# Content Providers – Trình cung cấp nội dung

## Content Providers

* Truy cập, chỉnh sửa dữ liệu (nội dung) có cấu trúc. Dữ liệu: **Databases, files, network…**
* Tạo khi cần **chia sẽ** dữ liệu với **ứng dụng khác**.
* Android cung cấp Content Providers cho
  + Audio
  + Video
  + Image
  + Personal Contact Infomation

## Content URIs – Cấu trúc dữ liệu

**<prefix>://<authority>/<data\_type>/<id>**

* **Prefix:** luôn là **content://**
* **<authority>**: Chỉ định tên cụ thể của **Content Provider** (VD: contacts, browser,…). Đối với một số Content Provider khác bạn sẽ phải chỉ định tên đầy đủ (VD: com.laptrinhtuduy.statusprovider).
* **<data\_type>**: Chỉ rõ kiểu dữ liệu (VD: Để lấy tất cả các liên hệ trong Contacts Content Provider thì kiểu dữ liệu là people và URI sẽ là: **content://contacts/people**.
* **<id>** : Chỉ định rõ một record (VD: Nếu bạn muốn lấy địa chỉ liên lạc thứ 5 trong Contacts Content Provider thì URI sẽ là: **content://contacts/people/5**.

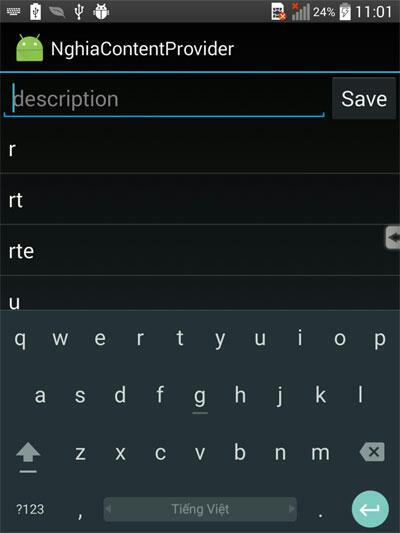
## Phương thức truy vấn:

* **onCreate()**: Phương thức này được gọi khi **Provider** được bắt đầu.
* **query()**: Truy vấn, lấy dữ liệu. Trả về**Cursor**.
* **insert()**: chèn một dòng.
* **delete()**: xóa một dòng.
* **update()**: cập nhật một dòng.
* **getType()**: Phương thức trả về kiểu **MIME** của dữ liệu tại các **URI**.

## Cấu trúc Package đề xuất:

* data
  + model
    - Note.java
  + AppProvider.java
  + BaseDataHandle.java
  + NoteData.java

## Tạo Content Provider:

1. Khởi tạo CSDL (SQLite) truy vấn.
   1. Tạo lớp CSDL
   2. Định nghĩa Database Schema
   3. Creating Database
   4. Updating Database
2. Tạo Content Provider Class, định nghĩa Content URL để truy xuất
3. Đăng kí trong Mainifesh
4. Kế thừa truy vấn, trong lớp kế thừa Content Provider
5. Test, chạy thử kiểm tra

### Bước 1: Khởi tạo CSDL (SQLite) truy vấn.

Class Note, binding bảng Note

/\*\* Hiển thị dòng trong Table Note \*/

public class Note {

private long \_id;

private String description;

// Getter, Setter, Constructor

}

Class NoteData, Xử lý truy vấn trong bảng Note

/\*\* Chứa hàm truy vấn bảng Note \*/

public class NoteData {

// TABLE NAME

public static final String TABLE\_NOTE = "notes";

// COLUMN NAME

public static final String COL\_ID = "\_id";

public static final String COL\_DESCRIPTION = "description";

// STATEMENT

public static final String NOTE\_CREATE\_STATEMENT =

"CREATE TABLE " + TABLE\_NOTE + "(" +

COL\_ID + " INTEGER PRIMARY KEY AUTOINCREMENT, " +

COL\_DESCRIPTION + " TEXT " +

")";

/\*

Các hàm truy vấn

\*/

/\* Đổi Note to ContentValues \*/

private static void noteToContentValues(Note note, ContentValues values) {

//bindding table common value

//BaseDataHandle.baseDataObjectToContentValues(note, values);

values.put(COL\_ID, note.get\_id());

values.put(COL\_DESCRIPTION, note.getDescription());

}

/\*\* Đổi Cursor to Note \*/

private static void cursorToNote(Cursor cursor, Note note) {

//BaseDataHandle.cursorToBaseDataObject(cursor, note);

int id = cursor.getInt(cursor.getColumnIndex(COL\_ID));

String description = cursor.getString(cursor.getColumnIndex(COL\_DESCRIPTION));

note.set\_id(id);

note.setDescription(description);

}

  /\*\* Tạo mới Note, sử dụng 2 hànm ở trên \*/

public static long add(Context context, Note note) {

ContentValues values = new ContentValues();

noteToContentValues(note, values);

return BaseDataHandle.insert(context, TABLE\_NOTE, values);

}

}

Class BaseDataHandle, Khởi tạo CSDL Sqlite

/\*\* Xử lý, khởi tạo SQLite Database \*/

public class BaseDataHandle extends SQLiteOpenHelper {

public static final int DATABASE\_VERSION = 1;

public static final String DATABASE\_NAME = "test\_db";

public BaseDataHandle(Context context) {

super(context, DATABASE\_NAME, null, DATABASE\_VERSION);

}

@Override

public void onCreate(SQLiteDatabase db) {

try {

// Tạo bảng NOTE

db.execSQL(NoteData.NOTE\_CREATE\_STATEMENT);

} catch (Exception ex) {

ex.printStackTrace();

}

}

@Override

public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {

// Xóa bảng NOTE

db.execSQL("DROP TABLE IF EXISTS " + NoteData.TABLE\_NOTE);

// Tạo lại

onCreate(db);

}

// Xử lý khác

}

### Bước 2: Tạo Content Provider Class, định nghĩa Content URL để truy xuất

Content Provider Class, xử lý khi nhận truy vấn từ Content Url

/\*\* Xử lý khi nhận Content Url \*/

public class AppProvider extends ContentProvider {

//<prefix>://<authority>/<data\_type>/<id>

private static final String AUTHORITY = "com.imark.nghia.nghiacontentprovider.provider";

private static final String NOTE\_BASE\_PATH = "notes";

private static final Uri NOTE\_CONTENT\_URI = Uri.parse(

"content://" + AUTHORITY + "/" + NOTE\_BASE\_PATH

);

private static final int CODE\_NOTES = 100;

private static final int CODE\_NOTE\_ID = 110;

private static final UriMatcher sUriMatcher = new UriMatcher(UriMatcher.NO\_MATCH);

static {

// để kiểm tra Loại URI

// Ví dụ: Nhận "NOTE\_BASE\_PATH" => loại CODE\_NOTES

sUriMatcher.addURI(AUTHORITY, NOTE\_BASE\_PATH, CODE\_NOTES);

sUriMatcher.addURI(AUTHORITY, NOTE\_BASE\_PATH + "/#", CODE\_NOTE\_ID);

}

// Loại trả về

private static final String TYPE\_NOTE\_CONTENT\_ITEM = ContentResolver.CURSOR\_ITEM\_BASE\_TYPE + "/mt-note";

private static final String TYPE\_NOTE\_CONTENT = ContentResolver.CURSOR\_DIR\_BASE\_TYPE + "mt-note";

//...

}

### Bước 3: Đăng kí trong Mainifesh

<manifest ...>

<application ...>

<!--Khai báo Provider-->

<provider

android:authorities="com.imark.nghia.nghiacontentprovider.provider"

android:name=".data.AppProvider"

android:multiprocess="true" />

</application>

</manifest>

### Bước 4: Kế thừa truy vấn, trong lớp kế thừa Content Provider

/\*\* Xử lý khi nhận Content Url \*/

public class AppProvider extends ContentProvider {

//...

private SQLiteDatabase mDB;

@Override

public boolean onCreate() {

// Khởi tạo Database

mDB = new BaseDataHandle(getContext()).getWritableDatabase();

return true;

}

@Nullable

@Override

public Cursor query(@NonNull Uri uri, String[] projection, String selection, String[] selectionArgs, String sortOrder) {

// Dùng QueryBuilder để truy vấn

SQLiteQueryBuilder queryBuilder = new SQLiteQueryBuilder();

queryBuilder.setTables(NoteData.TABLE\_NOTE); //add Select table

// Phân loại Uri

int uriType = sUriMatcher.match(uri);

switch (uriType) {

case CODE\_NOTES:

// not Where

break;

case CODE\_NOTE\_ID:

queryBuilder.appendWhere(NoteData.COL\_ID + "=" + uri.getLastPathSegment()); // add Where

break;

default:

throw new IllegalArgumentException("Unknown URI");

}

Cursor cursor = queryBuilder.query(mDB, projection, selection, selectionArgs, null, null, sortOrder);

// Notify change to Client

if (getContext() != null)

cursor.setNotificationUri(getContext().getContentResolver(), uri);

return cursor;

}

@Nullable

@Override

public String getType(Uri uri) {

// Mỗi loại Uri => loại khác nhau

int uriType = sUriMatcher.match(uri);

switch (uriType) {

case CODE\_NOTES:

return TYPE\_NOTE\_CONTENT;

case CODE\_NOTE\_ID:

return TYPE\_NOTE\_CONTENT\_ITEM;

default:

throw new IllegalArgumentException("Unknown URI");

}

}

@Nullable

@Override

public Uri insert(@NonNull Uri uri, ContentValues values) {

long rowID = mDB.insert(NoteData.TABLE\_NOTE, null, values);

if(rowID > 0){

// thêm thành công, trả về Uri

Uri resultUri = ContentUris.withAppendedId(NOTE\_CONTENT\_URI, rowID);

// Notify change to Client (Caller)

if (getContext() != null)

getContext().getContentResolver().notifyChange(resultUri, null);

return resultUri;

}

throw new SQLException("Failed to add a record into " + uri);

}

@Override

public int delete(@NonNull Uri uri, String selection, String[] selectionArgs) {

int rowAffected = 0; // dòng ảnh hưởng

int uriType = sUriMatcher.match(uri);

switch (uriType) {

case CODE\_NOTES:

rowAffected = mDB.delete(NoteData.TABLE\_NOTE, selection, selectionArgs);

break;

case CODE\_NOTE\_ID:

// build where, check selection

String id = uri.getLastPathSegment();

String where = NoteData.COL\_ID + "=" + id;

if (!TextUtils.isEmpty(selection)){

// Selection not null => Where ... and

where += " AND " + selection;

}

rowAffected = mDB.delete(NoteData.TABLE\_NOTE, where, selectionArgs);

break;

default:

throw new IllegalArgumentException("Unknown URI");

}

// Notify change to Client (Caller)

if (getContext() != null) {

getContext().getContentResolver().notifyChange(uri, null);

}

return rowAffected;

}

@Override

public int update(@NonNull Uri uri, ContentValues values, String selection, String[] selectionArgs) {

int rowAffected = 0; // dòng ảnh hưởng

int uriType = sUriMatcher.match(uri);

switch (uriType) {

case CODE\_NOTES:

rowAffected = mDB.update(NoteData.TABLE\_NOTE, values, selection, selectionArgs);

break;

case CODE\_NOTE\_ID:

// build where, check selection

String id = uri.getLastPathSegment();

String where = NoteData.COL\_ID + "=" + id;

if (!TextUtils.isEmpty(selection)){

// Selection not null => Where ... and

where += " AND " + selection;

}

rowAffected = mDB.update(NoteData.TABLE\_NOTE, values, where, selectionArgs);

break;

default:

throw new IllegalArgumentException("Unknown URI");

}

// Notify change to Client (Caller)

if (getContext() != null) {

getContext().getContentResolver().notifyChange(uri, null);

}

return rowAffected;

}

}

### Bước 5: Test, chạy thử kiểm tra

Hiển thị ListView trong App khác:

//ListView mListView\_data;

//ArrayAdapter mAdapterData;

//ArrayList mArrayData;

private void mListView\_data\_init() {

mArrayData = new ArrayList<>();

// Triệu hồi AppProvider lấy dữ liệu

Cursor cursor = getContentResolver().query(

Uri.parse("content://com.imark.nghia.nghiacontentprovider.provider/notes"),

null, null, null, null

);

// Uri.withAppendedPath(URI, ID); // thêm Id vào URI. Ví dụ: content://.../.../id

// Cursor to List

while (cursor.moveToNext()) {

String description = cursor.getString(cursor.getColumnIndex("description"));

mArrayData.add(description);

}

cursor.close();

mAdapterData = new ArrayAdapter<String>(getBaseContext(), android.R.layout.simple\_list\_item\_1, mArrayData);

mListView\_data.setAdapter(mAdapterData);

}

Thêm Note từ App khác:

//EditText mEditText\_description;

//Button mButton\_save;

View.OnClickListener mButton\_save\_onClick = new View.OnClickListener() {

@Override

public void onClick(View v) {

ContentValues values = new ContentValues();

values.put("description", mEditText\_description.getText() + ""); // description on Note

// triệu hồi AppProvider thực hiện qua ContentUrl

Uri uri = getContentResolver().insert(

Uri.parse("content://com.imark.nghia.nghiacontentprovider.provider/notes"),

values);

// hiển thị kết quả trả về

Toast.makeText(getBaseContext(), uri.toString(), Toast.LENGTH\_LONG).show();

mListView\_data\_init();

}

};

Lấy Note thông qua Id từ App khác:

// thêm Id vào URI. Ví dụ: content://.../.../id

// id lấy là "1"

Uri uri = Uri.withAppendedPath(

Uri.parse("content://com.imark.nghia.nghiacontentprovider.provider/notes"),

"1"

);

// Triệu hồi AppProvider lấy dữ liệu

Cursor cursor = getContentResolver().query(

uri,

null, null, null, null

);

while (cursor.moveToNext()) {

String description = cursor.getString(cursor.getColumnIndex("description"));

mArrayData.add(description);

}

cursor.close();

**Tham khảo:**

* <http://developer.android.com/guide/topics/providers/content-providers.html>
* <https://laptrinhtuduy.wordpress.com/2014/04/28/content-providers-trong-android/>
* <http://www.tutorialspoint.com/android/android_content_providers.htm>
* [http://code.tutsplus.com/tutorials/android-fundamentals-working-with-content-providers–mobile-5549](http://code.tutsplus.com/tutorials/android-fundamentals-working-with-content-providers--mobile-5549)