

Отправка данных в приложении Android

Андриенко Иван Сергеевич

Приамурский государственный университет имени Шолом-Алейхема

Студент

Аннотация

В данной статье рассмотрен процесс отправки данных в Android-приложении с использованием FileProvider. Описаны основные этапы настройки приложения, включая создание контента, его подготовку к отправке и вызов намерения для передачи данных через сторонние приложения. В качестве примера представлена реализация простого приложения, где пользователь может отправить изображение или текст с использованием Android-инструментов. Показано, как с помощью FileProvider можно реализовать функционал обмена данными между приложениями.

Ключевые слова: Android, FileProvider, отправка данных, intent, контент, интеграция, обмен данными.

Sending data in the Android app

Andrienko Ivan Sergeevich

Sholom-Aleichem Priamursky State University

Student

Abstract

This article discusses the process of sending data in an Android application using the Internet. The main steps of setting up an application are described, including creating content, preparing it for sending, and calling an intent to transfer data through third-party applications. As an example, the implementation of a simple application is presented, where the user can send an image or text using Android tools. It is shown how intent can be used to implement the functionality of data exchange between applications.

Keywords: Android, FileProvider, sending data, intent, content, integration, data exchange.

1 Введение

1.1 Актуальность

С увеличением количества мобильных приложений растет потребность в обеспечении удобства обмена данными между ними. В Android-интенты предоставляют разработчикам простой и удобный способ обмена информацией, позволяя передавать данные между различными приложениями и службами. Это делает интеграцию возможностей обмена данными более эффективной и гибкой. В условиях развития мобильных

технологий способность приложений к взаимодействию и обмену данными становится важным аспектом для улучшения пользовательского опыта.

1.2 Обзор исследований

В.В. Свиридов, П.И. Карасев и М.А. Ал-Амееди в своём исследовании анализируют существующие решения для безопасной передачи данных между Android-приложениями на одном устройстве. Авторы акцентируют внимание на проблемах, связанных с передачей данных через механизмы ContentProvider и FileProvider, и предлагают различные подходы к повышению безопасности при обмене данными между приложениями [1]. К. Гейер, Д.А. Эллис, Х. Шоу и Б.И. Дэвидсон в своём исследовании рассматривают разработку и использование открытых приложений и инструментов для измерения, количественного анализа и визуализации использования технологий. Авторы предлагают новые методы и инструменты для анализа данных, собираемых с мобильных устройств, что позволяет более глубоко изучить поведение пользователей и его влияние на безопасность данных [2]. Е.А. Трушкина и Д.Е. Намиот исследуют возможности обмена данными между мобильными устройствами без необходимости создания соединений. В своей работе они рассматривают альтернативные методы передачи данных, которые могут быть полезны в условиях ограниченной или отсутствующей сети, уделяя внимание вопросам безопасности и эффективности обмена [3]. К.А. Татарканов, А.В. Балкизова и Л.С. Мазанова в своём исследовании обсуждают роль механизма Intent и Intent Filter в мобильной разработке под Android. Авторы анализируют, как Intent и Intent Filter используются для взаимодействия между компонентами приложений, а также их влияние на безопасность передачи данных [4]. Мевлют И.Ш., Абляев М.Р., Ворожейкин Р.В. и Адильшаева Э.И. в своей статье выделяют преимущества использования механизма намерений Intent в Android-приложениях. Авторы подчеркивают, что Intent не только упрощает взаимодействие между компонентами приложений, но и может повысить безопасность передачи данных за счет точной настройки фильтров намерений [5].

1.3 Цель исследования

Цель исследования — разработать подход к интеграции функции отправки данных в Android-приложении с использованием FileProvider. Это включает создание контента, его подготовку и вызов намерения для отправки данных через сторонние приложения.

2 Материалы и методы

Для исследования использовались Android Studio, язык программирования Java и инструменты Android для работы с интентами. В ходе разработки был создан интерфейс для выбора и отправки данных, реализованы методы подготовки контента и вызова намерений.

3 Результаты и обсуждения

Использование FileProvider позволяет обеспечить безопасность и контроль над файлами, передаваемыми другим приложениям. Это особенно важно в условиях, когда приложения взаимодействуют с чувствительными данными. Благодаря FileProvider разработчики могут передавать файлы, не подвергая их риску несанкционированного доступа, что значительно повышает уровень безопасности Android-приложений.

Также этот класс позволяет создавать Uri для файла, который может быть передан другим приложениям, обеспечивая при этом контроль над доступом к файлам. Для настройки FileProvider в манифесте приложения необходимо добавить элемент `<provider>`, который определяет настройки провайдера. В нашем случае была указана следующая конфигурация (рис. 1). Данный код включает важные параметры, такие как `android:authorities`, который задает уникальный идентификатор для провайдера. Важно, чтобы идентификатор соответствовал формату `package_name.fileprovider`, так как это необходимо для правильного взаимодействия с другими приложениями. Параметр `android:grantUriPermissions="true"` позволяет предоставлять временные права доступа к файлам, что важно для обеспечения безопасности.

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <manifest xmlns:android="http://schemas.android.com/apk/res/android"
3    xmlns:tools="http://schemas.android.com/tools">
4
5
6    <application
7      android:allowBackup="true"
8      android:dataExtractionRules="@xml/data_extraction_rules"
9      android:fullBackupContent="@xml/backup_rules"
10     android:icon="@mipmap/ic_launcher"
11     android:label="@string/label_send_data"
12     android:roundIcon="@mipmap/ic_launcher_round"
13     android:supportRtl="true"
14     android:theme="@style/Theme.SendData"
15     tools:targetApi="31">
16      <activity
17        android:name=".MainActivity"
18        android:exported="true">
19        <intent-filter>
20          <action android:name="android.intent.action.MAIN" />
21
22          <category android:name="android.intent.category.LAUNCHER" />
23        </intent-filter>
24      </activity>
25
26      <provider
27        android:name="androidx.core.content.FileProvider"
28        android:authorities="${applicationId}.fileprovider"
29        android:exported="false"
30        android:grantUriPermissions="true">
31        <meta-data
32          android:name="android.support.FILE_PROVIDER_PATHS"
33          android:resource="@xml/file_paths" />
34      </provider>
35
36    </application>
37
38  </manifest>
```

Рис. 1. Настройка FileProvider в манифесте

Чтобы определить к каким именно директориям будет предоставлен доступ через FileProvider создадим файл `file_paths.xml`. Этот файл

размещается в каталоге `res/xml` и должен включать описание путей к файлам, которые могут быть доступны. В данном примере элемент `<files-path>` указывает на директорию `images` внутри каталога `files`, который относится к внутреннему хранилищу приложения. Параметр `name` определяет имя, которое будет использовано при создании `Uri` для доступа к файлам (рис. 2).

```

1 <paths xmlns:android="http://schemas.android.com/apk/res/android">
2   <files-path
3     name="shared_images"
4     path="." />
5 </paths>

```

Рис. 2. Создание `file_paths.xml`

В макете создадим изображение и кнопку. Кнопка "Поделиться" предназначена для инициации процесса отправки изображения через доступные приложения на устройстве, а само изображение демонстрирует содержимое, которое пользователь может отправить (рис. 3).

```

7   android:layout_height="match_parent"
8   android:background="@color/design_default_color_secondary_variant"
9   tools:context=".MainActivity">
10
11   <ImageView
12     android:id="@+id/imageView"
13     android:layout_width="201dp"
14     android:layout_height="171dp"
15     android:contentDescription="Sample Image"
16     android:src="@drawable/ic_launcher_foreground"
17     app:layout_constraintBottom_toBottomOf="parent"
18     app:layout_constraintEnd_toEndOf="parent"
19     app:layout_constraintHorizontal_bias="0.497"
20     app:layout_constraintStart_toStartOf="parent"
21     app:layout_constraintTop_toTopOf="parent"
22     app:layout_constraintVertical_bias="0.398" />
23
24   <Button
25     android:id="@+id/shareButton"
26     android:layout_width="wrap_content"
27     android:layout_height="wrap_content"
28     android:layout_gravity="center_horizontal"
29     android:text="Поделиться"
30     app:layout_constraintBottom_toBottomOf="parent"
31     app:layout_constraintEnd_toEndOf="parent"
32     app:layout_constraintStart_toStartOf="parent"
33     app:layout_constraintTop_toTopOf="parent"
34     app:layout_constraintVertical_bias="0.655" />
35
36 </androidx.constraintlayout.widget.ConstraintLayout>

```

Рис. 3. Реализация главного макета приложения

После настройки FileProvider был реализован функционал для отправки данных. Пользовательское взаимодействие начинается с нажатия на кнопку "Поделиться", которая инициирует создание Intent для отправки данных. В методе обработки нажатия создается Uri для выбранного файла с помощью FileProvider.getUriForFile(). После получения Uri создается Intent для отправки данных. Intent.ACTION_SEND указывает на то, что приложение хочет отправить данные, а Intent.EXTRA_STREAM добавляет Uri к отправляемым данным. Важно отметить использование флага, который временно предоставляет приложению-получателю права на чтение содержимого по указанному Uri (рис. 4).

```
21     private ImageView imageView; 1 usage
22     private Button shareButton; 2 usages
23
24     @Override
25     protected void onCreate(Bundle savedInstanceState) {
26         super.onCreate(savedInstanceState);
27         setContentView(R.layout.activity_main);
28
29         imageView = findViewById(R.id.imageView);
30         shareButton = findViewById(R.id.shareButton);
31
32         shareButton.setOnClickListener(new View.OnClickListener() {
33             @Override
34             public void onClick(View v) {
35                 Uri imageUri = FileProvider.getUriForFile(
36                     context, MainActivity.this,
37                     authority: "com.modeg.senddata.fileprovider",
38                     new File(getFilesDir(), child: "sample_image.jpg")
39                 );
40
41                 Intent shareIntent = new Intent();
42                 shareIntent.setAction(Intent.ACTION_SEND);
43                 shareIntent.putExtra(Intent.EXTRA_STREAM, imageUri);
44                 shareIntent.setType("image/jpeg");
45
46                 startActivity(Intent.createChooser(shareIntent, title: "Поделиться изображением"));
47             }
48         });
49     }
50 }
```

Рис. 4. Создание макета

Приложение было протестировано на нескольких устройствах с различными версиями Android. Результаты показали, что настройка и отправка данных работают корректно, без возникновения ошибок. Отправленные изображения успешно открывались в приложениях, поддерживающих получение данных через Intent. Тестирование подтвердило стабильность и безопасность предложенного решения. (рис. 5, 6).

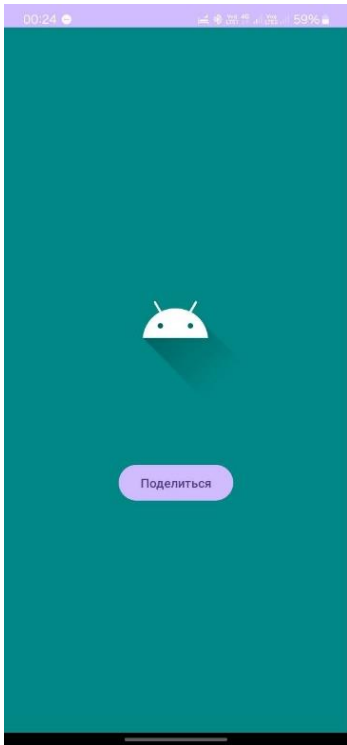


Рис. 5. Тестирование приложения

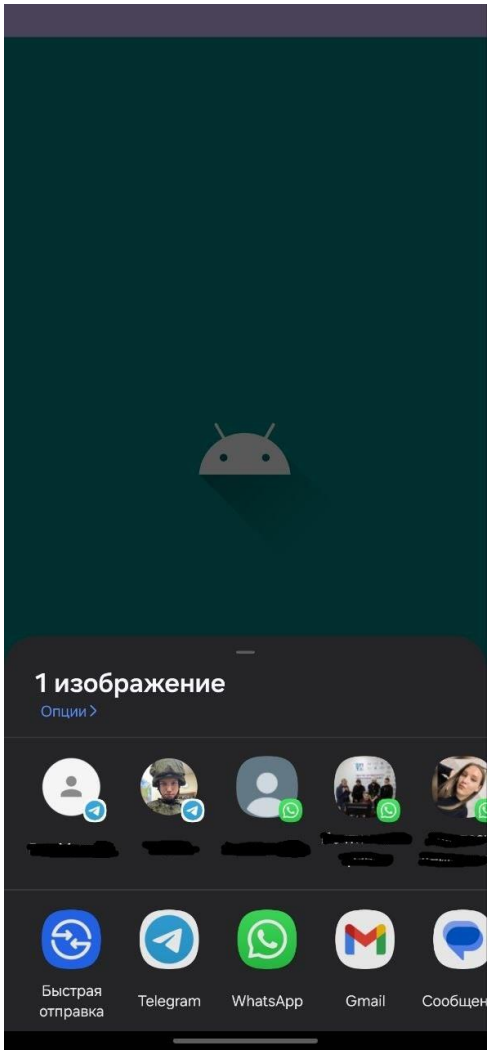


Рис. 6. Тестирование приложения

Выводы

В данной статье рассмотрен процесс отправки данных в Android-приложении с использованием FileProvider. Показаны основные этапы настройки и реализации, включая создание контента, настройку доступа к файлам и отправку данных. Данный подход позволяет интегрировать функции обмена данными в мобильные приложения, улучшая их взаимодействие и удобство использования.

Библиографический список

1. Свиридов В.В., Карасев П.И., Ал-Амееди М.А. Анализ решений для безопасной передачи данных между Android-приложениями на одном устройстве // В сборнике: Информационная безопасность и защита персональных данных. Проблемы и пути их решения. Брянск, 2023. С. 233-235.
2. Geyer K., Ellis D.A., Shaw H., Davidson B.I. Open-source smartphone app and tools for measuring, quantifying, and visualizing technology use // Behavior Research Methods. 2022. Vol. 54. No. 1. P. 1-12.
3. Трушкина Е.А., Намиот Д.Е. Обмен данными между мобильными устройствами без организации соединений // International Journal of Open Information Technologies. 2022. Т. 10. № 9. С. 99-114.
4. Татарканов К.А., Балкизова А.В., Мазанова Л.С. Роль Intent и Intent Filter в мобильной разработке под операционную систему Android // В сборнике: Новая наука - новые возможности. Сборник статей IV Международного научно-исследовательского конкурса. Петрозаводск, 2022. С. 53-58.
5. Мевлют И.Ш., Абляев М.Р., Ворожейкин Р.В., Адильшаева Э.И. Преимущества использования механизма намерений Intent // Таврический научный обозреватель. 2016. № 6 (11). С. 213-216.
6. URL: <https://developer.android.com/training/sharing/send> (дата обращения 12.08.2024)