Сохранение файлов

Этот урок научит вас

1. [Выберите внутренний или внешний накопитель](https://developer.android.com/training/basics/data-storage/files.html#InternalVsExternalStorage)
2. [Получение разрешений для внешних накопителей](https://developer.android.com/training/basics/data-storage/files.html#GetWritePermission)
3. [Сохранение файла внутренней памяти](https://developer.android.com/training/basics/data-storage/files.html#WriteInternalStorage)
4. [Сохранение файла во внешнем хранилище](https://developer.android.com/training/basics/data-storage/files.html#WriteExternalStorage)
5. [Запрос Свободное пространство](https://developer.android.com/training/basics/data-storage/files.html#GetFreeSpace)
6. [Удаление файла](https://developer.android.com/training/basics/data-storage/files.html#DeleteFile)

Вы также должны прочитать

* [Использование внутреннего хранилища](https://developer.android.com/guide/topics/data/data-storage.html#filesInternal)
* [Использование внешнего хранилища](https://developer.android.com/guide/topics/data/data-storage.html#filesExternal)

Android использует файловую систему , которая похожа на диск на основе файловых систем на других платформах. Этот урок описывает , как работать с файловой системой Android для чтения и записи файлов с [File](https://developer.android.com/reference/java/io/File.html) API.

[File](https://developer.android.com/reference/java/io/File.html)Объект подходит для чтения или записи больших объемов данных в начала до конца заказа , не пропуская вокруг. Например, это хорошо для файлов изображений или что - либо обменены по сети.

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

Выберите внутренний или внешний накопитель

Все Android устройства имеют две зоны для хранения файлов: "внутренний" и "внешний" хранения. Эти названия происходят из первых дней Android, когда большинство устройств, предлагаемых встроенной энергонезависимой памяти (внутренней памяти), а также съемный носитель информации, такие как микро SD-карт (внешнее запоминающее устройство).Некоторые устройства делят постоянное место хранения на "внутренних" и "внешних" разделов, так что даже без съемного носителя информации, всегда есть два помещения для хранения и поведение API одно и то же, является ли съемный внешний накопитель или нет. Ниже перечислены суммировать факты о каждом из дискового пространства.

**Внутреннее хранилище:**

* Это всегда доступно.
* Файлы, сохраненные здесь доступны только приложения.
* Когда пользователь удаляет приложение, система удаляет все файлы своего приложения из внутренней памяти.

Внутренняя память лучше всего, когда вы хотите быть уверены, что ни пользователь, ни другие приложения могут получить доступ к файлам.

**Внешнее хранилище:**

* Это не всегда доступно, так как пользователь может подключить внешнее запоминающее устройство USB в качестве хранилища, а в некоторых случаях удалить его из устройства.
* Это доступен для чтения, так что файлы, сохраненные здесь можно прочитать вне вашего контроля.
* Когда пользователь удаляет приложение, система удаляет файлы своего приложения отсюда , только если вы сохраните их в каталоге с [getExternalFilesDir()](https://developer.android.com/reference/android/content/Context.html#getExternalFilesDir(java.lang.String)).

Внешнее запоминающее устройство является лучшим местом для файлов, которые не требуют ограничения доступа и для файлов, которые вы хотите поделиться с другими приложениями или позволяют пользователю получить доступ к компьютеру.

**Примечание:** Перед тем какAndroid N, внутренние файлы могут быть сделаны доступными для других приложений с помощью релаксирующих разрешения файловой системы. Это уже не так.Если вы хотите , чтобы сделать содержание частной файла ,доступной для других приложений, ваше приложение может использовать [FileProvider](https://developer.android.com/reference/android/support/v4/content/FileProvider.html). См [общий доступ к файлам](https://developer.android.com/training/secure-file-sharing/index.html) .

**Совет:** Хотя приложения устанавливаются на внутреннюю память по умолчанию, вы можете указать [android:installLocation](https://developer.android.com/guide/topics/manifest/manifest-element.html" \l "install)атрибут в манифесте так ваше приложение может быть установлено на внешнем накопителе. Пользователи ценят эту опцию , когда размер APK очень велик , и они имеют внешнее пространство для хранения , что это больше , чем во внутренней памяти. Для получениядополнительной информации см [App Install Location](https://developer.android.com/guide/topics/data/install-location.html) .

Получение разрешений для внешних накопителей

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

<манифеста ... > <-использует разрешение для Android: имя = "android.permission.WRITE\_EXTERNAL\_STORAGE" />     ... </ манифеста>

**Внимание:** В настоящее время все приложения имеют возможность считывать внешнее запоминающее устройство без специального разрешения. Тем не менее, это изменится в будущих версиях. Если ваше приложение должно прочитать внешнее запоминающее устройство (но не писать в него), то вам нужно будет объявить [READ\_EXTERNAL\_STORAGE](https://developer.android.com/reference/android/Manifest.permission.html" \l "READ_EXTERNAL_STORAGE)разрешение. Для того, чтобы убедиться , что ваше приложение продолжает работать , как и ожидалось, вы должны объявить это разрешение сейчас, до того , как изменения вступят всилу.

<манифеста ... > <-использует разрешение для Android: имя = "android.permission.READ\_EXTERNAL\_STORAGE" />     ... </ манифеста>

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

Вам не нужно никаких разрешений, чтобы сохранить файлы на внутренней памяти. Ваше приложение всегда имеет разрешение на чтение и запись файлов в своем внутреннем каталоге хранения.

Сохранение файла внутренней памяти

При сохранении файла во внутреннее устройство хранения, вы можете приобрести соответствующий каталог в виде [File](https://developer.android.com/reference/java/io/File.html)позвонив поодному из двух методов:

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

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

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

Возвращает [File](https://developer.android.com/reference/java/io/File.html)представляющий внутренний каталог для временных файлов кэша вашего приложения. Обязательно удалите каждый файл , когда он больше не нужен и осуществить разумный предельный размер объема памяти , используемой в любой момент времени, например, 1 МБ. Если система начинает работать низко на хранении, он может удалять файлы кэша без предупреждения.

Чтобы создать новый файл в одном из этих каталогов, вы можете использовать [File()](https://developer.android.com/reference/java/io/File.html" \l "File(java.io.File, java.lang.String))конструктор, передавая [File](https://developer.android.com/reference/java/io/File.html)обеспечивается одним из указанных выше способов , что указывает ваш внутренний каталог хранения. Например:

Файл регистрировать и хранить в каком-л. Определенном порядке = новый файл ( контекст . GetFilesDir (), имя файла );

Кроме того , вы можете позвонить , [openFileOutput()](https://developer.android.com/reference/android/content/Context.html" \l "openFileOutput(java.lang.String, int))чтобы получить , [FileOutputStream](https://developer.android.com/reference/java/io/FileOutputStream.html) что пишет в файл в вашем внутреннем каталоге.Например, вот как написать какой - нибудь текст в файл:

Строка имени файла = "MYFILE" ;

строка снабжать струной , тетивой и т.п. = "Привет , мир!" ;

FileOutputStream OutputStream , попробуйте {   OutputStream = openFileOutput ( имя файла , контекст . MODE\_PRIVATE );

OutputStream . Написать ( снабжать струной , тетивой и т.п. . GetBytes ());

OutputStream . Близкий (); }

поймать ( Exception е ) {   е . printStackTrace (); }

Или, если вам нужно кэшировать некоторые файлы, вы должны использовать [createTempFile()](https://developer.android.com/reference/java/io/File.html" \l "createTempFile(java.lang.String, java.lang.String)). Например, следующий метод извлекает имя файла из [URL](https://developer.android.com/reference/java/net/URL.html)и создает файл с таким именем в директории внутреннего кэша вашего приложения:

общественного файла getTempFile ( Context контекст , строка URL ) { File файл ; попробуйте { Строка имя\_файла = Uri . разобрать ( URL ). getLastPathSegment ();         файл = Файл . createTempFile ( имя\_файла , нуль , контекст . getCacheDir ()); } поймать ( IOException е ) { // Ошибка при создании файла } возврата файла ; }   
      
      
          
   
      
          
   

**Примечание:** внутренний каталог хранения вашего приложения задается имя пакета вашего приложения в специальном месте файловой системы Android. Технически, другое приложение может читать ваши внутренние файлы , если вы установите режим доступа к файлу ,чтобы быть читаемыми. Тем не менее, другие приложения также должны были бы знать ваши приложения имя пакета и имена файлов. Другие приложения не могут просматривать ваши внутренние каталоги и не имеют чтения или записи доступа , если вы явно не установить файлы , чтобы быть для чтения или записи. Так что до тех пор , как вы используете [MODE\_PRIVATE](https://developer.android.com/reference/android/content/Context.html#MODE_PRIVATE) для ваших файлов на внутренней памяти, они никогда не были доступны для других приложений.

Сохранение файла во внешнем хранилище

Поскольку внешние устройства хранения данных могут быть недоступны, например, когда пользователь смонтировали хранения на ПК или снял карту памяти SD , которая обеспечивает внешний накопитель, вы всегда должны убедиться в том , что объем доступен перед доступом к ней. Вы можете запросить внешнее состояние памяти с помощью вызова [getExternalStorageState()](https://developer.android.com/reference/android/os/Environment.html" \l "getExternalStorageState()). Если возвращенное состояние равно [MEDIA\_MOUNTED](https://developer.android.com/reference/android/os/Environment.html#MEDIA_MOUNTED), то вы можете читать и писать файлы. Например, следующие методы могут быть использованы для определения доступности для хранения:

/ \* Проверяет , если внешнее запоминающее устройство доступно для чтения и записи \* / общественное логическое значение isExternalStorageWritable () { Строка состояния = окружающая среда . GetExternalStorageState (); если ( окружающая среда . MEDIA\_MOUNTED . Равенства ( состояние )) { вернуть истинный ; } возвращать ложь ; } / \* Проверяет , является ли внешнее запоминающее устройство доступно по крайней мере , для чтения \* / общественного логическое значение isExternalStorageReadable () { Строка состояния = окружающая среда . getExternalStorageState (); если ( окружающая среда . MEDIA\_MOUNTED . равенства ( состояние ) || Environment . MEDIA\_MOUNTED\_READ\_ONLY . равенства ( состояние )) { вернуть истинный ; } вернуться ложным ; }

Хотя внешний накопитель изменяемый пользователем и другими приложениями, есть две категории файлов, которые вы могли бы сохранить здесь:

Публичные файлы

Файлы, которые должны быть свободно доступны для других приложений и пользователю. Когда пользователь удаляет приложение, эти файлы должны оставаться доступными для пользователя.

Например, фотографии, сделанные приложения или другие загруженные файлы.

Собственные файлы

Файлы, которые по праву принадлежат к вашему приложению и должны быть удалены, когда пользователь удаляет приложение. Хотя эти файлы являются технически доступны пользователю и другие приложения, потому что они находятся на внешнем запоминающем устройстве, они представляют собой файлы, которые реально не обеспечивают ценность для пользователя за пределами вашего приложения. Когда пользователь удаляет приложение, система удаляет все файлы во внешней частной директории вашего приложения.

Например, дополнительные ресурсы загружены приложения или временных файлов медиа.

Если вы хотите сохранить доступа к файлам на внешнем запоминающем устройстве, используйте [getExternalStoragePublicDirectory()](https://developer.android.com/reference/android/os/Environment.html#getExternalStoragePublicDirectory(java.lang.String)) метод , чтобы получить [File](https://developer.android.com/reference/java/io/File.html) представляющий соответствующий каталог на внешнем запоминающем устройстве. Метод принимает аргумент , указывающий тип файла , который вы хотите сохранить , так что они могут быть логически организованы с другими общественными файлами, такими как [DIRECTORY\_MUSIC](https://developer.android.com/reference/android/os/Environment.html#DIRECTORY_MUSIC) или [DIRECTORY\_PICTURES](https://developer.android.com/reference/android/os/Environment.html#DIRECTORY_PICTURES). Например:

общественного файла getAlbumStorageDir ( Строка ALBUMNAME ) { // Получить каталог для каталога публичной фотографии пользователя. Файл файл = новый файл ( Environment . getExternalStoragePublicDirectory ( Environment . DIRECTORY\_PICTURES ), ALBUMNAME ); если (! файл . mkdirs ()) { Log . е ( LOG\_TAG , "Справочник не создан" ); } возврат файла ; }

Если вы хотите сохранить файлы , которые являются закрытыми для вашего приложения, вы можете приобрести соответствующий каталог, позвонив [getExternalFilesDir()](https://developer.android.com/reference/android/content/Context.html" \l "getExternalFilesDir(java.lang.String))и передав ей имя , указывающее тип директории , которую вы хотите. Каждый каталог создан таким образом , добавляется в родительский каталог , который инкапсулирует внешние файлы для хранения своего приложения, которые система удаляет , когда пользователь удаляет приложение.

Например, вот метод, который вы можете использовать, чтобы создать каталог для отдельного фотоальбома:

общественного файла getAlbumStorageDir ( Context контекст , строка ALBUMNAME ) { // Получить каталог для частного каталога фотографий приложения. Файл файл = новый файл ( контекст . getExternalFilesDir ( Environment . DIRECTORY\_PICTURES ), ALBUMNAME ); если (! файл . mkdirs ()) { Log . е ( LOG\_TAG , "Справочник не создал" ); } обратный файл ; }

Если ни один из предопределенных имен подкаталога не подходят ваши файлы, вы можете вместо этого позвонить [getExternalFilesDir()](https://developer.android.com/reference/android/content/Context.html" \l "getExternalFilesDir(java.lang.String))и передать null. Это возвращает корневой каталог для частной директории вашего приложения на внешнем запоминающем устройстве.

Помните , что [getExternalFilesDir()](https://developer.android.com/reference/android/content/Context.html" \l "getExternalFilesDir(java.lang.String)) создает каталог внутри каталога , который удаляется , когда пользователь удаляет приложение. Если файлы вы экономите должны оставаться доступными после того, как пользователь удалит ваше приложение-например, когда ваше приложение камеры и пользователь захочет сохранить фотографии, вы должны использовать [getExternalStoragePublicDirectory()](https://developer.android.com/reference/android/os/Environment.html" \l "getExternalStoragePublicDirectory(java.lang.String)).

Вне зависимости от того, используется ли [getExternalStoragePublicDirectory()](https://developer.android.com/reference/android/os/Environment.html" \l "getExternalStoragePublicDirectory(java.lang.String))для файлов , которые являются общими или[getExternalFilesDir()](https://developer.android.com/reference/android/content/Context.html#getExternalFilesDir(java.lang.String))для файлов , которые являются частными для вашего приложения, важно , что вы используете имена каталогов ,предоставляемые API констант , таких как [DIRECTORY\_PICTURES](https://developer.android.com/reference/android/os/Environment.html#DIRECTORY_PICTURES). Эти имена каталогов убедиться , что файлы обрабатываются должным образом системой. Например, файлы , сохраненные в [DIRECTORY\_RINGTONES](https://developer.android.com/reference/android/os/Environment.html#DIRECTORY_RINGTONES) классифицируются сканером медиа - системы в качестве мелодии вместо музыки.

Запрос Свободное пространство

Если вы знаете заранее , сколько данных вы спасительной, вы можете узнать, доступен ли не вызывая достаточное пространство[IOException](https://developer.android.com/reference/java/io/IOException.html), вызывая [getFreeSpace()](https://developer.android.com/reference/java/io/File.html" \l "getFreeSpace())или [getTotalSpace()](https://developer.android.com/reference/java/io/File.html" \l "getTotalSpace()). Эти методы обеспечивают текущее доступное пространство и общее пространство в объеме хранения, соответственно. Эта информация также полезна для предотвращения заполнения объема памяти выше определенного порогового значения.

Тем не менее, система не гарантирует , что вы можете написать столько байт , сколько обозначены [getFreeSpace()](https://developer.android.com/reference/java/io/File.html" \l "getFreeSpace()). Если номер возвращается в несколько мегабайт больше , чем размер данных , которые вы хотите сохранить, или если файловая система находится менее чем на 90%, то это, вероятно , безопасно продолжить работу . В противном случае, вы , вероятно , не стоит писать в хранилище.

**Примечание:** Вы не обязаны проверить объем доступного пространства , прежде чем вы сохраните файл. Вы вместо того, чтобы можно попробовать записать файл сразу, а затем поймать , [IOException](https://developer.android.com/reference/java/io/IOException.html) если происходит. Возможно , вам придется сделать это , если вы не знаете точно , сколько места нужно. Например, если изменить кодировку файла , прежде чем сохранить его путем преобразования PNG изображения в формате JPEG, вы не знаете размер файла заранее.

Удаление файла

Вы всегда должны удалить файлы , которые больше не нужны. Самый простой способ удалить файл должен иметь открытый файл ссылки на вызов [delete()](https://developer.android.com/reference/java/io/File.html" \l "delete())на себя.

MyFile . удалить ();

Если файл сохраняется во внутренней памяти, вы можете также спросить , [Context](https://developer.android.com/reference/android/content/Context.html)чтобы найти и удалить файл с помощью вызова[deleteFile()](https://developer.android.com/reference/android/content/Context.html#deleteFile(java.lang.String)):

myContext . DeleteFile ( имя\_файла );

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

* Все файлы, сохраненные во внутренней памяти
* Все файлы вы сохранили на внешнем накопителе , используя [getExternalFilesDir()](https://developer.android.com/reference/android/content/Context.html" \l "getExternalFilesDir(java.lang.String)).

Тем не менее, вы должны вручную удалить все кэшированные файлы , созданные с помощью [getCacheDir()](https://developer.android.com/reference/android/content/Context.html" \l "getCacheDir())на регулярной основе , а также регулярно удалять другие файлы , которые больше не нужны