, разработанный Google. Библиотека предназначена для проверки взаимодействий пользователя с элементами приложения, таких как текстовые поля, кнопки, выпадающие списки и прочие компоненты. Основное преимущество Espresso заключается в его интеграции с Android Studio, что делает процесс написания и запуска тестов удобным и быстрым. Кроме того, библиотека обеспечивает синхронизацию тестов с пользовательским интерфейсом, что особенно важно для тестирования приложений с асинхронной логикой.

Применение Espresso позволяет сократить время на ручное тестирование, минимизировать вероятность ошибок и обеспечить стабильность пользовательского интерфейса. Благодаря этому инструменту разработчики могут проверять функциональность приложения на разных устройствах и в различных состояниях.

Для демонстрации возможностей Espresso было создано приложение с интерфейсом. Приложение представляет собой классический пример взаимодействия с элементами пользовательского интерфейса.

Основной экран приложения содержит TextView с начальным текстом "Hello, Espresso!" и кнопку "Change Text" (рис. 1). Этот макет позволяет проверить базовые взаимодействия пользователя с элементами интерфейса. Код определяет структуру интерфейса. TextView задает начальный текст, а кнопка отвечает за выполнение действия при нажатии.



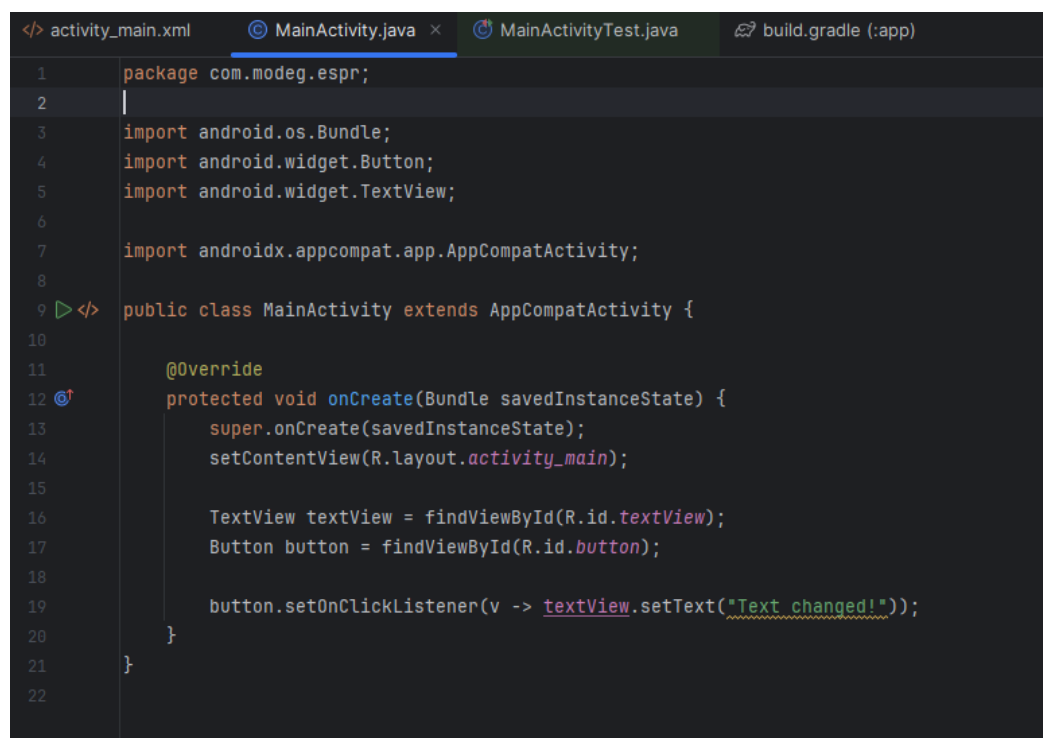
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:gravity="center"
    android:padding="16dp">

    <TextView
        android:id="@+id/textView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello Espresso!"
        android:textSize="18sp"
        android:layout_marginBottom="16dp" />

    <Button
        android:id="@+id/button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Change Text" />
</LinearLayout>
```

Рис. 1. Создание интерфейса

В коде MainActivity.java реализована логика, позволяющая изменить текст TextView при нажатии кнопки. Логика состоит из обработки события `setOnClickListener`, где с помощью метода `setText` устанавливается новый текст (рис. 2).



```
package com.modeg.espr;

import android.os.Bundle;
import android.widget.Button;
import android.widget.TextView;

import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        TextView textView = findViewById(R.id.textView);
        Button button = findViewById(R.id.button);

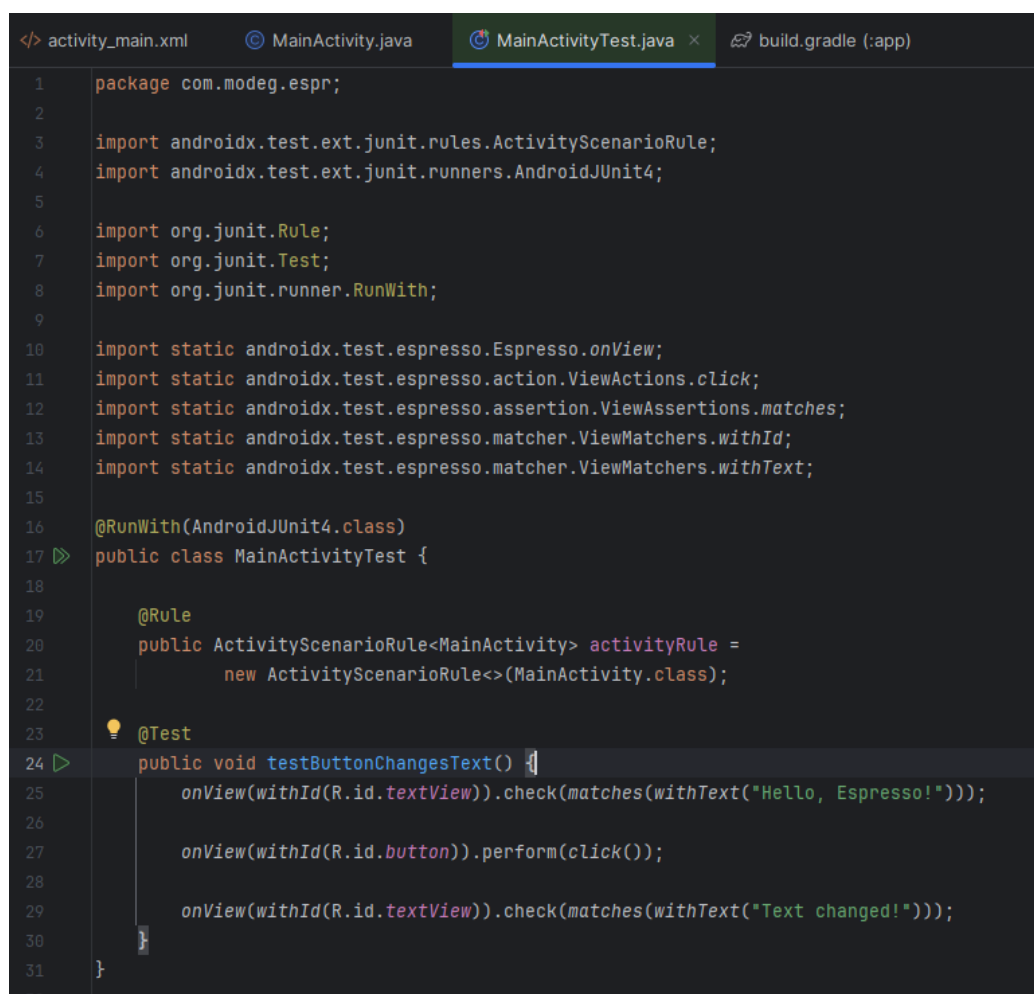
        button.setOnClickListener(v -> textView.setText("Text changed!"));
    }
}
```

Рис. 2. Код MainActivity.java

Для проверки работы приложения были написаны тесты с использованием Espresso. Тесты находятся в файле MainActivityTest.java, расположенном в директории `androidTest`. В тестах используются такие

методы, как `onView`, который позволяет найти элемент интерфейса по его ID, и `check`, который проверяет текущее состояние элемента. Например, тест на проверку текста `TextView` начинается с вызова `onView(withId(R.id.textview))`, который указывает на элемент `TextView`, после чего вызывается метод `check(matches(withText("Hello, Espresso!")))` для проверки начального состояния.

Тест также проверяет корректность изменения текста при нажатии кнопки. Для этого используется метод `perform(click())`, который имитирует нажатие на кнопку, а затем повторно проверяется состояние `TextView`, чтобы убедиться, что текст изменился на "Text changed!" (рис. 3).



```
</> activity_main.xml  MainActivity.java  MainActivityTest.java x  build.gradle (:app)
1  package com.moddeg.espr;
2
3  import androidx.test.ext.junit.rules.ActivityScenarioRule;
4  import androidx.test.ext.junit.runners.AndroidJUnit4;
5
6  import org.junit.Rule;
7  import org.junit.Test;
8  import org.junit.runner.RunWith;
9
10 import static androidx.test.espresso.Espresso.onView;
11 import static androidx.test.espresso.action.ViewActions.click;
12 import static androidx.test.espresso.assertion.ViewAssertions.matches;
13 import static androidx.test.espresso.matcher.ViewMatchers.withId;
14 import static androidx.test.espresso.matcher.ViewMatchers.withText;
15
16 @RunWith(AndroidJUnit4.class)
17 public class MainActivityTest {
18
19     @Rule
20     public ActivityScenarioRule<MainActivity> activityRule =
21         new ActivityScenarioRule<>(MainActivity.class);
22
23     @Test
24     public void testButtonChangesText() {
25         onView(withId(R.id.textview)).check(matches(withText("Hello, Espresso!")));
26
27         onView(withId(R.id.button)).perform(click());
28
29         onView(withId(R.id.textview)).check(matches(withText("Text changed!")));
30     }
31 }
```

Рис. 3. Написание теста

Результаты выполнения тестов показали, что приложение работает корректно, а тесты стабильно проходят на разных устройствах. Этот пример демонстрирует, как с помощью Espresso можно эффективно автоматизировать тестирование пользовательского интерфейса, даже для приложений с простыми сценариями взаимодействия. Такой подход может быть применен и для более сложных приложений, где тестируются другие элементы (рис. 4).

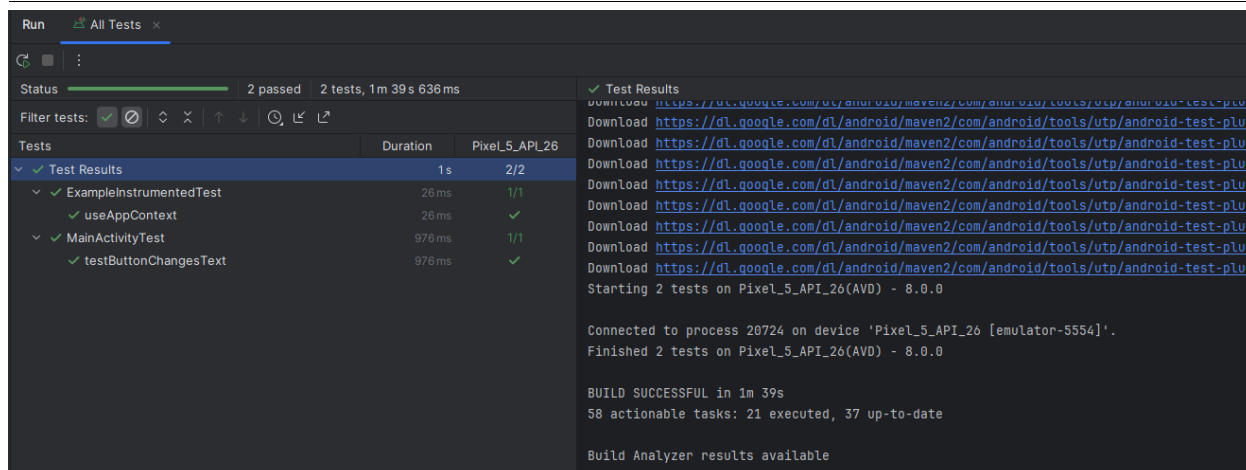


Рис. 4. Успешное тестирование приложения

4 Выводы

В ходе работы было продемонстрировано использование инструмента Espresso для проверки корректности взаимодействий с элементами интерфейса. Созданное тестовое приложение с базовой функциональностью показало, что Espresso позволяет эффективно автоматизировать процесс тестирования, минимизируя ручной труд и сокращая время проверки. С помощью метода синхронизации с пользовательским интерфейсом тесты выполняются стабильно даже в условиях асинхронной логики. Использование Espresso не только упрощает тестирование, но и предоставляет разработчикам возможность поддерживать высокое качество приложений в условиях постоянного обновления функциональности. Этот инструмент может быть рекомендован для интеграции в процесс разработки, как для небольших проектов, так и для сложных корпоративных приложений.

Библиографический список

1. Третьякова С. В., Свищёв А. В. Сравнение подходов к автоматическому тестированию Android приложений // Моя профессиональная карьера. 2023. Т. 3. № 55. С. 68-74.
2. Гонин С. А., Полевщиков И. С. Аналитический обзор методов и средств тестирования Android-приложений // Инновационные технологии: теория, инструменты, практика. 2017. Т. 1. С. 20-25.
3. Sinaga A. M., Pratama Y., Siburian F. O., S K. J. F. P. Comparison of Graphical User Interface Testing Tools // Journal of Computer Networks, Architecture and High Performance Computing. 2021. Т. 3. № 2. С. 123-134.
4. Воробьёв Н. А., Бурмин Л. Н., Степанов Ю. А. Сравнительный анализ средств тестирования мобильных приложений // Евразийский союз ученых. 2020. № 6-1 (75). С. 36-38.
5. Новичков Д. Е., Яковлева С. Д. Тестирование мобильных приложений: практики и методы обеспечения качества.