Разрешения

Разрешения приложений помогают защищать конфиденциальность пользователей, ограничивая доступ приложения к некоторым вещам:

- **Данные**, такие как состояние системы или контактная информация пользователей.
- **Действия**, такие как подключение к сопряженному устройству или запись звука.

Если приложение содержит функции, которым требуется доступ к ограничиваемым данным или действиям, прежде всего следует подумать нет ли других путей для выполнения этих функций без необходимости требовать разрешения. Если других путей нет, то нужно заявить соответствующие разрешения в манифесте приложения. Некоторые из них, известные как *разрешения времени установки*, автоматически предоставляются при установке приложения. Другие – *разрешения времени выполнения*, дополнительно требуют запроса разрешения во время выполнения.

Типы разрешений

Android разделяет разрешения на несколько типов, в зависимости от степени их влияния на пользовательские данные и другие приложения в системе.

- **Разрешения времени установки**, также известные как **обычные**. Такие разрешения имеют ограниченное влияние на доступ к данным или выполнение чувствительных действий, риск для конфиденциальности пользователя или влияния на другие приложения очень небольшой. Их достаточно объявить в манифесте, и при установке система автоматически предоставит их приложению. Пользователь может посмотреть список таких разрешений на странице приложения в магазине приложений.
- **Разрешения времени выполнения**, также известные как **опасные**, дают приложению доступ к ограниченным данным или позволяют выполнять действия, которые более существенно влияют на систему и другие приложения, поэтому необходимо запросить разрешение у пользователя на их выполнение.
- **Специальные разрешения** могут получать только сама платформа и OEM-производители. Как правило, это особо чувствительные действия, которые могут серьезно навредить конфиденциальности пользователя или работе других приложений, например, запись звука во время разговора по телефону или снимки экрана во время работы других приложений.

Использование разрешений

Если приложение требует какие-либо разрешения, их нужно объявить в файле манифеста, включив соответствующий элемент `<uses-permission>`. Например, приложение, которому требуется доступ к камере, должно иметь следующую строку в файле *AndroidManifest.xml*:

```
<manifest ...>
    <uses-permission android:name="android.permission.CAMERA"/>
    <application ...>
</manifest>
```

Некоторые разрешения, например `CAMERA`, позволяют приложению получать доступ к оборудованию, которое может присутствовать не на всех устройствах. Если приложение

декларирует одно из таких разрешений, связанных с оборудованием, стоит оценить сможет ли такое приложение работать на устройстве, у которого нет этого оборудования, или же будут ограничены только некоторые функции. Если присутствие оборудования является опциональным, то можно объявить его в манифесте следующим образом:

```
<manifest ...>
    <uses-permission android:name="android.permission.CAMERA"/>
    <uses-feature android:name="android.hardware.camera"
                  android:required="false" />
    <application ...>
</manifest>
```

Чтобы проверить, имеет ли устройство определенное оборудование, используется метод `hasSystemFeature()`:

```
if (applicationContext.packageManager.hasSystemFeature(
    PackageManager.FEATURE_CAMERA_FRONT)) {
    // Оборудование присутствует
}
else {
    // Оборудования нет, отключаются связанные с ним функции
}
```

Если ключевой функционал приложения зависит от наличия оборудования, и при его отсутствии приложение функционировать не может, то следует установить атрибут в `android:required="true"`, и в таком случае магазин приложений не даст установить приложение на устройства, в которых данное оборудование отсутствует.

Запрос разрешения времени выполнения

Запрос у пользователя разрешения времени выполнения возможен только в том случае, если данное разрешение объявлено в файле манифеста! Рекомендуемый порядок действий для запроса разрешения следующий:

1. Подождите, пока пользователь начнёт выполнять действие, требующее доступа к конкретным личным данным.
2. Проверьте, нет ли уже разрешения на выполнение такого действия, и если да, то просто выполните его. Следует проверять наличие разрешения каждый раз, поскольку разрешение может быть отозвано в любой момент! Чтобы проверить наличие разрешения следует использовать метод `checkSelfPermission()`, он возвращает константы `PERMISSION_GRANTED` или `PERMISSION_DENIED`, в зависимости от того имеет ли приложение разрешение:

   ```
   if (ContextCompat.checkSelfPermission(this,
       android.Manifest.permission.CAMERA) != PackageManager.PERMISSION_GRANTED) {
       // Разрешения нет, надо запросить
   }
   else {
       // Разрешение есть, выполнение действия
   }
   ```

3. Проверьте, должно ли ваше приложение показать пользователю обоснование, объясняющее, почему приложению необходимо предоставление разрешения. В диалоговом окне запроса разрешений, отображаемом системой, указано какое

разрешение требуется приложению, но не указано, почему. В некоторых случаях это может сбить пользователя с толку. Хорошей идеей будет объяснить пользователю, почему приложению нужны эти разрешения до их запроса. Если метод `checkSelfPermission()` возвращает значение `PERMISSION_DENIED`, вызовите метод `shouldShowRequestPermissionRationale()`. Если этот метод возвращает `true`, покажите пользователю обучающий пользовательский интерфейс. Обычно при первом запросе разрешения возвращается `false`, а при последующих (если пользователь не запретил явно) – `true`.

4. Запросите разрешение. Как и в случае с отображением другой активности, здесь есть два варианта – современный и устаревший:
   a. Современный вариант: создать слушателя ответа:

```
private val requestPermissionLauncher = registerForActivityResult(
    ActivityResultContracts.RequestPermission()) { isGranted ->
    if (isGranted)
        // Пользователь дал разрешение
    else
        // Пользователь отказал в разрешении
}
```

   И в нужный момент сделать запрос разрешения:

```
requestPermissionLauncher.launch(android.Manifest.permission.CAMERA)
```

   Ответ придёт в слушатель ответа, дальнейшие действия предпринимаются там.

   b. **Устаревший вариант**: использовать функцию `requestPermissions()`:

```
val MY_PERMISSION_REQUEST_CODE = 1
ActivityCompat.requestPermissions(this,
                                  arrayOf( список требуемых разрешений ),
                                  MY_PERMISSION_REQUEST_CODE)
```

   Когда пользователь подтверждает или отклоняет разрешение, система асинхронно вызывает функцию обратного вызова `onRequestPermissionsResult`. Все ответы на запросы разрешения поступают в эту функцию, поэтому для их различения используется числовая константа, передаваемая в функцию `requestPermissions` (в примере выше это константа `MY_PERMISSION_REQUEST_CODE`):

```
// В эту функцию система пришлёт ответ на запрос
override fun onRequestPermissionsResult(
                requestCode: Int,
                permissions: Array,
                grantResults: IntArray) {

    // Это наш запрос на камеру?
    if (requestCode == MY_CAMERA_REQUEST_CODE) {

        if ((grantResults.isNotEmpty() &&
             grantResults[0] == PackageManager.PERMISSION_GRANTED))
             // Разрешение получено, выполняем действие
```

```
                else
                    // Пользователь отказал в разрешении
                return
            }
            super.onRequestPermissionsResult(requestCode,
                                             permissions,
                                             grantResults)
        }
```

Система отобразит запрос, и пользователь сможет принять решение о предоставлении разрешения (или отказе).

Устаревший вариант больше не рекомендуется к применению, но важно понимать как он работает, потому что в старом унаследованном коде он по прежнему может встречаться.

Таким образом, полный цикл запроса разрешения на доступ к камере может выглядеть следующим образом:

```
// Проверка: вдруг разрешение уже есть?
if (ContextCompat.checkSelfPermission(this,
    android.Manifest.permission.CAMERA) != PackageManager.PERMISSION_GRANTED) {

    // Разрешения пока нет, нужно ли показать пояснения?
    if (ActivityCompat.shouldShowRequestPermissionRationale(this,
        android.Manifest.permission.CAMERA)) {
        // Да, показываем пояснения пользователю
        ...
    }
    else {
        // Пояснений не требуется, запрашиваем разрешение
        requestPermissionLauncher.launch(android.Manifest.permission.CAMERA)
    }
}
else {
    // Разрешение уже есть, выполняем действие
    ...
}


// В эту функцию система пришлёт ответ на запрос
private val requestPermissionLauncher = registerForActivityResult(
    ActivityResultContracts.RequestPermission()) { isGranted ->
    if (isGranted)
        // Пользователь дал разрешение
    else
        // Пользователь отказал в разрешении
}
```

Отказ от разрешения

Если какое-то разрешение больше не нужно для работы приложения, можно отозвать его – для этого нужно передать список отзываемых разрешений в функцию `revokeSelfPermissionsOnKill()`. После закрытия приложения разрешение будет отозвано, и в следующий раз его нужно будет запросить заново. Такое действие может понадобиться

если ситуации, в которых использовалось разрешение, стали кардинально иными, и нужно объяснить пользователю нововведения (например, после существенного обновления функционала программы при выходе новой версии).

Если приложение не запускалось в течение длительного времени (обычно около 6 месяцев), то система отзовёт у приложения все выданные разрешения. Это делается для того чтобы пользователь, который долго не пользовался приложением, смог заново принять решения о доверии приложению. Изменить такое поведение системы программно нельзя, но сам пользователь может в системных настройках отключить отзыв разрешений у конкретного приложения.

Пользователь также может в любой момент зайти в системные настройки и отозвать у приложения разрешения, которые ранее были выданы. Поэтому приложение никогда не должно рассчитывать что разрешения есть, проверка наличия должна проводиться всегда!

Если пользователь отозвал разрешение прямо во время работы программы, система завершит работу программы.

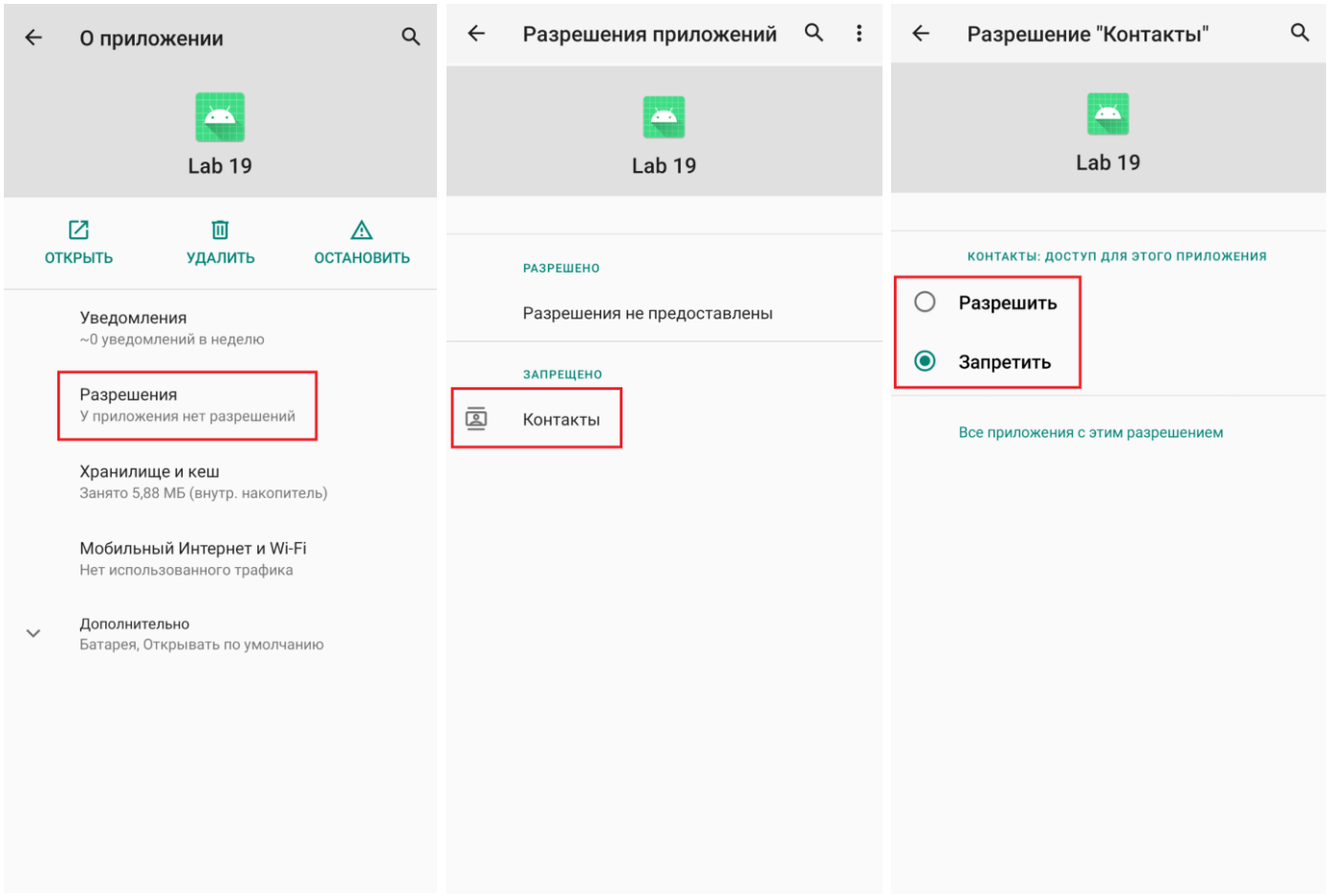
Разрешения в настройках приложения

В современных версиях Android если пользователь за всё время с момента установки дважды отказался предоставить приложению какое-либо разрешение, то система больше будет показывать ему соответствующий запрос – подразумевается, что пользователь не желает больше видеть такие запросы. Даже если приложение пытается сделать такой запрос, оно будет автоматически получать отказ.

В такой ситуации нужно показать пользователю объяснение что при отсутствии этого разрешения выполнение данной операции невозможно. При этом следует с уважением относиться к выбору пользователя. При желании можно предоставить ему возможность быстро попасть в системные настройки, где пользователь сможет вручную поменять статус разрешений. Для этого нужно использовать неявное намерение, которое обработает сама система:

```
val intent = Intent(
    Settings.ACTION_APPLICATION_DETAILS_SETTINGS,
    Uri.parse("package:$packageName"))
startActivity(intent)
```

Это намерение откроет страницу приложения, и там в разделе «Разрешения» пользователь сможет изменить свой выбор:



Задание

Разработать приложение для получения списка контактов из телефонной книги пользователя. Для этого необходимо разрешение `READ_CONTACTS`.

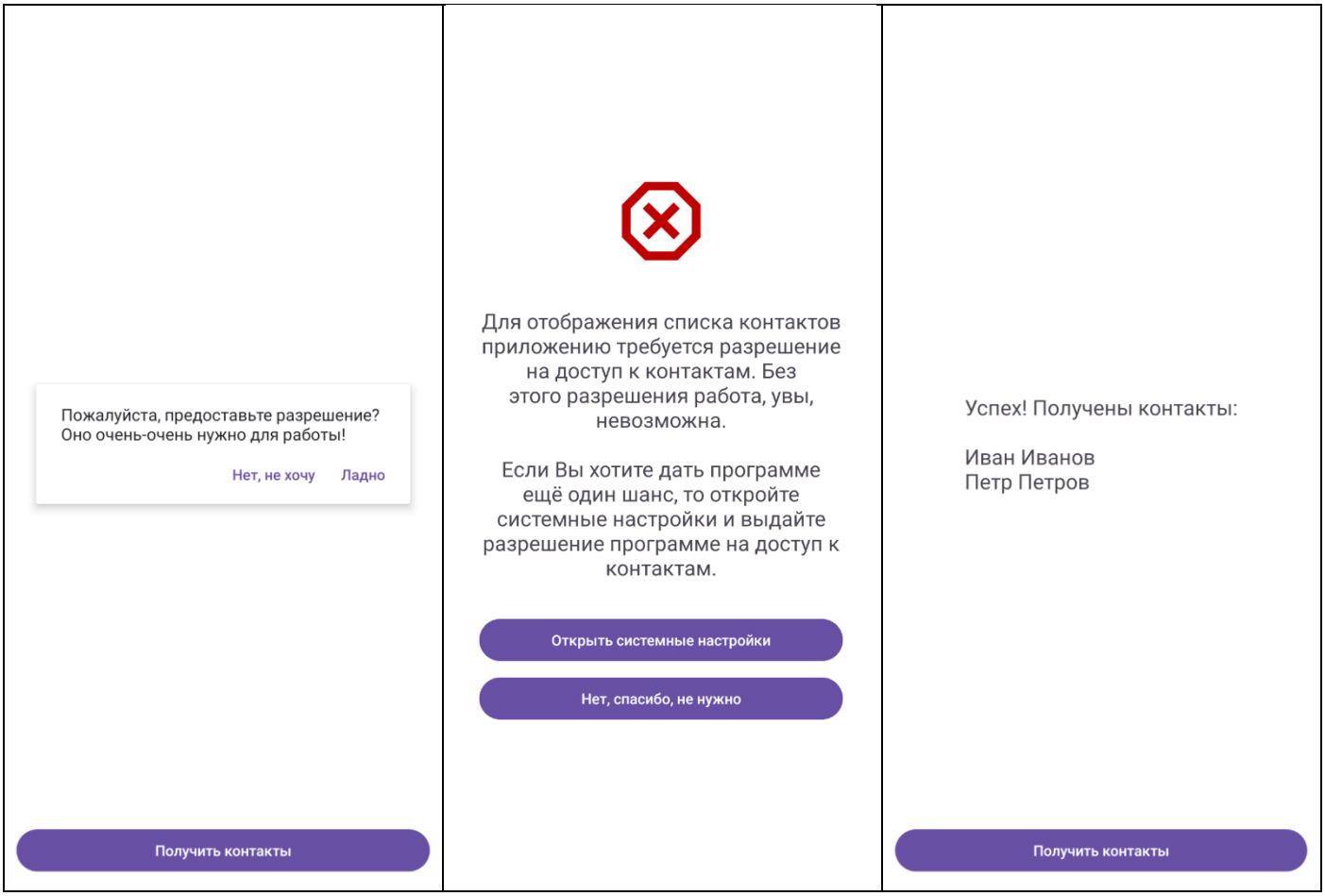
По нажатию кнопки «Получить контакты» приложение должно проверить наличие разрешения, и если оно отсутствует – начать процесс его получения.

Если система считает что нужно показать пользователю обоснование необходимости разрешения – показать такое обоснование в какой-либо форме.

Если пользователь отказал в разрешении – показать ему информацию о том, что без разрешения функционал приложения ограничен, и дать возможность перейти в системные настройки приложения.

Если разрешение получено – считать имена контактов из адресной книги и отобразить их на экране в какой-либо форме.

Экраны приложения могут выглядеть примерно таким образом:



Чтение контактов

Для чтения контактов (и многих других операций в Android) используется механизм контент-провайдеров. Он позволяет обмениваться данными между разными приложениями, и при этом не вникать в то где и как именно эти данные хранятся – в базе данных, в виде локальных файлов или вообще в облаке. Данные запрашиваются при помощи метода `query()`, и возвращаются в табличном виде, почти как в базе данных – строки с данными, столбцы определённого типа.

Изучение контент-провайдеров выходит за рамки данной темы, поэтому можно просто использовать следующую функцию для получения контактов:

```
@SuppressLint("Range")
fun getContacts(): List<String> {
    val result = mutableListOf<String>()

    // Запрос списка контактов
    val cur = contentResolver.query(ContactsContract.Contacts.CONTENT_URI,
        null, null, null, null)

    if (cur != null) {
        // Определение номера столбца, содержащего имя контакта
        val colName = cur.getColumnIndex(ContactsContract.Contacts.DISPLAY_NAME)

        while (cur.moveToNext()) {
            val name = cur.getString(colName)
            result.add(name)
        }
```

```
        cur.close()
    }

    return result
}
```

Функция предполагает что разрешение `READ_CONTACTS` уже получено, в противном случае будет выброшено исключение.