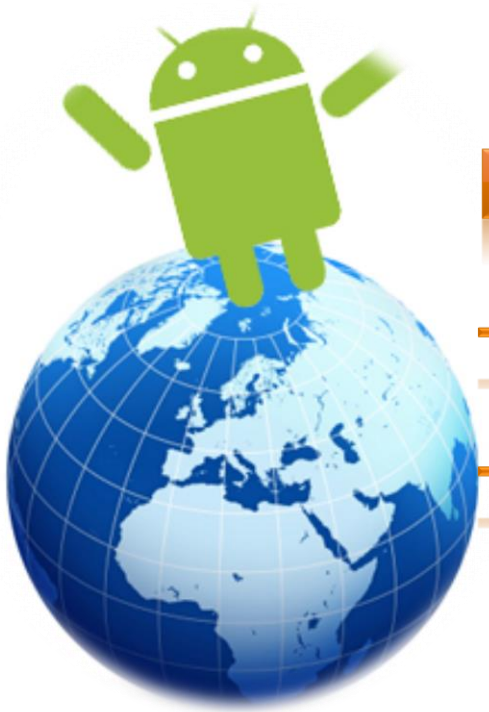
LẬP TRÌNH ANDROID NÂNG CAO



BÀI 4 :

- CONTENT PROVIDER
- BROADCAST RECEIVER

LẬP TRÌNH ANDROID NÂNG CAO

PHẦN I :
CONTENT PROVIDER

5.5.1 Giới thiệu

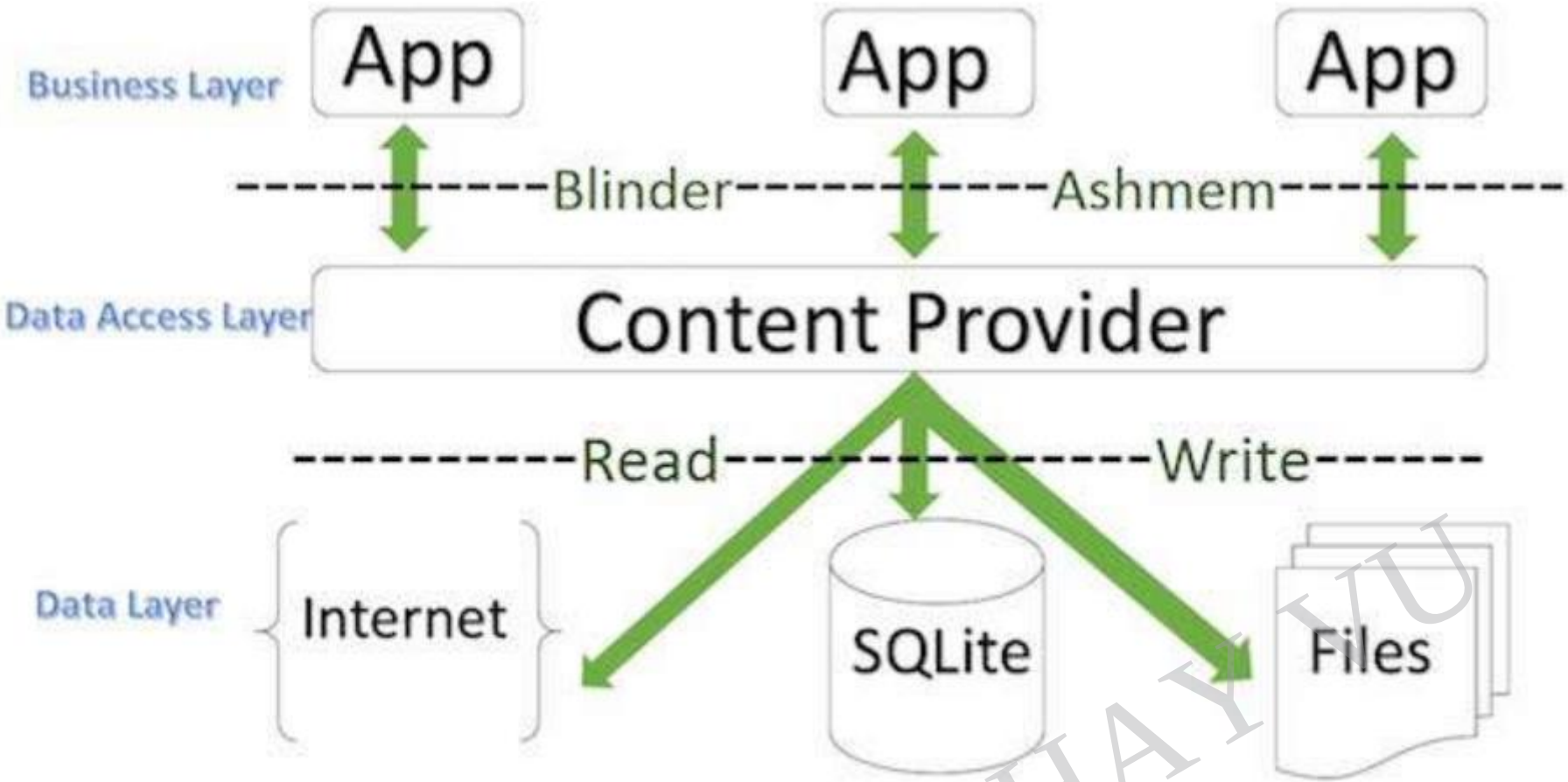
- Content Provider là 1 bộ cấp dữ liệu đặc biệt, cung cấp nhiều cách chuẩn hóa để lấy và thao tác trên dữ liệu được lưu trữ trong android
- Content Provider giúp chia sẽ dữ liệu giữa các ứng dụng trong android tiện lợi và nhanh chóng

```
Browser
CallLog
Contacts
    people
    phones
    photos
    groups
MediaStroe
    audio
        albums
        artists
        Geners
        playlists
    images
        thumbnails
    video
Settings
```

Các tiện ích dựa trên content providers

Content Provider	Intended Data
Browser	Browser bookmarks, browser history, etc.
CallLog	Missed calls, call details, etc.
Contacts	Contact details
MediaStore	Media files such as audio, video and images
Settings	Device settings and preferences

5.5.1 Giới thiệu



CONTENTPROVIDER

5.5.2 Sử dụng Content Provider

- Để sử dụng 1 content provider, ta cần tạo yêu cầu truy vấn dữ liệu theo dạng một đường dẫn URI:

  <prefix>://<authority>/<data_path>/<id>

  - **<prefix>**: Nó luôn được thiết lập 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_path>**: 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**.

5.5.2 Sử dụng Content Provider

- Ví dụ để lấy dữ liệu về bookmarks lưu trữ trong trình duyệt trong android, URI sẽ là

  content://browser/bookmarks
- Để lấy tất cả danh bạ điện thoại

  content://contacts/people
- Để lấy đúng contact thứ 3 trong danh bạ

  content://contacts/people/3

5.5.2 Sử dụng Content Provider

- Dữ liệu trả về bao gồm nhiều dòng. Để quản lý từng dòng, android dùng cursor (con trỏ)

```
CursorLoader loader=new
        CursorLoader(context, uri, null, null, null, null);
Cursor c=loader.loadInBackground();
```

- Hoặc có thể dùng

```
Cursor c = getContentResolver()
        .query(uri, null, null, null, null);
```

Tham số: URI, phép chiếu, SQLWHERE, ORDERBY

5.5.2 Sử dụng Content Provider

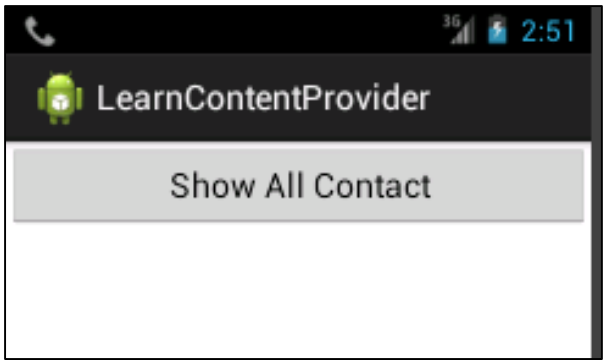
Lấy contacts:

```
public void showAllContacts()
{
    Uri uri=Uri.parse("content://contacts/people");
    ArrayList<String> list=new ArrayList<String>();
    CursorLoader loader=new
            CursorLoader(this, uri, null, null, null, null);
    Cursor c1=loader.loadInBackground();
    c1.moveToFirst();
    while(c1.isAfterLast()==false){
        String s="";
        String idColumnName=ContactsContract.Contacts._ID;
        int idIndex=c1.getColumnIndex(idColumnName);
        s=c1.getString(idIndex)+" - ";
        String nameColumnName=ContactsContract.Contacts.DISPLAY_NAME;
        int nameIndex=c1.getColumnIndex(nameColumnName);
        s+=c1.getString(nameIndex);
        c1.moveToNext();
        list.add(s);
    }
    c1.close();
    ListView lv=(ListView) findViewById(R.id.listView1);
    ArrayAdapter<String>adapter=new ArrayAdapter<String>(this,
            android.R.layout.simple_list_item_1, list);
    lv.setAdapter(adapter);
}
```

5.5.2 Sử dụng Content Provider

Hoặc có thể dùng getContentResolver thay cho CursorLoader

```
Uri uri=Uri.parse("content://contacts/people");
ArrayList<String> list=new ArrayList<String>();
Cursor c1=getContentResolver()
        .query(uri, null, null, null, null);
```




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

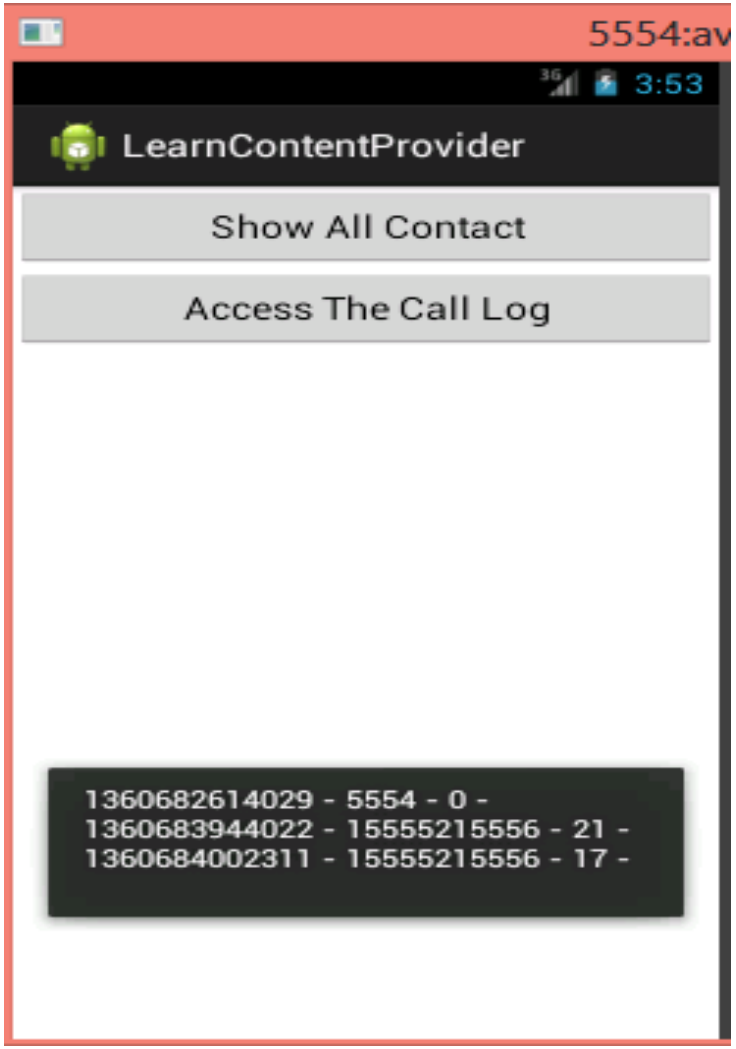
5.5.2 Sử dụng Content Provider

Truy xuất Call Log:

```
public void accessTheCallLog()
{
    String [] projection=new String[]{
            Calls.DATE,
            Calls.NUMBER,
            Calls.DURATION
            };
    Cursor c=getContentResolver().query(
            CallLog.Calls.CONTENT_URI,
            projection,
            Calls.DURATION+"<?",new String[]{"30"},
            Calls.DATE +" Asc");
    c.moveToFirst();
    String s="";
    while(c.isAfterLast()==false){
        for(int i=0;i<c.getColumnCount();i++){
            s+=c.getString(i)+" - ";
        }
        s+="\n";
        c.moveToNext();
    }
    c.close();
    Toast.makeText(this, s, Toast.LENGTH_LONG).show();
}
```

5.5.2 Sử dụng Content Provider

Truy xuất Call Log:




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

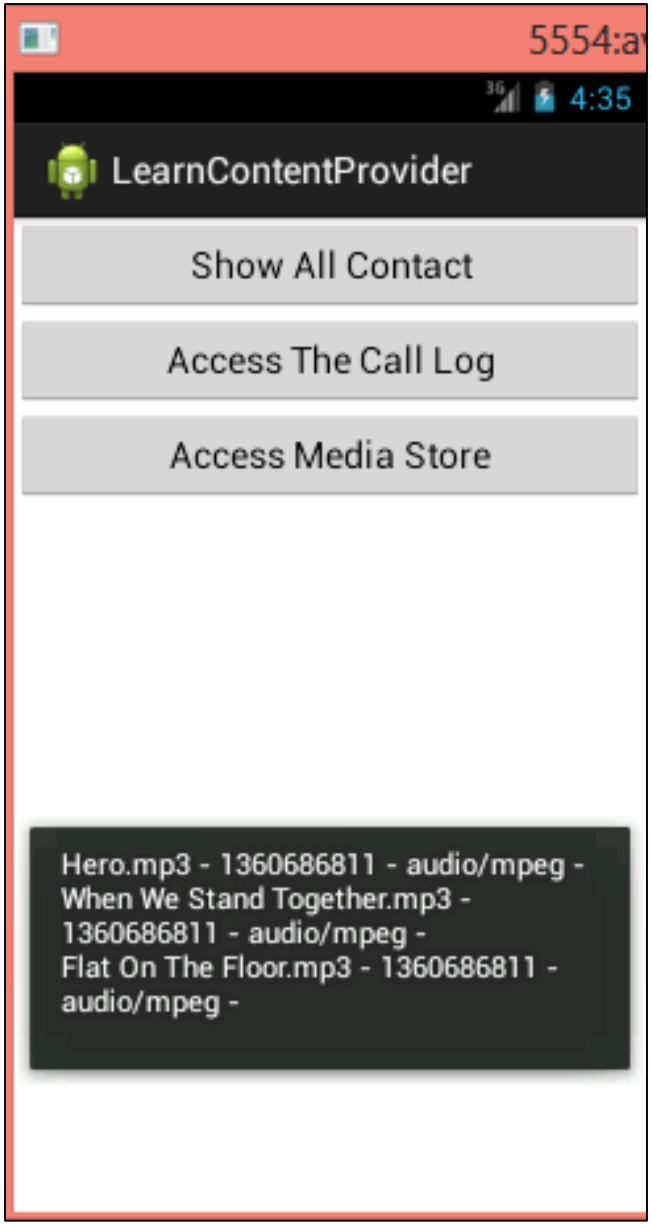
Giống như Contact, ta có thể dùng lớp **CursorLoader** để truy cập Call log

5.5.2 Sử dụng Content Provider

Truy xuất Media Store:

```
public void accessMediaStore()
{
    String []projection={
            MediaStore.MediaColumns.DISPLAY_NAME,
            MediaStore.MediaColumns.DATE_ADDED,
            MediaStore.MediaColumns.MIME_TYPE
            };
    CursorLoader loader=new CursorLoader
            (this, Media.EXTERNAL_CONTENT_URI,
                    projection, null, null, null);
    Cursor c=loader.loadInBackground();
    c.moveToFirst();
    String s="";
    while(!c.isAfterLast()){
        for(int i=0;i<c.getColumnCount();i++){
            s+=c.getString(i)+" - ";
        }
        s+="\n";
        c.moveToNext();
    }
    Toast.makeText(this, s, Toast.LENGTH_LONG).show();
    c.close();
}
```

5.5.2 Sử dụng Content Provider



Truy xuất Media Store:

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

Sử dụng **getContentResolver** để truy xuất

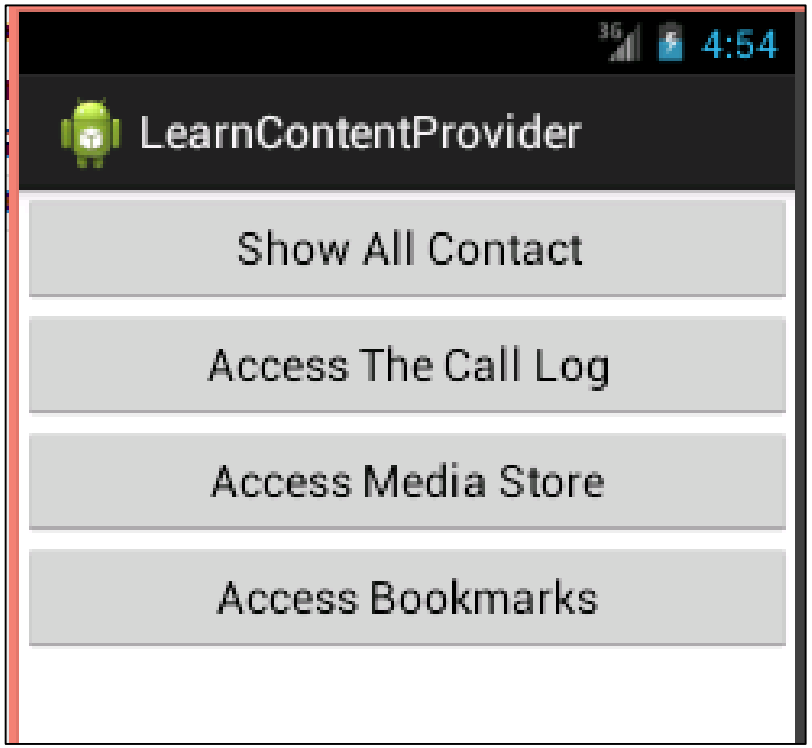
5.5.2 Sử dụng Content Provider

Truy xuất Book marks:

```
public void accessBookmarks()
{
    String []projection={
            Browser.BookmarkColumns.TITLE,
            Browser.BookmarkColumns.URL,
        };
    Cursor c=getContentResolver()
            .query(Browser.BOOKMARKS_URI, projection,
                    null, null, null);
    c.moveToFirst();
    String s="";
    int titleIndex=c.getColumnIndex
                    (Browser.BookmarkColumns.TITLE);
    int urlIndex=c.getColumnIndex
                    (Browser.BookmarkColumns.URL);
    while(!c.isAfterLast())
    {
        s+=c.getString(titleIndex)+" - "+
                c.getString(urlIndex);
        c.moveToNext();
    }
    c.close();
    Toast.makeText(this, s, Toast.LENGTH_LONG).show();
}
```

5.5.2 Sử dụng Content Provider

Truy xuất Book marks:




```
<uses-permission
    android:name=
        "com.android.browser.permission.READ_HISTORY_BOOKMARKS"
/>
```

5.5.3 Tự tạo Content Provider trong Android

- Để tạo 1 content provider cho riêng mình, ta cần kế thừa lớp ContentProvider và cài đặt lại một số phương thức
- Có 6 phương thức cần cài đặt lại:
  - **getType()**: trả về kiểu dữ liệu theo định dạng MIME từ URI đi kèm
  - **onCreate():** được gọi khi provider bắt đầu thực thi
  - **Query():** nhận yêu cầu từ người dùng. Kết quả trả về là 1 kiểu Cursor
  - **Insert():** thêm 1 dòng dữ liệu mới vào Content Provider
  - **Update():** cập nhật 1 dòng dữ liệu trong Content Provider
  - **Delete():** xóa 1 dòng dữ liệu trong Content Provider
- Chúng ta có thể chọn cách lưu trữ dữ liệu theo ý thích, như: kiểu file, XML, database, hoặc thông qua 1 web service

5.5.3 Tự tạo Content Provider trong Android

- Trước khi thực hiện, cần khai báo một vài hằng số:
  - Khai báo trong Manifest

    <provider name=".MyProvider“
    authorities="abc.com.MyProvider" . . >
  - Khai báo nội dung URI và đường dẫn:
  - Ví dụ:

```
public static final String AUTHORITY = "abc.com.MyProvider";
                    // giống khai báo trong file Manifest.xml
public static final String CONTENT_URI =
        "content://"+ AUTHORITY + "/demodb");
                    //URI để truy cập Content Provider
```

- Sử dụng UriMatcher

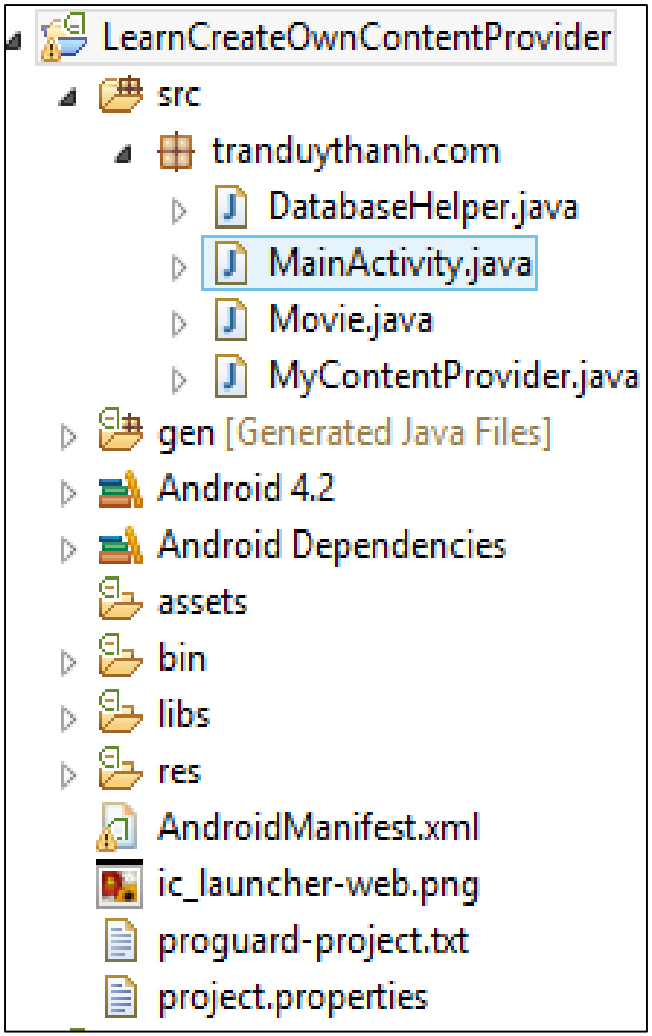
UriMatcher là 1 lớp hữu ích để so khớp nội dung trong URI và các tập mẫu trong ContentProvider. Nếu ta đưa nội dung không khớp với mẫu thì contentProvider sẽ không hiểu và không thể thực thi yêu cầu

5.5.4. Tự tạo Content Provider với 5 bước

1. Đầu tiên cần tạo một class **Conten Provider** kế thừa class **ContentProviderbase**.
2. Định nghĩa địa chỉ **URI** cho Content Provider, nó sẽ được dùng để truy xuất nội dung.
3. Tạo một **CSDL** của riêng để lưu các Content. Thông thường, Android sử dụng CSDL SQLite, cần phải Override phương thức **onCreate()** nó sẽ sử dụng phương thức **SQLite Open Helper** để tạo hoặc mở CSDL của provider. Khi ứng dụng khởi chạy, phương thức **onCreate()** sẽ xử lý các Content Providers của nó.
4. Tiếp theo cần phải **implement** các truy vấn của Content Provider để xử lý các yêu cầu khác nhau về dữ liệu.
5. Cuối cùng, **đăng ký** Content Provider trong manifest bằng cách sử dụng nhãn <provider>.

5.5.3 Tự tạo Content Provider

Ví dụ



Tạo project mới, đặt tên là "**LearnCreateOwnContentProvider**"

5.5.3 Tự tạo Content Provider

Ví dụ

Lớp DatabaseHelper

```
public class DatabaseHelper extends SQLiteOpenHelper {

    private static final String DATABASE_NAME = "movieDB";
    private static final int DATABASE_VERSION = 2;
    private static final String DATABASE_CREATE =
            "create table favorite_movie " +
            "( _id integer primary key," +
            "name text not null, " +
            "description text not null);";
    public DatabaseHelper(Context context, String name,
            CursorFactory factory,int version) {
        super(context, name, factory, version);
    }
    public DatabaseHelper(Context context){
        super(context, DATABASE_NAME, null, DATABASE_VERSION);
    }
    public void onCreate(SQLiteDatabase arg0) {
        arg0.execSQL(DATABASE_CREATE);
    }
    public void onUpgrade(SQLiteDatabase arg0, int arg1, int arg2) {
        arg0.execSQL("DROP TABLE IF EXISTS favorite_movie");
        onCreate(arg0);
    }
}
```

5.5.3 Tự tạo Content Provider

Ví dụ

Lớp Movie

```
public class Movie {
    private DatabaseHelper dbHelper;
    private SQLiteDatabase database;
    public final static String MOVIE_TABLE="favorite_movie";
    public final static String MOVIE_NAME="name";
    public final static String MOVIE_DESC="description";
    public final static String MOVIE_ID="_id";
    public Movie(Context context){
        dbHelper = new DatabaseHelper(context);
        database = dbHelper.getWritableDatabase();
    }
    public long createMovie(String name,String desc){
        ContentValues values = new ContentValues();
        values.put(MOVIE_NAME, name);
        values.put(MOVIE_DESC, desc);
        return database.insert(MOVIE_TABLE, null, values);
    }
    public void createMovie(ContentValues values){
        database.insert(MOVIE_TABLE, null, values);
    }
```

5.5.3 Tự tạo Content Provider

Ví dụ

Lớp Movie

```
public Cursor fetchAllMovies(){
    String[] cols = new String[] {MOVIE_NAME, MOVIE_DESC,MOVIE_ID};
    Cursor mCursor = database.query(true, MOVIE_TABLE,cols,null
            , null, null, null, null, null);
    if (mCursor != null) {
        mCursor.moveToFirst();
    }
    return mCursor;
}
public Cursor fetchOneMovie(String id){
    String[] cols = new String[]
            {MOVIE_NAME, MOVIE_DESC,MOVIE_ID};
    Cursor c=database.query(MOVIE_TABLE, cols,
            MOVIE_ID+"=?", new String[]{id},
            null, null, null);
    if (c != null) {
        c.moveToFirst();
    }
    return c;
}
public int delete(String id){
    return database.delete(MOVIE_TABLE,
            MOVIE_ID + "=" + id, null);
}
```

5.5.3 Tự tạo Content Provider

Ví dụ

Lớp Movie

```
    public int deleteall()
    {
        return database.delete(MOVIE_TABLE, null, null);
    }
    public int updateMovie(ContentValues values,String id){
        return database.update(MOVIE_TABLE, values,
                MOVIE_ID + "=?",new String[]{id});
    }
    public int updateMovie(ContentValues values,
            String selection,String[] selectionArgs)
    {
        return database.update(MOVIE_TABLE, values,
                selection, selectionArgs);
    }
}
```

5.5.3 Tự tạo Content Provider

Ví dụ

Lớp MyContentProvider

```
public Uri insert(Uri uri, ContentValues values) {
    m.createMovie(values);
    return null;
}
public boolean onCreate() {
    m = new Movie(this.getContext());
    // Setup the UriMatcher
    this.uriMatcher = new UriMatcher(UriMatcher.NO_MATCH);
    this.uriMatcher.addURI(AUTHORITY, "objects", MATCH_ALL);
    this.uriMatcher.addURI(AUTHORITY, "object/#", MATCH_ID);
    return true;
}
public Cursor query(Uri uri, String[] projection, String selection,
        String[] selectionArgs, String sortOrder) {
    Cursor cursor = null;
    switch (uriMatcher.match(uri)) {
    case MATCH_ALL:
        cursor= m.fetchAllMovies();
        break;
    case MATCH_ID:
        cursor=m.fetchOneMovie(uri.getLastPathSegment());
        break;
    }
    return cursor;
}
```

5.5.3 Tự tạo Content Provider

Ví dụ

Lớp ContentProvider

```
public class MyContentProvider extends ContentProvider {
    Movie m;
    public static final String CONTENT_URI =
            "content://tranduythanh.com.own.contentprovider";
    public static final int MATCH_ALL = 1;
    public static final int MATCH_ID = 2;
    public static final String AUTHORITY =
            "tranduythanh.com.own.contentprovider";
    private UriMatcher uriMatcher;
    public int delete(Uri arg0, String arg1, String[] arg2) {
        int ret=-1;
        switch(uriMatcher.match(arg0))
        {
            case MATCH_ALL:
                ret=m.deleteall();
                break;
            case MATCH_ID:
                ret=m.delete(arg0.getLastPathSegment());
                break;
        }
        return ret;
    }
    public String getType(Uri uri) {
        return "";
    }
```

5.5.3 Tự tạo Content Provider

Ví dụ

Lớp MyContentProvider

```
    @Override
    public int update(Uri uri, ContentValues values, String selection,
            String[] selectionArgs) {
        int ret=-1;
        switch (uriMatcher.match(uri)) {
        case MATCH_ALL:
            ret=m.updateMovie(values, null, null);
            break;
        case MATCH_ID:
            ret=m.updateMovie(values, uri.getLastPathSegment());
            break;
        default:
            break;
        }
        return ret;
    }
}
```

5.5.3 Tự tạo Content Provider

Ví dụ

```
<LinearLayout xmlns:android="http://schemas.a
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical" >
    <Button
        android:id="@+id/btninsert"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Insert" />
    <Button
        android:id="@+id/Update"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Update" />
    <Button
        android:id="@+id/btndelete"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="delete" />
    <Button
        android:id="@+id/display"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="display" />
    <Button
        android:id="@+id/displayOne"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="display one item" />
    <TextView
        android:id="@+id/lbl"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="Data ???" />
</LinearLayout>
```

Trong **MainActivityLayout XML**

5.5.3 Tự tạo Content Provider

Ví dụ

Trong MainActivity class

```
public class MainActivity extends Activity
            implements OnClickListener{
    TextView lbl;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        Button btn1=(Button)findViewById(R.id.btndelete);
        btn1.setOnClickListener(this);
        Button btn2=(Button)findViewById(R.id.btninsert);
        btn2.setOnClickListener(this);
        Button btn3=(Button)findViewById(R.id.display);
        btn3.setOnClickListener(this);
        Button btn4=(Button)findViewById(R.id.Update);
        btn4.setOnClickListener(this);
        Button btn5=(Button)findViewById(R.id.displayOne);
        btn5.setOnClickListener(this);
        lbl=(TextView)findViewById(R.id.lbl);
    }
```

5.5.3 Tự tạo Content Provider

Ví dụ

Trong MainActivity class

```
public void onClick(View v) {
    switch (v.getId()) {
    case R.id.btndelete:
        deleteAll();
        break;
    case R.id.btninsert:
        InsertData();
        break;
    case R.id.display:
        getData();

        break;
    case R.id.displayOne:
        getDataOne();

        break;
    case R.id.Update:
        update();
        break;
    }
}
```

5.5.3 Tự tạo Content Provider

Ví dụ

Trong MainActivity class

```
void deleteAll()
{
    Uri uri=Uri.parse(MyContentProvider.CONTENT_URI+"/objects");
    int ret=getContentResolver().delete(uri, null, null);
    String msg="Delete all are successful";
    if(ret<=0)
        msg="Delete all are failed";
    Toast.makeText(this, msg, Toast.LENGTH_LONG).show();

}
void update()
{
    Uri uri = Uri.parse(MyContentProvider.CONTENT_URI+"/object/2");
    ContentResolver resolver = this.getContentResolver();

    ContentValues newContent = new ContentValues();

    newContent.put(Movie.MOVIE_NAME, "namename_change");
    newContent.put(Movie.MOVIE_DESC,"Description_change");

    resolver.update(uri, newContent, null, null);
}
```

5.5.3 Tự tạo Content Provider

Ví dụ

Trong MainActivity class

```
void InsertData(){
    Uri uri = Uri.parse(MyContentProvider.CONTENT_URI+"/object");
    ContentResolver resolver = getContentResolver();
    ContentValues newContent = new ContentValues();
    newContent.put(Movie.MOVIE_NAME, "namename");
    newContent.put(Movie.MOVIE_DESC,"Description");
    resolver.insert(uri, newContent);
}
void getData(){
    Uri uri = Uri.parse(MyContentProvider.CONTENT_URI+"/objects");
    ContentResolver resolver = this.getContentResolver();
    Cursor cursor = resolver.query(uri, null, null, null, null);
    int nameIndex = cursor.getColumnIndex(Movie.MOVIE_NAME);
    int valueIndex = cursor.getColumnIndex(Movie.MOVIE_DESC);
    String str="";
    cursor.moveToFirst();
    while(!cursor.isAfterLast()){
        String name = cursor.getString(nameIndex);
        String value = cursor.getString(valueIndex);
        str += name + " " + value + "\n";
        cursor.moveToNext();
    }
    lbl.setText(str);
}
```

5.5.3 Tự tạo Content Provider

Ví dụ

Trong MainActivity class

```
void getDataOne()
{
    Uri uri = Uri.parse(MyContentProvider.CONTENT_URI+"/object/2");
    ContentResolver resolver = this.getContentResolver();

    Cursor cursor = resolver.query(uri, null, null, null, null);
    if(cursor.getCount() == 0)
    {
        return;
    }
    int nameIndex = cursor.getColumnIndex(Movie.MOVIE_NAME);
    int valueIndex = cursor.getColumnIndex(Movie.MOVIE_DESC);
    String str="";
    cursor.moveToFirst();
    do
    {
        String name = cursor.getString(nameIndex);
        String value = cursor.getString(valueIndex);
        str += name + " " + value + "\n";
        cursor.moveToNext();
    }while(!cursor.isAfterLast());
    lbl.setText(str);
}
}
```

5.5.3 Tự tạo Content Provider

Ví dụ

Trong AndroidManifest.xml

```
<application
    android:allowBackup="true"
    android:icon="@drawable/ic_launcher"
    android:label="@string/app_name"
    android:theme="@style/AppTheme">
    <activity..
    <provider android:name="tranduythanh.com.MyContentProvider"
        android:authorities="tranduythanh.com.own.contentprovider">
    </provider>
</application>
```

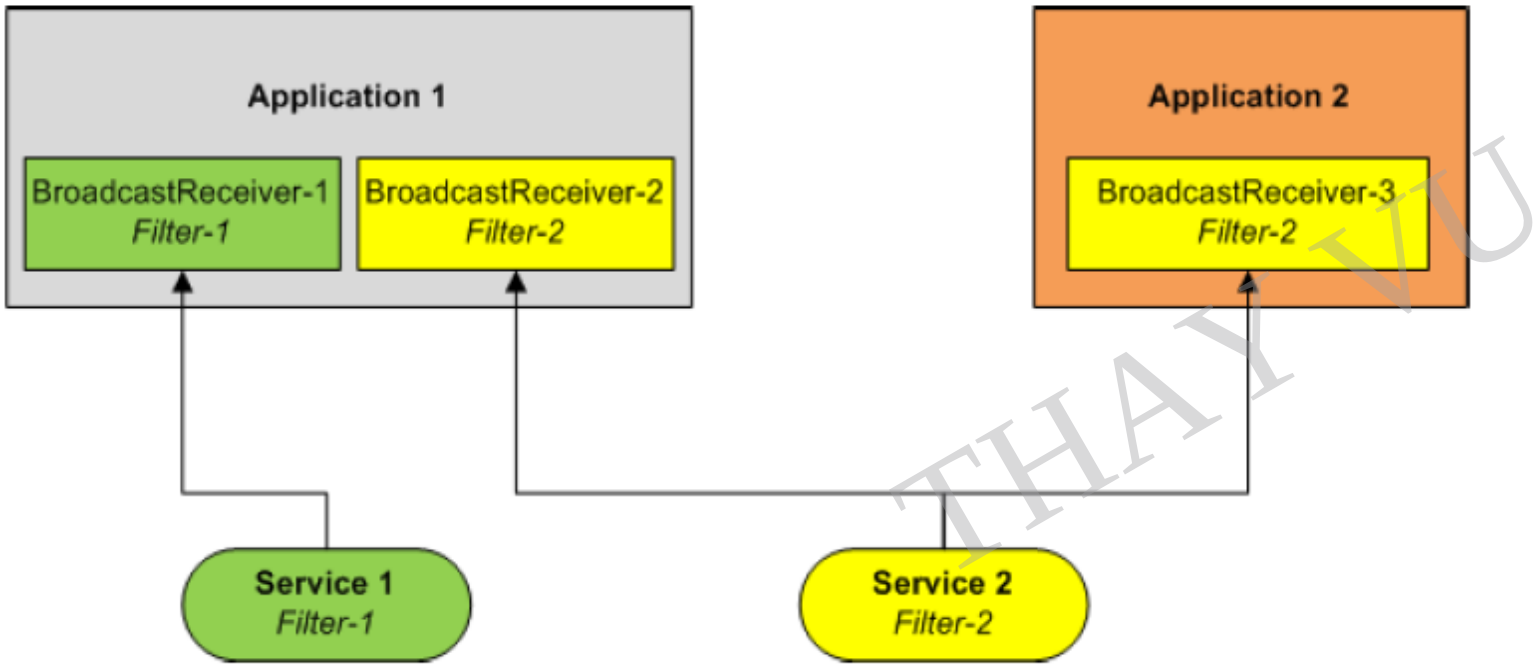
LẬP TRÌNH ANDROID NÂNG CAO

PHẦN II :

BROADCAST RECEIVER

6.2 BROADCAST RECEIVER

- **Broadcast Receiver** là thành phần phản hồi các thông báo phát ra từ các ứng dụng khác (intent) hoặc từ chính hệ thống (system).
- Ví dụ, hệ thống có thể tạo tín hiệu thông báo pin yếu, tin nhắn đến, … hoặc các ứng dụng cũng có thể khởi tạo các tín hiệu broadcast để thông báo cho ứng dụng khác, Broadcast Receiver sẽ thông dịch thông tin đó và khởi tạo hành động thích hợp.



6.2. BROADCAST RECEIVER

- Sau đây là hai bước quan trọng để làm Broadcast Receiver làm việc cho các Intent:
  1. Tạo Broadcast Receiver
  2. Đăng ký Broadcast Receiver

1. TẠO BROADCAST RECEIVER

- Triển khai Broadcast Receiver bằng việc ghi đè phương thức **onReceive(Context curContext, Intent broadcastMsg)**
- Ví dụ:

```
public class MyReceiver extends BroadcastReceiver {
    @Override
    public void onReceive(Context context, Intent intent) {
        Toast.makeText(context, "Intent Detected.", Toast.LENGTH_LONG).show();
    }
}
```

2. ĐĂNG KÝ BROADCAST RECEIVER

- Đăng ký một Broadcast Receiver để lắng nghe sự kiện, chúng ta có 2 cách:
  1. Đăng ký bằng code : **Context.registerReceiver()**
  2. Đăng ký trong AndroidManifest.xml file với thẻ **<receiver>**.
- Có 2 loại Broadcast như sau:
  1. Normal Broadcast (được gửi bởi **Context.sendBroadcast**): tất cả các receiver của broadcast này sẽ chạy cùng lúc (không có thứ tự)
  2. Orderd Broadcast (được gửi bởi **Context.sendOrderedBroadcast** : tại 1 thời điểm thì chỉ có 1 receiver nhận broadcast. Ta có thể quản lý thứ tự các receiver bằng thuộc tính **android:priority**

2. ĐĂNG KÝ BROADCAST RECEIVER

- Ví dụ: chúng ta đang đăng ký MyReceiver cho system event (sự kiện được tạo từ hệ thống) là ACTION_BOOT_COMPLETED, sự kiện này được kích hoạt bởi hệ thống một khi hệ điều hành Android đã hoàn thành tiến trình boot.

```
<application
   android:icon="@drawable/ic_launcher"
   android:label="@string/app_name"
   android:theme="@style/AppTheme" >
   <receiver android:name="MyReceiver">

      <intent-filter>
         <action android:name="android.intent.action.BOOT_COMPLETED">
         </action>
      </intent-filter>

   </receiver>
</application>
```

3. Các System Event phổ biến

Event	Miêu tả
android.intent.action.BATTERY_CHANGED	Thông báo này chứa trạng thái nạp, mức độ, và thông tin khác về pin
android.intent.action.BATTERY_LOW	Chỉ trạng thái low battery trên thiết bị
android.intent.action.BATTERY_OKAY	Chỉ rằng pin bây giờ là tốt sau khi low battery
android.intent.action.BOOT_COMPLETED	Đây là tín hiệu broadcast thông báo sau khi hệ thống đã kết thúc boot
android.intent.action.BUG_REPORT	Chỉ activity để báo cáo một bug
android.intent.action.CALL	Thông báo một lời gọi tới ai đó được xác định bởi dữ liệu
android.intent.action.CALL_BUTTON	Người dùng nhấn nút call để tới Dialer (trình gọi điện) hoặc giao diện UI thích hợp khác để tạo một cuộc gọi
android.intent.action.DATE_CHANGED	Date đã được thay đổi
android.intent.action.REBOOT	Reboot thiết bị

3. CÁC SYSTEM EVENT PHỔ BIẾN

Ngoài ra, còn một số System Event khác:

- *AIRPLANE_MODE*
- ACTION_TIME_TICK
- ACTION_TIME_CHANGED
- ACTION_TIMEZONE_CHANGED
- ACTION_PACKAGE_ADDED
- ACTION_PACKAGE_CHANGED
- ACTION_PACKAGE_REMOVED
- ACTION_PACKAGE_RESTARTED
- ACTION_PACKAGE_DATA_CLEARED
- ACTION_UID_REMOVED
- ACTION_POWER_CONNECTED
- ACTION_POWER_DISCONNECTED
- ACTION_SHUTDOWN

4. Ví dụ: Đăng ký 1 Broadcast Receiver bằng code

```
public class MainActivity extends Activity {
    BroadcastReceiver receiver=null;
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        IntentFilter filter=new IntentFilter
                    ("android.provider.Telephony.SMS_RECEIVED");
        receiver=new BroadcastReceiver() {
            public void onReceive(Context arg0, Intent arg1) {
                processReceive(arg0, arg1);
            }
        };
        registerReceiver(receiver, filter);
    }
    protected void onDestroy() {
        super.onDestroy();
        unregisterReceiver(receiver);
    }
    public void processReceive(Context context, Intent intent)
}
```

4. Ví dụ: Đăng ký 1 Broadcast Receiver bằng code

Lớp MyBroadCast

```
public class MyBroadCast extends
                    BroadcastReceiver {
    @Override
    public void onReceive(Context context,
            Intent intent) {
        //Process Intent here
    }
}
```

Đăng ký Broadcast Receiver bằng Manifest xml file

```
<application
    android:allowBackup="true"
    android:icon="@drawable/ic_launcher"
    android:label="@string/app_name"
    android:theme="@style/AppTheme" >
    <activity...
    <receiver android:name="MyBroadCast" >
        <intent-filter>
            <action android:name="android.provider.Telephony.SMS_RECEIVED" />
        </intent-filter>
    </receiver>
</application>
```

4. Ví dụ: Đọc tin nhắn bằng Broadcast Receiver

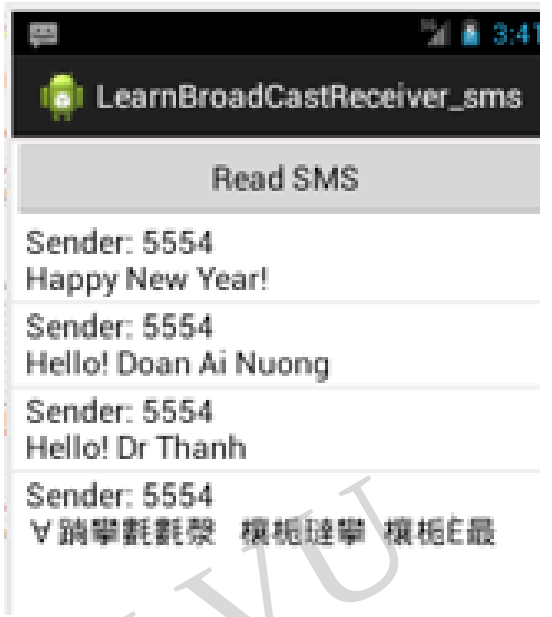
Đọc tất cả tin nhắn

Tự động hiển thị tin nhắn khi điện thoại nhận SMS

4. Ví dụ: Đọc tin nhắn bằng Broadcast Receiver

- Layout XML

```
<LinearLayout xmlns:android="http://schemas.and
    xmlns:tools="http://schemas.android.com/too
    android:id="@+id/LinearLayout1"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    tools:context=".MainActivity" >
    <Button
        android:id="@+id/btnreadsms"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Read SMS" />
    <ListView
        android:id="@+id/lvsms"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" >
    </ListView>
</LinearLayout>
```



4. Ví dụ: Đọc tin nhắn bằng Broadcast Receiver

- MySmsReceiver class

```
public class MySmsReceiver extends BroadcastReceiver {
    public static final String SMS_EXTRA="pdus";
    public static final String SMS_URI="content://sms/inbox";
    public static final String BODY = "body";
    public static final String ADDRESS = "address";
    public void onReceive(Context context, Intent intent) {
        Bundle extras = intent.getExtras();
        String messages = "";
        if ( extras != null )
        {
            Object[] smsExtra = (Object[]) extras.get( SMS_EXTRA );
            for ( int i = 0; i < smsExtra.length; ++i ){
                SmsMessage sms = SmsMessage.
                        createFromPdu((byte[])smsExtra[i]);
                String body = sms.getMessageBody().toString();
                String address = sms.getOriginatingAddress();
                messages += "SMS from " + address + " :\n";
                messages += body + "\n";
            }
            Toast.makeText( context, messages,
                    Toast.LENGTH_SHORT ).show();
        }
    }
}
```

4. Ví dụ: Đọc tin nhắn bằng Broadcast Receiver

- MainActivity class

```
    public void onClick(View v) {
        ContentResolver contentResolver = getContentResolver();
        Cursor cursor = contentResolver.query(
                    Uri.parse( MySmsReceiver.SMS_URI),
                        null, null, null, null);
        int indexBody = cursor.getColumnIndex( MySmsReceiver.BODY );
        int indexAddr = cursor.getColumnIndex( MySmsReceiver.ADDRESS);
        if ( indexBody < 0 || !cursor.moveToFirst() ) return;
        smsList.clear();
        do{
            String str = "Sender: " +
                         cursor.getString( indexAddr ) + "\n"+
                         cursor.getString( indexBody );
            smsList.add( str );
        }
        while( cursor.moveToNext() );
        adapter.notifyDataSetChanged();
    }
}
```

4. Ví dụ: Đọc tin nhắn bằng Broadcast Receiver

➢ Manifest XML

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="tranduythanh.com"
    android:versionCode="1"
    android:versionName="1.0" >
    <uses-sdk
        android:minSdkVersion="14"
        android:targetSdkVersion="17" />
    <uses-permission android:name="android.permission.RECEIVE_SMS" />
    <uses-permission android:name="android.permission.WRITE_SMS" />
    <uses-permission android:name="android.permission.READ_SMS" />
    <application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity..
        <receiver android:name="tranduythanh.com.MySmsReceiver" >
            <intent-filter>
                <action android:name="android.provider.Telephony.SMS_RECEIVED" />
            </intent-filter>
        </receiver>
    </application>
</manifest>
```

4. Ví dụ: Đọc tin nhắn bằng Broadcast Receiver

```
public class MySmsReceiver extends BroadcastReceiver {
public static final String SMS_EXTRA="pdus";
public static final String SMS_URI="content://sms/inbox";
public static final String BODY = "body";
public static final String ADDRESS = "address";
public void onReceive(Context context, Intent intent) {
    Bundle extras = intent.getExtras();
    String messages = "";
    if ( extras != null )
    {
        Object[] smsExtra = (Object[]) extras.get( SMS_EXTRA );
        for ( int i = 0; i < smsExtra.length; ++i ){
            SmsMessage sms = SmsMessage.
            createFromPdu((byte[])smsExtra[i]);
            String body = sms.getMessageBody().toString();
            String address = sms.getOriginatingAddress();
            messages += "SMS from " + address + " :\n";
            messages += body + "\n";
        }
    Toast.makeText( context, messages,
    Toast.LENGTH_SHORT ).show();
    }
    }
```