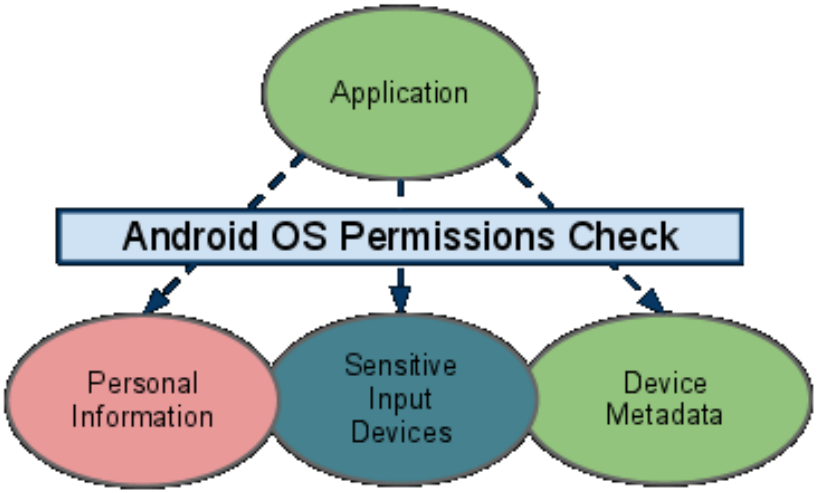
LẬP TRÌNH TRÊN THIẾT BỊ DI ĐỘNG
(MOBILE PROGRAMMING)

BÀI 6
MỘT SỐ XỬ LÝ MỨC HỆ THỐNG TRÊN ANDROID

Mục tiêu, yêu cầu

- Mục tiêu
  - Cung cấp các kỹ thuật lập trình xử lý ở mức hệ thống trên hệ điều hành Android.
  - Phương pháp khai thác các dịch vụ hệ thống Android.



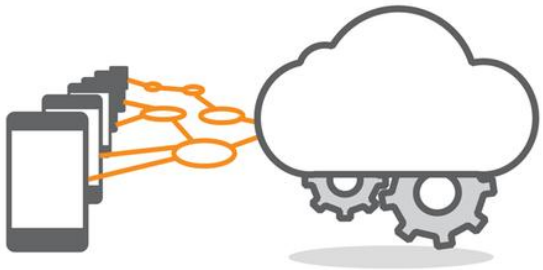
- Yêu cầu
  - Sinh viên nắm vững các cách thức xử lý và khai thác dịch vụ ở mức hệ thống trên Android.
  - Biết cách lập trình với Services, Notification, Media/Camera, Telephony, Manage Network/WiFi connections,…

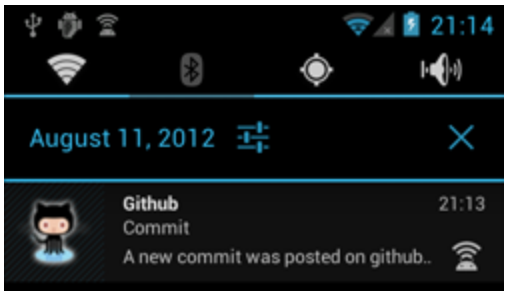
Tài liệu học tập & tham khảo

- Tài liệu học tập
  - Lập trình trên mobile, sách giáo trình của Khoa CNTT – Viện Đại học Mở Hà Nội, 2014.
- Tài liệu tham khảo
  - Reto Meier, Professional Android™Application Development, Wiley Publishing, Inc., 2009.
  - Beginning J2me: From Novice To Professional, Jonathan Knudsen, Sing Li, April 25 – 2005
  - Website: http://www.java2s.com/ & http://developer.android.com/
  - R. Rogers, J. Lombardo, Z. Mednieks, and B. Meike, Android Application Development, O"Reilly Media, Inc., 2009.
  - Z.Mednieks, L.Dornin, G.B.Meike, M.Nakamura, Programming Android: Java Programming for the New Generation of Mobile Devices, O'Reilly Media, 2011.

Nội dung chính của bài 6

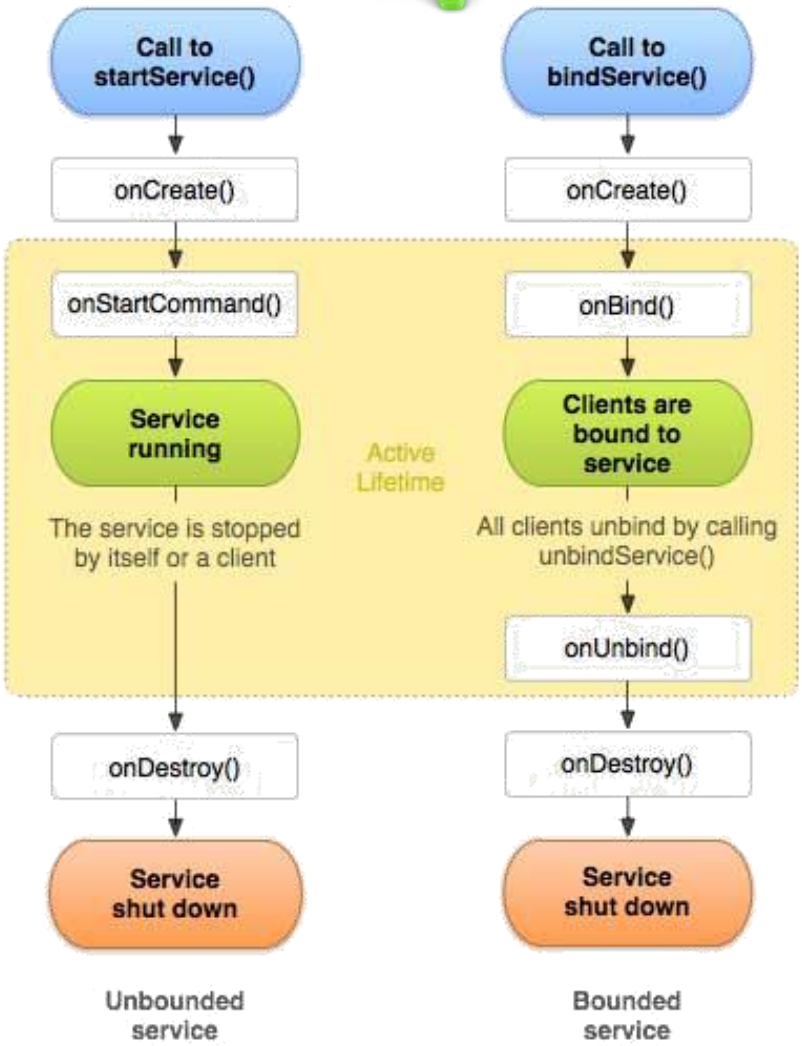
- Lập trình dịch vụ - Services
- Lập trình thông báo - Notifications
- Lập trình Telephony, Sms,…
- Lập trình với Camera và Media
- …





6.1. Lập trình dịch vụ - Services

- Giới thiệu Services
  - Chạy ngầm (không có giao diện).
  - Cập nhật Content Provider, kích hoạt Intent, đưa ra cảnh báo - Notifications.
  - Services được khởi động, kết thúc và điều khiển bởi các Services, Activities hoặc Broadcast Receivers của một ứng dụng.
  - Có 2 loại:
    - Unbound services: chạy độc lập với các thành phần ứng dụng.
    - Bound services: gắn với một thành phần ứng dụng (sẽ bị kết thúc nếu thành phần đó dừng).



6.1. Lập trình dịch vụ - Services

- Lập trình với Unbound-Services
  - B1) Tạo lớp kế thừa từ "Service" và ghi đè hàm "onBind" như sau:

```
public IBinder onBind(Intent  arg0) {
        //...
        return null;
}
```

Hàm này không chạy đối với lệnh startService

  - B2) Thường ghi đè thêm hàm "onStartCommand" để khởi động

```
public int onStartCommand(Intent arg0, int flags, int startId) {
        int kq = super.onStartCommand(arg0, flags, startId);
        //...
        return kq;
}
```

  - B3) Khởi chạy / kết thúc Services

```
♥ = startService(new  Intent( theContext, theService.class));
stopService(new  Intent(theContext, ♥.getClass()));
```

ComponentName

6.1. Lập trình dịch vụ - Services

- Lập trình với Unbound-Services
  - Ví dụ: Tạo service hiện thông báo vào Logcat

```
 8 public class MainActivity extends Activity {
 9     protected void onCreate(Bundle ts) {
10         super.onCreate(ts);
11         ComponentName sv = startService(new  Intent(this, myService.class));
12         stopService(new  Intent(this,sv.getClass()));
13     }
14 }
```

Khởi chạy Service

Kết thúc Service

```
 8 public class myService extends Service{
 9     public int onStartCommand(Intent a0, int flgs, int sId) {
10         int kq=super.onStartCommand(a0, flgs, sId);
11         Log.d("LHLK", "onStart service");
12         return kq;
13     }
14     public IBinder onBind(Intent arg0) {
15         Log.d("LHLK", "onBind service");
16         return null;
17     }
18 }
```

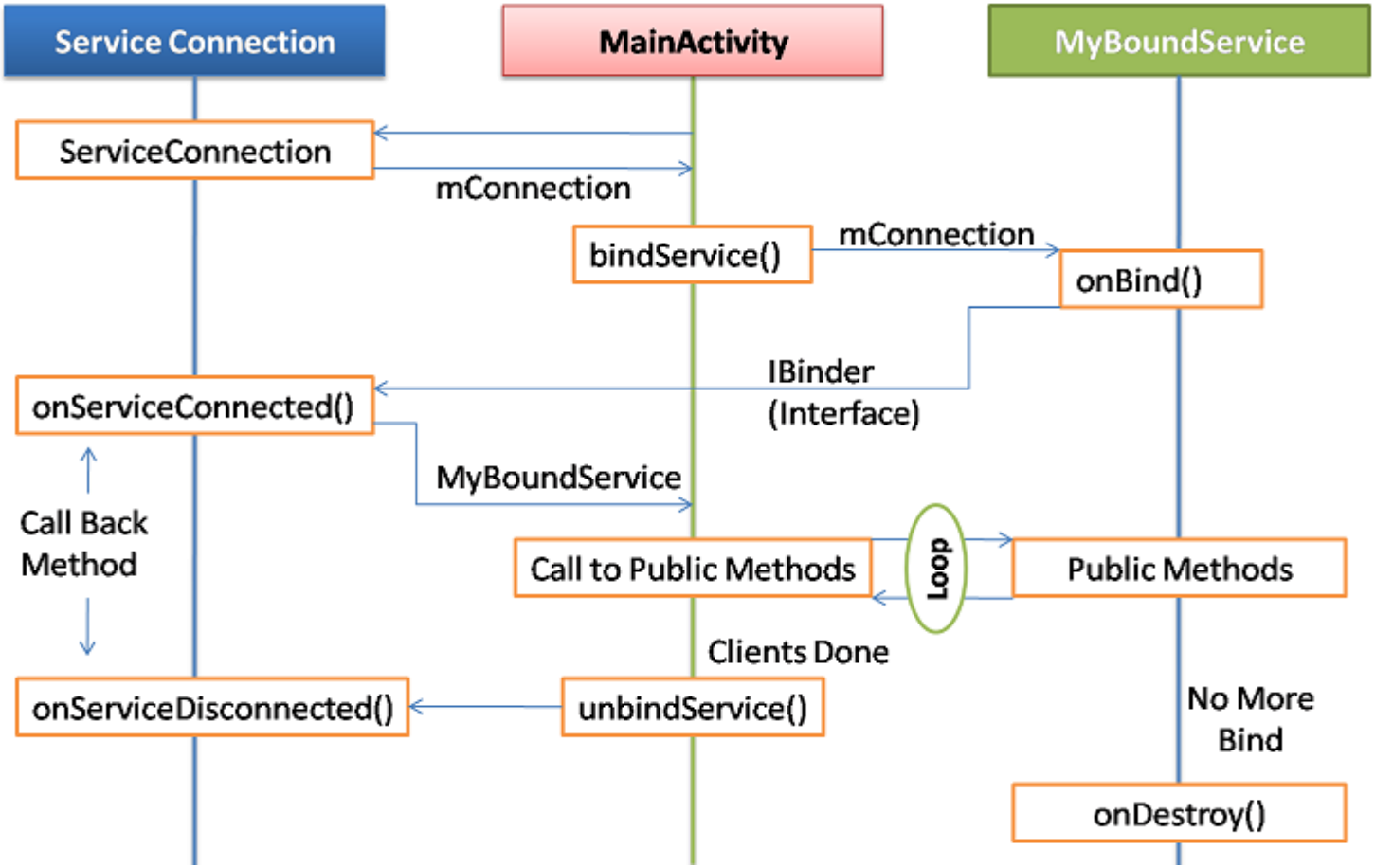
Hàm khởi chạy

Khai báo Service trong thẻ <application> của tệp AndroidManifest.xml

```
<service android:name="com.example.ltc_c6_hk2_1415.myService"/>
```

6.1. Lập trình dịch vụ - Services

- Lập trình với Bound-Services
  - ServiceConnection: kết nối giữa Activity và Bound-Service



6.1. Lập trình dịch vụ - Services

- Lập trình với Bound-Services
  - B1) Tạo lớp kế thừa từ "Service" và ghi đè hàm "onBind" như sau:

```
public IBinder onBind(Intent  arg0) {
        //...
        return null;
}
```

Hàm chạy khi có lệnh "bindService"

  - B2) Lập trình đối tượng "ServiceConnection" để tạo kết nối

```
♣ = new ServiceConnection() {
    public void onServiceDisconnected(ComponentName name) { ... }
    public void onServiceConnected(ComponentName n, IBinder s) {…}
};
```

Đơn giản, bỏ trống nội dung 2 hàm này

  - B3) Khởi chạy Services

```
bindService(new  Intent( theContext, theService.class), ♣,
                    Context.BIND_AUTO_CREATE);
```

6.1. Lập trình dịch vụ - Services

- Lập trình với Bound-Services
  - Ví dụ: Tạo service để hiện thông báo trên Log khi kết nối.

Lớp này trả về đối tượng Service đang thực thi

```
9  public class myService extends Service{
10     private final IBinder mBinder = new LocalBinder();
11     public IBinder onBind(Intent arg0) {
12         Log.d("LHLK", "onBind service");
13         return mBinder;
14     }
15     class LocalBinder extends Binder {
16         public myService getService() {
17             return myService.this;
18         }
19     }
20 }
```

```
12 public class MainActivity extends Activity {
13     protected void onCreate(Bundle ts) {
14         super.onCreate(ts);
15         ServiceConnection  sc=new ServiceConnection() {
16             public void onServiceDisconnected(ComponentName name) {}
17             public void onServiceConnected(ComponentName name, IBinder service) {
18                 Log.d("LHLK", "Service Connected");
19             }
20         };
21         bindService(new  Intent(this, myService.class), sc, Context.BIND_AUTO_CREATE);
22     }
23 }
```

Các hàm trong này không chạy nếu hàm "onBind" trong Service return null

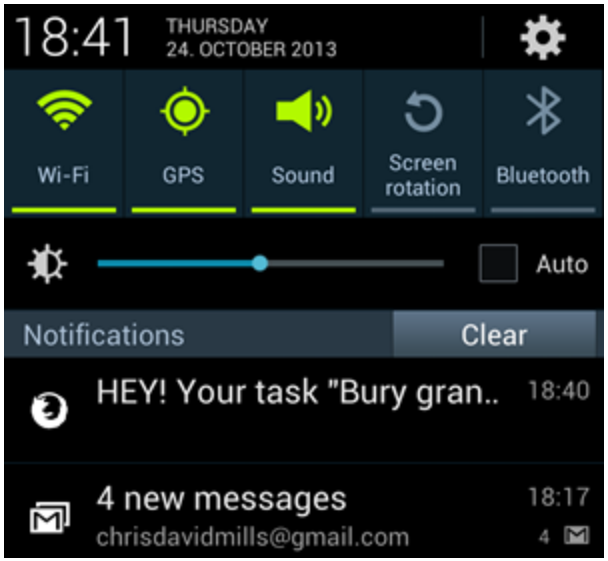
Tự động liên kết & chạy Service

AndroidManifest.XML

```
<service android:name="com.example.ltc_c6_hk2_1415.myService"/>
```

6.2. Lập trình thông báo - Notifications

- Giới thiệu Notifications
  - Notifications là dùng để chủ động thông báo/cảnh báo đến người dùng bằng các hình thức:
    - Tạo biểu tượng trên thanh trạng thái (phía trên màn hình)
    - Hiển thị các thông tin chi tiết trong cửa sổ mở rộng từ thanh trạng thái
    - Nháy đèn flash
    - Nhịp rung
    - Âm thanh
  - Notifications rất hữu ích khi ứng dụng đang chạy dạng nền như Service, Broadcast Receiver, Activity dạng ẩn.



6.2. Lập trình thông báo - Notifications

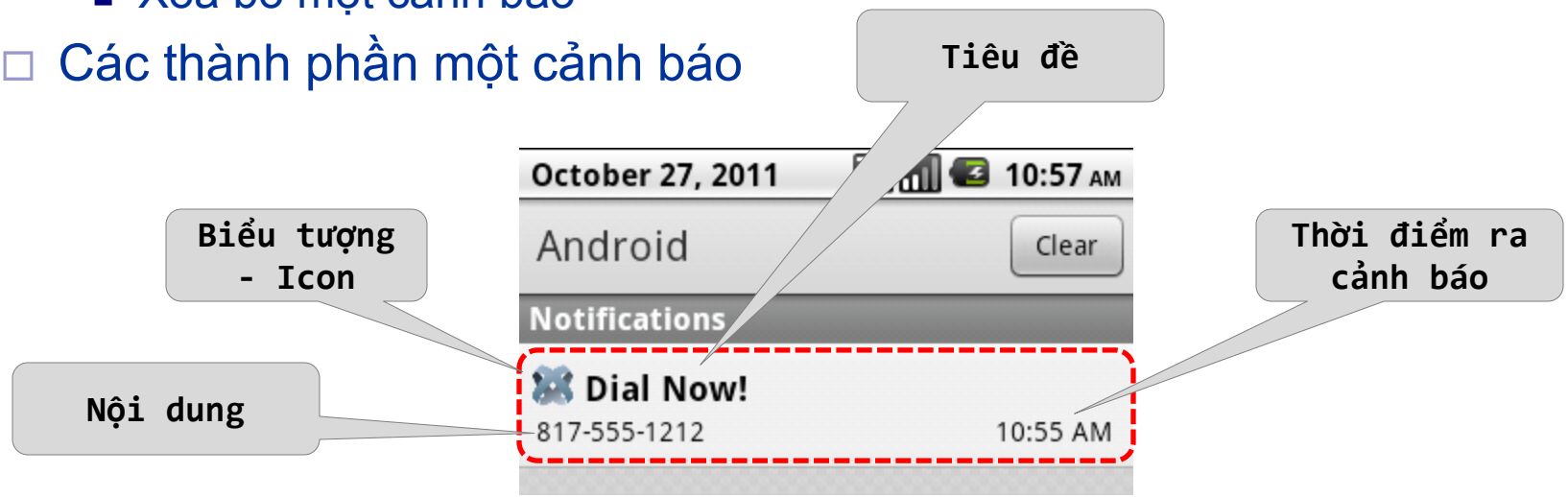
- Notifications trên thanh trạng thái
  - NotificationManager: lớp đối tượng quản lý cảnh báo

```
♥ = (NotificationManager)getSystemService(
                    Context.NOTIFICATION_SERVICE );
```

có thể thực hiện với ♥:

- Thêm một cảnh báo mới
- Sửa đổi cảnh báo đang có
- Xóa bỏ một cảnh báo

  - Các thành phần một cảnh báo



6.2. Lập trình thông báo - Notifications

- Tạo mới một Notification
  - B1) Tạo nội dung cảnh báo: sử dụng "Notification.Builder"

```
♣ = new Notification.Builder( mContext )
      .setContentTitle(" tiêu đề cảnh báo ")
      .setContentText(" nội dung cảnh báo ")
      .setSmallIcon( R.drawable.biểu_tượng )
      .setLargeIcon( ảnh_biểu_tượng_lớn );
```

API 16

  - B2) Tạo đối tượng cảnh báo: "Notification"

```
Notification  ♠ = ♣.build();
```

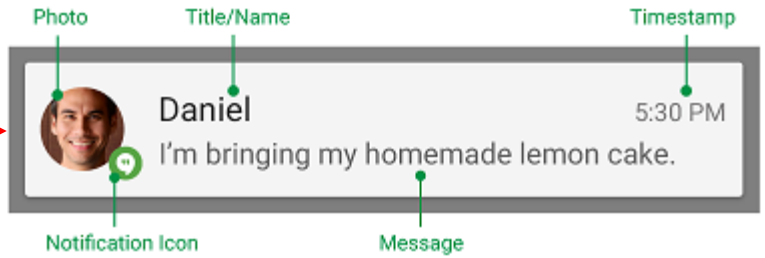
  - B3) Hiển thị cảnh báo lên màn hình

```
♥.notify( nID, ♠);
```

nID để định danh cho việc thay đổi cảnh báo

NotificationManager

Thiết lập thêm các tham số: ♣.set…(…);



6.2. Lập trình thông báo - Notifications

- Đặt kích hoạt Activity khi tương tác trên Notifications
  - Thiết lập cho đối tượng nội dung cảnh báo (Notification.Builder) các nội dung như sau:
    - B1) Tạo Intent khởi động Activity

```
♦ = new   Intent( theContext, theActivity.class );
```

    - B2) Đặt trạng thái mới & rỗng khi chạy Activity, hoặc các tùy chọn

```
♦.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK
                          | Intent.FLAG_ACTIVITY_CLEAR_TASK);
♦.set...(...);
```

    - B3) Tạo "PendingIntent" từ Intent khởi động Activity

```
∇ = PendingIntent.getActivity( theContext, 0, ♦,
                          PendingIntent.FLAG_UPDATE_CURENT );
```

    - B4) Đặt vào đối tượng nội dung cảnh báo (Notification.Builder)

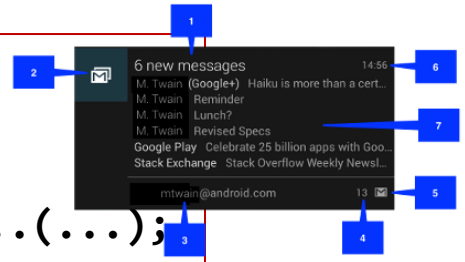
```
♣.setContentIntent( ∇ );
```

Notification.Builder

6.2. Lập trình thông báo - Notifications

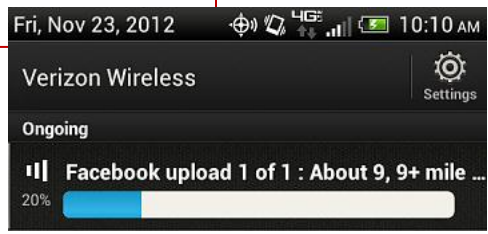
- Thay đổi hoặc xóa bỏ Notifications
  - Thay đổi nội dung cảnh báo: thiết lập lại các tham số trong Notification.Builder và thực hiện lệnh "*.notify(…)"

```
♣ = new Notification.Builder( mContext )
      .setContentTitle(" tiêu đề cảnh báo ")
      .setContentText(" nội dung cảnh báo ")
      .setSmallIcon( R.drawable.biểu_tượng ).set...(...);
♥.notify( nID, ♣.build() );
```



nID là định danh của cảnh báo cần sửa

```
.setStyle( Notification.InboxStyle  ? );
.setProgress( … );
…
```



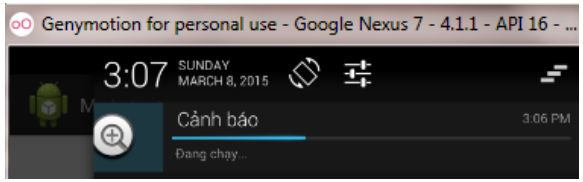
  - Xóa bỏ cảnh báo: dùng lệnh "cancel" trên NotificationManager

```
♥.cancel( nID );          //xóa bỏ cảnh báo có định danh nID
♥.cancelAll();            //xóa bỏ hết các cảnh báo trước đó
```

NotificationManager

6.2. Lập trình thông báo - Notifications

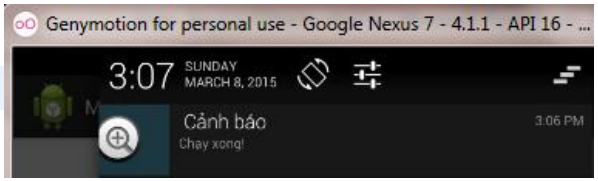
- Ví dụ: Tạo thông báo và hiển thị tiến trình chạy 30 giây, khi hết tiến trình cho phép chạy một Activity nếu tác động.
  - Trước hết, thiết lập các tham số nội dung cảnh báo (15,22,27-31)




```
14 public class MyActivity extends Activity {
15     Notification.Builder b;
16     EditText  e;
17     NotificationManager nm;
18     protected void onCreate(Bundle ts) {
19         super.onCreate(ts);
20         nm = (NotificationManager)getSystemService(
21                                     Context.NOTIFICATION_SERVICE);
22         b = new Notification.Builder( this );
23         e = new  EditText(this);
24         e.setHint("NOT COMPLETE");
25         setContentView(e);
26
27         b.setContentTitle(" Cảnh báo ");
28         b.setContentText(" Đang chạy... ");
29         b.setSmallIcon( android.R.drawable.btn_plus );
30         b.setLargeIcon( BitmapFactory.decodeResource(
31                 Resources.getSystem(), android.R.drawable.btn_star));
```

6.2. Lập trình thông báo - Notifications

- Tạo luồng (33,50) để thiết lập & chạy tiến trình (35-39), kết thúc tiến trình trong luồng sẽ đặt thêm Activity khi tác động vào cảnh báo (41-48).




```
33          new Thread(){
34              public void run(){
35                  for(int i=0;i<30;i++){
36                      b.setProgress(30, i, false);
37                      nm.notify(0,b.build());
38                      try{ Thread.sleep(1000); }catch(Exception e){}
39                  }
40
41                  Intent  it = new Intent(MyActivity.this, MyActivity.class);
42                  it.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK
43                                  | Intent.FLAG_ACTIVITY_CLEAR_TASK);
44                  PendingIntent pi=PendingIntent.getActivity(MyActivity.this,
45                                  0, it, PendingIntent.FLAG_UPDATE_CURRENT);
46                  b.setContentIntent(pi);
47                  b.setContentText("Chạy xong!").setProgress(0, 0, false);
48                  nm.notify(0,b.build());
49              }
50          }.start();
51      }
52  }
```

Hoặc tạo Uri từ tài nguyên âm thanh

```
b.setSound(RingtoneManager.getDefaultUri(RingtoneManager.TYPE_NOTIFICATION));
```

6.3. Lập trình Telephony, Sms

- Telephony
  - Thư viện Telephony API cho phép truy cập dịch vụ phần cứng điện thoại như tạo cuộc gọi, tiếp nhận và xử lý cuộc gọi đến,…
  - Tạo cuộc gọi, sử dụng "Intent" với ứng dụng gốc trên Android:
    - Intent.ACTION_CALL: cuộc gọi được thực hiện tức thời,
    - Intent.ACTION_DIAL: mở ứng dụng gọi điện mặc định và hiện số.

```
∇ = new Intent(Intent.ACTION_DIAL, Uri.parse("tel:1234567"));
startActivity( ∇ );
```

  - Đặt quyền trong tệp AndroidManifest.XML:

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

  - Một số "Intent" với ứng dụng gốc khác:
    - ACTION_ANSWER: trả lời cuộc gọi đến.
    - ACTION_EDIT / VIEW: mở một nội dụng để sửa hoặc hiển thị.
    - ACTION_WEB_SEARCH: tìm kiếm nội dung trên web,…

6.3. Lập trình Telephony, Sms

- Telephony
  - Giám sát trạng thái điện thoại:
    - B1) Đặt các quyền giám sát trong tệp AndroidManifest.XML:

```
android.permission.READ_PHONE_STATE          //đọc trạng thái
android.permission.ACCESS_COARSE_LOCATION    //đọc tọa độ
```

    - B2) Lập trình đối tượng nghe & xử lý sự kiện:

```
♦ = new PhoneStateListener() {
   public void onCallStateChanged(int sta, String incomNum) {}
        //xử lý khi trạng thái cuộc gọi thay đổi
   public void onCellLocationChanged(CellLocation location) {}
        //xử lý khi thay đổi vị trí
   public void onDataActivity(int direction) {}
        //xử lý khi có trao đổi dữ liệu
   public void onServiceStateChanged(ServiceState srvState) {}
        //xử lý khi thay đổi trạng thái dịch vụ
   public void onSignalStrengthChanged(int asu) {}
        //xử lý khi thay đổi tín hiệu
};
```

6.3. Lập trình Telephony, Sms

- Telephony
  - Giám sát trạng thái điện thoại:
    - B3) Lấy đối tượng quản lý:

```
TelephonyManager ♥ = (TelephonyManager)
                     getSystemService(Context.TELEPHONY_SERVICE);
```

    - B4) Đăng ký nghe sự kiện:

```
♥.listen( ♦,
        PhoneStateListener.LISTEN_CALL_STATE |
        PhoneStateListener.LISTEN_CELL_LOCATION |
        PhoneStateListener.LISTEN_DATA_ACTIVITY |
        PhoneStateListener.LISTEN_SERVICE_STATE |
        PhoneStateListener.LISTEN_SIGNAL_STRENGTH );
```

    - B5) Hủy đăng ký nghe & giám sát trạng thái trên điện thoại:

```
♥.listen( ♦, PhoneStateListener.LISTEN_NONE);
```

6.3. Lập trình Telephony, Sms

- Đọc trạng thái, các tham số của điện thoại
  - Sử dụng đối tượng "TelephonyManager":

```
TelephonyManager ♥ = (TelephonyManager)
                     getSystemService(Context.TELEPHONY_SERVICE);
```

gồm các lệnh sau:

```
String  ? = ♥.getLine1Number();           //lấy số SIM
String  ? = ♥.getNetworkCountryIso();     //lấy thông tin mạng
String  ? = ♥.getNetworkOperator();       //lấy hoạt động mạng
String  ? = ♥.getNetworkOperatorName();   //lấy tên hoạt động
int     ? = ♥.getNetworkType();           //lấy kiểu mạng
String  ? = ♥.getSimCountryIso()          //lấy mã SIM quốc gia
String  ? = ♥.getSimSerialNumber()        //lấy số định danh SIM
String  ? = ♥.getSimOperator()            //lấy mã nước+mã mạng
String  ? = ♥.getDeviceId()               //lấy số EMEI
int     ? = ♥.getCallState()              //trạng thái cuộc gọi
...
```

6.3. Lập trình Telephony, Sms

- Ví dụ: Lập trình Activity theo dõi trạng thái điện thoại và hiện cảnh báo số giây ngay khi kết thúc mỗi cuộc gọi.

```
15 public class MyService_Notification extends Activity{
16   TelephonyManager tm;
17   NotificationManager nm;
18   Notification.Builder nb;
19
20   public void onCreate(Bundle ts){
21     super.onCreate(ts);
22     TextView  tv=new TextView(this);
23     setContentView(tv);
24     tv.setText("Nghe trạng thái cuộc gọi!");
25
26     TelephonyManager  tm = (TelephonyManager)
27             getSystemService(Context.TELEPHONY_SERVICE);
28     nb = new Notification.Builder(this)
29             .setContentTitle("Số giây vừa gọi:")
30             .setContentText("")
31             .setSmallIcon(android.R.drawable.star_on);
32     nm = (NotificationManager)getSystemService(
33                     Context.NOTIFICATION_SERVICE);
34     tm.listen(tListener, PhoneStateListener.LISTEN_CALL_STATE);
35   }
```

6.3. Lập trình Telephony, Sms

- Tiếp nội dung của ví dụ…
  - Hàm lấy dữ liệu về lịch sử cuộc gọi

```
37  String[]  getCallDuration(){
38    Uri contacts = Calls.CONTENT_URI;
39    Cursor mCursor = getApplication().getContentResolver().query(
40                  contacts, null, null, null, Calls.DATE +" desc");
41    int num = mCursor.getColumnIndex(Calls.NUMBER);
42    int dur = mCursor.getColumnIndex(Calls.DURATION);
43    String[] kq=null;
44    if( mCursor.moveToFirst() == true ) {
45      String pn = mCursor.getString( num );
46      String cd = mCursor.getString( dur );
47      kq = new String[]{pn,cd};
48    }
49    mCursor.close();
50    return kq;
51  }
```

6.3. Lập trình Telephony, Sms

- Tiếp nội dung của ví dụ…
  - Đối tượng nghe & xử lý sự kiện trạng thái điện thoại

```
53  PhoneStateListener tListener=new PhoneStateListener(){
54    boolean dem=false;
55    public void onCallStateChanged(int sta, String incomNum) {
56      if(sta==TelephonyManager.CALL_STATE_OFFHOOK){
57        dem=true;
58        nm.cancel(0);
59      }else if(sta==TelephonyManager.CALL_STATE_IDLE && dem){
60        dem=false;
61        String[] kq = getCallDuration();
62          if(kq!=null && kq.length==2){
63            nb.setContentText("Last call ["+kq[0]+"]=["+kq[1]+"]ms");
64            nm.notify(0, nb.build());
65        }
66      }
67    }
68  };
69 }
```

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

6.3. Lập trình Telephony, Sms

- Sms
  - Thiết lập các quyền:

```
<uses-permission android:name="android.permission.SEND_SMS" />
<uses-permission android:name="android.permission.RECEIVE_SMS"/>
<uses-permission android:name="android.permission.WRITE_SMS"/>
<uses-permission android:name="android.permission.READ_SMS"/>
```

  - Lập trình gửi tin nhắn:

```
SmsManager   ∇ = SmsManager.getDefault();
∇.sendTextMessage( number2send, null, text2send, null, null);
```

Số điện thoại gửi

PendingIntent chạy khi tin đã được gửi

PendingIntent chạy khi gửi tin đi

6.3. Lập trình Telephony, Sms

- Sms
  - Lập trình nhận tin:
    - Lập trình kế thừa lớp "BroadcastReceiver"

```
class  IncomingSMSReceiver  extends  BroadcastReceiver{
        public void onReceive(Context  ct, Intent  it){
                //nội dung xử lý nhận SMS
        }
}
```

Hàm "onReceive" có tham số "Intent" (∇) để lấy các thông tin:

```
∇.getAction(): lấy hành động (Telephony.SMS_RECEIVED),
∇.getExtras(): lấy thông tin mở rộng với đối tượng "Bundle",
∇.get("pdus"): lấy mảng các đối tượng nội dung tin (dạng byte[]),
...
```

6.3. Lập trình Telephony, Sms

- Sms
  - Đăng ký nhận tin: 2 cách
    - Thiết lập trong AndroidManifest.XML:

Tên của lớp đối tượng BroadcastReceiver

```
<receiver android:name=" …?… ">
   <intent-filter>
      <action android:name =
                  "android.provider.Telephony.SMS_RECEIVED" />
   </intent-filter>
</receiver>
```

    - Đăng ký bằng lệnh:

```
♦ = new  IntentFilter(
                  "android.provider.Telephony.SMS_RECEIVED");
registerReceiver( ♠ , ♦ );
```

Đối tượng lớp BroadcastReceiver

6.3. Lập trình Telephony, Sms

- Sms
  - Đọc nội dung tin trong Inbox, Draf, Sent, Contacts
    - B1) Mở một con trỏ đọc tin

```
content://sms/sent
content://sms/draft
ContactsContract.Contacts.CONTENT_URI
```

```
Cursor ♥ = getContentResolver().
                        query(Uri.parse("content://sms/inbox"),
                        null, null, null, null );
```

    - B2) Thực hiện đọc nội dung tin trên từng dòng bản ghi

```
if (♥.moveToFirst()) {
   do {
       String msgData = "";
       for( int i=0; i<♥.getColumnCount(); i++ ) {
                msgData += " " + ♥.getColumnName(i)
                                + ":" + ♥.getString(i);
       }
   } while (♥.moveToNext());
} else {
   // empty box, no SMS
}
```

Phải nhận biết cột *i* ứng với thông tin gì?

6.3. Lập trình Telephony, Sms

- Ví dụ: Lập trình cho phép nhập số điện thoại, nội dung tin nhắn và gửi đi. Đọc và hiển thị toàn bộ các tin của Inbox.

```
13 public class MySMS_Send_ReadInbox extends Activity{
14   EditText  et1,et2;
15   Button    bt1,bt2;
16   TextView  tv3;
17   public void onCreate(Bundle t){
18     super.onCreate(t);
19     setContentView(R.layout.mysms_send_readinbox);
20     bt1=(Button)findViewById(R.id.sms_bt1);
21     bt2=(Button)findViewById(R.id.sms_bt2);
22     et1=(EditText)findViewById(R.id.sms_et1);
23     et2=(EditText)findViewById(R.id.sms_et2);
24     tv3=(TextView)findViewById(R.id.sms_tv3);
25     bt1.setOnClickListener(XLSK);
26     bt2.setOnClickListener(XLSK);
27   }
28
29   View.OnClickListener  XLSK=new  View.OnClickListener() {
30     public void onClick(View v) {
31         if(v==bt1){      send_sms();
32         }else if(v==bt2){    read_inbox();     }
33     }
34   };
```

6.3. Lập trình Telephony, Sms

```
36  void  send_sms(){
37      String  sms = et1.getText().toString();
38      String  num = et2.getText().toString();
39      SmsManager  sm=SmsManager.getDefault();
40      sm.sendTextMessage(num, null, sms, null, null);
41  }
42
43  void  read_inbox(){
44      String  kq="";
45      Cursor  cs=getContentResolver().query(
46              Uri.parse("content://sms/inbox"), null, null, null, null);
47      if(cs.moveToFirst()){
48          do{
49              kq = kq + "\n" + cs.getString(cs.getColumnIndexOrThrow("address"))
50                      + " *** " + cs.getString(cs.getColumnIndexOrThrow("body"));
51          }while(cs.moveToNext());
52      }else  kq="Inbox is empty!";
53      tv3.setText(kq);
54  }
55 }
```

6.4. Lập trình Camera / Media

- Thiết lập sử dụng Camera
  - Thiết lập quyền sử dụng Camera

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

  - Truy cập dịch vụ camera, sử dụng đối tượng "Camera"

```
Camera ♥ = android.hardware.Camera.open();
        // … Do somethings with the camera - ♥ …
♥.release();
```

  - Thiết lập các tham số cho Camera

```
Camera.Parameters ♣ = camera.getParameters();
        ♣.setPictureFormat(PixelFormat.JPEG);
        ♣.setFlashMode(FLASH_MODE_RED_EYE);
        ♣.setZoom( (int)zVal );
        ♣.setRotation( (int)rotAngle );
        …
♥.setParameters(♣);
```

6.4. Lập trình Camera / Media

- Lập trình chụp ảnh
  - Hiển thị ảnh cho Camera

```
B1) Tạo một đối tượng "SurfaceView" trên giao diện màn hình
B2) Xác định tham chiếu của SurfaceView trên: ∇ = findView…
B3) Đặt hiển thị ảnh lên màn hình:
        ♥.setPreviewDisplay( ∇.getHolder() );
B4) Bắt đầu hiển thị:
        ♥.startPreview();
        [ … làm một số việc khác … ]
B5) Kết thúc:
        ♥.stopPreview();
```

  - Nghe & xử lý sự kiện hiển thị ảnh: sử dụng "PreviewCallback"

```
♥.setPreviewCallback( new PreviewCallback() {
    public void onPreviewFrame(byte[] _data, Camera _camera){
        // TODO Do something with the preview image.
    }
});
```

6.4. Lập trình Camera / Media

- Lập trình chụp ảnh
  - Chụp hình ảnh từ Camera

```
B1) Lập trình đối tượng xử lý chụp (nếu cần):
    ◊ = new ShutterCallback() {
        public void onShutter() {  …  }
    };
    ♠ = new PictureCallback() {
      public void onPictureTaken(byte[] _dat, Camera _cam) {
          // TODO Do something with the image RAW data.
      }
    };
    ♦ = new PictureCallback() {
      public void onPictureTaken(byte[] _dat, Camera _cam) {
          // TODO Do something with the image JPEG data.
      }
    };
B2) Thực hiện chụp:
    ♥.takePicture( ◊, ♠, ♦);
```

Xử lý chụp

XL dữ liệu ảnh thô

XL dữ liệu ảnh chuẩn

Các tham số này có thể không dùng - null

6.4. Lập trình Camera / Media

- Lập trình chụp ảnh
  - Chụp hình ảnh từ Camera bằng ứng dụng sẵn có

Intent

Activity hiện tại

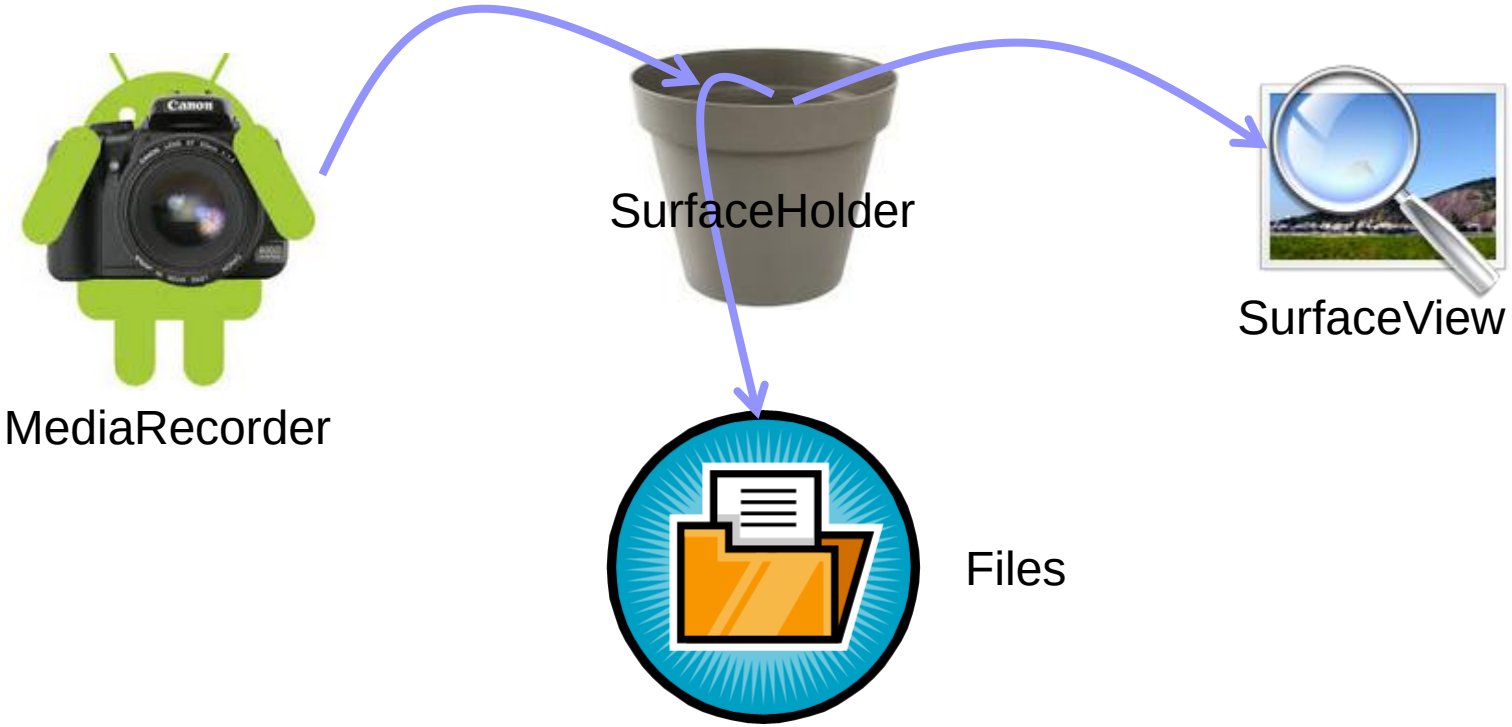
```
B1) Tạo một Intent để chạy ứng dụng chụp ảnh:
    ◊ = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);

B2) Kiểm tra sự sẵn có ứng dụng chụp ảnh:
    Nếu ◊.resolveActivity(getPackageManager()) != null
    thì tiếp tục,

B3) Thực hiện khởi chạy ứng dụng chụp ảnh:
   ?.startActivityForResult( ◊, 1);
```

  - Hệ thống Android phiên bản sau cung cấp thêm phương pháp chụp hình bằng thư viện camera2 (chi tiết xem thêm ở các tài liệu tham khảo):

```
android.hardware.camera2
```

6.4. Lập trình Camera / Media

- Lập trình quay video
  - MediaRecorder: lớp đối tượng để thực hiện ghi hình
  - SurfaceHolder: lớp đối tượng quản lý hình ảnh đang ghi
  - SurfaceView: lớp đối tượng hiển thị hình ảnh đang ghi



6.4. Lập trình Camera / Media

- Lập trình quay video

```
◊.setPreviewDisplay(∇.getSurface());
◊.prepare();
```

B1) Tạo giao diện chứa SurfaceView
và lấy đối tượng "SurfaceHolder" của nó:
∇ = ((SurfaceView)findViewById(?))).getHolder();

B2) Lập trình đối tượng xử lý quay video (♣) từ giao diện SurfaceHolder.Callback, với các hàm:
surfaceCreated(SurfaceHolder ?)…
surfaceChanged(SurfaceHolder h, int f, int w, int h)…
surfaceDestroyed(SurfaceHolder h)…

B3) Tạo đối tượng để quay video:
◊ = new MediaRecorder();

```
◊.stop;
◊.release();
```

B4) Khởi tạo các tham số quay video
◊.setAudioSource/setVideoSource/setProfile
/setOutputFile/setMaxDuration/setMaxFileSize/…

B5) Đăng ký xử lý quay video
∇.addCallback(♣);

B6) Bắt đầu / kết thúc quay
◊.start() / stop();

Sau mỗi lần quay video, chúng ta khởi tạo lại tham số

6.4. Lập trình Camera / Media

- Ví dụ: Lập trình tạo nút lệnh để xem và quay video

```
15 public class MyVideoRecord extends Activity
16                 implements SurfaceHolder.Callback{
17     Button  bt1;
18     MediaRecorder  mr;
19     SurfaceHolder  sh;
20     boolean  recording=false;
21     protected void onCreate(Bundle s) {
22         super.onCreate(s);
23         setContentView(R.layout.activity_my_video_record);
24         bt1=(Button)findViewById(R.id.vr_bt1);
25         bt1.setOnClickListener(XLSK);
26
27         mr=new MediaRecorder();
28         initRecorder();
29
30         SurfaceView cv = (SurfaceView) findViewById(R.id.surfaceView1);
31         sh = cv.getHolder();
32         sh.addCallback(this);
33         sh.setType(SurfaceHolder.SURFACE_TYPE_PUSH_BUFFERS);
34     }
```

Giao diện lập trình các hàm xử lý quay video

6.4. Lập trình Camera / Media

```
36      View.OnClickListener XLSK=new View.OnClickListener() {
37          public void onClick(View v) {
38              if (recording) {
39                  mr.stop();
40                  recording = false;
41
42                  initRecorder();
43                  prepareRecorder();
44              } else {
45                  recording = true;
46                  mr.start();
47              }
48          }
49      };
50      void  initRecorder(){
51          mr.setAudioSource(MediaRecorder.AudioSource.DEFAULT);
52          mr.setVideoSource(MediaRecorder.VideoSource.DEFAULT);
53          CamcorderProfile cp=CamcorderProfile.get(CamcorderProfile.QUALITY_HIGH);
54          mr.setProfile(cp);
55          mr.setOutputFile(
56              Environment.getExternalStorageDirectory().getPath()+"/myvideo.mp4");
57          mr.setMaxDuration(60000);
58          mr.setMaxFileSize(6000000);
59      }
```

Dừng quay video

Bắt đầu quay video

6.4. Lập trình Camera / Media

```
61      @Override
62      public void surfaceCreated(SurfaceHolder holder) {
63          prepareRecorder();
64      }
65      @Override
66      public void surfaceChanged(SurfaceHolder holder, int f, int w, int h) {}
67      @Override
68      public void surfaceDestroyed(SurfaceHolder holder) {
69          if (recording) {
70              mr.stop();
71              recording = false;
72          }
73          mr.release();
74      }
75      void prepareRecorder() {
76          mr.setPreviewDisplay(sh.getSurface());
77          try {  mr.prepare();  } catch (Exception e) {
78              e.printStackTrace();
79              finish();
80          }
81      }
82  }
```

Hàm chạy khi tạo đối tượng SurfaceView trên giao diện

Hàm chạy khi thay đổi định dạng, kích thước SurfaceView

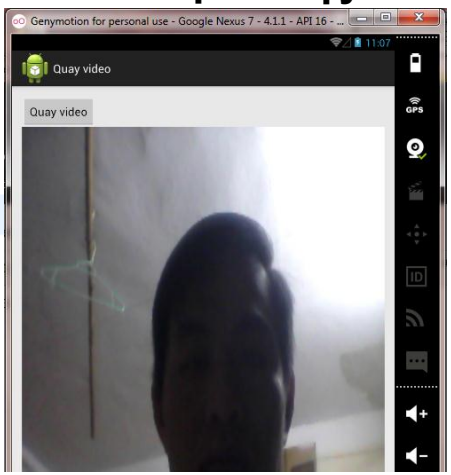
Hàm chạy khi đối tượng SurfaceView bị xóa bỏ

6.4. Lập trình Camera / Media

- Định nghĩa giao diện

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context="com.example.apk_vidu.MyTakePicture" >
    <Button
        android:id="@+id/vr_bt1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentLeft="true"
        android:layout_alignParentTop="true"
        android:text="Quay video" />
    <SurfaceView
        android:id="@+id/surfaceView1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/vr_bt1"
        android:layout_alignParentBottom="true"
        android:layout_alignParentRight="true"
        android:layout_below="@+id/vr_bt1" />
</RelativeLayout>
```

Kết quả chạy



Tóm tắt bài học

- Bài học này đã cung cấp các kiến thức và phương pháp lập trình xử lý về các vấn đề:
  - Lập trình dịch vụ - Services
  - Lập trình thông báo - Notifications
  - Lập trình Telephony, Sms
  - Lập trình với Camera.
- Qua mỗi phần đều có các minh họa thể hiện nội dung, phương pháp trình bày và ứng dụng vào một chương trình hoặc bài toán cụ thể.

Chúc các bạn học tốt và thành công!