**Activity**

Intent intent = **new** Intent();  
intent.setClass(MainActivity.**this**, Activity.**class**);  
startActivity(intent);

**Activity For Result**

Intent intent = **new** Intent();  
intent.setClass(MainActivity.**this**, Activity.**class**);  
startActivityForResult(intent, 0);

Activity.**this**.setResult(123);  
Activity.**this**.finish();

@Override  
**protected void** onActivityResult(**int** requestCode, **int** resultCode, @Nullable Intent data) {  
 **super**.onActivityResult(requestCode, resultCode, data);  
 **if** (requestCode == 0) {  
 **if** (resultCode == 123) {  
 Toast.*makeText*(**this**, **"OK "** + resultCode, Toast.***LENGTH\_LONG***).show();  
 } **else** {  
 Toast.*makeText*(**this**, **"No result code"**, Toast.***LENGTH\_LONG***).show();  
 }  
 }  
}

**Intent**

**public class** NguoiDung **implements** Serializable

NguoiDung nd = **new** DatabaseHelper(getApplicationContext()).find(i);  
Intent intent = **new** Intent(MainActivity.**this**, ChiTiet.**class**);  
intent.putExtra(**"MODE"**,**"sua"**);  
intent.putExtra(**"ND"**,nd);  
startActivity(intent);

Intent i = getIntent();  
String MODE = i.getStringExtra(**"MODE"**);

NguoiDung n = (NguoiDung) getIntent().getSerializableExtra(**"ND"**);

**Implicit Intent**

Intent intent = **new** Intent(Intent.***ACTION\_DIAL***);

Intent intent = **new** Intent(Intent.***ACTION\_DIAL***, Uri.*parse*(**"tel:0936461828"**));

Intent intent = **new** Intent(Intent.***ACTION\_VIEW***, Uri.*parse*(**"http://www.ptit.edu.vn"**));

Intent intent = **new** Intent(Intent.***ACTION\_VIEW***, Uri.*parse*(**"geo:0,0?q= Vo chi Cong, Tay Ho, Ha Noi"**));

startActivity(intent);

**External Storage**

<**uses-permission android:name="android.permission.WRITE\_EXTERNAL\_STORAGE"**/>  
<**uses-permission android:name="android.permission.READ\_EXTERNAL\_STORAGE"**/>

File myExternalFile;  
String filename = **"text.txt"**;  
String filepath = **"MyFileStorage"**;  
**if** (*isExternalStorageAvailable*()) {  
 *//write* myExternalFile = **new** File(getExternalFilesDir(filepath), filename);  
 **try** {  
 FileOutputStream fos;  
 fos = **new** FileOutputStream(myExternalFile);  
 String textToWrite = **"Write to External Storage N05"**;  
 **try** {  
 fos.write(textToWrite.getBytes());  
 fos.close();  
 } **catch** (IOException ioe) {  
 ioe.printStackTrace();  
 }  
 } **catch** (FileNotFoundException fnf) {  
 fnf.printStackTrace();  
 }  
 *//read* String myData = **""**;  
 **try** {  
 FileInputStream fis = **new** FileInputStream(myExternalFile);  
 DataInputStream in = **new** DataInputStream(fis);  
 BufferedReader br = **new** BufferedReader(  
 **new** InputStreamReader(in));  
 String strLine;  
 **while** ((strLine = br.readLine()) != **null**) {  
 myData = myData + strLine;  
 }  
 in.close();  
 Toast.*makeText*(MainActivity.**this**, **"data read:"** + myData, Toast.***LENGTH\_SHORT***).show();  
 } **catch** (IOException e) {  
 e.printStackTrace();  
 }  
  
}  
  
**if** (*isExternalStorageReadOnly*()) {  
 Toast.*makeText*(MainActivity.**this**, **"Read Only"**, Toast.***LENGTH\_SHORT***).show();  
}

**private static boolean** isExternalStorageReadOnly() {  
 String extStorageState = Environment.*getExternalStorageState*();  
 **if** (Environment.***MEDIA\_MOUNTED\_READ\_ONLY***.equals(extStorageState)) {  
 **return true**;  
 }  
 **return false**;  
}  
**private static boolean** isExternalStorageAvailable() {  
 String extStorageState = Environment.*getExternalStorageState*();  
 **if** (Environment.***MEDIA\_MOUNTED***.equals(extStorageState)) {  
 **return true**;  
 }  
 **return false**;  
}

**Internal Storage**

*// Write the string to the file* **final** String TESTSTRING = **new** String(**"Hello Android N04, abcd"**);  
FileOutputStream fOut = openFileOutput(**"samplefile.txt"**, ***MODE\_PRIVATE***);  
 OutputStreamWriter osw = **new** OutputStreamWriter(fOut);  
 osw.write(TESTSTRING);  
 osw.flush();  
 osw.close();  
 *//Read from file* FileInputStream fIn = openFileInput(**"samplefile.txt"**);  
 InputStreamReader isr = **new** InputStreamReader(fIn);  
 **char**[] inputBuffer = **new char**[TESTSTRING.length()];  
 isr.read(inputBuffer);  
 String readString = **new** String(inputBuffer);  
 Toast.*makeText*(MainActivity.**this**,readString,Toast.***LENGTH\_SHORT***).show();

**SharedPreference**

*// write*Context context= MainActivity.**this**;  
SharedPreferences sharedPreferences1=context.getSharedPreferences(**"N04"**,***MODE\_PRIVATE***);  
SharedPreferences.Editor editor1= sharedPreferences1.edit();  
editor1.putString(**"hello"**,**"xin chao cac ban nhom N04 abcd"**);  
editor1.putInt(**"Number"**,1000);  
editor1.putBoolean(**"co\_dieu\_hoa"**,**true**);  
Set<String> stringSet = **new** HashSet<>();  
stringSet.add(**"SetOne"**);  
stringSet.add(**"SetTwo"**);  
stringSet.add(**"SetThree"**);  
editor1.putStringSet(**"set"**,stringSet);  
editor1.commit();  
*//read***int** realAge= sharedPreferences1.getInt(**"Number"**,0);  
String s=sharedPreferences1.getString(**"hello"**,**""**);

**SQLite**

**public final class** DatabaseHelper **extends** SQLiteOpenHelper {  
  
 **public static final** String ***DATABASE\_NAME*** = **"student\_list"**;  
 **private static final** String ***TABLE\_NAME*** = **"student"**;  
 **private static final** String ***ID*** = **"id"**;  
 **private static final** String ***NAME*** = **"name"**;  
 **private static final** String ***EMAIL*** = **"email"**;  
 **private static final** String ***NUMBER*** = **"number"**;  
 **private** Context **context**;  
  
 **public** DatabaseHelper(Context context) {  
 **super**(context, ***DATABASE\_NAME***, **null**, 1);  
 **this**.**context** = context;  
 }  
  
 @Override  
 **public void** onCreate(SQLiteDatabase db) {  
 String sqlQuery = **"CREATE TABLE "** + ***TABLE\_NAME*** + **" ("** +  
 ***ID*** + **" integer primary key AUTOINCREMENT, "** +  
 ***NAME*** + **" TEXT, "** +  
 ***EMAIL*** + **" TEXT, "** +  
 ***NUMBER*** + **" TEXT)"**;  
 db.execSQL(sqlQuery);  
 }  
  
 @Override  
 **public void** onUpgrade(SQLiteDatabase db, **int** oldVersion, **int** newVersion) {  
 db.execSQL(**"DROP TABLE IF EXISTS "** + ***TABLE\_NAME***);  
 onCreate(db);  
 }  
  
 **public void** addStudent(Student student) {  
 SQLiteDatabase db = **this**.getWritableDatabase();  
 ContentValues values = **new** ContentValues();  
 values.put(***NAME***, student.getName());  
 values.put(***NUMBER***, student.getNumber());  
 values.put(***EMAIL***, student.getEmail());  
 *//Neu de null thi khi value bang null thi loi* db.insert(***TABLE\_NAME***, **null**, values);  
 db.close();  
 }  
  
 **public** Student getSdtudentById(**int** id) {  
 SQLiteDatabase db = **this**.getReadableDatabase();  
 Cursor cursor = db.query(***TABLE\_NAME***, **new** String[]{***ID***,  
 ***NAME***, ***EMAIL***, ***NUMBER***}, ***ID*** + **"=?"**,  
 **new** String[]{String.*valueOf*(id)}, **null**, **null**, **null**, **null**);  
 **if** (cursor != **null**)  
 cursor.moveToFirst();  
 Student student = **new** Student(cursor.getString(1), cursor.getString(2), cursor.getString(3));  
 cursor.close();  
 db.close();  
 **return** student;  
 }  
  
 **public int** Update(Student student) {  
 SQLiteDatabase db = **this**.getWritableDatabase();  
 ContentValues values = **new** ContentValues();  
 values.put(***NAME***, student.getName());  
 values.put(***NUMBER***, student.getNumber());  
 values.put(***EMAIL***, student.getEmail());  
 **return** db.update(***TABLE\_NAME***, values, ***ID*** + **"=?"**, **new** String[]{String.*valueOf*(student.getId())});  
 }  
  
 **public int** Delete(Student student) {  
 SQLiteDatabase db = **this**.getWritableDatabase();  
 ContentValues values = **new** ContentValues();  
 values.put(***NAME***, student.getName());  
 values.put(***NUMBER***, student.getNumber());  
 values.put(***EMAIL***, student.getEmail());  
 **return** db.delete(***TABLE\_NAME***, ***ID*** + **"=?"**, **new** String[]{String.*valueOf*(student.getId())});  
 }  
  
 **public** List<Student> getAllStudent() {  
 List<Student> listStudent = **new** ArrayList<Student>();  
 *// Select All Query* String selectQuery = **"SELECT \* FROM "** + ***TABLE\_NAME***;  
 SQLiteDatabase db = **this**.getWritableDatabase();  
 Cursor cursor = db.rawQuery(selectQuery, **null**);  
 **if** (cursor.moveToFirst()) {  
 **do** {  
 Student student = **new** Student();  
 student.setId(cursor.getInt(0));  
 student.setName(cursor.getString(1));  
 student.setEmail(cursor.getString(2));  
 student.setNumber(cursor.getString(3));  
 listStudent.add(student);  
 } **while** (cursor.moveToNext());  
 }  
 cursor.close();  
 db.close();  
 **return** listStudent;  
 }  
  
 **public int** getStudentsCount() {  
 String countQuery = **"SELECT \* FROM "** + ***TABLE\_NAME***;  
 SQLiteDatabase db = **this**.getReadableDatabase();  
 Cursor cursor = db.rawQuery(countQuery, **null**);  
 cursor.close();  
 *// return count* **return** cursor.getCount();  
 }  
}

**Fragment**

**public class** DetailFrag **extends** Fragment {  
 **public** DetailFrag() {  
 *// Required empty public constructor* }  
 @Override  
 **public** View onCreateView(LayoutInflater inflater, ViewGroup container,  
 Bundle savedInstanceState) {  
 *// Inflate the layout for this fragment* **return** inflater.inflate(R.layout.***fragment\_detail***, container, **false**);  
 }  
}

FragmentManager manager = **this**.getSupportFragmentManager();  
manager.beginTransaction()  
 .show(manager.findFragmentById(R.id.***detailFrag***))  
 .hide(manager.findFragmentById(R.id.***listFrag***))  
 .addToBackStack(**null**)  
 .commit();

**Simple ListView**

**final** String[] students = {**"Nam"**, **"Lan"**, **"Hoa"**, **"Hanh"**, **"Thanh"**};  
ArrayAdapter<String> adapter1 = **new** ArrayAdapter<String>(MainActivity.**this**, android.R.layout.***simple\_list\_item\_1***, students);  
**listView1**.setAdapter(adapter1);  
**listView1**.setOnItemClickListener

**Spinner**

**final** String goods[] = {**"TV"**,**"Labtop"**,**"SmartPhone"**, **"Watch"**,**"Book"**};  
ArrayAdapter<String> adapter1 = **new** ArrayAdapter<String>(**this**,android.R.layout.***simple\_list\_item\_1***,goods);  
adapter1.setDropDownViewResource(android.R.layout.***simple\_list\_item\_single\_choice***);  
**spinner1**.setAdapter(adapter1);  
**spinner1**.setOnItemSelectedListener(**new** AdapterView.OnItemSelectedListener() {  
 @Override  
 **public void** onItemSelected(AdapterView<?> parent, View view, **int** position, **long** id) {  
 }  
 @Override  
 **public void** onNothingSelected(AdapterView<?> parent) {  
 }  
});

**Custom Adapter**

**public class** RoomGridViewAdapter **extends** BaseAdapter {  
 Activity **activity**;  
 List<Room> **rooms**;  
  
 **public** RoomGridViewAdapter(Activity activity, List<Room> rooms) {  
 **this**.**activity** = activity;  
 **this**.**rooms** = rooms;  
 }  
  
 @Override  
 **public int** getCount() {  
 **return rooms**.size();  
 }  
  
 @Override  
 **public** Object getItem(**int** position) {  
 **return rooms**.get(position);  
 }  
  
 @Override  
 **public long** getItemId(**int** position) {  
 **return** position;  
 }  
  
 @Override  
 **public** View getView(**int** position, View convertView, ViewGroup parent) {  
 LayoutInflater inflater = **activity**.getLayoutInflater();  
 convertView = inflater.inflate(R.layout.***room\_in\_grid\_view***, **null**);  
 TextView roomName = convertView.findViewById(R.id.***room\_name***);  
 roomName.setText(**rooms**.get(position).getName());  
 **if** (**rooms**.get(position).getStt().equals(**"no"**)) {  
 roomName.setTextColor(Color.***RED***);  
 }  
 **return** convertView;  
 }  
}

**DateTimePicker**

*//Date*Calendar calendar = Calendar.*getInstance*();  
**int** year = calendar.get(Calendar.***YEAR***);  
**int** month = calendar.get(Calendar.***MONTH***);  
**int** day = calendar.get(Calendar.***DAY\_OF\_MONTH***);  
DatePickerDialog datePickerDialog = **new** DatePickerDialog(NoteDetailActivity.**this**, **new** DatePickerDialog.OnDateSetListener() {  
 @Override  
 **public void** onDateSet(DatePicker view, **int** year, **int** monthOfYear, **int** dayOfMonth) {  
 monthOfYear += 1;  
 String day = dayOfMonth < 10 ? (**"0"** + dayOfMonth) : dayOfMonth + **""**;  
 String month = monthOfYear < 10 ? (**"0"** + monthOfYear) : monthOfYear + **""**;  
 setDate.setText(day + **"/"** + month + **"/"** + year);  
 **timerDate** = day + **"/"** + month + **"/"** + year;  
 }  
}, year, month, day);  
datePickerDialog.show();

*//Time*Calendar calendar = Calendar.*getInstance*();  
**int** h = calendar.get(Calendar.***HOUR\_OF\_DAY***);  
**int** m = calendar.get(Calendar.***MINUTE***);  
TimePickerDialog timePickerDialog = **new** TimePickerDialog(NoteDetailActivity.**this**, **new** TimePickerDialog.OnTimeSetListener() {  
 @Override  
 **public void** onTimeSet(TimePicker view, **int** hourOfDay, **int** minute) {  
 String h = hourOfDay < 10 ? (**"0"** + hourOfDay) : hourOfDay + **""**;  
 String m = minute < 10 ? (**"0"** + minute) : minute + **""**;  
 setTime.setText(h + **":"** + m);  
 **timerTime** = h + **":"** + m;  
 }  
}, h, m, **true**);  
timePickerDialog.show();

**Auto Complete**

AutoCompleteTextView auto1 = findViewById(R.id.***autoCompleteTextView***);  
 MultiAutoCompleteTextView auto2 = findViewById(R.id.***multiAutoCompleteTextView***);  
  
 String[] s= **new** String[]{**"Lam"**,**"Dat"**,**"Tien"**};  
*// ArrayAdapter<String> adapter = new ArrayAdapter<>(this,android.R.layout.simple\_dropdown\_item\_1line,s);* ArrayAdapter<String> adapter = **new** ArrayAdapter<>(**this**,android.R.layout.***simple\_list\_item\_1***,s);  
 auto1.setAdapter(adapter);  
 auto1.setThreshold(1);  
 auto2.setAdapter(adapter);  
 auto2.setThreshold(1);  
 auto2.setTokenizer(**new** MultiAutoCompleteTextView.CommaTokenizer());

**Alarm Manager - Broadcast Receiver – Service – Notification – Pending Intent**

AlarmManager alarmManager = (AlarmManager) getSystemService(***ALARM\_SERVICE***);  
Intent i = **new** Intent(NoteDetailActivity.**this**, ReminderReceiver.**class**);  
i.putExtra(**"id"**, **note**.getId());  
PendingIntent pendingIntent = PendingIntent.*getBroadcast*(NoteDetailActivity.**this**, 0, i, PendingIntent.***FLAG\_UPDATE\_CURRENT***);  
Objects.*requireNonNull*(alarmManager).set(AlarmManager.***RTC\_WAKEUP***, cal.getTimeInMillis(), pendingIntent);

**public class** ReminderReceiver **extends** BroadcastReceiver {  
 @Override  
 **public void** onReceive(Context context, Intent intent) {Intent i = **new** Intent(context, ReminderService.**class**);**int** id = intent.getIntExtra(**"id"**, -1);  
 Log.*v*(**"receiver"**, id + **""**);  
 i.putExtra(**"id"**, id);  
 context.startService(i);  
 }  
}

**public class** ReminderService **extends** Service {  
 MediaPlayer **mediaPlayer**;  
  
 **public** ReminderService() {  
 }  
  
 @Override  
 **public** IBinder onBind(Intent intent) {  
 *//* ***TODO: Return the communication channel to the service.* return null**;  
 }  
  
 @Override  
 **public int** onStartCommand(Intent intent, **int** flags, **int** startId) {  
 **mediaPlayer** = MediaPlayer.*create*(**this**, R.raw.***nhac\_chuong\_nyancat***);  
 **mediaPlayer**.start();  
*// Toast.makeText(getApplicationContext(), "hihihihi", Toast.LENGTH\_LONG).show();* Log.*v*(**"ac"**, **"ab"**);  
 **int** id = intent.getIntExtra(**"id"**, -1);  
 Log.*v*(**"service"**, id + **""**);  
 DbHelper dbHelper = **new** DbHelper(**this**);  
 Note note = dbHelper.getNoteById(id);  
 NotificationCompat.Builder mBuilder =  
 **new** NotificationCompat.Builder(**this**)  
 .setSmallIcon(R.mipmap.***ic\_launcher***)  
 .setContentTitle(note.getTitle())  
 .setContentText(note.getContent());  
  
 NotificationManager notificationManager = (NotificationManager) getSystemService(Context.***NOTIFICATION\_SERVICE***);  
  
 **if** (Build.VERSION.***SDK\_INT*** >= Build.VERSION\_CODES.***O***) {  
 String channelId = startId + **""**;  
 NotificationChannel channel = **new** NotificationChannel(  
 channelId,  
 startId + **""**,  
 NotificationManager.***IMPORTANCE\_HIGH***);  
 notificationManager.createNotificationChannel(channel);  
 mBuilder.setChannelId(channelId);  
 }  
  
 Intent resultIntent = **new** Intent(**this**, NoteDetailActivity.**class**);  
 resultIntent.putExtra(**"id"**, note.getId());  
 TaskStackBuilder stackBuilder = TaskStackBuilder.*create*(**this**);  
 stackBuilder.addNextIntentWithParentStack(resultIntent);  
 PendingIntent resultPendingIntent =  
 stackBuilder.getPendingIntent(0, PendingIntent.***FLAG\_UPDATE\_CURRENT***);  
 mBuilder.setContentIntent(resultPendingIntent);  
 notificationManager.notify(startId, mBuilder.build());  
  
 **return *START\_NOT\_STICKY***;  
 }  
}

**ContentProvider**

**public class** StudentProvider **extends** ContentProvider {  
 **static final** String ***PROVIDER\_NAME*** = **"com.example.provider.College"**;  
 **static final** String ***URI*** = **"content://"** + ***PROVIDER\_NAME*** + **"/students"**;  
 **static final** Uri ***CONTENT\_URI*** = Uri.*parse*(***URI***);  
 **static final** String ***\_ID*** = **"\_id"**;  
 **static final** String ***NAME*** = **"name"**;  
 **static final** String ***GRADE*** = **"grade"**;\  
 **private static** HashMap<String, String> *STUDENTS\_PROJECTION\_MAP*;  
 **static final int *STUDENTS*** = 1;  
 **static final int *STUDENTS\_ID*** = 2;  
 **static final** UriMatcher ***uriMatcher***;  
 **static** {  
 ***uriMatcher*** = **new** UriMatcher(UriMatcher.***NO\_MATCH***);  
 ***uriMatcher***.addURI(***PROVIDER\_NAME***, **"students"**, ***STUDENTS***);  
 ***uriMatcher***.addURI(***PROVIDER\_NAME***, **"students/#"**, ***STUDENTS\_ID***);  
 }  
 **private** SQLiteDatabase **db**;  
 **static final** String ***DATABASE\_NAME*** = **"college"**;  
 **static final** String ***STUDENTS\_TABLE\_NAME*** = **"students"**;  
 **static final int *DATABASE\_VERSION*** = 1;  
 **static final** String ***CREATE\_DB\_TABLE*** =  
 **"CREATE TABLE "** + ***STUDENTS\_TABLE\_NAME*** +  
 **" (\_id INTEGER PRIMARY KEY AUTOINCREMENT, "** +  
 **"name TEXT NOT NULL, "** +  
 **"grade TEXT NOT NULL);"**;  
  
 **private static class** DatabaseHelper **extends** SQLiteOpenHelper {  
 DatabaseHelper(Context context) {  
 **super**(context, ***DATABASE\_NAME***, **null**, ***DATABASE\_VERSION***);  
 }  
 @Override  
 **public void** onCreate(SQLiteDatabase db) {  
 db.execSQL(***CREATE\_DB\_TABLE***);  
 }  
 @Override  
 **public void** onUpgrade(SQLiteDatabase db, **int** oldVersion, **int** newVersion) {  
 db.execSQL(**"DROP TABLE IF EXISTS "** + ***STUDENTS\_TABLE\_NAME***);  
 onCreate(db);  
 }  
  
 }  
  
 @Override  
 **public boolean** onCreate() {  
 Context context = getContext();  
 DatabaseHelper dbHelper = **new** DatabaseHelper(context);  
 **db** = dbHelper.getWritableDatabase();  
 **return** (**db** == **null**) ? **false** : **true**;  
 }  
 @Nullable  
 @Override  
 **public** Cursor query(Uri uri, String[] projection, String selection,  
 String[] selectionArgs, String sortOrder) {  
 SQLiteQueryBuilder qb = **new** SQLiteQueryBuilder();  
 qb.setTables(***STUDENTS\_TABLE\_NAME***);  
  
 **switch** (***uriMatcher***.match(uri)) {  
 **case *STUDENTS***:  
 qb.setProjectionMap(*STUDENTS\_PROJECTION\_MAP*);  
 **break**;  
 **case *STUDENTS\_ID***:  
 qb.appendWhere(***\_ID*** + **"="** + uri.getPathSegments().get(1));  
 **break**;  
 **default**:  
 **throw new** IllegalArgumentException(**"Unknown URI "** + uri);  
 }  
 **if** (sortOrder == **null** || sortOrder == **""**) {  
 sortOrder = ***NAME***;  
 }  
 Cursor c = qb.query(**db**, projection, selection, selectionArgs, **null**, **null**, sortOrder);  
 c.setNotificationUri(getContext().getContentResolver(), uri);  
 **return** c;  
 }  
 @Nullable  
 @Override  
 **public** String getType(@NonNull Uri uri) {  
 **switch** (***uriMatcher***.match(uri)) {  
 **case *STUDENTS***:  
 **return "vnd.android.cursor.dir/vnd.students"**;  
 **case *STUDENTS\_ID***:  
 **return "vnd.android.cusor.item/vnd.students"**;  
 **default**:  
 **throw new** IllegalArgumentException(**"Unsupport URIL "** + uri);  
 }  
 }  
 @Nullable  
 @Override  
 **public** Uri insert(@NonNull Uri uri, @Nullable ContentValues values) {  
 **long** rowID = **db**.insert(***STUDENTS\_TABLE\_NAME***, **""**, values);  
 **if** (rowID > 0) {  
 Uri \_uri = ContentUris.*withAppendedId*(***CONTENT\_URI***, rowID);  
 getContext().getContentResolver().notifyChange(\_uri, **null**);  
 **return** \_uri;  
 }  
 **throw new** SQLException(**"Failed to add a record into "** + uri);  
 }  
 @Override  
 **public int** delete(@NonNull Uri uri, @Nullable String selection, @Nullable String[] selectionArgs) {  
 **int** count = 0;  
  
 **switch** (***uriMatcher***.match(uri)) {  
 **case *STUDENTS***:  
 count = **db**.delete(***STUDENTS\_TABLE\_NAME***, selection, selectionArgs);  
 **break**;  
 **case *STUDENTS\_ID***:  
 count = **db**.delete(***STUDENTS\_TABLE\_NAME***, ***\_ID*** + **"="** + uri.getPathSegments().get(1) +  
 (!TextUtils.*isEmpty*(selection) ? **" AND ("** +  
 selection + **')'** : **""**), selectionArgs);  
 **break**;  
 **default**:  
 **throw new** IllegalArgumentException(**"Unknown URI "** + uri);  
 }  
 getContext().getContentResolver().notifyChange(uri, **null**);  
 **return** count;  
 }  
 @Override  
 **public int** update(Uri uri, ContentValues values, String selection,  
 String[] selectionArgs) {  
 **int** count = 0;  
  
 **switch** (***uriMatcher***.match(uri)) {  
 **case *STUDENTS***:  
 count = **db**.update(***STUDENTS\_TABLE\_NAME***, values, selection, selectionArgs);  
 **break**;  
 **case *STUDENTS\_ID***:  
 count = **db**.update(***STUDENTS\_TABLE\_NAME***, values, ***\_ID*** + **"="** + uri.getPathSegments().get(1) +  
 (!TextUtils.*isEmpty*(selection) ? **" AND ("** +  
 selection + **')'** : **""**), selectionArgs);  
 **break**;  
 **default**:  
 **throw new** IllegalArgumentException(**"Unknown URI "** + uri);  
 }  
 getContext().getContentResolver().notifyChange(uri, **null**);  
 **return** count;  
 }  
}

**public class** MainActivity **extends** AppCompatActivity {  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_main***);  
 }  
  
 **public void** onClickAddName(View view) {  
 ContentValues values = **new** ContentValues();  
  
 values.put(StudentProvider.***NAME***, ((EditText) findViewById(R.id.***txtName***)).getText().toString());  
 values.put(StudentProvider.***GRADE***, ((EditText) findViewById(R.id.***txtGrade***)).getText().toString());  
  
 Uri uri = getContentResolver().insert(StudentProvider.***CONTENT\_URI***, values);  
  
 Toast.*makeText*(getBaseContext(), uri.toString(), Toast.***LENGTH\_LONG***).show();  
 }  
  
 **public void** onClickRetrieveStudents(View view) {  
 String URI = **"content://com.example.provider.College/students"**;  
 Uri students = Uri.*parse*(URI);  
 Cursor c = managedQuery(students, **null**, **null**, **null**, **"name"**);  
 **if** (c.moveToFirst()) {  
 **do** {  
 Toast.*makeText*(**this**, c.getString(c.getColumnIndex(StudentProvider.***\_ID***)) +  
 **", "** + c.getString(c.getColumnIndex(StudentProvider.***NAME***)) +  
 **", "** + c.getString(c.getColumnIndex(StudentProvider.***GRADE***)), Toast.***LENGTH\_LONG***).show();  
 } **while** (c.moveToNext());  
 }  
 }  
}

**Menu**

*<?***xml version="1.0" encoding="utf-8"***?>*<**menu xmlns:android="http://schemas.android.com/apk/res/android"**>  
<**item android:title="Setting" android:id="@+id/menuSetting"**></**item**>  
 <**item android:title="Share" android:id="@+id/menuShare"**>  
 <**menu**>  
 <**item android:title="Share Pic" android:id="@+id/sharePicMenu"**></**item**>  
 <**item android:title="Share Info" android:id="@+id/ShareInfoMenu"**></**item**>  
 </**menu**>  
  
 </**item**>  
 <**item android:title="Search" android:id="@+id/menuSearch"**></**item**>  
 <**item android:title="Exit" android:id="@+id/menuExit"**></**item**>  
</**menu**>

@Override  
**public boolean** onCreateOptionsMenu(Menu menu) {  
 getMenuInflater().inflate(R.menu.***menu\_demo***, menu);  
 **return super**.onCreateOptionsMenu(menu);  
}  
  
@Override  
**public boolean** onOptionsItemSelected(MenuItem item) {  
 **switch** (item.getItemId()) {  
 **case** R.id.***menuSetting***:  
 Toast.*makeText*(MainActivity.**this**, **"Setting selected"**, Toast.***LENGTH\_SHORT***).show();  
 **break**;  
 }  
 **return super**.onOptionsItemSelected(item);  
}

**Skill**

*//set image*catAvt.setImageResource(**imageViews**.get(position));

**byte**[] byteArray = getIntent().getByteArrayExtra(**"image"**);  
Bitmap bitmap = BitmapFactory.*decodeByteArray*(byteArray, 0, byteArray.**length**);  
**fullImage**.setImageBitmap(bitmap);

*//check box***cb1**.setOnClickListener

*//Toggle Button***toggleButton1**.setOnCheckedChangeListener

<**RadioGroup  
 android:id="@+id/group1"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"**>  
 <**RadioButton**

*//Radio Group***int** idCheck = **group1**.getCheckedRadioButtonId();  
**if** (idCheck==R.id.***rb1***)  
**group1**.setOnCheckedChangeListener

*//get current time***public static** String getCurrentTime() {  
 Date date = Calendar.*getInstance*().getTime();  
 DateFormat dateFormat = **new** SimpleDateFormat(**"hh:mm:ss dd/MM/yyyy"**);  
 **return** dateFormat.format(date);  
}

*//string to date***public static** Date StringTimeToDate(String time) {  
 SimpleDateFormat sdf = **new** SimpleDateFormat(**"HH:mm dd/MM/yyyy"**, Locale.***ENGLISH***);  
 **try** {  
 **return** sdf.parse(time);  
 } **catch** (ParseException e) {  
 e.printStackTrace();  
 **return null**;  
 }  
}

*//get id from name***public static int** getIdResource(Context context, String name, String defType, String packageName) {  
 **return** context.getResources().getIdentifier(name, defType, packageName);  
}

BaseUtil.*getIdResource*(activity, **"album\_"** + catShortName, **"drawable"**, activity.getPackageName());

*//fast custom dialog***final** AlertDialog.Builder builder = **new** AlertDialog.Builder(activity);  
LayoutInflater layoutInflater = activity.getLayoutInflater();  
View view1 = layoutInflater.inflate(R.layout.***gridview\_cat\_info***, **null**);  
**final** AlertDialog alertDialog = builder.create();  
alertDialog.setView(view1);  
alertDialog.show();  
alertDialog.getWindow().setLayout(800, 660);

*//Photo***final** CharSequence[] options = {**"Take Photo"**, **"Choose from Gallery"**, **"Cancel"**};  
AlertDialog.Builder builder = **new** AlertDialog.Builder(NoteDetailActivity.**this**);  
builder.setTitle(**"Choose picture to attach"**);   
builder.setItems(options, **new** DialogInterface.OnClickListener() {  
  
 @Override  
 **public void** onClick(DialogInterface dialog, **int** item) {  
 **if** (options[item].equals(**"Take Photo"**)) {  
 Intent takePicture = **new** Intent(android.provider.MediaStore.***ACTION\_IMAGE\_CAPTURE***);  
 startActivityForResult(takePicture, ***CAPTURE\_IMAGE\_REQUEST\_CODE***);  
 } **else if** (options[item].equals(**"Choose from Gallery"**)) {  
 Intent pickPhoto = **new** Intent(Intent.***ACTION\_PICK***, android.provider.MediaStore.Images.Media.***EXTERNAL\_CONTENT\_URI***);  
 startActivityForResult(pickPhoto, ***GALLERY\_IMAGE\_REQUEST\_CODE***);  
 } **else if** (options[item].equals(**"Cancel"**)) {  
 dialog.dismiss();  
 }  
 }  
});  
builder.show();

**switch** (requestCode) {  
 **case *CAPTURE\_IMAGE\_REQUEST\_CODE***:  
 **if** (resultCode == ***RESULT\_OK*** && data != **null**) {  
 **bitmap** = (Bitmap) data.getExtras().get(**"data"**);  
 Log.*v*(**"rc0"**, **bitmap**.toString());  
 }  
  
 **frameLayout**.setVisibility(View.***VISIBLE***);  
 **frameLayout**.setEnabled(**true**);  
 **attachImage**.setImageBitmap(**bitmap**);  
  
 **break**;  
 **case *GALLERY\_IMAGE\_REQUEST\_CODE***:  
 **if** (resultCode == ***RESULT\_OK*** && data != **null**) {  
 Uri selectedImage = data.getData();  
 String[] filePathColumn = {MediaStore.Images.Media.***DATA***};  
 **if** (selectedImage != **null**) {  
 Cursor cursor = getContentResolver().query(selectedImage,  
 filePathColumn, **null**, **null**, **null**);  
 **if** (cursor != **null**) {  
 cursor.moveToFirst();  
  
 **int** columnIndex = cursor.getColumnIndex(filePathColumn[0]);  
 String picturePath = cursor.getString(columnIndex);  
 Log.*v*(**"rc0"**, picturePath);  
  
  
 checkPermission(Manifest.permission.***READ\_EXTERNAL\_STORAGE***, 101);  
 BitmapFactory.Options options = **new** BitmapFactory.Options();  
 options.**inSampleSize** = 2;**bitmap** = BitmapFactory.*decodeFile*(picturePath, options);  
 **bitmap** = Bitmap.*createScaledBitmap*(**bitmap**, 600, 600, **false**);  
 cursor.close();  
  
 **frameLayout**.setVisibility(View.***VISIBLE***);  
 **frameLayout**.setEnabled(**true**);  
 **attachImage**.setImageBitmap(**bitmap**);  
 }  
 }  
  
 }  
 **break**;  
 }

*//Base64*

ByteArrayOutputStream stream = **new** ByteArrayOutputStream();String temp = Base64.*encodeToString*(byteArray, Base64.***DEFAULT***);

**byte**[] byteArray = stream.toByteArray();

**byte**[] byteArray = Base64.*decode*(**attachment**.getLink(), Base64.***DEFAULT***);

*//permission*

<**uses-permission android:name="android.permission.READ\_CONTACTS"**></**uses-permission**>

**public class** MainActivity **extends** AppCompatActivity {  
  
 **public static final int *REQUEST\_READ\_CONTACTS*** = 79;  
 ListView **list**;  
 ArrayList **mobileArray**;  
 *//@TargetApi(Build.VERSION\_CODES.LOLLIPOP)* @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_main***);  
 **if** (ActivityCompat.*checkSelfPermission*(**this**, android.Manifest.permission.***READ\_CONTACTS***)  
 == PackageManager.***PERMISSION\_GRANTED***) {  
 **mobileArray** = getAllContacts();  
 } **else** {  
 requestPermission();  
 }  
 **list** = findViewById(R.id.***list***);  
 ArrayAdapter adapter = **new** ArrayAdapter<String>(**this**,  
 android.R.layout.***simple\_list\_item\_1***, android.R.id.***text1***, **mobileArray**);  
 **list**.setAdapter(adapter);  
 }  
  
 **private void** requestPermission() {  
 **if** (ActivityCompat.*shouldShowRequestPermissionRationale*(**this**, android.Manifest.permission.***READ\_CONTACTS***)) {  
 *// show UI part if you want here to show some rationale !!!* } **else** {  
 ActivityCompat.*requestPermissions*(**this**, **new** String[]{android.Manifest.permission.***READ\_CONTACTS***},  
 ***REQUEST\_READ\_CONTACTS***);  
 }  
 **if** (ActivityCompat.*shouldShowRequestPermissionRationale*(**this**, android.Manifest.permission.***READ\_CONTACTS***)) {  
 } **else** {  
 ActivityCompat.*requestPermissions*(**this**, **new** String[]{android.Manifest.permission.***READ\_CONTACTS***},  
 ***REQUEST\_READ\_CONTACTS***);  
 }  
 }  
  
 @Override  
 **public void** onRequestPermissionsResult(**int** requestCode,  
 String permissions[], **int**[] grantResults) {  
 **switch** (requestCode) {  
 **case *REQUEST\_READ\_CONTACTS***: {  
 **if** (grantResults.**length** > 0 && grantResults[0] == PackageManager.***PERMISSION\_GRANTED***) {  
 **mobileArray** = getAllContacts();  
 } **else** {  
 *// permission denied,Disable the  
 // functionality that depends on this permission.* }  
 **return**;  
 }  
 }  
 }  
  
 **private** ArrayList getAllContacts() {  
 ArrayList<String> nameList = **new** ArrayList<>();  
 ContentResolver cr = getContentResolver();  
 Cursor cur = cr.query(ContactsContract.Contacts.***CONTENT\_URI***,  
 **null**, **null**, **null**, **null**);  
 **if** ((cur != **null** ? cur.getCount() : 0) > 0) {  
 **while** (cur != **null** && cur.moveToNext()) {  
 String id = cur.getString(  
 cur.getColumnIndex(ContactsContract.Contacts.***\_ID***));  
 String name = cur.getString(cur.getColumnIndex(  
 ContactsContract.Contacts.***DISPLAY\_NAME***));  
 nameList.add(name);  
 **if** (cur.getInt(cur.getColumnIndex( ContactsContract.Contacts.***HAS\_PHONE\_NUMBER***)) > 0) {  
 Cursor pCur = cr.query(  
 ContactsContract.CommonDataKinds.Phone.***CONTENT\_URI***,  
 **null**,  
 ContactsContract.CommonDataKinds.Phone.***CONTACT\_ID*** + **" = ?"**,  
 **new** String[]{id}, **null**);  
 **while** (pCur.moveToNext()) {  
 String phoneNo = pCur.getString(pCur.getColumnIndex(  
 ContactsContract.CommonDataKinds.Phone.***NUMBER***));  
 }  
 pCur.close();  
 }  
 }  
 }  
 **if** (cur != **null**) {  
 cur.close();  
 }  
 **return** nameList;  
 }  
}