РАБОТА с ФАЙЛАМИ

В этом уроке

1. [Выберите внутреннее или внешнее хранилище](http://www.fandroid.info/sohranenie-fajlov/#1)
2. [Получите разрешения для внешних накопителей](http://www.fandroid.info/sohranenie-fajlov/#2)
3. [Сохраните файл во внутреннем хранилище](http://www.fandroid.info/sohranenie-fajlov/#3)
4. [Сохраните файл на внешнем накопителе](http://www.fandroid.info/sohranenie-fajlov/#4)
5. [Запросите свободное пространство](http://www.fandroid.info/sohranenie-fajlov/#5)
6. [Удаление файла](http://www.fandroid.info/sohranenie-fajlov/#6)

Вы также должны прочитать

* [Использование внутренней памяти](http://developer.android.com/guide/topics/data/data-storage.html#filesInternal)
* [Использование внешних накопителей](http://developer.android.com/guide/topics/data/data-storage.html#filesExternal)

Android использует файловую систему, похожую на систему на основе дисковых файловых систем на других платформах. Этот урок описывает, как работать с файловой системой Android, чтобы читать и записывать файлы, используя [File](http://developer.android.com/reference/java/io/File.html) API.

[File](http://developer.android.com/reference/java/io/File.html) объект подходит для чтения или записи больших объемов данных с начала до конца, не пропуская ничего вокруг. Например, это хорошо подходит для файлов изображений или для обмена чем-нибудь по сети.

Этот урок показывает, как выполнять основные задачи связанные с файлами в вашем приложении. Урок предполагает, что вы знакомы с основами файловой системы Linux и стандартным API файлового ввода/вывода в [java.io](http://developer.android.com/reference/java/io/package-summary.html).

Выберите внутреннее или внешнее хранилище

Все Android устройства имеют две области хранения файлов: «внутреннее» и «внешнее» хранилище. Эти названия пришли из первых дней Android, когда предлагалось, что большинство устройств предоставляют встроенную постоянную неизменную память (внутреннее хранилище), а также съемный носитель, такое как микро SD карта(внешнее хранилище). Некоторые устройства делят места постоянного хранения на «внутренние» и «внешние» разделы, т.е. даже без съемного носителя, всегда есть два хранилища, и поведение API для внешнего хранилища такое же, не зависимо от того, является ли оно съемным или нет. Следующие списки обобщают факты о каждом хранилище.

**Внутреннее хранилище:**

* Всегда доступно.
* Файлы, сохраненные здесь, по умолчанию, доступны лишь вашему приложению.
* Когда пользователь удаляет ваше приложение, система удаляет все файлы вашего приложения из внутренней памяти.

Внутреннее хранилище лучше, когда вы хотите быть уверены, что ни пользователь, ни другие приложения не могут получить доступ к файлам.

**Внешнее хранилище:**

* Не всегда доступно, так как пользователь может смонтировать внешний накопитель в качестве USB хранилища, и в некоторых случаях вынуть его из устройства.
* Чтение разрешено всем, поэтому файлы, сохраненные здесь, можно прочитать без вашего контроля.
* Когда пользователь удаляет ваше приложение, система удаляет файлы вашего приложения отсюда, только если вы сохраните их в каталоге из[getExternalFilesDir()](http://developer.android.com/reference/android/content/Context.html#getExternalFilesDir(java.lang.String)).

Внешние накопители лучшее место для файлов, которые не требуют ограничения доступа, и для файлов, которыми вы хотите поделиться с другими приложениями, или хотите позволить пользователю получить доступ к ним на компьютере.

**Полезный совет:** Хотя приложения устанавливаются во внутреннее хранилище по умолчанию, можно указать [android:installLocation](http://developer.android.com/guide/topics/manifest/manifest-element.html" \l "install) атрибут в манифесте, чтобы ваше приложение могло быть установлено на внешний накопитель. Пользователи ценят эту опцию, когда размер APK очень большой, и у них есть внешнее пространство для хранения, превышающее внутреннюю память. Для получения дополнительной информации, см. [Путь установки приложения](http://developer.android.com/guide/topics/data/install-location.html).

Получите разрешения для внешних накопителей

Для записи на внешний накопитель, вы должны запросить [WRITE\_EXTERNAL\_STORAGE](http://developer.android.com/reference/android/Manifest.permission.html#WRITE_EXTERNAL_STORAGE) разрешение в вашем [файле манифеста](http://developer.android.com/guide/topics/manifest/manifest-intro.html):

|  |  |
| --- | --- |
| 1  2  3  4 | <manifest ...>      <uses-permission android:name="android.permission.WRITE\_EXTERNAL\_STORAGE" />      ...  </manifest> |

**Внимание:** В настоящее время, все приложения имеют возможность читать с внешних накопителей без специального разрешения. Тем не менее, это изменится в будущем релизе. Если ваше приложение должно прочитать внешний накопитель (но не писать в него), то вам нужно будет объявить [READ\_EXTERNAL\_STORAGE](http://developer.android.com/reference/android/Manifest.permission.html#READ_EXTERNAL_STORAGE) разрешение. Чтобы убедиться, что ваше приложение продолжало работать, как и ожидалось, вы должны объявить это разрешение сейчас, прежде чем изменения вступят в силу.

|  |  |
| --- | --- |
| 1  2  3  4 | <manifest ...>      <uses-permission android:name="android.permission.READ\_EXTERNAL\_STORAGE" />      ...  </manifest> |

Однако, если ваше приложение использует [WRITE\_EXTERNAL\_STORAGE](http://developer.android.com/reference/android/Manifest.permission.html#WRITE_EXTERNAL_STORAGE) разрешение, то также неявно имеет разрешение на чтение внешнего хранилища.

Вам не нужно никаких разрешений для сохранения файлов во внутреннее хранилище. Ваше приложение всегда имеет разрешение на чтение и запись файлов в каталоге внутреннего хранилища.

Сохраните файл во внутреннем хранилище

При сохранении файла во внутреннее хранилище, вы можете запросить соответствующий каталог, в виде объекта [File](http://developer.android.com/reference/java/io/File.html) , вызвав один из двух методов:

[**getFilesDir()**](http://developer.android.com/reference/android/content/Context.html#getFilesDir())

Возвращает [File](http://developer.android.com/reference/java/io/File.html) представляющий собой внутренний каталог вашего приложения.

[**getCacheDir()**](http://developer.android.com/reference/android/content/Context.html#getCacheDir())

Возвращает [File](http://developer.android.com/reference/java/io/File.html) представляющий собой внутренний каталог для временных файлов кэша вашего приложения. Будьте уверены, удалять файлы по одному больше не нужено, и реализуйте разумный предел размера объема памяти, который вы будете использовать в какой-либо момент времени, например, 1 Мб. Если в системе возникает нехватка места, она может удалить ваши файлы кэша без предупреждения.

Чтобы создать новый файл в одном из этих каталогов, вы можете использовать [File()](http://developer.android.com/reference/java/io/File.html" \l "File(java.io.File,%20java.lang.String))конструктор, передав [File](http://developer.android.com/reference/java/io/File.html)предоставленный одним из выше указанных методов, который указывает на ваш внутренний каталог. Например:

|  |  |
| --- | --- |
| 1 | File file = new File(context.getFilesDir(), filename); |

В качестве альтернативы, вы можете вызвать [openFileOutput()](http://developer.android.com/reference/android/content/Context.html" \l "openFileOutput(java.lang.String,%20int)) для получения [FileOutputStream](http://developer.android.com/reference/java/io/FileOutputStream.html) , который пишет в файл в вашем внутреннем каталоге. Например, вот как записать текст в файл:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11 | String filename = "myfile";  String string = "Hello world!";  FileOutputStream outputStream;    try {    outputStream = openFileOutput(filename, Context.MODE\_PRIVATE);    outputStream.write(string.getBytes());    outputStream.close();  } catch (Exception e) {    e.printStackTrace();  } |

Или, если вам нужно кэшировать некоторые файлы, то следует использовать [createTempFile()](http://developer.android.com/reference/java/io/File.html" \l "createTempFile(java.lang.String,%20java.lang.String)). Например, следующий метод извлекает имя файла из [URL](http://developer.android.com/reference/java/net/URL.html) и создает файл с таким же именем во внутренней каталоге кэша вашего приложения:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10 | public File getTempFile(Context context, String url) {      File file;      try {          String fileName = Uri.parse(url).getLastPathSegment();          file = File.createTempFile(fileName, null, context.getCacheDir());      catch (IOException e) {          // Error while creating file      }      return file;  } |

**Примечание:** Каталог внутреннего хранилища вашего приложения использует имя пакета вашего приложения, располагаясь в специальном месте файловой системы Android. Технически, другое приложение может читать ваши внутренние файлы, если вы установите для файла режим доступа на чтение. Тем не менее, другие приложения также должны знать ваше имя пакета и имена файлов. Другие приложения не могут просматривать ваши внутренние каталоги, и не имеют доступа для чтения или записи, если явно не установить разрешения для чтения или записи. Так что, пока вы используете [MODE\_PRIVATE](http://developer.android.com/reference/android/content/Context.html#MODE_PRIVATE) для ваших файлов во внутренней памяти, они не являются доступными для других приложений.

Сохраните файл на внешнем накопителе

Поскольку внешнее хранилище может быть недоступно — например, когда пользователь монтируется хранилище к ПК, или вынул SD карту, которая обеспечивает внешнее хранилище — вы всегда должны убедиться, что раздел доступен перед доступом к нему. Вы можете запросить состояния внешнего хранилища, вызвав [getExternalStorageState()](http://developer.android.com/reference/android/os/Environment.html" \l "getExternalStorageState()). Если возвращенное состояние равно [MEDIA\_MOUNTED](http://developer.android.com/reference/android/os/Environment.html#MEDIA_MOUNTED), то вы можете читать и писать файлы. Например, следующие методы полезны для определения доступности хранилища:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18 | /\* Checks if external storage is available for read and write \*/  public boolean isExternalStorageWritable() {      String state = Environment.getExternalStorageState();      if (Environment.MEDIA\_MOUNTED.equals(state)) {          return true;      }      return false;  }    /\* Checks if external storage is available to at least read \*/  public boolean isExternalStorageReadable() {      String state = Environment.getExternalStorageState();      if (Environment.MEDIA\_MOUNTED.equals(state) ||          Environment.MEDIA\_MOUNTED\_READ\_ONLY.equals(state)) {          return true;      }      return false;  } |

Хотя внешнее запоминающее устройство может изменяться пользователем и другими приложениями, есть две категории файлов, которые вы могли бы сохранить здесь:

**Общедоступные файлы**

Файлы, которые должны быть в свободном доступе для других приложений и для пользователя. Когда пользователь удаляет ваше приложение, эти файлы должны оставаться доступными для пользователя.Например, фотографии, сделанные с помощью приложения, или другие загруженные файлы.

**Личные файлы**

Файлы, которые по праву принадлежат вашему приложению, и их следует удалить, когда пользователь удаляет ваше приложение. Хотя эти файлы являются технически доступными пользователю, и другим приложениям, т.к. они находятся на внешнем накопителе, это файлы, которые реально не представляют собой ценности для пользователя вне вашего приложения. Когда пользователь удаляет ваше приложение, система удаляет все файлы с внешнего приватного каталога вашего приложения.Например, дополнительные ресурсы загруженные вашим приложения или временные медиа-файлы.

Если вы хотите сохранить общедоступные файлы на внешнем устройстве хранения, используйте[getExternalStoragePublicDirectory()](http://developer.android.com/reference/android/os/Environment.html#getExternalStoragePublicDirectory(java.lang.String)) метод для получения [File](http://developer.android.com/reference/java/io/File.html) представляющего соответствующий каталог на внешнем накопителе. Метод принимает аргумент, указывающий тип файла, который вы хотите сохранить, чтобы они могли быть логически организованы с другими общедоступными файлами, такими как [DIRECTORY\_MUSIC](http://developer.android.com/reference/android/os/Environment.html#DIRECTORY_MUSIC) или [DIRECTORY\_PICTURES](http://developer.android.com/reference/android/os/Environment.html#DIRECTORY_PICTURES). Например:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | public File getAlbumStorageDir(String albumName) {      // Get the directory for the user's public pictures directory.      File file = new File(Environment.getExternalStoragePublicDirectory(              Environment.DIRECTORY\_PICTURES), albumName);      if (!file.mkdirs()) {          Log.e(LOG\_TAG, "Directory not created");      }      return file;  } |

Если вы хотите сохранять файлы, которые являются приватными для вашего приложения, вы можете запросить соответствующий каталог, вызвав [getExternalFilesDir()](http://developer.android.com/reference/android/content/Context.html" \l "getExternalFilesDir(java.lang.String)) и передать ему имя, указывающее тип необходимого каталога. Каждый каталог, созданный таким способом, добавляется в родительский каталог, который инкапсулирует все файлы внешнего хранилища вашего приложения, которые система удаляет, когда пользователь удаляет ваше приложение.

Например, вот метод, который можно использовать для создания каталога отдельного фотоальбома:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | public File getAlbumStorageDir(Context context, String albumName) {      // Get the directory for the app's private pictures directory.      File file = new File(context.getExternalFilesDir(              Environment.DIRECTORY\_PICTURES), albumName);      if (!file.mkdirs()) {          Log.e(LOG\_TAG, "Directory not created");      }      return file;  } |

Если ни одно из предопределенных имен подкаталогов не удовлетворяет вашим файлам, вы можете вместо этого вызвать [getExternalFilesDir()](http://developer.android.com/reference/android/content/Context.html" \l "getExternalFilesDir(java.lang.String)) и передать null. При это возвращается корневой каталог для приватных каталогов вашего приложения на внешнем накопителе.

Помните, что [getExternalFilesDir()](http://developer.android.com/reference/android/content/Context.html" \l "getExternalFilesDir(java.lang.String)) создает каталог внутри каталога, который удаляется, когда пользователь удаляет ваше приложение. Если сохраненные файлы должны оставаться доступными после того как пользователь удаляет ваше приложение — например, когда ваше приложение представляет собой камеру, и пользователь захочет сохранить фотографии — то следует использовать [getExternalStoragePublicDirectory()](http://developer.android.com/reference/android/os/Environment.html" \l "getExternalStoragePublicDirectory(java.lang.String)).

Независимо от того, используете ли вы [getExternalStoragePublicDirectory()](http://developer.android.com/reference/android/os/Environment.html" \l "getExternalStoragePublicDirectory(java.lang.String)) для файлов, которые являются общими или [getExternalFilesDir()](http://developer.android.com/reference/android/content/Context.html" \l "getExternalFilesDir(java.lang.String)) для файлов, которые являются приватными для вашего приложения, важно, что вы используете имена каталогов, предоставляемые константами API, такими как [DIRECTORY\_PICTURES](http://developer.android.com/reference/android/os/Environment.html#DIRECTORY_PICTURES). Эти имена каталогов гарантируют, что файлы будут интерпретироваться системой должным образом. Например, файлы, сохраненные в[DIRECTORY\_RINGTONES](http://developer.android.com/reference/android/os/Environment.html#DIRECTORY_RINGTONES) классифицируются по системе медиа сканера как мелодии звонка вместо музыки.

Запросите свободное пространство

Если вы знаете заранее, сколько данных вы будете сохранять, вы можете выяснить, есть ли в наличие достаточно места, не вызывая [IOException](http://developer.android.com/reference/java/io/IOException.html) с помощью вызова [getFreeSpace()](http://developer.android.com/reference/java/io/File.html" \l "getFreeSpace()) или[getTotalSpace()](http://developer.android.com/reference/java/io/File.html#getTotalSpace()). Эти методы предоставляют информацию о текущем доступном пространстве и общем пространстве раздела, соответственно. Эта информация также полезна, чтобы избежать заполнения раздела накопителя выше определенного порога.

Тем не менее, система не гарантирует, что вы можете записать столько байт, сколько обозначено[getFreeSpace()](http://developer.android.com/reference/java/io/File.html#getFreeSpace()). Если возвращаемое число на несколько МБ больше, чем размер данных, которые вы хотите сохранить, или если файловая система заполнена меньше чем на 90%, то, наверное, можно продолжить. В противном случае, вероятно, не стоит записывать в хранилище.

**Примечание:** Вы не обязаны проверять количество свободного места, прежде чем вы сохраняете файл. Вместо этого, вы можете попробовать записать файл, и поймать [IOException](http://developer.android.com/reference/java/io/IOException.html)если оно произойдет. Возможно, вам придется сделать так, если вы не знаете точно, сколько места нужно. Например, если вы измените кодировку файла перед сохранением путем преобразования PNG изображения в JPEG, вы не будете знать размер файла заранее.

Удаление файла

Вы всегда должны удалить файлы, которые вам больше не нужны. Самый простой способ удалить файл это вызвать [delete()](http://developer.android.com/reference/java/io/File.html" \l "delete()).

|  |  |
| --- | --- |
| 1 | myFile.delete(); |

Если файл сохранен во внутреннем хранилище, вы также можете попросить [Context](http://developer.android.com/reference/android/content/Context.html) найти и удалить файл, вызвав [deleteFile()](http://developer.android.com/reference/android/content/Context.html" \l "deleteFile(java.lang.String)):

|  |  |
| --- | --- |
| 1 | myContext.deleteFile(fileName); |

**Примечание:** Когда пользователь удаляет ваше приложение, Android система удаляет следующее:

* Все файлы, сохраненные во внутреннем хранилище
* Все файлы, сохраненные на внешнем накопителе, используя [getExternalFilesDir()](http://developer.android.com/reference/android/content/Context.html" \l "getExternalFilesDir(java.lang.String)).

Тем не менее, вы должны вручную удалить все файлы из кэша, созданные с помощью[getCacheDir()](http://developer.android.com/reference/android/content/Context.html#getCacheDir()) на регулярной основе, а также регулярно удалять другие файлы, которые вам больше не нужны.