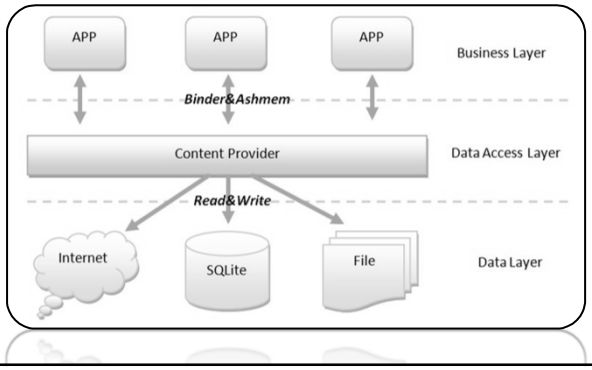
Proveedores de contenido (1)

Un proveedor de contenidos (***ContentProvider***) permite a una aplicación compartir sus datos (bd, archivo, *shared preferences*, etc.) con otra aplicación.

Android ofrece varios *ContentProvider* accesibles para las aplicaciones: audio, video, imágenes, contactos…

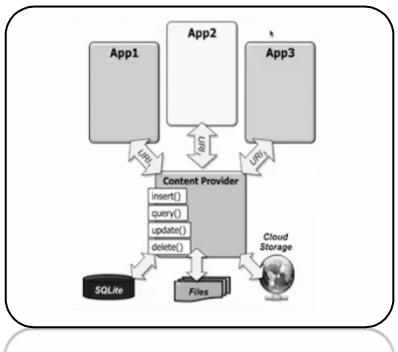


44

Proveedores de contenido (2)

Componentes del *ContentProvider*:

- URI: enlace al proveedor de contenidos.
- Métodos: permiten gestionar los datos (*insert*, *query*, *update*, *delete*).



45

Proveedores de contenido (3)

Para acceder a un *ContentProvider* debemos utilizar su URI, que se construye de la siguiente forma:

- **Prefijo**: indica que los datos son controlados por un *ContentProvider*.
- **Autoridad**: aplicación responsable del *ContentProvider*.
- **Ruta**: determina el tipo de datos que se van a utilizar, habitualmente una tabla.
- **Id**: identificador del dato solicitado.

46

Proveedores de contenido (4)

- Crear un *ContentProvider*

Para crear un *ContentProvider* hemos de seguir los siguientes pasos:

1) Implementar un sistema para almacenar los datos: base de datos, archivo, *SharedPreferences*, etc.
2) Crear una clase que extienda a *ContentProvider*, y sobrecargar los métodos: *onCreate*, *getType*, *query*, *insert*, *update* y *delete*.
3) Declarar el *ContentProvider* en el manifiesto de la aplicación.

47

Proveedores de contenido (5)

- Crear un *ContentProvider*

- *onCreate*: permite iniciar el *ContentProvider*.
- *getType*: devuelve el tipo **MIME** correspondiente al *ContentProvider*.
- *query*: devuelve un **Cursor** y permite obtener un dato almacenado en el *ContentProvider*.
- *insert*: inserta un dato en el *ContentProvider*.
- *update*: actualiza un dato del *ContentProvider*.
- *delete*: elimina un dato del *ContentProvider*.

48

Proveedores de contenido (6)

- Utilizar un *ContentProvider*

Una vez que hemos creado el *ContentProvider* podemos acceder a los datos desde otra aplicación.

El acceso a los datos del *ContentProvider* se realiza mediante un objeto *ContentResolver*, a través del cual podremos invocar a los métodos: *query*, *insert*, etc.

El proceso para obtener los datos del *ContentProvider* puede ser lento, por lo que se aconseja realizar la tarea en segundo plano y para ello se utiliza el cargador *CursorLoader* implementando la interfaz *LoaderManager.LoaderCallbacks <Cursor>* en la actividad o fragmento desde el que vayamos a acceder a los datos.

49

Proveedores de contenido (7)

- Utilizar un *ContentProvider*

Esta interfaz dispone de tres métodos:

- *Loader<Cursor> onCreateLoader (int id, Bundle arg)*

  Se encarga de iniciar el cargador cuyo id se recibe como parámetro.

- *void onLoaderFinished (Loader<Cursor> loader, Cursor data)*

  Se invoca al finalizar la carga de datos, disponiendo del cursor "data" para recorrer los datos obtenidos.

- *void onLoaderReset (Loader<Cursor> loader)*

  Se invoca cuando el cargador se ha reiniciado quedando los datos anteriores inaccesibles.

50

Proveedores de contenido (8)

- Utilizar un *ContentProvider*

Una vez implementada la interfaz podremos utilizar *getLoaderManager()* para obtener una referencia al cargador, y a partir de ella invocar al método *initLoader* para iniciar el proceso de carga de datos.

Mediante *getContentResolver()* podemos obtener una instancia del *ContentResolver* que nos va a permitir realizar las operaciones *query*, *insert*, *update* y *delete*, sobre los datos del *ContentProvider*.

51

Proveedores de contenido (9)

- Acceder a datos de otras aplicaciones

Android ofrece varios *ContentProvider* accesibles para las aplicaciones: imágenes, video, contactos, etc.

Para utilizar uno de estos *ContentProvider* hemos de seguir los siguientes pasos:

1) Solicitar el permiso adecuado en función de los datos y la operación a realizar.

   Por ejemplo, para acceder a los contactos solicitaremos permiso de lectura sobre ellos:

```
<uses-permission android:name="android.permission.READ_CONTACTS" />
```

52

Proveedores de contenido (10)

- Acceder a datos de otras aplicaciones

2) Obtener un *ContentResolver* invocando al método *getContentResolver()*.

```
ContentResolver cr = getContentResolver();
```

3) Invocar a uno de los métodos facilitados por el *ContentProvider*: *query*, *insert*, *delete*, *update*.

   Lo más habitual es que accedamos para leer algún dato y por tanto el método a utilizar será *query*.

53

Proveedores de contenido (11)

- Acceder a datos de otras aplicaciones

Parámetros del método *query*:

- URI: ruta del *ContentProvider*.
- *proyection* (*String*[]): lista de columnas.
- *selection* (*String*): filtro de selección.
- selectionArgs (*String*[]): lista de argumentos del filtro de selección.
- *sortOrder* (*String*): criterio de ordenación.

Ejemplo: obtener todos los valores de columna1 y columna2

```
String proyection[] = { columna1, columna2 };
Cursor cur = cr.query(URI, proyection, null, null, null);
```

54

Proveedores de contenido (12)

- Acceder a datos de otras aplicaciones

Para poder realizar estas peticiones tendremos que conocer la URI y la estructura de las tablas.

ContactProvider ofrece tres tablas:

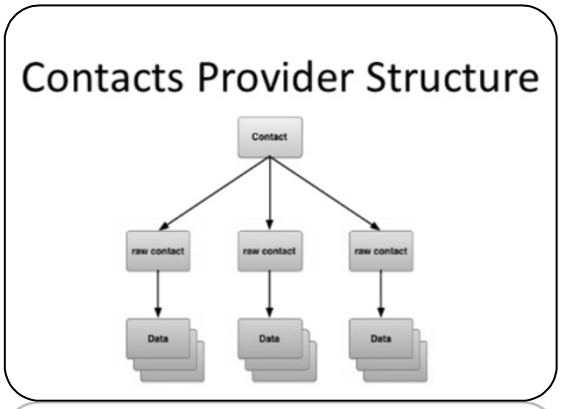
- *ContactsContract.Contacts*: contiene una fila por cada contacto.
- *ContactsContract.RawContacts*: contiene una fila por cada cuenta de cada contacto.
- *ContactsContract.Data*: contiene una fila por cada dato de cada cuenta de cada contacto.

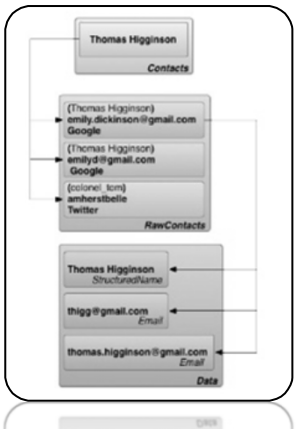
https://developer.android.com/reference/android/provider/ContactsContract.html

55

Proveedores de contenido (13)

- Acceder a datos de otras aplicaciones





56

Proveedores de contenido (14)

- Acceder a datos de otras aplicaciones

Ejemplo: obtener id y nombre a mostrar de todos los contactos.

```
String proyeccion[] = {
    ContactsContract.Contacts._ID,
    ContactsContract.Contacts.DISPLAY_NAME
};
ContentResolver cr = getContentResolver();
Cursor cur = cr.query(ContactsContract.Contacts.CONTENT_URI, proyeccion,
    null, null, null);
```

57

Proveedores de contenido (15)

- Acceder a datos de otras aplicaciones

5) El método *query* nos devuelve un cursor con el que podemos recorrer los registros que hemos obtenido mediante nuestra consulta.

```
if (cur.getCount() > 0) {
  while (cur.moveToNext()) {
    String id = cur.getString
      (cur.getColumnIndex(ContactsContract.Contacts._ID));
    String nombre = cur.getString
      (cur.getColumnIndex(ContactsContract.Contacts.DISPLAY_NAME));
  }
}
cur.close();
```

58

Proveedores de contenido (16)

- Acceder a datos de otras aplicaciones

También podemos acceder a datos individuales mediante la clase *ContactsContract.CommonDataKinds*.

```
Cursor curTlf = cr.query
(ContactsContract.CommonDataKinds.Phone.CONTENT_URI,
    null, ContactsContract.CommonDataKinds.Phone.CONTACT_ID +" = ?",
    new String[]{id}, null);
while (curTlf.moveToNext()) {
  String tlf = cursorTelefono.getString(cursorTelefono.getColumnIndex
    (ContactsContract.CommonDataKinds.Phone.DATA));
```

59