6 DE DICIEMBRE DE 2019

ALMACENAMIENTO DE DATOS EN APLICACIONES MÓVILES Y PROVEEDORES DE CONTENIDO

PROFESORA: MTE. NOEMI GUADALUPE CASTILLO SOSA

ARMANDO MORA HERRERA

INGENIERÍA EN SISTEMAS COMPUTACIONALES

DESARROLLO DE APLICACIONES PARA DISPOSITIVOS MÓVILES

INSTITUTO TECNOLÓGICO SUPERIOR DE FELIPE CARRILLO PUERTO

ALMACENAMIENTO DE DATOS:

Android utiliza un sistema de archivos similar a los sistemas de archivos basados en disco en otras plataformas. El sistema proporciona varias opciones para que guarde los datos de su aplicación:

- **Almacenamiento específico de la aplicación:** almacene archivos destinados solo para el uso de su aplicación, ya sea en directorios dedicados dentro de un volumen de almacenamiento interno o en diferentes directorios dedicados dentro del almacenamiento externo. Use los directorios dentro del almacenamiento interno para guardar información confidencial a la que otras aplicaciones no deberían acceder.
- **Almacenamiento compartido:** almacene archivos que su aplicación pretende compartir con otras aplicaciones, incluidos medios, documentos y otros archivos.
- **Preferencias:** Almacenar datos primitivos privados en pares clave-valor.
- **Bases de datos:** almacene datos estructurados en una base de datos privada utilizando la biblioteca de persistencia de Room.

	Type of content	Access method	Permissions needed	Can other apps access?	Files removed on app uninstall?
App-specific files	Files meant for your app's use only	From internal storage, `getFilesDir()` or `getCacheDir()` From external storage, `getExternalFilesDir()` or `getExternalCacheDir()`	Never needed for internal storage Not needed for external storage when your app is used on devices that run Android 4.4 (API level 19) or higher	No, if files are in a directory within internal storage Yes, if files are in a directory within external storage	Yes
Media	Shareable media files (images, audio files, videos)	`MediaStore` API	`READ_EXTERNAL_STORAGE` or `WRITE_EXTERNAL_STORAGE` when accessing other apps' files on Android 10 (API level 29) or higher Permissions are required for all files on Android 9 (API level 28) or lower	Yes, though the other app needs the `READ_EXTERNAL_STORAGE` permission	No
Documents and other files	Other types of shareable content, including downloaded files	Storage Access Framework	None	Yes, through the system file picker	No
App preferences	Key-value pairs	Jetpack Preferences library	None	No	Yes
Database	Structured data	Room persistence library	None	No	Yes

PROVEEDORES DE CONTENIDO:

Un proveedor de contenido administra el acceso a un repositorio central de datos. Un proveedor forma parte de una aplicación de Android y a menudo proporciona su propia IU para trabajar con los datos. No obstante, los proveedores de contenido están principalmente orientados a que los usen otras aplicaciones que acceden al proveedor usando un objeto de cliente del proveedor. Juntos, los proveedores y clientes de proveedores ofrecen una interfaz estándar y uniforme para los datos que también manipula la comunicación dentro del proceso y el acceso seguro a los datos.

Por lo general, son dos las situaciones en las que se trabaja con proveedores de contenido: cuando quieres implementar código para acceder a un proveedor de contenido existente en otra aplicación o cuando decides crear un proveedor de contenido nuevo en tu aplicación a fin de compartir datos con otras aplicaciones.

Un proveedor de contenido organiza el acceso a la capa de almacenamiento de los datos en tu aplicación para una serie de API y componentes diferentes (como se detalla en la figura 1) e incluye lo siguiente:

- Compartir con otras aplicaciones el acceso a los datos de tu aplicación.
- Enviar datos a un widget.
- Devolver sugerencias personalizadas de búsqueda para tu aplicación mediante el marco de trabajo de búsqueda usando un SearchRecentSuggestionsProvider.
- Sincronizar los datos de la aplicación con tu servidor mediante una implementación de AbstractThreadedSyncAdapter.
- Cargar datos en tu IU usando un CursorLoader.

Cuando quieres acceder a datos en un proveedor de contenido, usas el objeto ContentResolver en el Context de tu aplicación para comunicarte con el proveedor como cliente. El objeto ContentResolver se comunica con el objeto del proveedor, una instancia de una clase que implementa ContentProvider. El objeto del proveedor recibe solicitudes de datos de clientes, realiza la acción solicitada y devuelve resultados. Este objeto tiene métodos que llaman a métodos con el mismo nombre en el objeto del proveedor, una instancia de una

de las subclases concretas de ContentProvider. Los métodos ContentResolver proporcionan las funciones básicas "CRUD" (proveniente de los términos en inglés que equivalen a crear, recuperar, actualizar y eliminar) del almacenamiento persistente.

Para recuperar datos de un proveedor, tu aplicación necesita "permiso de acceso de lectura" correspondiente al proveedor. No puedes solicitar este permiso en tiempo de ejecución, sino que debes especificar que necesitas este permiso en tu manifiesto usando el elemento <uses-permission> y el nombre exacto del permiso definido por el proveedor. Cuando especificas este elemento en tu manifiesto, estás "solicitando" este permiso para tu aplicación. Cuando los usuarios instalan tu aplicación, conceden implícitamente esta solicitud.

¿CÓMO CREAR UN PROVEEDOR DE CONTENIDO?

Antes de que empieces a crear un proveedor, haz lo siguiente:

- Decide si necesitas un proveedor de contenido. Debes crear un proveedor de contenido si quieres proporcionar una o más de las siguientes funciones:
- Te recomendamos que ofrezcas datos o archivos complejos para otras aplicaciones.
- Te recomendamos que permitas que los usuarios copien datos complejos de tu aplicación en otras aplicaciones.
- Te recomendamos que proporciones sugerencias personalizadas usando el marco de trabajo de búsqueda.
- Quieres exponer los datos de tu aplicación a widgets.
- Quieres implementar la clase AbstractThreadedSyncAdapter, CursorAdapter o CursorLoader.

A continuación, sigue estos pasos para crear un proveedor:

1. Diseña el almacenamiento sin formato para tus datos. Un proveedor de contenido ofrece datos de dos maneras:
    - Datos de archivos
    - Datos estructurados

2. Define una implementación concreta de la clase ContentProvider y sus métodos necesarios. Esta clase es la interfaz entre tus datos y el resto del sistema Android. Para obtener más información acerca de esta clase, consulta la sección Cómo implementar la clase ContentProvider.
3. Define la string de autoridad del proveedor, sus URI de contenido y nombres de columnas. Si quieres que la aplicación del proveedor aborde intents, define acciones de intent, datos extra e indicadores. Define también los permisos que requerirás para las aplicaciones que quieran acceder a tus datos. Considera definir todos estos valores como constantes en una clase Contract independiente; posteriormente, podrás exponer esa clase a otros desarrolladores. Para obtener más información acerca de los URI de contenido, consulta la sección Cómo diseñar URI de contenido. Para obtener más información acerca de las intents, consulta la sección Intents y acceso a datos.
4. Agrega otras piezas opcionales, como datos de muestra o una implementación de AbstractThreadedSyncAdapter que pueda sincronizar datos entre el proveedor y datos basados en la nube.