




Android 存储空间的最佳实践



谷歌开发者 
已认证的官方帐号

已关注

12 人赞同了该文章

为了提高文件的规整程度并让用户可以更好地控制他们的文件，Android 10 为应用引入了名为 "分区存储" 的新范式。分区存储改变了应用在外置存储中保存和访问文件的方式，为了帮您迁移应用并支持分区存储，我们概括了常见用例的最佳实践并分享给大家。

处理媒体文件

这部分内容描述了处理媒体文件 (如视频、图片、音频文件) 的一些常见用例，并概要说明了应用可以使用的方法。我们制作了一个简单的图片，列出了每种用例以及其在不同系统版本的实践总结：

处理媒体文件用例	摘要
展示所有图片和视频文件	所有 Android 版本使用相同的方法。
展示特定文件夹内的文件和视频	所有 Android 版本使用相同的方法。
访问照片内的位置信息	使用分区存储和停用分区存储时使用不同的方法。
在一次操作中修改和删除多个媒体文件	在 Android 11 中使用一种方法；对于 Android 10 则需要停用分区存储并使用 Android 9 及更低版本中使用的方法。
导入已存在的单个图片	所有 Android 版本使用相同的方法。
拍摄单张图片	所有 Android 版本使用相同的方法。
与其他应用共享媒体文件	所有 Android 版本使用相同的方法。
与特定应用共享媒体文件	所有 Android 版本使用相同的方法。
从代码或依赖库中使用直接文件路径访问文件	在 Android 11 中使用一种方法；对于 Android 10 则需要停用分区存储并使用 Android 9 及更低版本中使用的方法。

展示多个文件夹中的图片和视频文件

使用 [query\(\)](#) API [查询媒体集合](#)。您可以通过调整 `projection`、`selection`、`selectionArgs` 与 `sortOrder` 参数来对媒体文件进行过滤和排序。

展示特定文件夹内的文件和视频

使用以下方法:

1. 使用 "请求应用权限" 一文中描述的最佳实践来请求 [READ_EXTERNAL_STORAGE](#) 权限。
2. 使用 [MediaColumns.DATA](#) 中的值来检索媒体文件，其中包含了磁盘中媒体文件的绝对文件系统路径。

访问照片内的位置信息

如果您的应用使用了分区存储，可以遵循媒体存储指南中的 "[照片中的位置信息](#)" 部分进行操作。

注意: 就算您选择停用分区存储，在使用 [MediaStore](#) API 访问图像并读取未修改的位置信息时，您也需要请求 [ACCESS_MEDIA_LOCATION](#) 权限。

在单个操作中修改或删除多个媒体文件

您需要根据运行应用的 Android 版本来整合逻辑。

在 Android 11 上运行

使用以下方法:

1. 使用 [MediaStore.createWriteRequest\(\)](#) 或 [MediaStore.createTrashRequest\(\)](#) 为应用的写入或删除请求创建待定 intent，然后通过调用该 intent 提示用户授予修改一组文件的权限。
2. 评估用户的响应:
 - 如果获得了权限，执行修改或删除操作；
 - 如果未能获得权限，向用户解释为什么您的应用需要此权限。

详细了解如何使用 Android 11 引入的[这些方法执行批量操作](#)。

在 Android 10 上运行

如果您的应用目标 API 为 Android 10 (API level 29), 请停用分区存储并继续使用 Android 9 及更低版本所使用的方法来执行这类操作。

在 Android 9 及更低版本上运行

使用以下方法:

1. 使用 "请求应用权限" 一文中描述的最佳实践来请求 WRITE_EXTERNAL_STORAGE 权限。
2. 使用 MediaStore API 来修改和删除媒体文件。

导入已存在的单个图片

如果您想要导入一张已存在的图片 (例如将照片用于用户个人资料), 您的应用可以使用自己的 UI 或者系统图片选择器来执行这一操作。

提供您自己的用户界面

使用以下方法:

1. 使用 "请求应用权限" 一文中描述的最佳实践来请求 READ_EXTERNAL_STORAGE 权限。
2. 使用 query() API 查询媒体集合。
3. 将结果显示到您的 UI 上。

使用系统选择器

使用 ACTION_GET_CONTENT intent 来要求用户选择要导入的图片。如果您希望过滤系统选择器向用户展示的图片类型, 可以使用 setType() 或 EXTRA_MIME_TYPES。

拍摄单张图片

当您想要拍摄一张图片并用于您的应用时 (例如将照片用于用户个人资料), 使用 ACTION_IMAGE_CAPTURE intent 来要求用户使用设备的相机拍摄一张照片。系统会将拍摄的照片存储于 MediaStore.Images 表中。

与其他应用共享媒体文件

使用 [insert\(\)](#) 方法将记录直接加入 MediaStore。详细信息，可以参阅媒体存储指南中 "[添加项目](#)" 部分。

与特定应用共享媒体文件

使用 Android FileProvider 组件，相关内容在 "[设置文件分享](#)" 指南中有描述。

从代码或依赖库中使用直接文件路径访问文件

您需要根据运行应用的 Android 版本来整合逻辑。

在 Android 11 上运行

使用以下方法：

1. 使用 "请求应用权限" 一文中描述的最佳实践来请求 READ_EXTERNAL_STORAGE 权限。
2. 使用直接文件路径访问文件。

详细信息，请参阅 "[使用原始路径访问文件](#)"。

在 Android 10 上运行

如果您的应用目标 API 为 Android 10 (API level 29)，请停用分区存储并继续使用 Android 9 及更低版本所使用的方法来执行这类操作。

在 Android 9 及更低版本上运行

使用以下方法：

1. 使用 "[请求应用权限](#)" 一文中描述的最佳实践来请求 [WRITE_EXTERNAL_STORAGE](#) 权限。
2. 使用直接文件路径访问文件。

打开文档文件

使用 [ACTION_OPEN_DOCUMENT](#) intent 来要求用户通过系统选择器选择需要打开的文件。如果您想要过滤系统选择器展示给用户的文件类型，可以使用 [setType\(\)](#) 或 [EXTRA_MIME_TYPES](#) 。

举例来说，您可以通过以下代码找到所有的 PDF、ODT 和 TXT 文件：

Kotlin 代码

```
startActivityForResult(  
    Intent(Intent.ACTION_OPEN_DOCUMENT).apply {  
        addCategory(Intent.CATEGORY_OPENABLE)  
        type = "*/*"  
        putExtra(Intent.EXTRA_MIME_TYPES, arrayOf(  
            "application/pdf", // .pdf  
            "application/vnd.oasis.opendocument.text", // .odt  
            "text/plain" // .txt  
        ))  
    },  
    REQUEST_CODE  
)
```

Java 代码

```
Intent intent = new Intent(Intent.ACTION_OPEN_DOCUMENT);  
intent.addCategory(Intent.CATEGORY_OPENABLE);  
intent.setType("*/*");  
intent.putExtra(Intent.EXTRA_MIME_TYPES, new String[] {  
    "application/pdf", // .pdf  
    "application/vnd.oasis.opendocument.text", // .odt  
    "text/plain" // .txt  
});  
startActivityForResult(intent, REQUEST_CODE);
```

从旧的存储位置迁移现有文件

如果一个目录既不是特定应用的目录，也不是公开共享目录，那么它就会被视为旧版存储位置。如果您的应用创建或使用位于旧版存储位置的文件，我们建议您将应用的文件迁移至可被分区存储访问的位置，同时对应用进行必要的修改，以使用分区存储中的文件。

保留对旧版存储位置的访问以进行数据迁移

您的应用需要保留对旧版存储位置的访问，以便可以将任何文件迁移至可被分区存储访问的位置。您应该使用的方法取决于您应用的目标 API 级别。

如果您的应用目标平台为 Android 11

1. 使用 `preserveLegacyExternalStorage` 标志来 保留旧版存储模型，这样您的应用就可以在用户将应用升级为目标平台为 Android 11 的新版本时迁移用户数据。

注意: 如果您使用 `preserveLegacyExternalStorage`，保持旧存储模型的效果只会在用户卸载应用之前有效。如果用户在运行 Android 11 的设备上安装或重装您的应用，则无论 `preserveLegacyExternalStorage` 的值是什么，您的应用都无法停用分区存储模型。

1. 继续 停用分区存储，以便您的应用可以继续访问运行 Android 10 的设备上旧版存储位置中的文件。

如果您的应用目标平台为 Android 10

停用分区存储，以便您可以更轻松地在多个 Android 版本间保持应用行为不变。

迁移应用数据

当您的应用已经做好迁移的准备时，使用以下方法：

1. 检查您应用在工作中是否使用了位于 `/sdcard/` 目录或其任何子目录中的文件；
2. 将应用的所有私有文件从现在的 `/sdcard/` 下的目录中移动至 `getExternalFilesDir()` 方法所返回的目录中；
3. 将所有共享的非媒体文件从现在的 `/sdcard/` 下的目录中移动至 `/sdcard/` 目录下的一个应用专用子目录；
4. 从 `/sdcard/` 目录移除应用程序的旧存储目录。

与其他应用共享内容

您可以使用 `FileProvider` 分享应用的文件给某个其他应用。而对于那些需要互相之间分享文件的所有应用，我们推荐为每一个应用使用 内容提供程序，然后在将应用添加到集合中时同步数据。

缓存非媒体文件

您应使用的方法取决于需要缓存的文件类型。

- 小型文件或者包含敏感信息的文件：使用 `Context#getCacheDir()`
- 大型文件或者不包含敏感信息的文件：使用 `Context#getExternalCacheDir()`

暂时停用分区存储

在您的应用完全兼容分区存储之前，您可以通过以下方法之一停用分区存储：

- 目标平台设置为 Android 9 (API level 28) 或更低。
- 如果您的目标平台为 Android 10 (API level 29) 或者更高版本，将您应用 manifest 中的 `requestLegacyExternalStorage` 属性设置为 "true"：

```
<manifest ... >
<!-- 该属性在目标 API 为 Android 10 或更高版本的应用中默认为 "false" -->
  <application android:requestLegacyExternalStorage="true" ... >
    ...
  </application>
</manifest>
```

注意：在您将应用的目标 API 更新为 Android 11 (API level 30) 后，如果应用运行在 Android 11 的设备上，系统会忽略 `requestLegacyExternalStorage` 属性。所以您的应用必须为支持分区存储做好准备，并为使用该设备的用户 迁移数据。

为了测试目标 API 为 Android 9 及更低版本的应用在使用分区存储时的行为，您可以通过设置 `requestLegacyExternalStorage` 的值设置为 `false` 来使应用选择启用行为。如果要在 Android 11 设备上进行测试，则还可以使用 应用兼容性标志 在使用或不使用分区存储的情况下测试应用的行为。

了解有关 Android 平台文件存储与访问的详细信息，请参阅以下资源：

- [数据和文件存储概览](#)

如果您想了解更多最新关于使用存储空间的最佳实践，请查阅 Android 官方中文文档网站中 [Android 存储用例和最佳做法](#) 部分。

发布于 10-23