Android Essentials

Manejo de datos

Fernando Berti

SQLite

FOLDER IT

Las siguientes son algunas clases que debemos conocer y entender para poder utilizar de manera efectiva la API de SQLite

```
android.database.sqlite.SQLiteDatabase
android.database.sqlite.SQLiteCursor
android.database.sqlite.SQLiteQueryBuilder
android.content.ContentValues
android.database.Cursor
android.database.SQLException
android.database.sqlite.SQLiteOpenHelper
```

SQLite

FOLDER IT

`SQLiteDatabase`: representa la base de datos por lo general referenciando a un archivo `*.db` en el sistema de archivos.

- Se puede utilizar este objeto para consultar, insertar, actualizar, o eliminar una tabla en la base de datos.
- También se puede utilizar para ejecutar una única sentencia SQL, realizar transacciones o definir tablas mediante DDL
- Por lo general hay una sola instancia de este objeto en su aplicación que representa su base de datos.

`SQLiteCursor`: colección de filas que se devuelven desde una SQLiteDatabase.

- También implementa la interfaz `android.database.Cursor`
- Este objeto tiene métodos para recorrer filas una a la vez como un cursor hacia adelante y recuperar las filas sólo cuando sea necesario.
- También puede saltar hacia adelante o hacia atrás si es necesario mediante la aplicación de cualidades de ventanas.
- Y se va a utilizar para leer los valores de columna para cualquier fila actual.

SQLite

FOLDER IT

`SQLiteQueryBuilder`:

- Se usa para construir una consulta SQLite incrementalmente especificando los nombres de tabla, nombres de columna, cláusula where, etc., como campos separados.
- Esta clase tiene un número de métodos set para construir gradualmente la consulta en lugar de especificar toda la consulta SQL como una sola cadena.

`ContentValues`:

- Un objeto Java de esta clase contiene un conjunto de pares clave/valor que son utilizados por un número de clases de SQLite para insertar o actualizar una fila de datos.

`SQLException`:

- API de base de datos utilizada Android SQLite lanzar excepciones cuando hay errores.

SQLite

FOLDER IT

`SQLiteOpenHelper`:

- Proporciona acceso a una `SQLiteDatabase`, dado un nombre de archivo de la DB, este objeto se comprueba si esa DB ya está instalada y disponible.
- Si está disponible realiza una comprobación para ver si la versión es la misma. Si es así, proporciona una referencia a la `SQLiteDatabase` que representa esa base de datos. Si la versión es diferente, ofrece una función de callback para migrar la base de datos antes de proporcionar una referencia válida para la misma.
- Si la base de datos no existe, entonces ofrece una función de callback para de crear y llenar la base de datos.

SQLite nullColumnHack

FOLDER IT

Es un String, opcional, puede ser nulo.

SQL no permite insertar una fila completamente vacía sin darle un nombre al menos a una columna.

Si no provees valores, no hay nombre de columnas y no se puede insertar la fila.
Si no se setea como nulo, el parámetro `nullColumnHack` provee un nombre de una columna `nullable` para insertar explícitamente un `null`

Content Provider

FOLDER IT

Administran el acceso a un conjunto estructurado de datos.
Encapsulan los datos y proporcionan mecanismos para definir la seguridad de los datos.

Los `ContentProvider` son la interfaz estándar que conecta datos en un proceso con código que se ejecuta en otro proceso.

No hace falta desarrollar tu propio `ContentProvider` si no vas a compartir tus datos con otras aplicaciones. Pero, sí necesitas tu propio proveedor para proporcionar sugerencias de búsqueda personalizadas en tu aplicación o si quieres copiar y pegar datos o archivos complejos de tu aplicación en otras aplicaciones, etc

Android incluye proveedores de contenido que administran datos como video, audio, imágenes e información de contacto personal.

Content Provider

FOLDER IT

Estos nos que permiten ocultar detalles de implementación para el acceso a conjuntos estructurados de datos.

Encapsulan los datos y proveen mecanismos para definir la forma de accederlos.

Son interfaces estándares ejecutadas por un proceso, con el objetivo de bridar datos a otro proceso (servicio o actividad) conectado.

Cuando se desea acceder a datos en un content provider, se usa un objeto `ContentResolver`, de nuestro `Context`, para comunicarse con el provider como un cliente.

Content Provider

FOLDER IT

A cada proveedor de contenido se debe registrar en el archivo manifest de Android

Tenemos un identificador
llamado **authorities**
Es como el dominio de una URL

```
<manifest>
      ....
      <provider
         android:name=".clase4.MyContentProvider"
         android:authorities="com.folderit.net.MyProvider"
         android:enabled="true"
         android:exported="true" />
      ....
</manifest>
```

Un `Uri` de Android es de la forma:

```
content://<authority-name>/<path-segment1>/<path-segment2>/...
```

Content Provider

FOLDER IT

Algunos ejemplos de URLs de contenido en contents providers brindados por android.

```
content://media/internal/images
content://contacts/people/
content://contacts/people/23
```

Estos providers no tienen un nombre de autoridad completo
Los providers ofrecidos por Android pueden no llevar un nombre completo

Teniendo en cuenta estos URI de contenido, se espera que un proveedor pueda recuperar las filas que representan los URI.

También se espera que el proveedor pueda alterar el contenido en este URI utilizando cualquiera de los métodos de cambios de estado: insertar, actualizar o eliminar

Content Provider

FOLDER IT

Implementar un Content Provider requiere seguir los siguientes pasos:

- Planificar su base de datos, URIs, nombres de columna, y así sucesivamente, y crear una clase de metadatos que defina constantes para todos estos elementos de metadatos
- Extender de la clase abstracta ContentProvider.
- Implementar estos métodos: `query(), insert(), update(), delete() y getType()`
- Registrar el Provider en el manifest
- Utilizar el Content Provider.