

MỤC LỤC

MỤC LỤC	1
GIỚI THIỆU MÔN HỌC	5
1. Nội dung môn học:	5
2. Đề cương môn học:	5
3. Tài liệu học tập:	5
3.1. Giáo trình chính	5
3.2. Tài liệu tham khảo	5
4. Phương pháp đánh giá học phần:.....	6
5. Danh sách đề tài gợi ý:	6
LAB 1. ỨNG DỤNG ANDROID ĐẦU TIÊN	7
1. MỤC TIÊU:.....	7
2. THỰC HIỆN:	7
2.1. Bài 1: Ứng dụng Android đầu tiên.	7
2.2. Bài 2: Chương trình giả lập máy ảo Genymotion	11
2.3. Bài 3: Cài đặt Vysor – phần mềm Ánh xạ điện thoại vào máy tính	13
3. Bài tập:.....	15
LAB 2. CÁC LAYOUT CƠ BẢN.....	16
1. MỤC TIÊU:.....	16
2. THỰC HIỆN:	16
2.1. Bài 1: Constraint Layout	16
2.2. Bài 2: Frame Layout.....	18
2.3. Bài 3: Linear Layout.....	19
2.4. Bài 4: Table Layout	22
2.5. Bài 5: Relative layout	24
3. Bài tập:.....	28
LAB 3. FIND VIEW BY ID, ON CLICK XML, ANOMOUS LISTENER	Error!
Bookmark not defined.	
1. MỤC TIÊU:.....	32
2. THỰC HIỆN:	32
2.1. Bài 1: Viết chương trình gọi điện thoại đơn giản. Giao diện như hình bên dưới.	
.....	32

2.2. Bài 2: Viết chương trình khi bấm vào nút Ok sẽ có dòng Text xuất hiện nội dung “Bạn vừa bấm vào nút OK”.....	32
2.3. Bài 3:.....	Error! Bookmark not defined.
3. BÀI TẬP:.....	38
LAB 4. LAB 4: TEXT VIEW, EDIT TEXT, BUTTON.....	43
1. MỤC ĐÍCH	43
2. LÝ THUYẾT:.....	43
2.1. TextView	43
2.2. EditText	45
2.3. Button	46
3. THỰC HÀNH.....	Error! Bookmark not defined.
3.1. Bài 1: Thiết kế giao diện như hình	48
3.2. Bài 2: Xây dựng chương trình máy tính.....	64
4. BÀI TẬP:.....	50
LAB 5. CHECKBOX, RADIO BUTTON, IMAGE BUTTON, IMAGE VIEW ..	51
1. MỤC ĐÍCH:	53
2. THỰC HIỆN:	53
2.1. Bài 1: Trong Android Studio, thiết kế giao diện như hình bên dưới.....	53
2.2. Bài 2: Tiếp tục sử dụng Project CheckBoxRadioButton trong Bài 1 để thực hiện.	56
2.3. Bài 3: Trong Android Studio, thiết kế giao diện như hình bên dưới. Yêu cầu: khi bấm vào nút Đổi hình, hình h1 sẽ chuyển thành hình h2 và ngược lại. Khi bấm vào ImageButton hoặc imageView sẽ đóng chương trình.....	59
3. BÀI TẬP:.....	63
LAB 6. TOAST, ALERT DIALOG, CUSTOM DIALOG, NOTIFICATION	Error!
Bookmark not defined.	
1. MỤC ĐÍCH	65
2. THỰC HIỆN.....	Error! Bookmark not defined.
2.1. Bài 1: Trong Android Studio, thiết kế giao diện như hình bên dưới.....	65
2.2. Bài 2: Tiếp tục sử dụng Project ToastAlertDialog trong Bài 1 để thực hiện.	66
3. BÀI TẬP:.....	73
LAB 7. ADAPTER, LIST VIEW ..	76
1. MỤC TIÊU:	76
2. THỰC HIỆN:	76

2.1. Bài 1: ListView cơ bản sử dụng mảng.	Error! Bookmark not defined.
2.2. Bài 2: Xây dựng ListView sử dụng String Array.....	80
2.3. Bài 3: Custom ListView	82
2.4. Bài 4: Recycle View	83
3. BÀI TẬP.....	85
LAB 8. SPINNER	87
1. MỤC TIÊU:.....	87
2. THỰC HIỆN:	87
2.1. Bài 1: Spinner	87
2.2. Bài 2: Kết hợp Spinner và ListView: Phần mềm quản lý sản phẩm	89
3. BÀI TẬP:.....	92
LAB 9. AUTOCOMPLETE TEXTVIEW, DATE VÀ TIME PICKER, TAB SELECTOR	93
1. MỤC ĐÍCH	93
2. THỰC HIỆN.....	93
2.1. Bài 1: Autocomplete textview	93
2.2. Bài 2: Date và time picker	93
2.3. Bài 3: Tab selector.....	93
3. BÀI TẬP:.....	93
LAB 10. ACTIVITY VÀ INTENT.....	94
1. MỤC TIÊU:.....	94
2. THỰC HIỆN:	94
2.1. Bài 1: Vòng đời Activity và Intent	94
2.2. Bài 2: Truyền và nhận dữ liệu giữa các Activity.....	96
2.3. BÀI 3: INTENT RESULT	98
3. BÀI TẬP.....	100
LAB 11. OPTION MENU – CONTEXT MENU – MENU ĐIỀU KHIỂN	101
1. MỤC TIÊU:.....	101
2. THỰC HIỆN.....	101
2.1. Bài 1: Option menu	101
2.2. Bài 2: Context Menu	102
2.3. Bài 3: Menu tìm kiếm.....	104
3. BÀI TẬP.....	106

LAB 12. ÔN TẬP: PHẦN MỀM QUẢN LÝ NHÂN VIÊN Error! Bookmark not defined.

1. MỤC TIÊU Error! Bookmark not defined.
2. THỰC HIỆN: Error! Bookmark not defined.
 - 2.1. Bài 1: Option menu Error! Bookmark not defined.
 - 2.2. Bài 2: Context menu Error! Bookmark not defined.
3. Bài tập Error! Bookmark not defined.

LAB 13. CƠ SỞ DỮ LIỆU SQLITE **109**

1. MỤC TIÊU 109
2. THỰC HIỆN: 109
 - 2.1. Bài 1: SQLite DB Browser 109
 - 2.2. Bài 2: Sao chép CSDL từ Asset vào thiết bị Android 112
 - 2.3. Bài 3: Truy vấn SQLite trong Android 115
 - 2.4. Bài 4: Thêm dữ liệu vào SQLite trong Android 116
3. BÀI TẬP: 124
 - 3.1. Viết phần mềm karaoke cho cơ sở dữ liệu karaoke bằng sqlite, viết chương trình hiển thị như hình dưới đây: 124
 - 3.2. Gợi ý cách làm: 125

LAB 14. GOOGLE MAP **130**

1. MỤC TIÊU 130
2. THỰC HÀNH 130
 - 2.1. Bài tập: Phần mềm sô tay nhà hàng 130
3. BÀI TẬP 139

LAB 15. SENSOR **140**

1. MỤC TIÊU: 140
2. THỰC HIỆN: 140
 - 2.1. Bài 1: 140
3. BÀI TẬP: 140

GIỚI THIỆU MÔN HỌC

1. Nội dung môn học:

- Tổng quan các kiến thức cơ bản về lập trình trên thiết bị di động.
- Lập trình trên thiết bị di động smart phone với Android.

2. Đề cương môn học:

Buổi	Nội dung	Số tiết
1	lab 1: ứng dụng android đầu tiên	4
2	lab 2: linear layout	4
3	lab 3: find view by id – on click xml – anomous listener	4
4	lab 4: text view – edit text – button – đồ án 1: phần mềm máy tính	4
5	lab 5: checkbox - radio button – image button – image view	4
6	lab 6: toast – alert dialog	4
7	lab 7: list view	4
8	lab 8: spinner – đồ án 2: phần mềm quản lý sản phẩm	4
9	lab 9: activity	4
10	lab 10: option menu – context menu – menu điều khiển	4
11	lab 11: đồ án 3: phần mềm quản lý nhân viên	4
12	lab 12: cơ sở dữ liệu sqlite	4
13	lab 13: google map	4
14	Dự trũ	
15	Báo cáo đồ án	

3. Tài liệu học tập:

3.1. Giáo trình chính

[1] Android Developer Fundamentals, Google Developer Training team, 2017

3.2. Tài liệu tham khảo

[2] Tập “Bài tập thực hành Lập trình thiết bị di động”, Khoa CNTT, 2019

[3] Beginning Android 4 Application Development, Wei-Meng Lee, John Wiley & Sons, 2012

4. Phương pháp đánh giá học phần:

STT	Thành phần	Trọng số	Quy định
1	Chuyên cần	10%	Vắng >4 buổi = cấm thi. Làm các bài tập trên lớp.
2	Giữa kỳ	30%	Viết một ứng dụng nhỏ trên Android.
3	Cuối kỳ	60%	Trình bày đồ án môn học

5. Danh sách đề tài gợi ý:

STT	TÊN ĐỀ TÀI	GHI CHÚ
1	Ứng dụng quản lý thư viện nhạc	
2	Ứng dụng sổ tay quản lý chi tiêu cá nhân	
3	Ứng dụng game Tetris	
4	Ứng dụng sổ tay ghi chú cá nhân	
5	Ứng dụng Order trong nhà hàng	
6	Ứng dụng từ điển Anh-Việt, Việt-Anh	
7	Ứng dụng quản lý kho hàng	
8	Ứng dụng Game Line	
9	Ứng dụng Game Matching (tìm cặp)	
10	Ứng dụng tra cứu mã số bài hát Karaoke 6 số	
11	Ứng dụng hỗ trợ học và luyện thi LT bằng lái xe B2	
12	Ứng dụng đọc tin tức online	
13	Ứng dụng đặt xe trực tuyến	
14	Ứng dụng đo tốc độ xe bằng GPS	
15	Ứng dụng hướng dẫn tập GYM	
16	Ứng dụng Game Cờ tướng	
17	Ứng dụng Game Flappy bird	
18	Ứng dụng Game cờ vua	
19	Ứng dụng tra cứu văn bản pháp luật	
20	Ứng dụng tra cứu lỗi vi phạm giao thông đường bộ	
21	Ứng dụng sổ tay sinh viên	
22	Ứng dụng game Caro (chơi với máy)	
23	Mô phỏng trò chơi Lines 98	
24	Viết chương trình vẽ đồ thị toán học	
25	Viết game phi thuyền bằng sensor	
26	Viết chương trình nhận diện khuôn mặt và dự đoán trong vòng 20 năm nữa thì khuôn mặt đó sẽ thay đổi như thế nào	

Lưu ý: nếu hai nhóm sinh viên cũng tìm hiểu một đề tài, thì ứng dụng demo phải hoàn toàn khác nhau!

LAB 1. ỨNG DỤNG ANDROID ĐẦU TIÊN

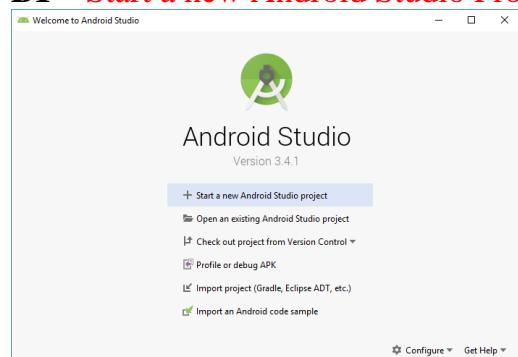
1. MỤC TIÊU:

- Cài đặt môi trường làm việc Android Studio.
- Hiểu cấu trúc cơ bản của Android project.
- Cài đặt máy ảo Genymotion.
- Chạy ứng dụng Android đầu tiên.

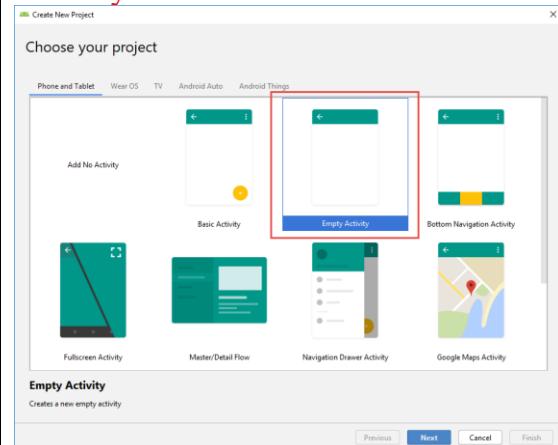
2. THỰC HIỆN:

2.1. Bài 1: Ứng dụng Android đầu tiên.

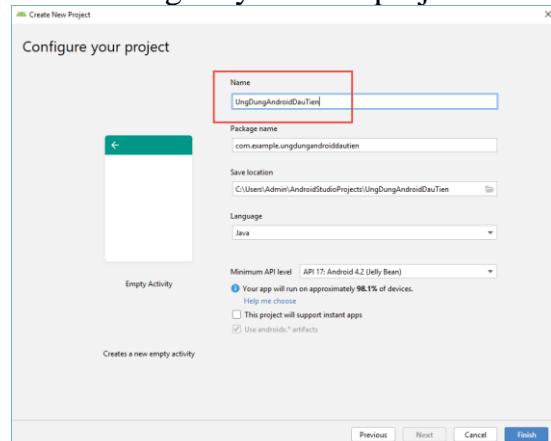
B1 – Start a new Android Studio Project



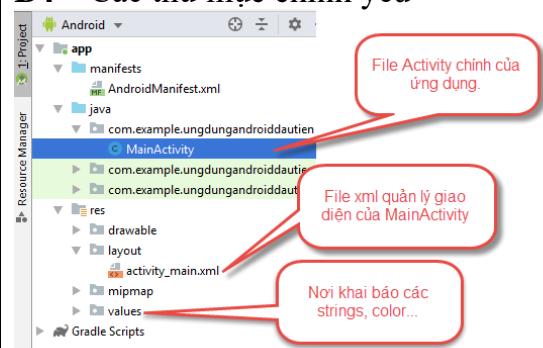
B2 – Choose your project → Empty Activity



B3 – Configure your new project



B4 – Các thư mục chính yếu



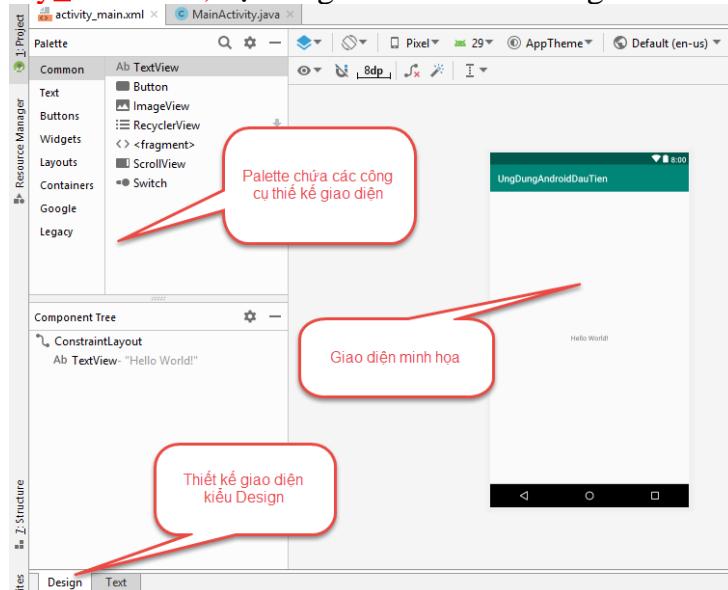
B5 – Trong MainActivity.java



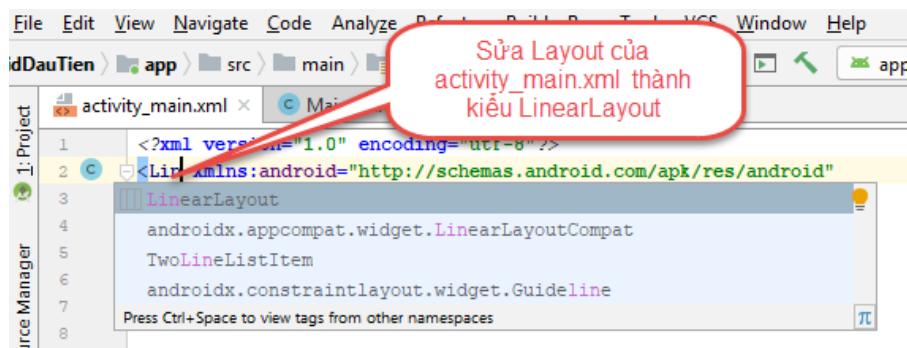
B6 – Trong activity_main.xml, có 2 kiểu thiết kế: Text và Design



B7 - Trong activity_main.xml, bật sang kiểu thiết kế Design:



B8 - Trong activity_main.xml, bật sang kiểu thiết kế Text → sửa Layout của activity_main.xml thành LinearLayout



B9 – Trong activity_main.xml, xóa phần TextView có sẵn để thay thế bằng một TextView tự thiết kế.



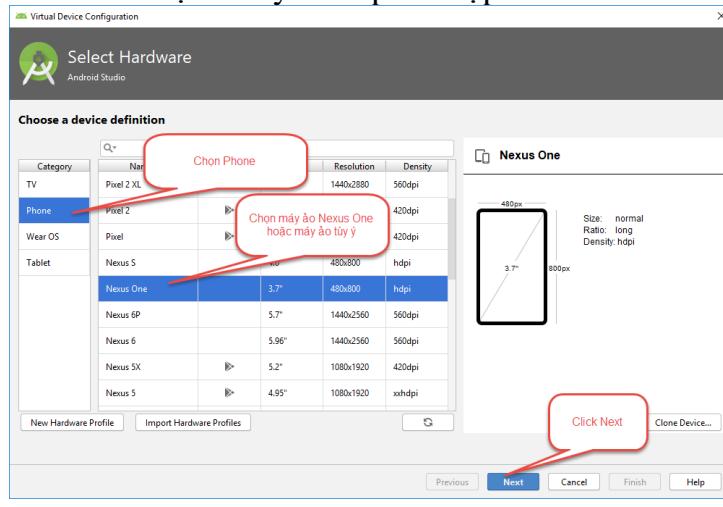
B10 – Trong `activity_main.xml`, thiết kế một `TextView` có nội dung: “Đây là ứng dụng android đầu tiên của tui !”, cỡ chữ `30sp`, màu đỏ, canh giữa chữ và canh chữ nằm giữa layout.
→ Cập nhật đoạn code sau.

```
<TextView
    android:text="Đây là ứng dụng Android đầu tiên của tui!"
    android:textSize="30sp"
    android:textColor="@android:color/holo_red_light"
    android:layout_center_in_parent="center"
    android:layout_width="match_parent"
    android:layout_height="wrap_content" />
```

B11 – Run ‘app’

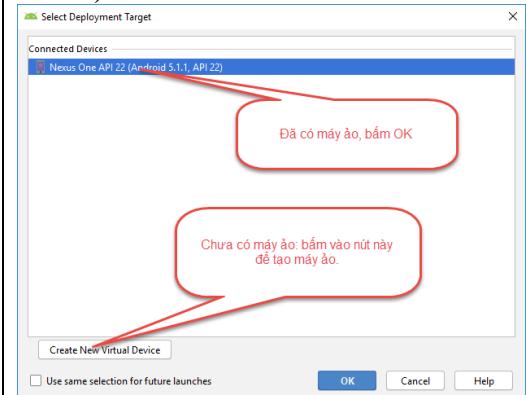


B13 – Chọn máy ảo phù hợp → bấm Next

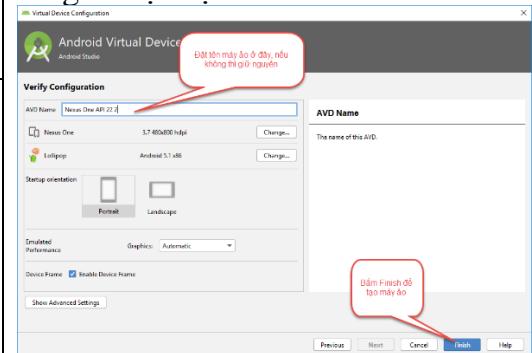


B14 – Chọn hệ điều hành → Next

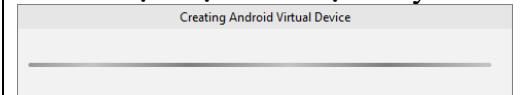
B12 – Kiểm tra xem đã có máy ảo chưa, nếu chưa có → Bấm vào nút **Create New Virtual Device** để tạo máy ảo. (Chỉ tạo máy ảo 1 lần, những lần sau sẽ dùng lại máy ảo này để demo)

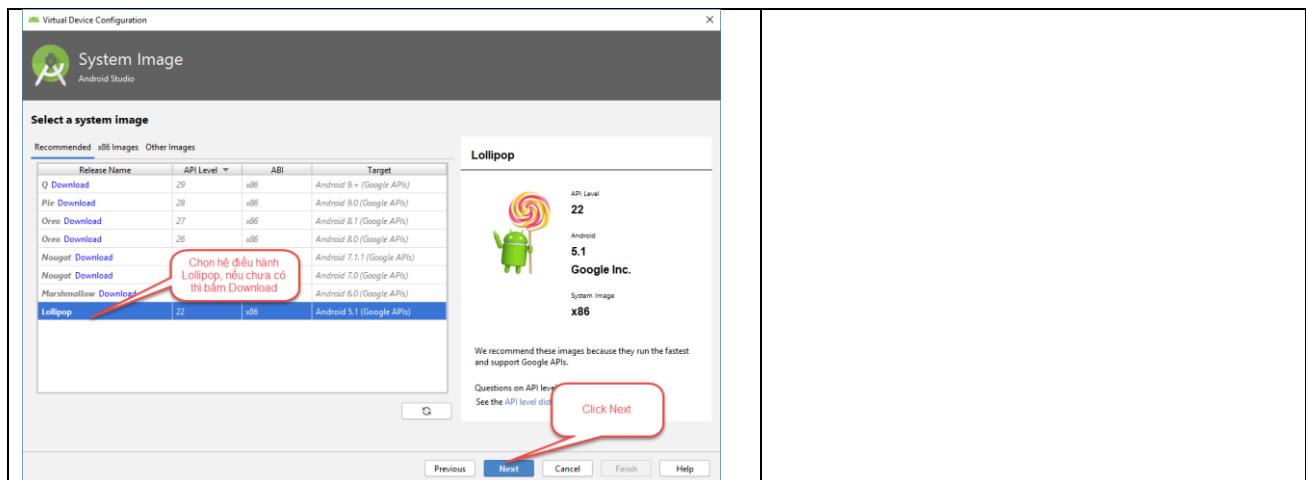


B15 – Đặt tên cho máy ảo, nếu không thì giữ mặc định → Finish.

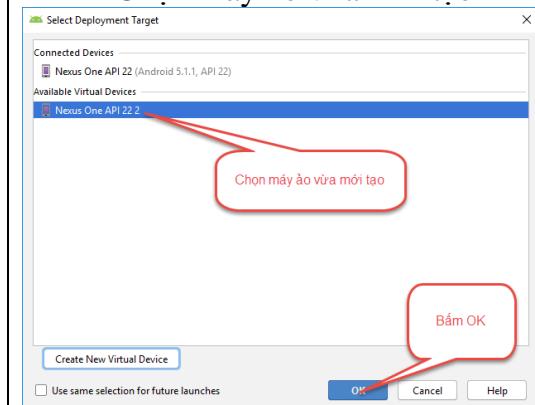


B16 – Đợi một lát để tạo máy ảo

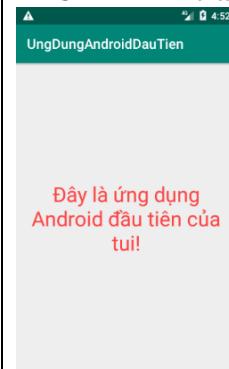




B17 – Chọn máy ảo vừa mới tạo → OK



B18 – Kiểm tra



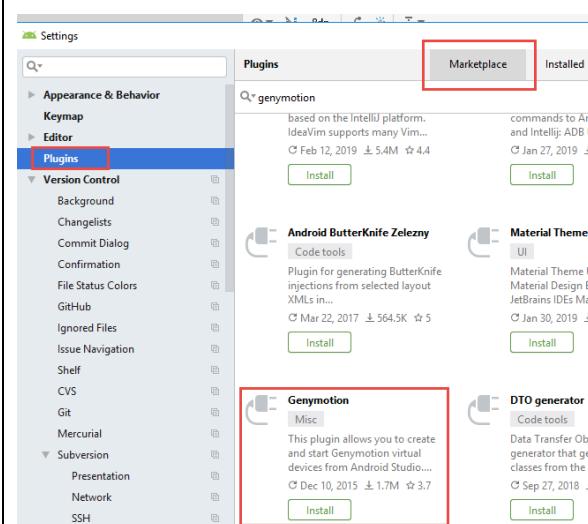
2.2. Bài 2: Chương trình giả lập máy ảo Genymotion

B1 – Đăng ký và cài đặt Genymotion

- Truy cập địa chỉ: <https://www.genymotion.com/account/create/>
- Điện thoại tin và nhấn Create Account.
- Check mail để kích hoạt tài khoản Genymotion.
- Tải Genymotion bản Personal with VirtualBox.
- Cài đặt Genymotion.

B2 – Cài đặt plugin Genymotion

Trong Android Studio → File → Settings → Plugins → Marketplace → Tìm “genymotion” → Install → Restart IDE.



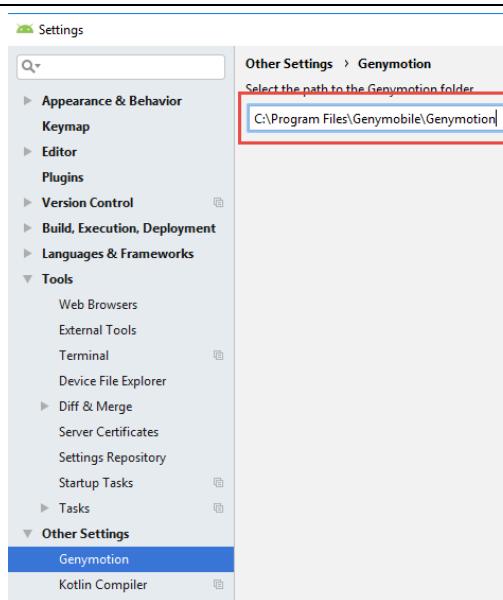
B4 – Cài đặt điện thoại ảo trong Genymotion.

Tìm “samsung galaxy” → Samsung Galaxy S3 → Install.

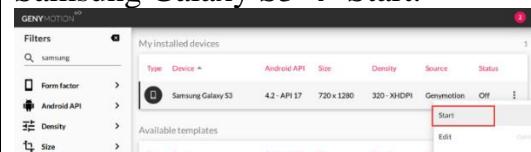


B3 – Chọn đường dẫn cho Plugin Genymotion.

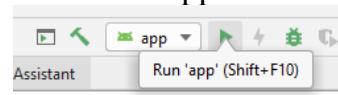
Trong Android Studio → File → Settings → Other Settings → Genymotion → Browse tới đường dẫn của Genymotion.



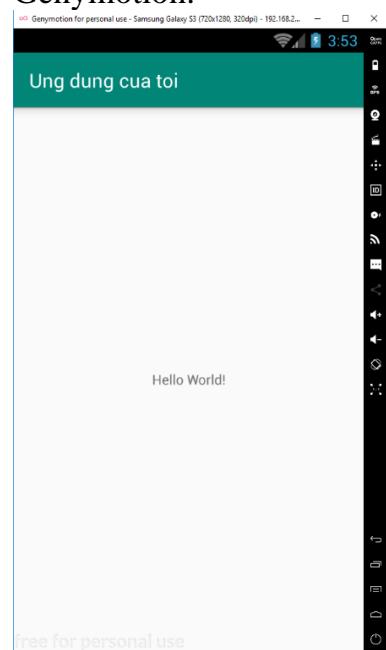
B5 – Khởi động máy ảo Genymotion. Samsung Galaxy S3 → Start.



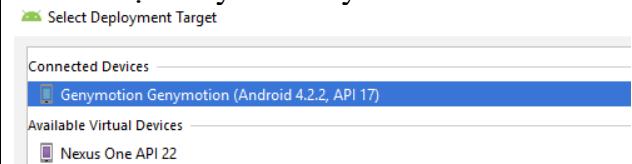
B6 – Run ‘app’



B8 – Quan sát màn hình kết quả trên máy ảo Genymotion.



B7 – Chọn máy ảo Genymotion



B9 – Quan sát màn hình ứng dụng máy ảo Genymotion.

Thấy có “Ung dung cua toi”.



2.3. Bài 3: Ứng dụng Vysor – phần mềm Ánh xạ điện thoại vào máy tính

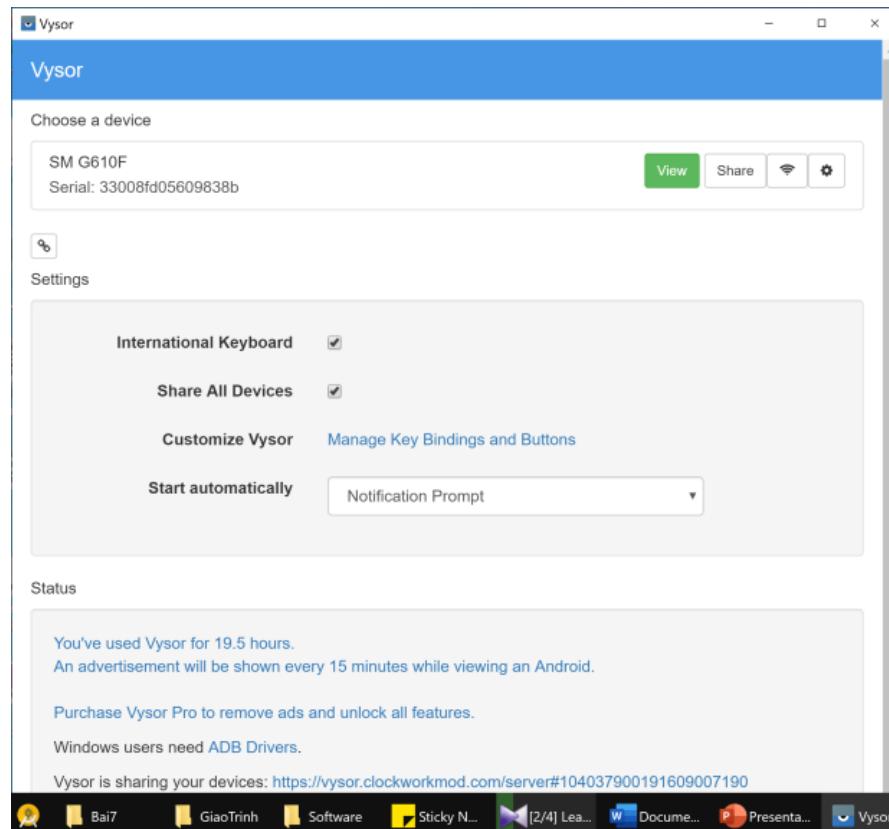
Phần mềm Vysor rất hay, nó cho phép chúng ta tương tác 2 chiều. Có thể từ Vysor trong máy tính điều khiển điện thoại và ngược lại.

Rất tiện lợi cho Giảng Viên trình chiếu giảng dạy, các bạn Sinh Viên trình chiếu báo cáo đồ án, các team work làm việc nhóm, triển khai dự án.

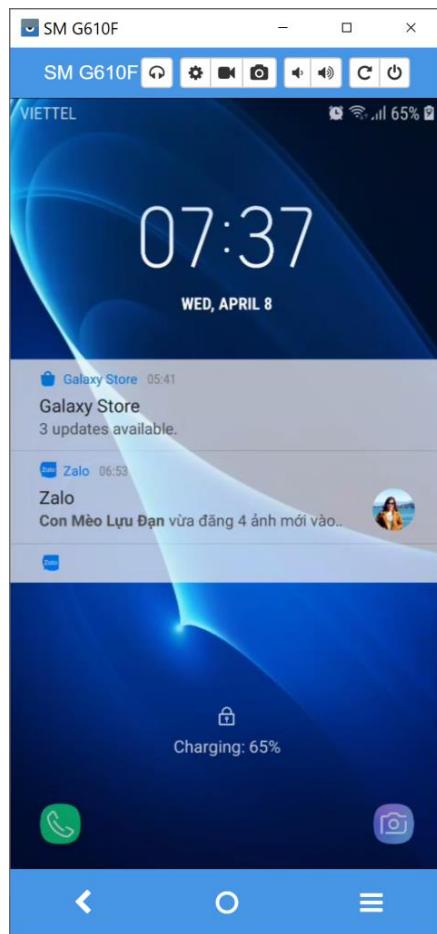


Nhấn Download, chương trình sẽ hỏi phiên bản cài đặt.

Tải xong thì bấm vào nó để cài đặt, vô cùng đơn giản, bấm rồi chờ xíu là xong, Vysor chạy lên và ta sẽ thấy các thiết bị điện thoại sẽ được hiển thị như hình bên dưới, dĩ nhiên là chỉ điện thoại nào đã cấu hình USB Debugging.



Ta muốn Ánh xạ điện thoại nào vào máy tính thì chọn View ở thiết bị đó, kết quả:



Tới đây ta có thể điều khiển điện thoại ngay trên Máy tính của mình, có thể trình chiếu... rất hay. Xem màn hình toàn cảnh:



3. Bài tập:

Bài 1: Trình bày cách tạo máy ảo Android Emulator

Bài 2: Trình bày cách sử dụng Android Emulator: cấu hình, gọi điện, nhắn tin, xoay màn hình, giả lập cảm biến...

Bài 3: So sánh sự khác biệt về tốc độ, chức năng giữa máy ảo Android Emulator và máy ảo Android Genymotion

Bài 4: Trình bày cấu hình Debugging bằng USB cho thiết bị di động

Bài 5: Muốn máy tính kết nối với thiết bị di động để trình chiếu và tương tác qua lại thì dùng phần mềm nào? Hãy trình bày chi tiết cách tải và sử dụng.

Bài 6: Hãy trình bày các bước để một phần mềm thiết kế trong Android Studio có thể chạy được trên thiết bị di động.

LAB 2. CÁC LAYOUT CƠ BẢN

Layout là nơi để tổ chức sắp xếp control/View trên giao diện và thường được sử dụng đầu tiên khi ta thiết kế giao diện. Các Layout này được Android Studio bố trí trong Palette/Layouts:

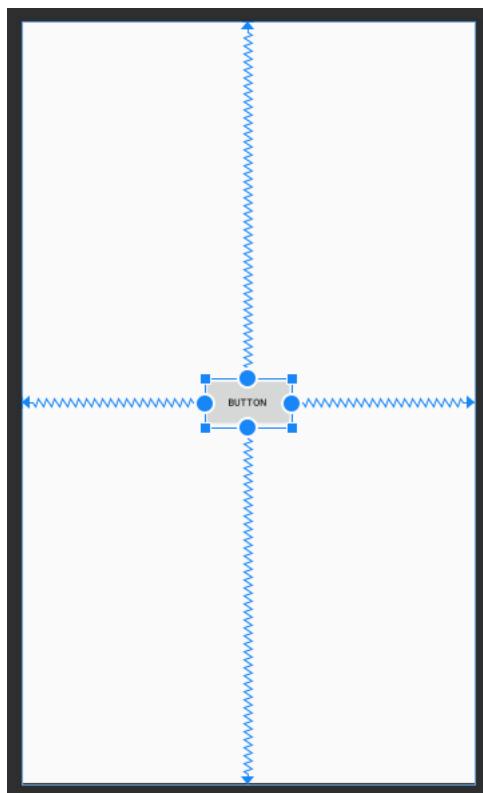
1. MỤC TIÊU:

- Trong bài này sẽ trình bày về các layout:
 - o ConstraintLayout
 - o Frame Layout
 - o Linear Layout
 - o Table Layout
 - o Relative layout
 - o LinearLayout
- Đọc thêm về các loại layout ở đây:
<https://developer.android.com/guide/topics/ui/layout>

2. LÝ THUYẾT:

2.1. Constraint Layout

Constraint layout dùng để định vị trí tương đối của Một control/View trên màn hình với các phần tử khác trong layout:



Ta có thể xác định một constraint cho một hay nhiều mặt của một view bằng chế độ kết nối bất kỳ sau đây :

- Điểm neo nằm trên một View khác.
- Một cạnh của layout.
- Một guideline.

Android cung cấp các loại Constraint sau: Relative positioning, Margins, Centering positioning, Visibility behavior, Dimension constraints, Chains.

Layout này được áp dụng để thiết kế các màn hình Responsive không lệ thuộc vào độ phân giải.

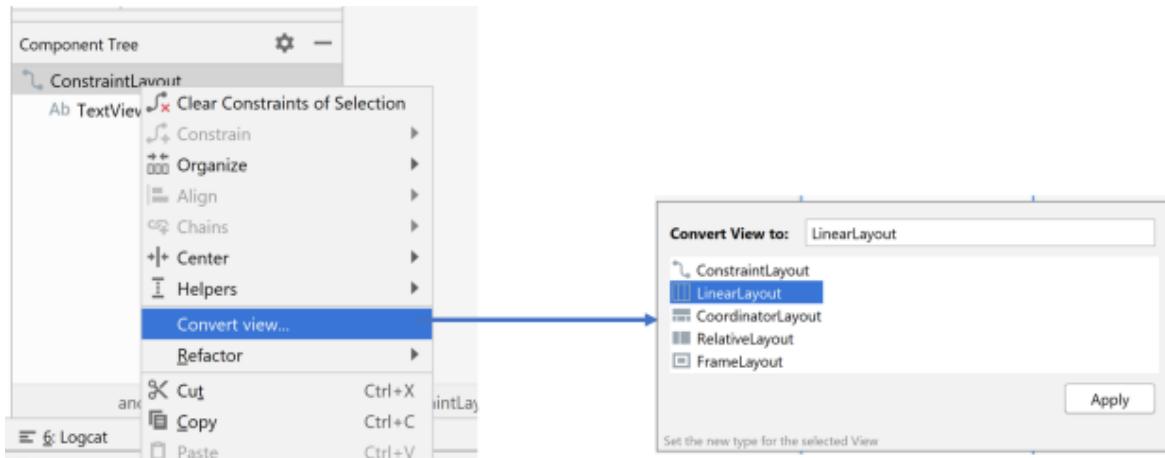
Cấu trúc XML của ConstraintLayout được mô tả như sau:

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <Button
        android:id="@+id/button2"
        android:layout_width="123dp"
        android:layout_height="70dp"
        android:text="Button"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

Android Studio còn cung cấp chức năng chuyển đổi từ ConstraintLayout qua các layout khác(và ngược lại), bằng cách bấm chuột phải vào Layout bất kỳ/chọn Convert View ==>Chọn layout đích mà ta muốn chuyển đổi:



2.2. Frame Layout

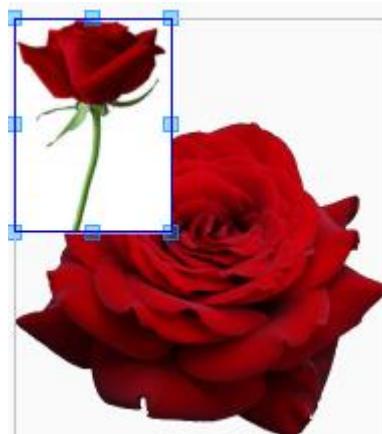
Sử dụng trong các trường hợp xây dựng bố cục tổ chức hiển thị một đối tượng duy nhất.

Đối tượng mặc định vị trí top-left trên FrameLayout, View khai báo sau sẽ chồng lên View khai báo trước, có thể sử dụng thuộc tính Gravity để thiết lập lại vị trí.

Cấu trúc xml của Frame Layout được mô tả như sau:

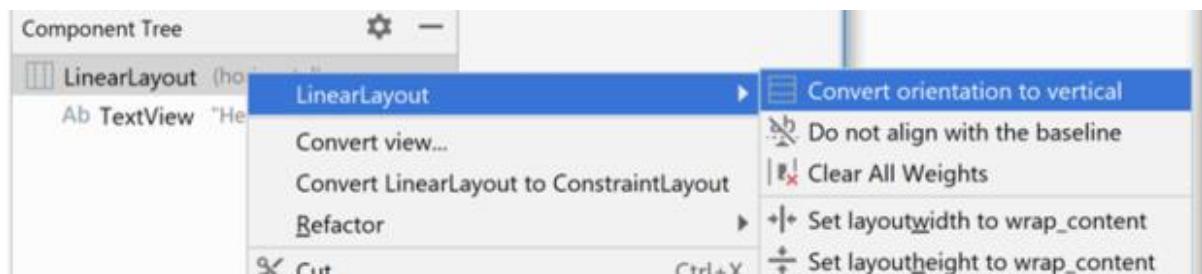
```
<FrameLayout android:id="@+id/activity_main"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <ImageView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        app:srcCompat="@drawable/h0"
        android:id="@+id/imageView3"/>
    <ImageView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        app:srcCompat="@drawable/h1"
        android:id="@+id/imageView4" />
</FrameLayout>
```

Ta thấy hình h1 khai báo sau nên h1 sẽ chồng lên h0, cả hai hình này đều được neo ở góc top-left của màn hình:



2.3. Linear Layout

Linear Layout lại vừa có Horizontal và Vertical, ta có thể bấm chuột phải vào nó để đổi tiếp:



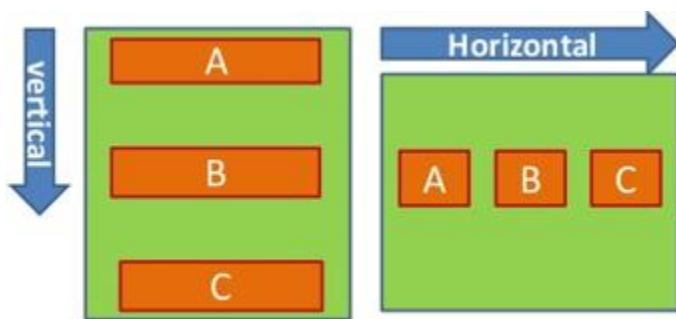
Sử dụng trong các trường hợp xây dựng bố cục tổ chức hiển thị các đối tượng theo một chiều duy nhất (chiều dọc-vertical hoặc ngang-horizontal).

Đối tượng mặc định vị trí top left trên LinearLayout , có thể sử dụng thuộc tính Gravity để thiết lập lại vị trí.

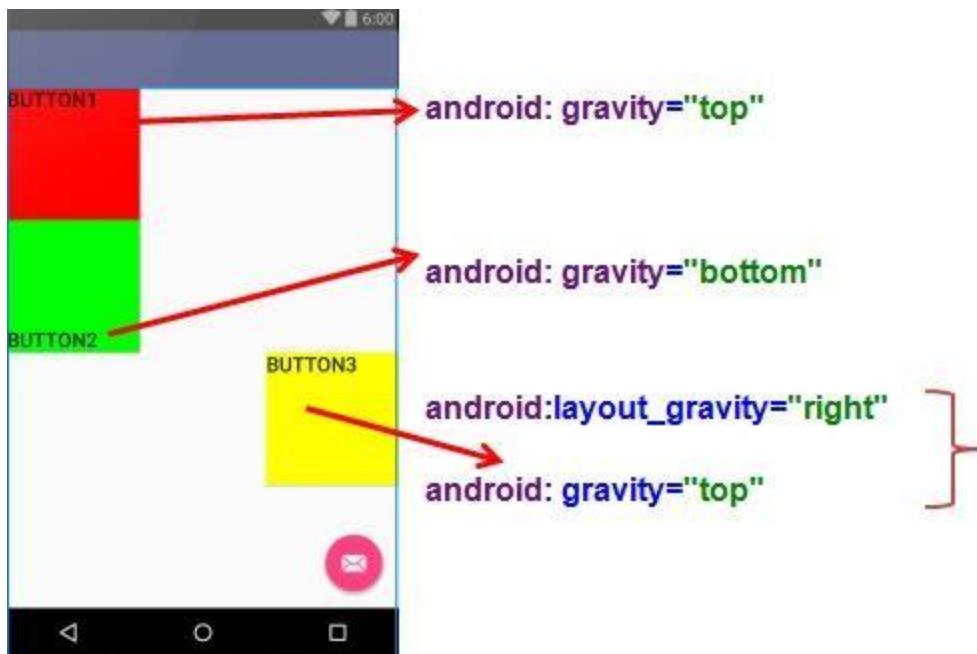
Cấu trúc XML của LinearLayout có dạng như sau:

```
<LinearLayout  
    xmlns:tools="http://schemas.android.com/tools"  
    android:id="@+id/linearLayout1"  
    android:orientation="horizontal"  
    android:layout_width="match_parent"  
    android:layout_height="match_parent">
```

Ta thay đổi chiều dọc (android:orientation= “vertical”), chiều ngang (android:orientation= “horizontal”) các View sẽ tự động sắp xếp:

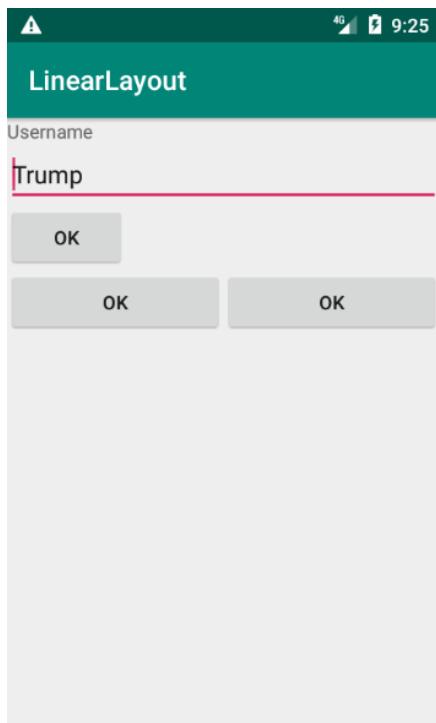


LinearLayout hỗ trợ thuộc tính `text android:gravity` để căn chỉnh chữ trong View, `android:layout_gravity` để căn chỉnh vị trí tương đối của View trong layout.



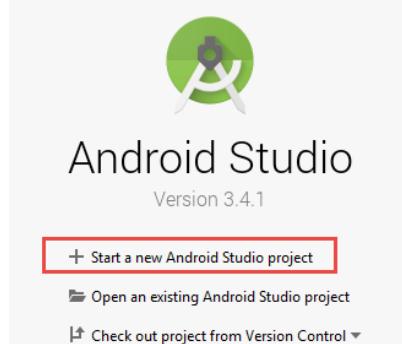
Ngoài ra LinearLayout còn hỗ trợ nhiều thuộc tính khác như `android:padding` để tăng độ dày của View, `android:layout_margin` để tăng/giảm khoảng cách giữa các View.

Ví dụ: Sử dụng Linear Layout, thiết kế giao diện như hình bên dưới.

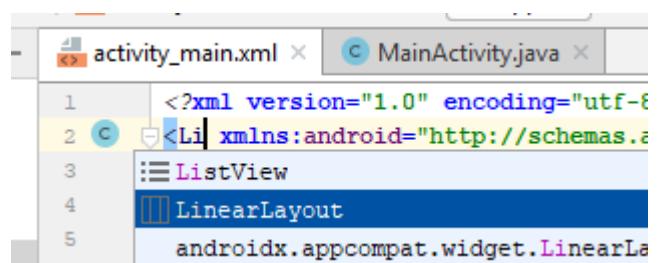


Hướng dẫn:

B1 - Mở Android Studio → New Project → Empty Activity → Đặt tên Project: LinearLayout.



B2 – Mở `activity_main.xml` sửa kiểu layout sử dụng thành `LinearLayout`.



B3 – Trong `activity_main.xml` thêm vào `TextView`, `EditText`, `Button` như sau:

```
<TextView  
    android:text="Username"  
    android:layout_gravity="center"  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content" />  
<EditText  
    android:text="Trump"  
    android:id="@+id/edtName"  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content" />  
<Button  
    android:id="@+id/btnOk"  
    android:text="OK" />
```

```
    android:layout_width="wrap_content"
    android:layout_height="wrap_content" />
```

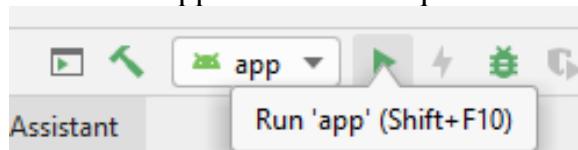
B4 - Trong `activity_main.xml` tiếp tục thêm vào LinearLayout, TextView, EditText, Button như sau:

```
<LinearLayout
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="wrap_content">
    <Button
        android:layout_weight="1"
        android:text="OK"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"/>
    <Button
        android:layout_weight="1"
        android:text="OK"
        android:id="@+id	btnOk2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"/>
</LinearLayout>
```

B6 - Kết quả



B5 – Run ‘app’ kiểm tra kết quả.



2.4. Table Layout

TableLayout kế thừa từ LinearLayout, cho phép hiển thị các đối tượng theo nhiều dòng (TableRow). Mỗi dòng có thể chứa nhiều View, mỗi View được xem là một cột.

Đặc biệt TableLayout lấy dòng có số lượng View nhiều nhất làm số cột, tức là nếu TableLayout có 3 dòng: dòng đầu tiên có 2 View, dòng thứ hai có 5 View và dòng cuối cùng có 4 View thì cột được xác định cho TableLayout này là 5.

<TableLayout

```
xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <Tablerow>
        <Button/>
    </Tablerow>
</TableLayout>
```

TableLayout cho phép ta Meger(trộn) các ô trong TableRow bằng cách thiết lập thuộc tính layout_span cho View trong TableRow:

```
<TableRow>
    <TextView android:text="URL:" />
    <EditText
        android:id="@+id/entry"
        android:layout_span="3" />
</TableRow>
```

Hay layout_column để di chuyển vị trí của View tới một cột nào đó trong TableRow:

Label (URL)	EditText	EditText-span	
Column 0	Column 1	Column 2 Button Cancel	Column 3 Button OK

Dùng stretchColumns để dãn đều các control, các cell (ta thường dùng dấu "*" để cấu hình cho toàn bộ các cột, hoặc nhập số để cấu hình cho cột được chỉ định):

```
<TableLayout
```

```
    android:stretchColumns="*"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
```

Ví dụ: thiết kế giao diện đăng nhập bằng table layout



Cấu trúc xml của giao diện như sau:

```
<?xml version="1.0" encoding="utf-8"?>
<TableLayout android:layout_height="match_parent"
    android:layout_width="match_parent"
    android:stretchColumns="1"
    android:layout_margin="8dp"

xmlns:android="http://schemas.android.com/apk/res/android"
">

    <TableRow>
        <TextView
            android:text="ĐĂNG NHẬP"
            android:layout_span="2"
            android:textStyle="bold"
            android:textSize="20sp"
            android:gravity="center"/>
    </TableRow>
    <TableRow>
        <TextView android:text="Username"
        android:textStyle="bold" />
        <EditText android:hint="nhập id" />
    </TableRow>
    <TableRow>
        <TextView android:text="Password"
        android:textStyle="bold" />
        <EditText android:inputType="textPassword"
            android:hint="nhập mật khẩu"
            />
    </TableRow>

```

```
<LinearLayout  
    android:gravity="center"  
    android:orientation="horizontal">  
    <Button  
  
        style="@style/Widget.AppCompat.Button.Colored"  
            android:text="Ok"  
            android:layout_width="wrap_content"  
            android:layout_height="wrap_content" />  
    <Button  
  
        style="@style/Widget.AppCompat.Button.Colored"  
            android:backgroundTint="#291010"  
            android:text="Hủy"  
            android:layout_width="wrap_content"  
            android:layout_height="wrap_content" />  
    <Button  
        android:text="Quên mật khẩu"  
  
    style="@style/Widget.AppCompat.Button.Colored"  
        android:backgroundTint="#291010"  
        android:layout_width="wrap_content"  
        android:layout_height="wrap_content" />  
  
    </LinearLayout>  
</TableLayout>
```

2.5. Relative layout

RelativeLayout cho phép sắp xếp các View theo vị trí tương đối giữa các View khác trên giao diện (kể cả View chứa nó). Thường nó dựa vào Id của các View khác để sắp xếp theo vị trí tương đối.

Do đó khi làm RelativeLayout ta phải chú ý là đặt Id của View cho chuẩn xác, nếu sau khi Layout xong mà ta lại đổi Id của các View thì giao diện sẽ bị xáo trộn (do đó nếu đổi ID thì phải đổi luôn các tham chiếu khác sao cho khớp với Id mà ta mới đổi).

```

<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:background="#ff0000ff"
    android:padding="10px" >

    <TextView android:id="@+id/label"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:background="#ffff0077"
        android:text="Type here:" />

    <EditText android:id="@+id/entry"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_below="@+id/label" />

```

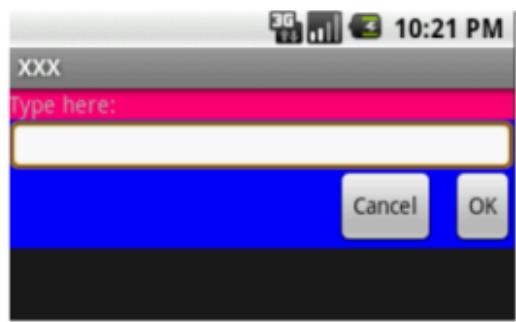
```

<Button
    android:id="@+id/ok"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_below="@+id/entry"
    android:layout_alignParentRight="true"
    android:layout_marginLeft="10px"
    android:text="OK" />

<Button
    android:text="Cancel"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_toLeftOf="@+id/ok"
    android:layout_alignTop="@+id/ok" />

</RelativeLayout>

```



Trên giao diện lúc kéo thả các View trên RelativeLayout ta tự căn chỉnh, lúc này các View sẽ tự động sắp xếp theo cách ta kéo thả trên màn hình.

Ví dụ: thiết kế giao diện đăng nhập sau bằng relative layout



Cấu trúc xml của giao diện như sau:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_margin="8dp"
    android:gravity="center"
    android:background="#e2e3eb">

    <TextView
        android:id="@+id/textview"
        android:text="Đăng nhập"
        android:layout_centerHorizontal="true"
        android:textSize="20sp"
        android:textAllCaps="true"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

    <EditText
        android:id="@+id/username"
        android:hint="Username"
        android:layout_below="@+id/textview"
        android:layout_marginTop="40dp"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />
    <EditText
        android:id="@+id/password"
        android:hint="passwords"
        android:inputType="textPassword"
        android:layout_below="@+id/username"
        android:layout_marginTop="8dp"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />
    <Button
        android:id="@+id/btnLogin"
        android:layout_below="@+id/password"
        android:layout_toLeftOf="@+id/recoverypassword"
        android:text="Đăng nhập"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />
    <Button
        android:id="@+id/recoverypassword"
        android:text="Quên mật khẩu"
        android:layout_centerHorizontal="true"
        android:layout_below="@+id/password"
```

```
    android:layout_width="wrap_content"
    android:layout_height="wrap_content" />

<Button
    android:id="@+id/help"
    android:text="Trợ giúp"
    android:layout_below="@+id/password"
    android:layout_toRightOf="@+id/recoverypassword"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content" />

</RelativeLayout>
```

3. Bài tập:

Bài 1: Để sắp xếp các view trên giao diện theo chiều đứng và chiều nằm ngang thì ta nên dùng Layout nào?

Bài 2: ConstraintLayout dùng để làm gì ?

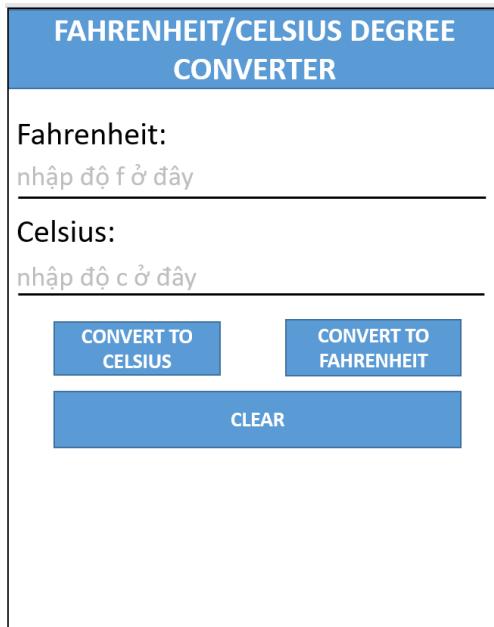
Bài 3: Muốn sắp xếp các View theo dạng dòng và cột thì ta dùng Layout nào ?

Bài 4: Các thuộc tính nào bắt buộc phải khai báo trong XML cho các View ?

Bài 6: Sử dụng layout phù hợp và các view để thiết kế màn hình sau:



Bài 7: Viết chương trình chuyển đổi độ C qua độ F và ngược lại. Yêu cầu sử dụng LinearLayout để sắp xếp các View trên giao diện



Bài 8: Viết chương trình giải phương trình bậc 1 với giao diện dưới đây:

GIẢI PHƯƠNG TRÌNH BẬC 1

Nhập a:

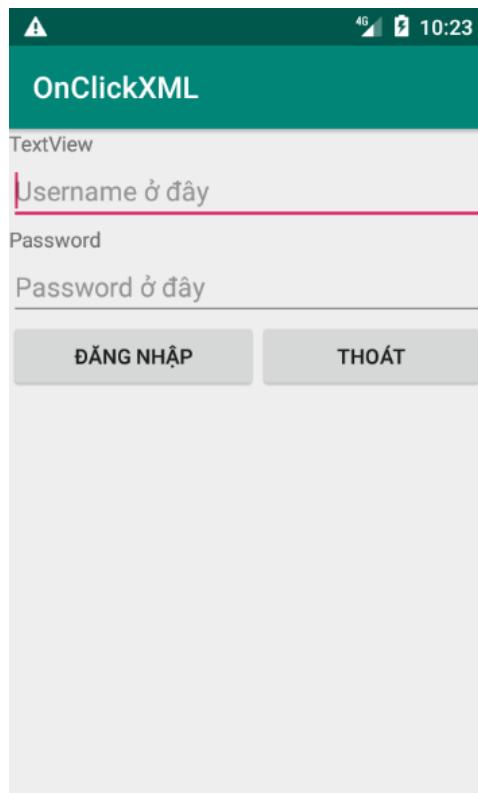
Nhập b:

Kết quả:

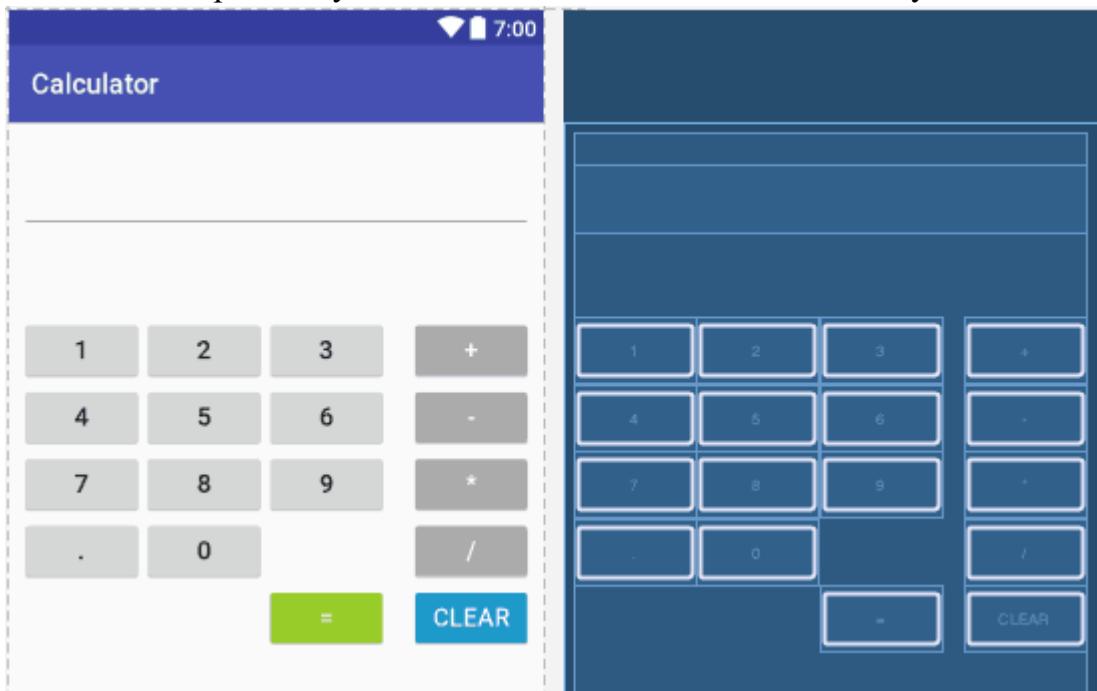
RESET **GIẢI**

THOÁT

Bài 9: Viết giao diện đăng nhập như sau. Yêu cầu sử dụng LinearLayout



Bài 10: Kết hợp các Layout và các View để thiết kế màn hình máy tính bỏ túi như sau:



LAB 3. CÁC KỸ THUẬT XỬ LÝ SỰ KIỆN

1. MỤC TIÊU:

Xử lý sự kiện trên view trong Android rất quan trọng, nó dùng để lắng nghe các hành vi của người dùng trên phần mềm. Có nhiều kỹ thuật xử lý sự kiện.

2. LÝ THUYẾT:

2.1. FindViewById

Khi lập trình Android, để tương tác với các View/Control trong giao diện chúng ta thường thông qua thuộc tính Id của các view/control để truy xuất thay đổi dữ liệu. Vì Android chia màn hình (Activity) thành hai phần: Phần thiết kế giao diện, phần xử lý nghiệp vụ.

Do đó để truy xuất được tới các View trong phần giao diện Android cung cấp hàm findViewById

Phần giao diện:

```
<Button
    android:id="@+id/btnOk"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/my_button_text"/>
```

Phần xử lý nghiệp vụ:

```
Button btnOk = (Button)findViewById(R.id.btnOk);
```

2.2. OnClickXML

OnClickXML là một loại sự kiện được gán cho các View lúc thiết kế giao diện, sự kiện này chỉ áp dụng cho những View được kéo thả sẵn trong màn hình, nếu View được tạo ra trong quá trình thực thi thì không thể sử dụng được.

Đầu tiên trong giao diện (activity_main.xml) ta dùng thuộc tính android:onClick để gán sự kiện cho các View.

```
<Button
    android:id="@+id/btnThoat"
    android:onClick="xulyThoat"
    android:text="Thoát"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"/>
```

Sau đó ta vào ActivityMain.java để khai báo hàm sự kiện như bên dưới hoặc ngay đằng sau xuLyThoat trong xml ta nhấn tổ hợp phím Alt+Enter thì Android Studio sẽ tự động phát sinh ra sự kiện này:

```
public void xuLyThoat(View view) {  
    //thoát chương trình  
    finish();  
}
```

Ví dụ: viết chương trình gọi điện thoại đơn giản. Giao diện như hình bên dưới.



Bước 1: Thiết kế giao diện. Cấu trúc xml như sau:

```
<?xml version="1.0" encoding="utf-8"?>  
<androidx.constraintlayout.widget.ConstraintLayout  
    xmlns:android="http://schemas.android.com/apk/res/android"  
    xmlns:app="http://schemas.android.com/apk/res-auto"  
    xmlns:tools="http://schemas.android.com/tools"  
    android:layout_width="match_parent"  
    android:layout_height="match_parent"  
    tools:context=".MainActivity">  
  
    <TextView
```

```

        android:id="@+id/textView"
        android:layout_width="0dp"
        android:layout_height="21dp"
        android:background="#FFEB3B"
        android:text="Mời bạn nhập số phone:"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"      />

<EditText
        android:id="@+id edtPhoneNumber"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:ems="10"
        android:inputType="phone"
        android:text="0987773061"

        app:layout_constraintTop_toBottomOf="@+id/textView"
        tools:layout_editor_absoluteX="74dp"      />

<Button
        android:id="@+id btnCall"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Gọi điện"
        android:onClick="xuLyGoiDien"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"

        app:layout_constraintTop_toBottomOf="@+id edtPhoneNumber"
        />
</androidx.constraintlayout.widget.ConstraintLayout>
```

Trên View/Control Edittext (để nhập liệu) ta đặt id là “`edtPhoneNumber`”, code bên dưới sẽ truy suất vào control Edittext dựa vào id này.

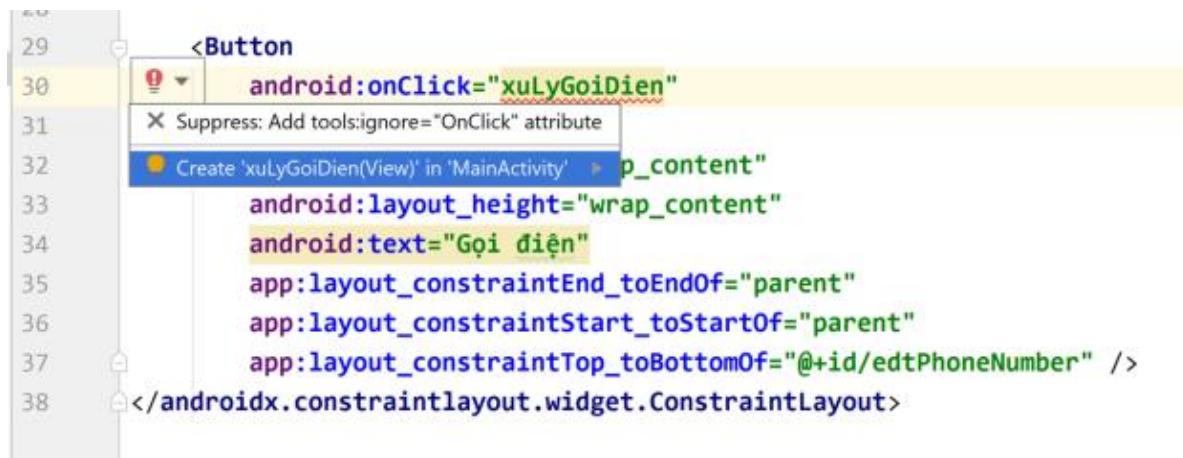
Bước 2: Bây giờ ta bổ sung sự kiện cho nút Gọi điện, thêm thuộc tính `android:onClick`, đặt tên hàm bên trong:

```

<Button
        android:onClick="xuLyGoiDien"
        android:id="@+id btnCall"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Gọi điện"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id edtPhoneNumber" />
```

Ví dụ ta đặt tên hàm là xuLyGoiDien, lúc này nó sẽ báo lỗi vì chưa được khai báo trong Java code

Bước 3: giờ ta làm cho nó tự động phát sinh mã lệnh như sau: Click chuột vào hàm, nó xô ra cái bóng đèn, nhấn vào bóng đèn chọn -> Create ‘xuLyGoiDien(View)’ in ‘MainActivity’ (Hoặc bạn cũng có thể nhấn tổ hợp phím Alt+ Enter)

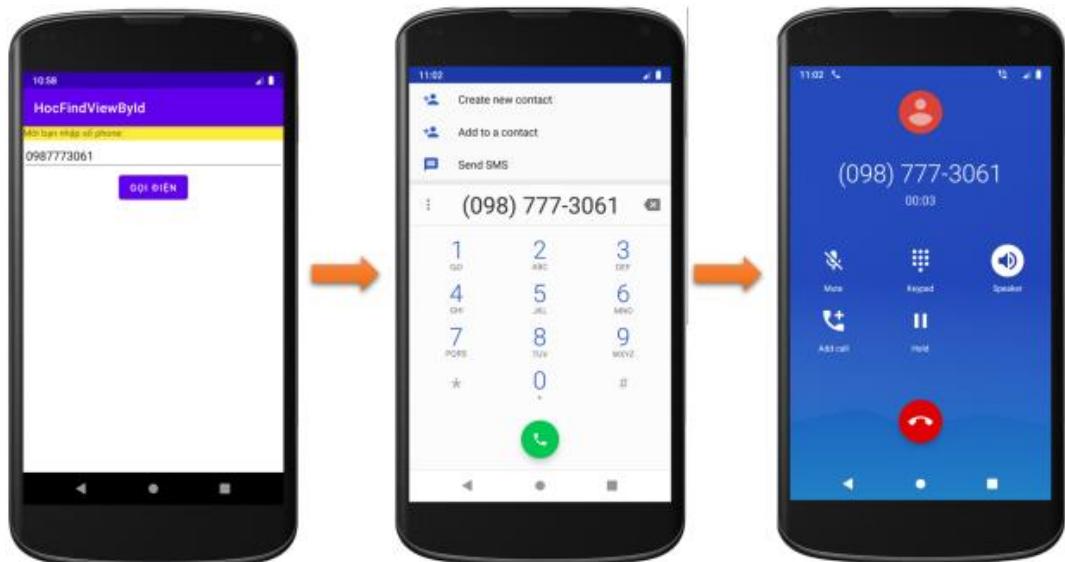


Lúc này MainActivity sẽ phát sinh ra một hàm xuLyGoiDien.

Bước 4: Vậy giờ ta bổ sung mã lệnh cho hàm xuLyGoiDien:

```
public void xuLyGoiDien(View view) {
    //lệnh findViewById để truy suất tới control có id tương ứng
    //truyền vào:
    EditText edtPhone=findViewById(R.id.edtPhoneNumber);
    //Lấy số phone ra:
    String phoneNumber=edtPhone.getText().toString();
    //Khai báo Intent để quay số gọi điện thoại
    Intent intent=new Intent(Intent.ACTION_DIAL);
    //Cú pháp gọi điện thoại
    Uri uri=Uri.parse("tel:"+phoneNumber);
    //gán uri cho intent:
    intent.setData(uri);
    //Gọi chức năng gọi điện thoại
    startActivity(intent);
}
```

Kết quả:



2.3. Anomous Listener

Là một loại gán sự kiện dạng nặc danh, loại này được ưa chuộng nhất khi lập trình xử lý sự kiện trong Android. Trong lớp xử lý nghiệp vụ (ActivityMain.java) các view sẽ được gán sự kiện tương tự như sau:

```
private void addEvents() {
    btnDo.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            txtMau.setBackgroundColor(Color.RED);
        }
    });
}
```

Ta chú ý mỗi lần đặt Id cho bất kỳ một View nào đó thì Id này sẽ tự động được phát sinh ra trong lớp R của Android. Vì vậy khi truy xuất tới View ta dùng R.id.idView.

Ta gọi hàm setOnClickListener cho View, bên trong hàm ta gõ new rồi nhấn tổ hợp phím Ctrl+Space: Android Studio sẽ tự động phát sinh ra sự kiện bên trong hàm cho chúng ta. Ta chú ý là loại sự kiện này không tái sử dụng được, mỗi View sẽ sở hữu sự kiện anomous riêng của mình.

Ví dụ: viết chương trình tô màu. Xử lý:

- Khi nhấn vào nút Tô màu đỏ → TextView sẽ chuyển sang màu đỏ.
- Khi nhấn lâu (Long click) vào nút Tô màu vàng → TextView sẽ chuyển sang màu vàng.
- Dùng kĩ thuật **Anomous Listener** để lắng nghe sự kiện.

B1 – Khởi động Android Studio → Start a new android project → Empty Activity → Name: AnomousListener.

B3 - Trong **MainActivity.java**, tại hàm **onCreate** → thêm 2 hàm
addControl();
addEvents();

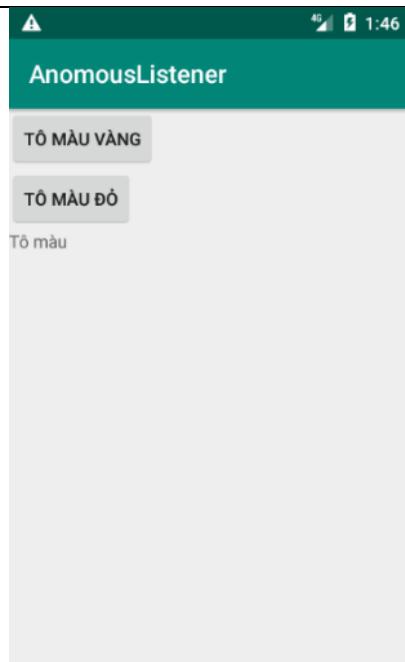
➔ Alt + Enter để phát sinh ra hàm mới.

```
protected void onCreate(Bundle savedInstanceState)
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    addControl();
    addEvents();
```

B2 – Sử dụng chức năng text để thiết kế. Trong **activity_main.xml** thêm vào các thành phần của giao diện (**không được sử dụng kéo thả**)

B4 – Trong `MainActivity.java`, tại hàm `addControl()`; cập nhật:

```
private void addControl () {
    btnDo = findViewById(R.id.btnDo);
    btnVang = findViewById(R.id.btnVang);
    txtMau = findViewById(R.id.txtMau);
}
```

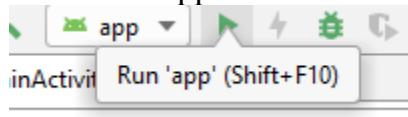


B5 – Trong `MainActivity.java`, tại hàm `addEvents()`; cập nhật:

```
private void addEvents () {
    btnDo.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            txtMau.setBackgroundColor(Color.RED);
        }
    });
}
```

Ctrl + Space

B6 – Run ‘app’



B7 – Quan sát kết quả, nhận xét so sánh với cách dùng On Click XML.

2.4. Variable as Listener

Variable as Listener là kỹ thuật gán sự kiện cho View thông qua tên biến, có nghĩa là chúng ta khai báo một biến có khả năng sinh sự kiện sau đó gán biến này cho View bất kỳ

Loại sự kiện này được dùng để chia sẻ cho các View trong màn hình, ví dụ bên dưới ta thấy `btnDangNhap`, `btnThoat` sử dụng một sự kiện `myClick`.

```

package com.example.myapplication;

import androidx.appcompat.app.AppCompatActivity;
import android.graphics.Color;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;

public class MainActivity extends AppCompatActivity {

    Button btnDangNhap, btnThoat;
    View.OnClickListener myClick=new
View.OnClickListener() {
        @Override
        public void onClick(View v) {
            if(view.equals(btnDangNhap)){
                //xử lý nút đăng nhập
            }else
                //xử lý nút thoát
        }
    };
}

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    addControls();
    addEvents();
}

private void addControls() {
    btnDangNhap=findViewById(R.id.btnDangNhap);
    btnThoat=findViewById(R.id.btnThoat);
}

private void addEvents() {
    btnDangNhap.setOnClickListener(myClick);
    btnThoat.setOnClickListener(myClick);
}
}

```

Ta chú ý biến myClick là một biến có khả năng sinh sự kiện (lắng nghe được người dùng nhấn vào view trên giao diện). Các view muốn sử dụng chỉ cần gọi hàm

setOnClickListener(myClick). Vì biến sự kiện này có khả năng chia sẻ tức là nhiều View có thể cùng sử dụng myClick. Vì vậy ta luôn phải kiểm tra xem View nào đang sử dụng biến sự kiện này thông qua lệnh equals.

3. BÀI TẬP:

Bài 1: Có bao nhiêu kỹ thuật gán sự kiện? Kể tên.

Bài 2: So sánh trường hợp sử dụng của các loại sự kiện

Bài 3: Viết chương trình giải phương trình bậc 1

GIẢI PHƯƠNG TRÌNH BẬC 1	
Nhập a:	<input type="text" value="nhập a ở đây"/>
Nhập b:	<input type="text" value="nhập b ở đây"/>
Kết quả:	<input type="text" value="x="/>
RESET	GIẢI
THOÁT	

Yêu cầu:

- Nút Reset: xóa dữ liệu cũ trên giao diện và cho focus lên ô nhập a.
- Nút Giải: nhấn vào để giải phương trình, kết quả hiển thị bên dưới.
- Nút Thoát: nhấn vào để thoát phần mềm

Bài 4: Viết chương trình minh họa các kỹ thuật gán sự kiện như hình bên dưới:

CÁC KIỂU LẬP TRÌNH SỰ KIỆN	
Số a:	<input type="text" value="nhập số a ở đây"/>
Số b:	<input type="text" value="nhập số b ở đây"/>
Kết quả:	<input type="text" value="hiển thị kết quả ở đây"/>
Tổng 2 số	
Hiệu 2 số	
Tích 2 số	
Thương 2 số	
Thoát chương trình	

Yêu cầu:

- Mỗi lần click chuột vào các Button thì sẽ thực hiện phép toán tương ứng. Ví dụ: khi click chuột vào “tổng 2 số” thì sẽ in ra kết quả của tổng 2 số.
- Riêng nút thoát chương trình: cho phép đóng chương trình.
- Tổng 2 số, hiệu 2 số → OnClick XML
- Tích 2 số, thương 2 số viết theo → Anonymous Listener.
- Thoát chương trình viết theo → Variable as Listener

Bài 5: Viết chương trình chuyển đổi độ C qua độ F và ngược lại.

FAHRENHEIT/CELSIUS DEGREE
CONVERTER

Fahrenheit:
nhập độ f ở đây

Celsius:
nhập độ c ở đây

CONVERT TO
CELSIUS CONVERT TO
FAHRENHEIT

CLEAR

Yêu cầu:

- Nút reset: xoá trống dữ liệu cũ và di chuyển focus tới ô nhập a.
- Nút Convert to Celsius: đổi độ sang độ C
- Nút Convert to Fahrenheit: đổi độ sang độ F

-HẾT-

LAB 4. TEXT VIEW, EDIT TEXT, BUTTON

1. MỤC ĐÍCH

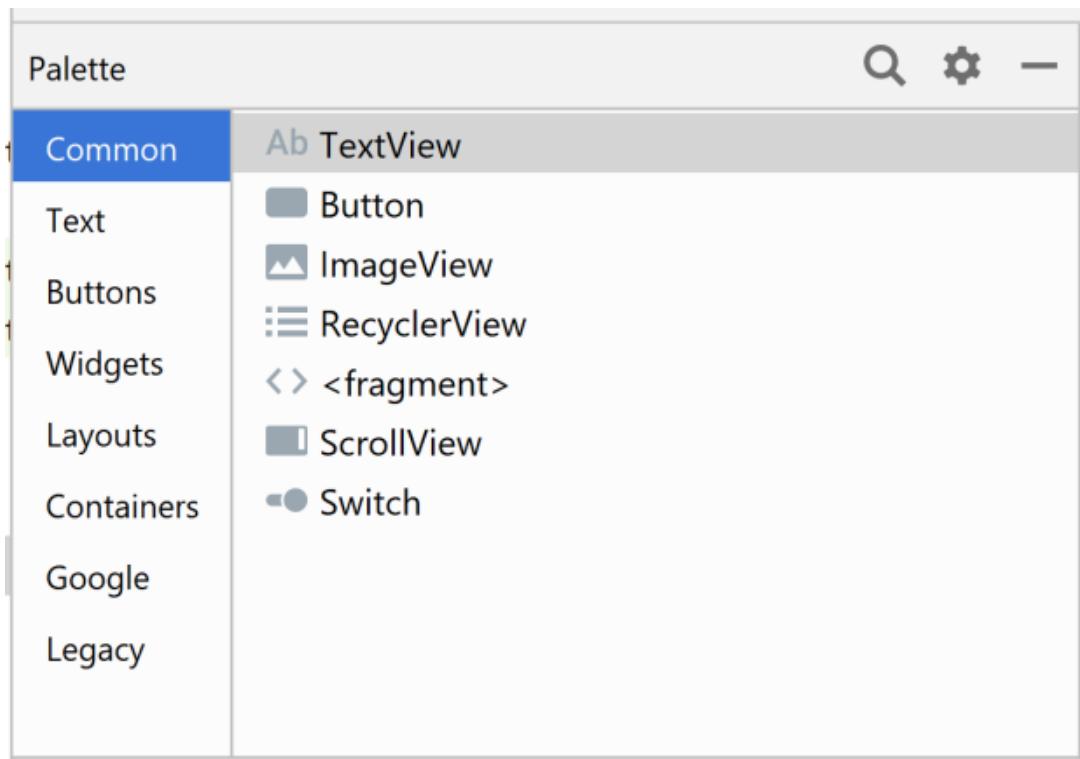
Android cung cấp rất nhiều View/Control để ta có thể tùy chọn thiết kế giao diện tương tác người dùng.

- TextView dùng để hiển thị dữ liệu,
- EditText để hiển thị và cho phép thay đổi thông tin,
- Button để ra lệnh,

2. LÝ THUYẾT:

2.1. TextView

TextView là view dùng để chỉ hiển thị dữ liệu không cho phép người sử dụng thay đổi dữ liệu. TextView thuộc nhóm Common và Text trong công cụ Palette:



Ta có thể kéo thả các View vào giao diện thiết kế một cách dễ dàng.

Bảng 4.1. Các thuộc tính/sự kiện quan trọng của TextView

Các thuộc tính/sự kiện	Mô tả
android:id	Id của TextView
android:layout_width	Độ rộng
android:layout_height	Độ cao
android:text	Chữ hiển thị lên giao diện
android:textColor	Màu chữ
android:textSize	Cỡ chữ
android:background	Màu nền
android:hint	Chữ gợi ý khi android:text rỗng

Để truy Xuất TextView ta cần đặt Id cho nó trong thanh công cụ Properties hoặc trong XML. Hàm findViewById đã được giới thiệu trước đó dùng để truy xuất View.

`TextView txtMessage=findViewById(R.id.txtMessage);`

Để thiết lập nội dung hiển thị ta có hai cách:

- Trong Java code (Activity):

`txtMessage.setText ("Khoa Công Nghệ Thông Tin");`

- Trong XML:

`android:text="Khoa Công Nghệ Thông Tin"`

Cấu trúc XML của textview

```
<TextView
    android:id="@+id/txtMessage"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:background="@android:color/holo_red_light"
    android:textColor="@android:color/holo_blue_light"
    android:text="Khoa Công Nghệ Thông Tin"
    android:textSize="30sp"
/>
```

Ngoài id, text thì TextView còn có các thuộc tính quan trọng khác như: layout_width(thiết lập độ rộng), layout_height(thiết lập chiều cao), background(màu

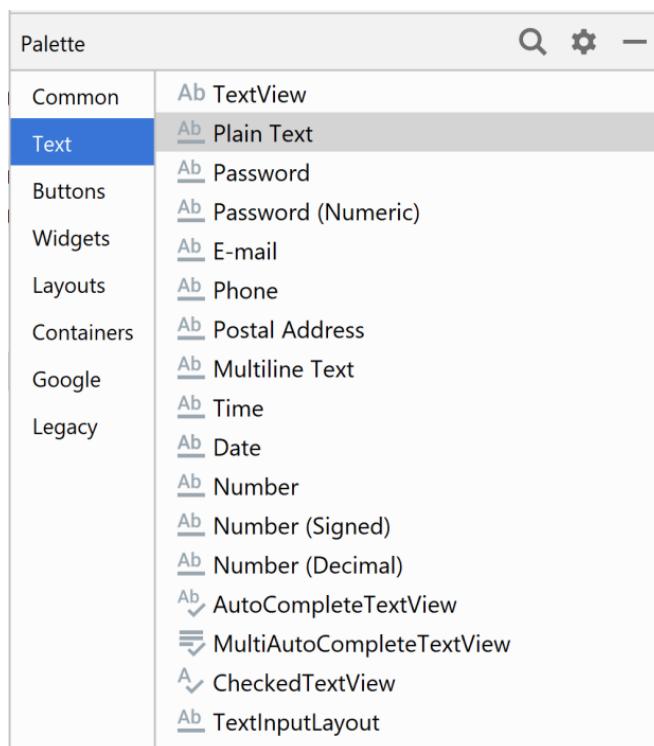
nền), textColor(màu chữ), textSize(cỡ chữ). Các thuộc tính cho TextView rất nhiều và được cung cấp sẵn trong công cụ Properties ta có thể vào đây để thao tác.

2.2. EditText

EditText là lớp kế thừa từ TextView, được dùng thay đổi nội dung text, chứa tất cả thuộc tính của TextView nên TextView hoạt động như thế nào thì EditText hoạt động như vậy, nằm trong nhóm Text của Palette.

Đặt Id nên bắt đầu là edt hoặc txt

Android đã tự động phân EditText thành nhiều loại khác nhau: Nhập chuỗi, số, điện thoại, email, mật khẩu ... dựa vào thuộc tính inputType. Nhờ thuộc tính inputType mà trải nghiệm người dùng được tốt hơn, điện thoại sẽ hiển thị bàn phím tương ứng với inputType mà ta cấu hình giúp người dùng thao tác một cách dễ dàng và nhanh chóng:



Bảng 4.2. EditText có nhiều thuộc tính quan trọng ta cần quan tâm:

Các thuộc tính/sự kiện	Mô tả
android:id	Id của EditText
android:layout_width	Độ rộng
android:layout_height	Độ cao
android:text	Chữ hiển thị lên giao diện
android:textColor	Màu chữ
android:textSize	Cỡ chữ

android:background	Màu nền
android:hint	Chữ gợi ý khi android:text rỗng
android:inputType	Thiết lập loại dữ liệu nhập để có bàn phím phù hợp

Ta thấy trong nhóm Text rất nhiều View (EditText), tùy vào nhu cầu sử dụng mà ta kéo thả các EditText phù hợp ra. Cấu trúc XML của Edittext:

```
<EditText
    android:id="@+id edtPassword"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:ems="1"
    android:textColor="@android:color/holo_blue_light"
    android:inputType="textPassword"
    android:text="hoilamgi"
    />
```

Thuộc tính inputType= “textPassword” giúp ta khi nhập dữ liệu vào thì phần mềm tự động mã hóa thành các ký tự * để bảo mật. Edittext ngoài ra còn sử dụng Hint để nhắc nhớ người dùng, ta có thể bổ sung android:hint = “nhập mật khẩu ở đây”, người dùng sẽ biết ô này dùng để làm gì.

Để truy xuất ta cũng dùng hàm findViewById:

```
EditText edtPassword=findViewById(R.id.edtPassword);
```

Để thay đổi nội dung hiển thị ta dùng:

```
edtPassword.setText("obama")
```

Để lấy nội dung người dùng nhập liệu:

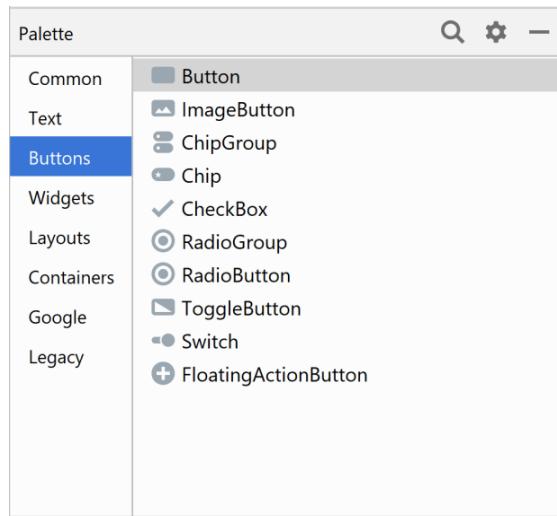
```
edtPassword.getText().toString()
```

2.3. Button

Đối tượng Button được xây dựng từ TextView, cho phép thể hiện các nội dung văn bản, hình ảnh – nhận và phản hồi tương tác nhấn từ người dùng.

Nên đặt Id cho Button bắt đầu bằng btn

Button thuộc nhóm Buttons hoặc Common trong công cụ Palette:



Các thuộc tính trong Button tương tự như trong TextView và EditText.

Bảng 4.3. Các thuộc tính/sự kiện quan trọng của Button

Các thuộc tính/sự kiện	Mô tả
android:id	Id của EditText
android:layout_width	Độ rộng
android:layout_height	Độ cao
android:text	Chữ hiển thị lên giao diện
android:textColor	Màu chữ
android:textSize	Cỡ chữ
android:background	Màu nền
android:onClick	Gán sự kiện cho Button

Cấu trúc XML của Button:

```
<Button
    android:id="@+id/btnDangNhap"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Đăng Nhập"
/>
```

Button thường dùng để cho phép người dùng ra lệnh thực thi một nghiệp vụ nào đó, sự ra lệnh này gọi là các sự kiện (event).

Button ngoài hiển thị dữ liệu còn có nhiệm vụ lắng nghe các sự kiện người dùng, hai cách nhanh nhất để lắng nghe sự kiện:

- Lắng nghe trong XML:

```
android:onClick="tenPhuongThuc"
```

- Lắng nghe trong java code:

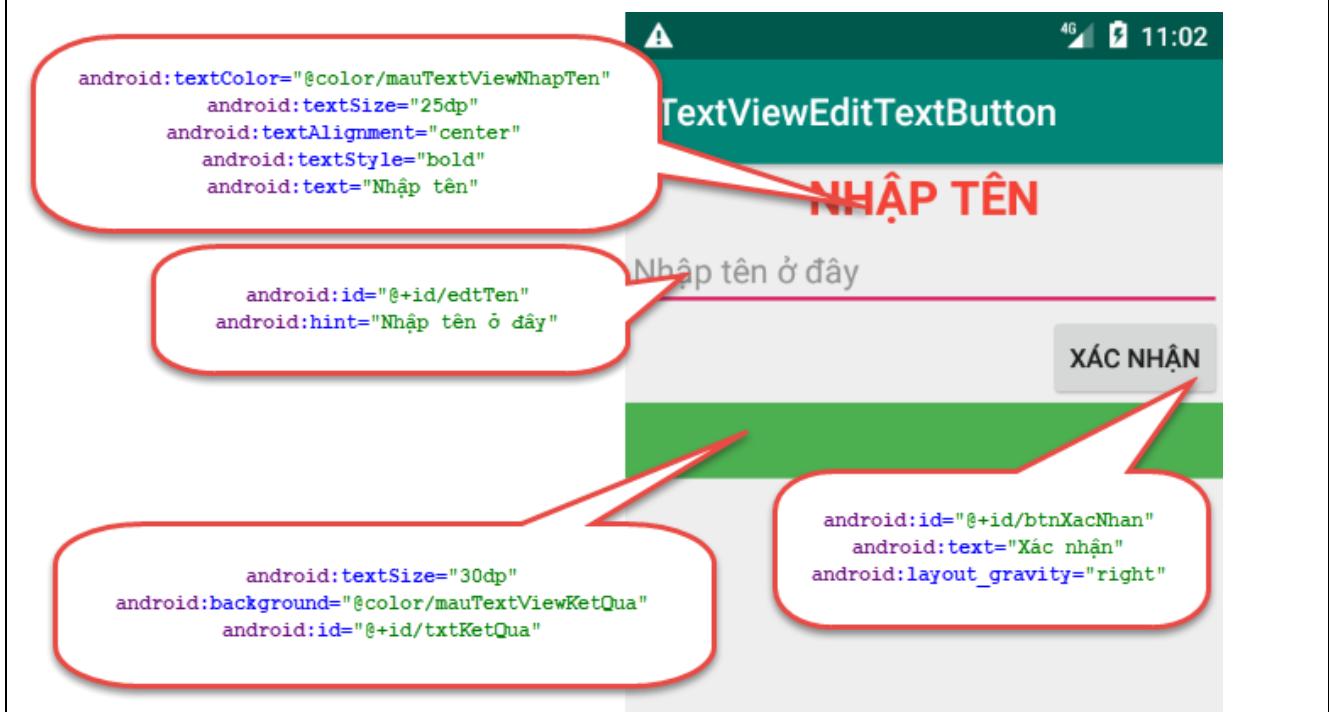
```
btnDangNhap.setOnClickListener(new OnClickListener() {
    @Override
    public void onClick(View v) {
        xuLyDangNhap();
    }
});
```

Ví dụ: Thiết kế giao diện nhập tên và xác nhận

- **Yêu cầu:** khi nhấn vào nút xác nhận sẽ lấy giá trị từ EditText truyền xuống TextView.

B1 – Khởi động Android Studio → Start a new android project → Empty Activity → Name: TextViewEditTextButton

B2 – Sử dụng text trong activity_main.xml thêm vào các thành phần của giao diện



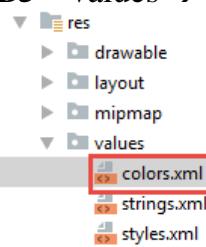
B3 – Thêm giá trị màu cho nút nhập tên

:textColor="@color/mauTextviewNhapTen"
:textSize="25dp"
:textAlignment="center"
:textStyle="bold"

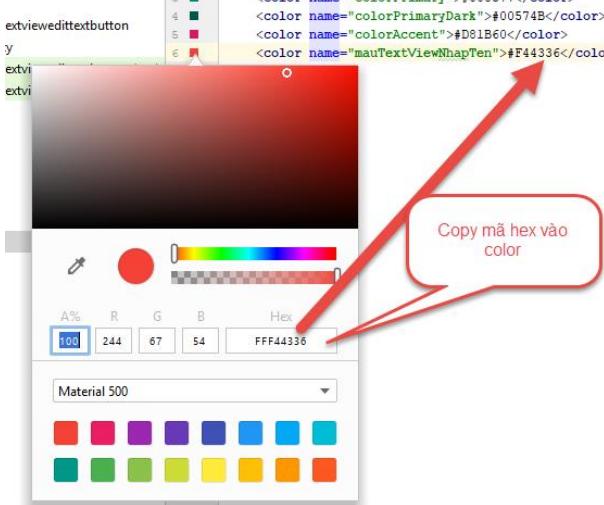
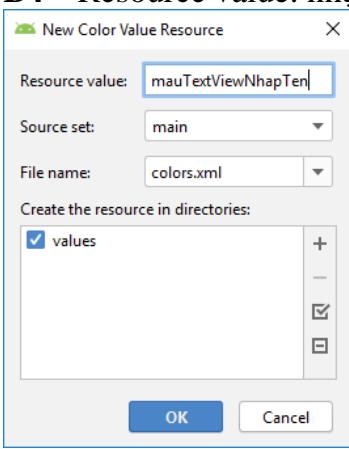
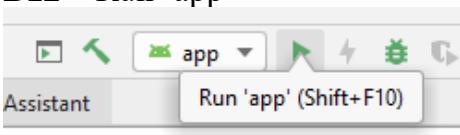
ALT + ENTER

Create color resource file 'mau1'
Create color value resource 'mau'

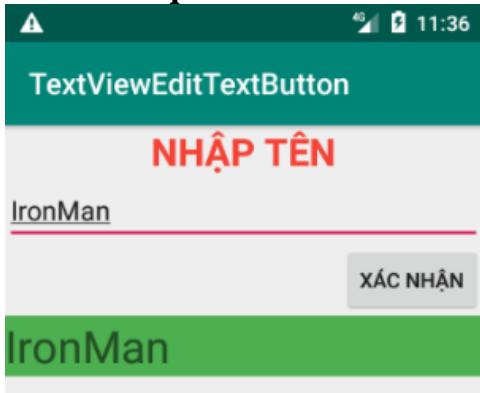
B5 – values → colors.xml



B6 – Trong colors.xml → bấm vào biểu tượng màu kế dòng 6, chọn màu.

	<p>B4 – Resource value: nhập bất kì → OK</p> 
<p>B7 – Trong activity_main.xml, tại nút Xác nhận, thêm sự kiện “xuLyXacNhan”</p> <pre> android:layout_width="match_parent" android:layout_height="wrap_content" <Button android:onClick="xuLyXacNhan" android:id="@+id btnXacNhan" android:text="Xác nhận" android:layout_gravity="right" ></pre> <p style="text-align: center;">ALT + ENTER</p>	<p>B8 – Trong MainActivity.java, khai báo các biến:</p> <p>EditText edtTen; Button btnXacNhan; TextView txtKetQua;</p>
<p>B9: Trong MainActivity.java, tại hàm OnCreate, thêm hàm addControl(): → ALT + ENTER để khởi tạo hàm.</p> <pre>protected void onCreate(Bundle savedInstanceState) { super.onCreate(savedInstanceState) setContentView(R.layout.activity_main); addControl(); }</pre> <p style="text-align: center;">ALT + ENTER</p> <p>Create method 'addControl' in 'MainActivity' Rename reference</p>	<p>B10 – Trong MainActivity.java, tại hàm addControl(); khai báo các Control sau:</p> <p>edtTen = findViewById(R.id.edtTen); txtKetQua = findViewById(R.id.txtKetQua); btnXacNhan = findViewById(R.id.btnXacNhan);</p>
<p>B11 – Trong MainActivity.java, tại hàm xuLyXacNhan(), cập nhật đoạn code xử lý sự kiện sao cho khi bấm vào nút Xác nhận sẽ lấy dữ liệu từ ô EditText truyền xuống ô TextView:</p> <pre>txtKetQua.setText(edtTen.getText() .toString());</pre>	<p>B12 – Run ‘app’</p> 

B13 – Kết quả:



3. BÀI TẬP:

Bài 1: Viết chương trình tính chỉ số cân đối cơ thể (Body Mass Index – BMI), giao diện như hình dưới đây:



CHƯƠNG TRÌNH TÍNH CHỈ SỐ BMI	
Nhập tên:	IronMan
Chiều cao:	170
Cân nặng:	65
TÍNH BMI	
BMI:	<u>20.5</u>
Kết luận:	<u>Bạn bình thường</u>

Gọi W là khối lượng của một người (tính bằng kilogram) và H là chiều cao của người đó (tính bằng mét), chỉ số khối cơ thể được tính theo công thức:

$$BMI = \frac{W}{H^2}$$

Phân loại để đánh giá như sau:

- BMI<18: người gầy
- BMI=18-24,9: người bình thường
- BMI=25-29,9: người béo phì độ I
- BMI=30-34,9: người béo phì độ 2
- BMI>35: người béo phì độ 3

Bài 2: Viết phần mềm chuyển đổi năm dương lịch qua năm âm lịch, giao diện như dưới đây

CHUYỂN ĐỔI NĂM DƯƠNG LỊCH

Năm dương lịch	<input type="text"/>
Năm âm lịch	<input type="text"/>
<input type="button" value="CHUYỂN ĐỔI"/> <input type="button" value="THOÁT"/>	

Khi người dùng chọn nút chuyển đổi thì chương trình sẽ tự động hiển thị năm âm lịch, ví dụ nhập năm 2013 thì năm âm lịch là Quý Tỵ.

Hướng dẫn cách làm:

Năm Âm = Can + Chi

Can = năm dương % 10:

0	1	2	3	4	5	6	7	8	9
Canh	Tân	Nhâm	Quý	Giáp	Ất	Bính	Đinh	Mậu	Kỷ

Chi = năm dương % 12:

0	1	2	3	4	5	6	7	8	9	10	11
Thân	Dậu	Tuất	Hợi	Tý	Sửu	Dần	Mẹo	Thìn	Tỵ	Ngọ	Mùi

Bài 3: Viết chương trình giải phương trình bậc 2 với giao diện như sau:

GIẢI PHƯƠNG TRÌNH BẬC 2

Nhập a:	<input type="text" value="nhập a ở đây"/>
Nhập b:	<input type="text" value="nhập b ở đây"/>
Nhập c:	<input type="text" value="nhập c ở đây"/>
Kết quả:	<input type="text" value="x="/>
<input type="button" value="RESET"/> <input type="button" value="GIẢI"/>	
<input type="button" value="THOÁT"/>	

Yêu cầu:

- Nút reset: xoá trống dữ liệu cũ và di chuyển focus tới ô nhập a.
- Nút giải: tiến hành giải phương trình bậc 2 và hiển thị kết quả xuống bên dưới.
- Nút thoát: thoát phần mềm.

LAB 5. CHECKBOX, RADIO BUTTON, IMAGE BUTTON, IMAGE VIEW

1. MỤC ĐÍCH:

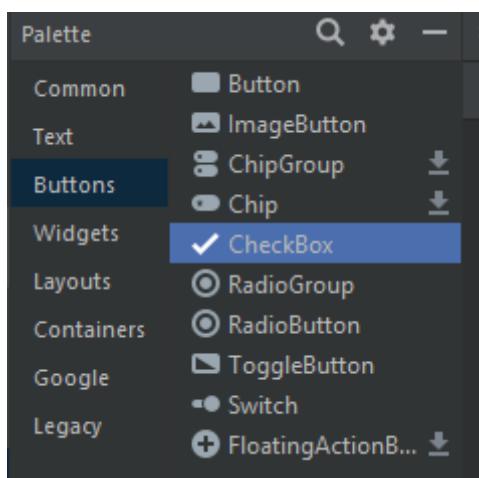
- CheckBox để chọn nhiều lựa chọn.
- RadioButton để chọn một lựa chọn.
- ImageButton là dạng button có hình ảnh, dùng để ra lệnh.
- ImageView để hiển thị hình ảnh.

2. LÝ THUYẾT:

2.1. Checkbox:

Checkbox là view cho phép người dùng chọn nhiều lựa chọn, nên đặt ID bắt đầu với chk.

Check box thuộc nhóm Button trong công cụ Palette:



Các thuộc tính sự kiện quan trọng của checkbox:

Các thuộc tính/sự kiện	Mô tả
Android:id	Id của checkbox
Android:layout_width	Độ rộng
Android:layout_height	Độ cao
Android:text	Chữ hiển thị lên giao diện
Android:textColor	Màu chữ
Android:textSize	Cỡ chữ
Android:background	Màu nền
Android:checked	Thiết lập chọn/bỏ chọn checkbox (true/false)

Cấu trúc xml của checkbox:

```
<CheckBox
    android:id="@+id/checkBox"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="CheckBox" />
```

Để kiểm tra xem checkbox nào được người sử dụng chọn có hai cách:

- Kiểm tra bằng hàm isChecked()
- Hoặc kiểm tra sự kiện khi người dùng lựa chọn ngay trên checkbox

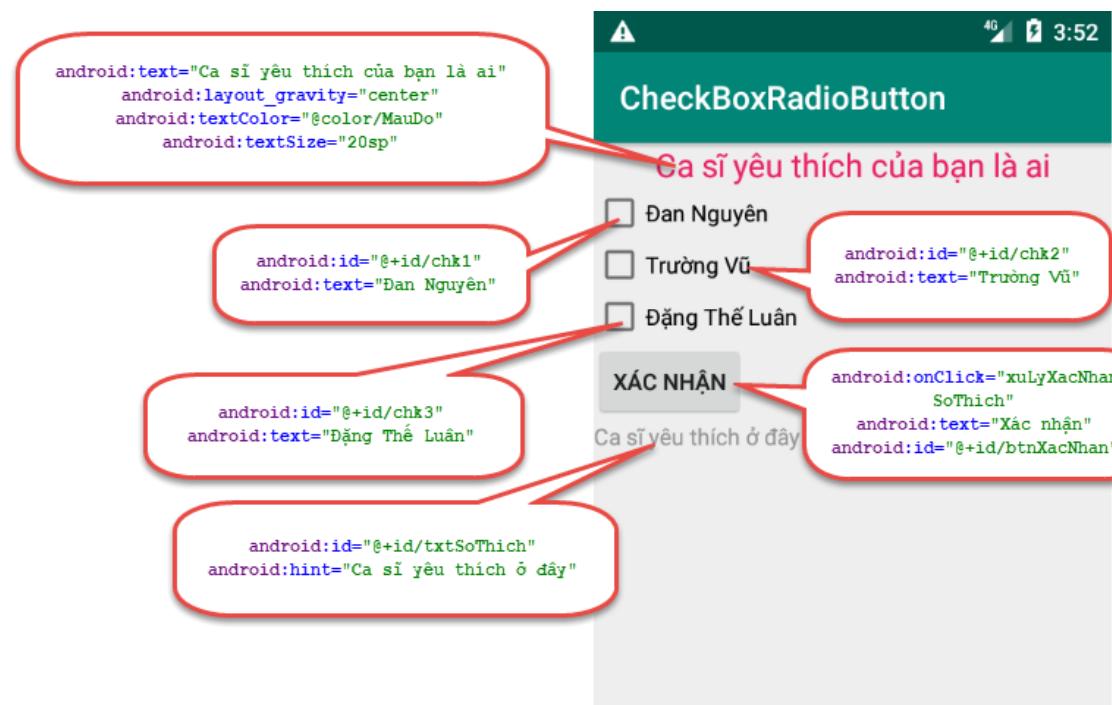
Ví dụ: thiết kế chương trình chọn nhiều lựa chọn sử dụng checkbox

- **Yêu cầu:**

- Khi check vào ô checkbox bất kì và bấm nút Xác nhận, kết quả sẽ được truyền xuống ô TextView.
- Sử dụng On Click XML để xử lý sự kiện.

B1 – Khởi động Android Studio → Start a new android project → Empty Activity → Name: **CheckBoxRadioButton**

B2 – Sử dụng text trong **activity_main.xml** thêm vào các thành phần của giao diện



B3 – Trong **activity_main.xml**, tại nút **Xác nhận**, thêm sự kiện “**xuLyXacNhanSoThich**

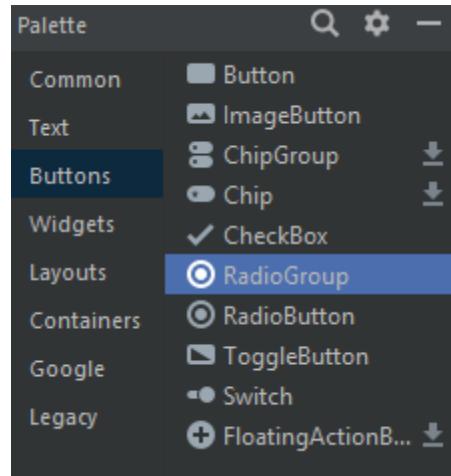
B4 – Trong **MainActivity.java**, khai báo các biến:

CheckBox **chk1**, **chk2**, **chk3**;
TextView **txtSoThich**;

<p>B5 - Trong MainActivity.java, tại hàm OnCreate, thêm hàm addControl(): → ALT + ENTER để khởi tạo hàm.</p> <pre>protected void onCreate(Bundle savedInstanceState) { super.onCreate(savedInstanceState); setContentView(R.layout.activity_main); addControls(); }</pre> <p>B6 – Trong MainActivity.java, tại hàm addControl(); khai báo các Control sau:</p> <pre>chk1 = findViewById(R.id.chk1); chk2 = findViewById(R.id.chk2); chk3 = findViewById(R.id.chk3); txtSoThich = findViewById(R.id.txtSoThich);</pre>	
<p>B7 – Trong MainActivity.java, tại hàm xuLyXacNhanSoThich(), cập nhật đoạn code xử lý sự kiện.</p> <pre>public void xuLyXacNhanSoThich(View view) { String msg = ""; if(chk1.isChecked()) { msg+=chk1.getText().toString() +"\n"; } if (chk2.isChecked()) { msg+=chk2.getText().toString() +"\n"; } if (chk3.isChecked()) { msg+=chk3.getText().toString() +"\n"; } txtSoThich.setText(msg); }</pre>	<p>B8 – Run ‘app’</p> <p>B9 – Kết quả:</p>

2.2. RadioButton

RadioButton là View cho phép người dùng chọn một lựa chọn, nên đặt Id bắt đầu với rad. Để sử dụng được RadioButton trước tiên ta phải vào Buttons kéo RadioGroup ra trước, nó dùng để gom nhóm các RadioButton. Sau đó ta mới kéo RadioButton ra



Các thuộc tính, sự kiện quan trọng của RadioButton:

Các thuộc tính/sự kiện	Mô tả
Android:id	Id của RadioButton
Android:layout_width	Độ rộng
Android:layout_height	Độ cao
Android:text	Chữ hiển thị lên giao diện
Android:textColor	Màu chữ
Android:textSize	Cỡ chữ
Android:background	Màu nền
Android:checked	Thiết lập chọn/bỏ chọn RadioButton (true/false)

Cấu trúc XML của Radio Button

```

RadioGroup
    android:id="@+id/radioGroup"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:orientation="vertical">
    <RadioButton
        android:id="@+id/radNam"
        android:text="Nam"
        android:layout_width="match_parent"
        android:layout_height="wrap_content">
    </RadioButton>
</RadioGroup>
  
```

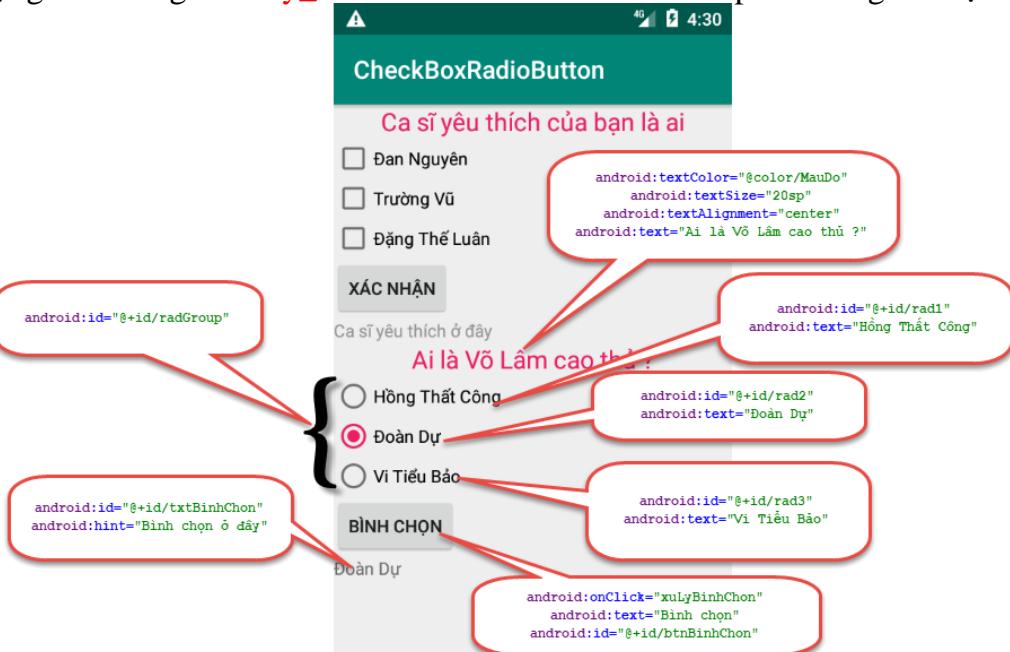
RadioGroup có thuộc tính android:orientation để sắp xếp các View bên trong theo hai hướng: vertical (nằm dọc), horizontal (nằm ngang).

Để kiểm tra xem người sử dụng lựa chọn RadioButton nào ta có ba cách:

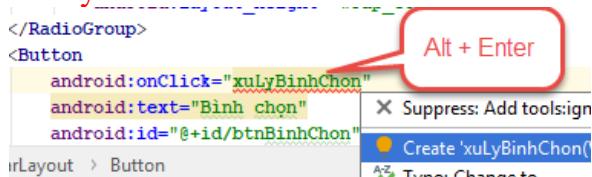
- Kiểm tra hàm isChecked() cho từng RadioButton
- Lấy id từ RadioGroup
- Kiểm tra sự kiện khi người dùng lựa chọn ngay trên RadioGroup.

Ví dụ: chương trình bình chọn sử dụng RadioButton (dùng tiếp project ở ví dụ 2.1)

B1 – Sử dụng text trong activity_main.xml thêm vào các thành phần của giao diện



B2 – Trong activity_main.xml, tại nút Bình chọn, thêm sự kiện On Click XML “xuLyBinhChon”



B5 – Trong MainActivity.java, tại hàm xuLyBinhChon();, cập nhật đoạn code xử lý sự kiện.

```
public void xuLyBinhChon(View view)
{
    if (rad1.isChecked())
    {
        txtBinhChon.setText(rad1.getText());
    }
    else if (rad2.isChecked())
    {
        ...
    }
}
```

B3 - Trong MainActivity.java, khai báo các biến:

```
TextView txtBinhChon;
RadioButton rad1, rad2,
rad3;
```

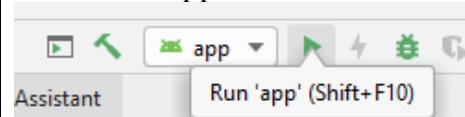
B4 – Trong MainActivity.java, tại hàm addControl(); khai báo các Control sau:

```
rad1 = findViewById(R.id.rad1);
;
rad2 = findViewById(R.id.rad2);
;
rad3 = findViewById(R.id.rad3);
;
txtBinhChon = findViewById(R.id.txtBinhChon);
```

```

{
    txtBinhChon.setText(rad2.getText());
}
else if (rad3.isChecked())
{
    txtBinhChon.setText(rad3.getText());
}
}

```

B6 – Run ‘app’**B7 – Kết quả:**

Ngoài cách sử dụng ***if else*** như trong Bước 5, chúng ta còn có thể sử dụng ***RadioGroup*** để lấy id của ***RadioButton*** như sau:

B5b – Trong **MainActivity.java**, tại hàm **xuLyBinhChon()**, cập nhật đoạn code xử lý sự kiện.

```

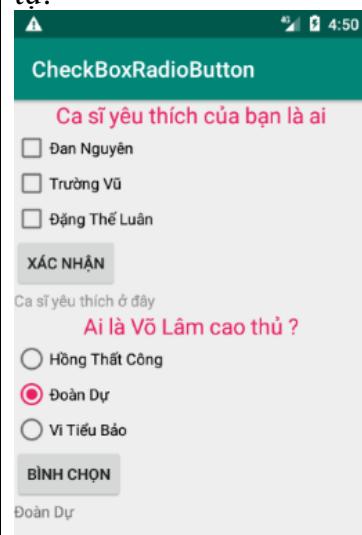
public void xuLyBinhChon(View
view)
{
    RadioGroup radGroup = =
findViewById(R.id.radGroup);
    int id = =
radGroup.getCheckedRadioButtonI
d();
    if (id>0)
    {
        RadioButton radioButton =
findViewByI
d(id);

    txtBinhChon.setText(radioButton
.getText());
    }
}

```

B6b – Run ‘app’

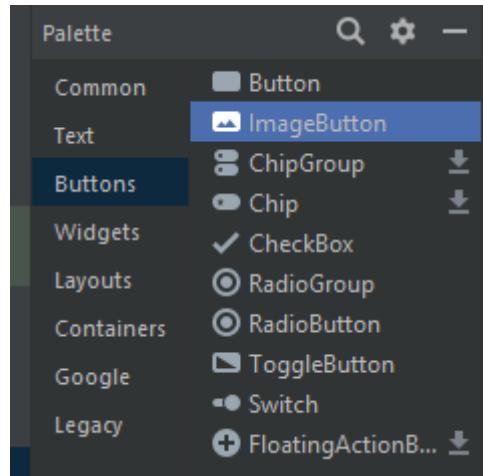
B7b- Kiểm tra thấy kết quả tương tự.



2.3. ImageButton

ImageButton là một Button có thể chứa hình ảnh, có thể đặt prefix là btn hoặc imgBtn. ImageButton hoạt động hoàn toàn giống Button, chỉ khác ở chỗ ImageButton có hiển thị hình ảnh.

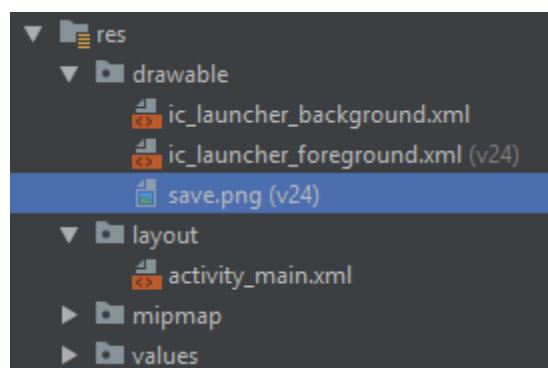
ImageButton thuộc nhóm Buttons trong công cụ Palette:



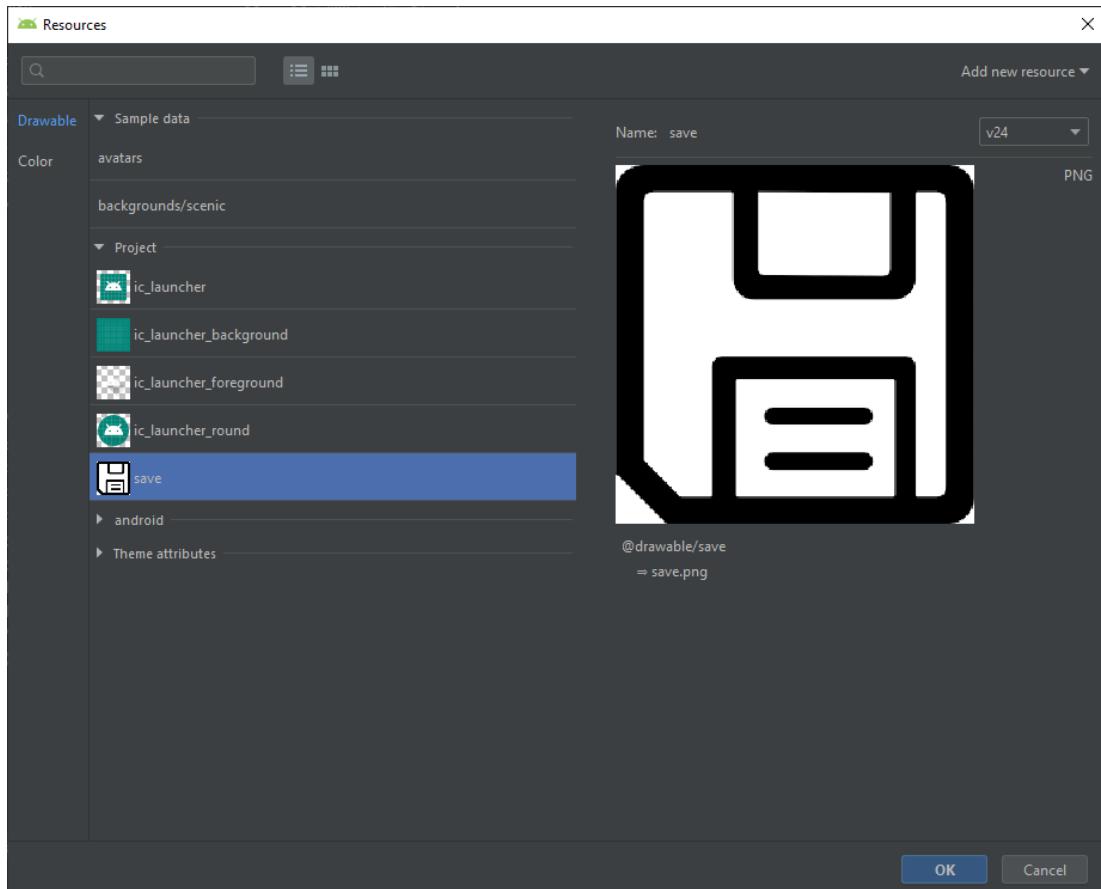
Các thuộc tính, sự kiện quan trọng của ImageButton:

Các thuộc tính/sự kiện	Mô tả
Android:id	Id của RadioButton
Android:layout_width	Độ rộng
Android:layout_height	Độ cao
App:srcCompat	Thiết lập hình ảnh cho ImageButton

Để sử dụng ImageButton, trước tiên ta phải dán hình ảnh vào thư mục drawable trong ứng dụng (chú ý quy tắc đặt tên hình giống như tên biến, nếu sai sẽ bị báo lỗi).



Khi kéo thả ImageButton vào giao diện, chương trình sẽ hiển thị cửa sổ yêu cầu ta chọn hình ảnh cho ImageButton



Cấu trúc XML của ImageButton:

```
<ImageButton
    android:id="@+id/imageButton"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    app:srcCompat="@drawable/save" />
```

Các thuộc tính, sự kiện của ImageButton giống Button.

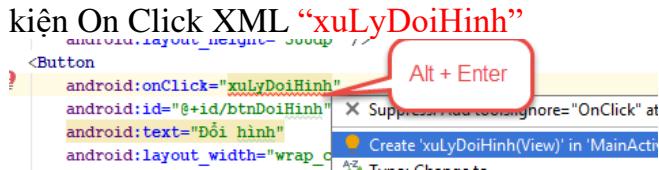
Ví dụ: thiết kế chương trình khi bấm vào nút Đổi hình, hình h1 sẽ chuyển thành hình h2 và ngược lại. Khi bấm vào ImageButton hoặc imageView sẽ đóng chương trình.

B1 – Khởi động Android Studio → Start a new android project → Empty Activity → Name: ImageButtonImageView

B2 – Sử dụng text trong activity_main.xml thêm vào các thành phần của giao diện



B2 – Trong `activity_main.xml`, tại nút **Đổi hình**, thêm sự kiện On Click XML “`xuLyDoiHinh`”



B3 – Trong `activity_main.xml`, tại `imageButton` và `imageView`, thêm sự kiện On Click XML “`xuLyDong`”



B4 – Trong `MainActivity.java`, tại hàm `xuLyDoiHinh();`, cập nhật đoạn code xử lý sự kiện.

```
public void xuLyDoiHinh(View view) {
    ImageView imgHinh = findViewById(R.id.imgHinh);
    if (imgHinh.getTag() == null || imgHinh.getTag().equals("h1"))
        imgHinh.setImageResource(R.drawable.h2);
    imgHinh.setTag("h2");
```

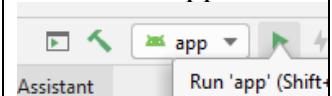
B5 - Trong `MainActivity.java`, tại hàm `xuLyDong();`, cập nhật đoạn code.

```
public void xuLyDong(View view) {
    finish();}
```

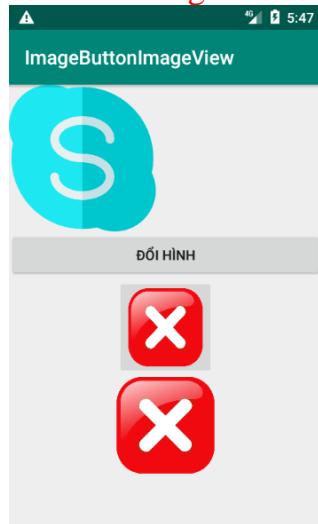
```

    }
else
{
    imgHinh.setImageResource(R.drawable.h1);
    imgHinh.setTag("h1");
}
}

```

B6 – Run ‘app’

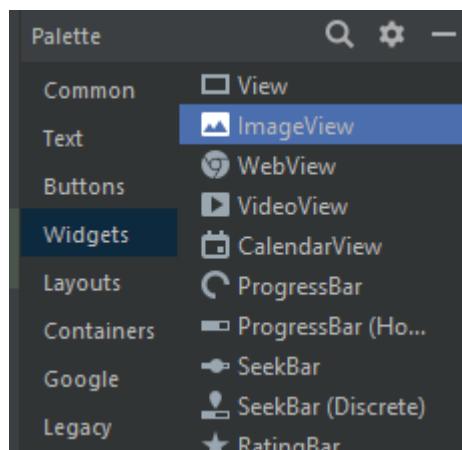
B7 – Kiểm tra: khi bấm vào nút **Đổi hình**, h1 sẽ chuyển thành h2 và ngược lại. Khi bấm vào **imageButton** hoặc **imaveView** sẽ đóng chương trình.



2.4. ImageView

ImageView dùng để hiển thị hình ảnh, Id nên đặt là img. Giống ImageButton, ta cần phải có hình ảnh trong thư mục drawable để gán cho ImageView. Ta chọn hình ảnh mong muốn rồi dán vào thư mục drawable.

Sau khi có hình ảnh, ta vào nhóm Images trong công cụ Palette để kéo thả ImageView ra màn hình.



Cấu trúc xml của ImageView:

```
<ImageView
    android:id="@+id/imageView"
    android:layout_width="wrap_content"
```

```
    android:layout_height="wrap_content"
    tools:srcCompat="@drawable/save" />
```

3. BÀI TẬP:

Bài 1: Viết phần mềm xử lý thông tin với màn hình như sau:

THÔNG TIN CÁ NHÂN

Nhập tên: _____

CMND: _____

Bằng cấp

Trung cấp Cao đẳng Đại học

Sở thích

Đọc báo Đọc sách Lập trình

Kết quả:

Xác nhận Reset

About Thoát

Xử lý:

- Nút xác nhận: kiểm tra bằng cấp, sở thích để xuất thông tin cùng tên và chứng minh nhân dân ra textView ở khu vực kết quả.
- Nút reset: reset các thông tin trên form về mặc định.
- Nút about: hiển thị thông tin tác giả lên alert dialog.
- Nút thoát: thoát chương trình

Bài 2: Xây dựng chương trình máy tính

- Thiết kế đúng giao diện (tùy ý sử dụng loại layout).
- Thực hiện các phép toán cộng, trừ, nhân, chia trên hai số nguyên với nhau.
- Kết quả hiển thị có dạng: $15 + 20 = 35$



LAB 6. CÁC CỬA SỔ THÔNG BÁO THƯỜNG DÙNG

1. MỤC ĐÍCH

Android cung cấp nhiều cửa sổ thông báo người dùng: Toast, AlertDialog, CustomDialog và Notification. Mục đích sử dụng của từng cửa sổ này sẽ khác nhau, tùy vào từng trường hợp mà ta sử dụng.

2. LÝ THUYẾT

2.1. Toast

Toast có thể được tạo và hiển thị trong Activity hoặc trong Service. Không cho phép người sử dụng tương tác, hiển thị sau khoảng thời gian nào đó sẽ tự đóng lại.

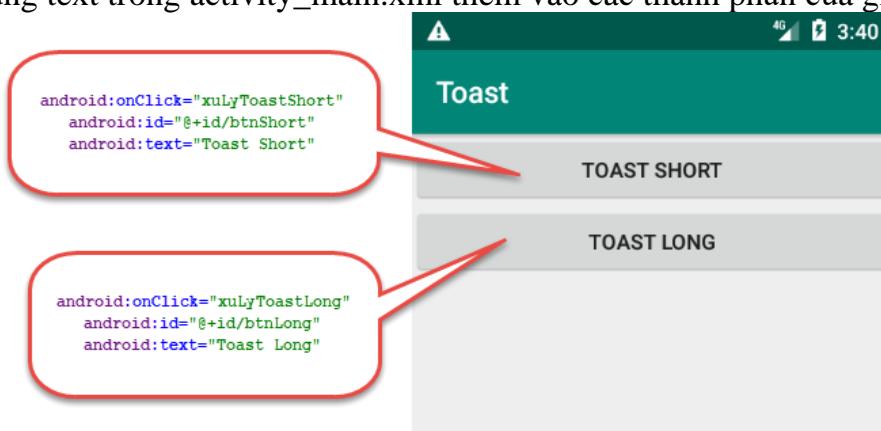
Có hai giá trị mặc định (ta nên sử dụng hai giá trị này, không nên gõ con số cụ thể vào): hằng số Toast.LENGTH_SHORT hiển thị khoảng 1.5 giây, Toast.LENGTH_LONG hiển thị khoảng 3.5 giây.

```
public void xuLyToastShort(View view) {
    Toast toast=Toast.makeText(MainActivity.this,"Đây là
Toast Short",Toast.LENGTH_LONG);
    toast.show();
}
```

Ví dụ: thiết kế giao diện thông báo sử dụng toast. Khi bấm vào nút Toast Short → xuất hiện 1 đoạn Toast nội dung “Đây là Toast Short”. Tương tự với nút Toast Long.

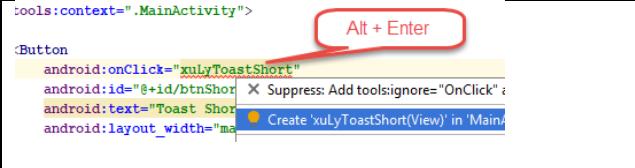
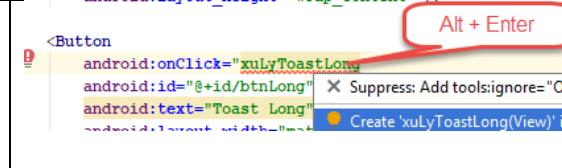
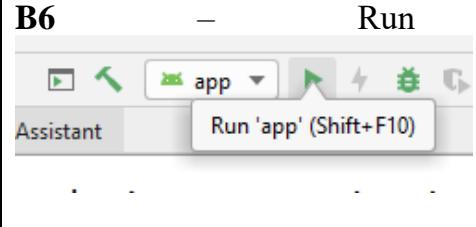
B1 – Khởi động Android Studio → Start a new android project → Empty Activity → Name: ToastAlertDialog

B2 – Sử dụng text trong activity_main.xml thêm vào các thành phần của giao diện



B2 – Trong activity_main.xml, tại nút ToastShort thêm sự kiện On Click XML “xuLyToastShort”

B3 – Trong activity_main.xml, tại nút ToastLong thêm sự kiện On Click XML “xuLyToastLong”

 <p>Alt + Enter</p> <pre><code>tools:context=".MainActivity"> <Button android:onClick="xulyToastShort" android:id="@+id btnShort" android:text="Toast Short" android:layout_width="ma</code></pre> <p>Create 'xulyToastShort(View)' in 'MainActivity'</p>	 <p>Alt + Enter</p> <pre><code><Button android:onClick="xulyToastLong" android:id="@+id btnLong" android:text="Toast Long" android:layout_width="ma</code></pre> <p>Create 'xulyToastLong(View)' in 'MainActivity'</p>
<p>B4 – Trong MainActivity.java, tại hàm xuLyToastShort();, cập nhật đoạn code xử lý sự kiện.</p> <pre><code>public void xuLyToastShort (View view) { Toast toast=Toast.makeText(MainActivity.this, "Đây là Toast Short", Toast.LENGTH_LONG); toast.show (); }</code></pre>	<p>B5 – Trong MainActivity.java, tại hàm xuLyToastLong();, cập nhật đoạn code xử lý sự kiện. Lưu ý: trong hàm xuLyToastLong, chúng ta dùng cách không cần khai báo biến để thực hiện. Quan sát sự khác nhau giữa 2 cách viết ở B4 và B5. Nhận xét.</p> <pre><code>public void xuLyToastLong (View view) { Toast.makeText(MainActivity.this, "Đây là Toast Long", Toast.LENGTH_SHORT) .show (); }</code></pre>
<p>B6 – Run ‘app’</p>  <p>Run 'app' (Shift+F10)</p>	

2.2. AlertDialog

AlertDialog là cửa sổ hiển thị và cho phép người dùng tương tác với hệ thống. Ta thường dùng cửa sổ này để nhận các quyết định từ phía người dùng.

Để sử dụng được AlertDialog ta khởi tạo một đối tượng Builder như sau:

```
AlertDialog.Builder builder = new
AlertDialog.Builder(MainActivity.this);
```

Đối tượng builder này cung cấp cho ta các phương thức:

- setTitle: thiết lập tiêu đề cho Dialog.
- setMessage: thiết lập nội dung cho Dialog
- setIcon: để thiết lập Icon
- setPositiveButton, setNegativeButton: thiết lập hiển thị nút chọn cho Dialog.
- Create(): để tạo Dialog.
- Show(): để hiển thị Dialog

```
- private void taoCuaSoHoiThoat() {
    AlertDialog.Builder builder = new
    AlertDialog.Builder(MainActivity.this);
    builder.setTitle("Xác nhận Thoát");
    builder.setIcon(android.R.drawable.ic_dialog_info);
    builder.setMessage("Bạn muốn thoát không?");
```

```

//thiết lập các nút lệnh để người dùng tương tác
builder.setPositiveButton("Có chứ", new
DialogInterface.OnClickListener() {
    @Override
    public void onClick(DialogInterface dialogInterface, int i) {
        finish();
    }
});
builder.setNegativeButton("Không", new
DialogInterface.OnClickListener() {
    @Override
    public void onClick(DialogInterface dialogInterface, int i) {
        dialogInterface.dismiss();
    }
});
//tạo cửa sổ dialog:
AlertDialog dialog=builder.create();
//không cho người dùng nhấn ở ngoài để đóng cửa sổ
dialog.setCancelableOnTouchOutside(false);
//hiển thị cửa sổ này lên:
dialog.show();
}

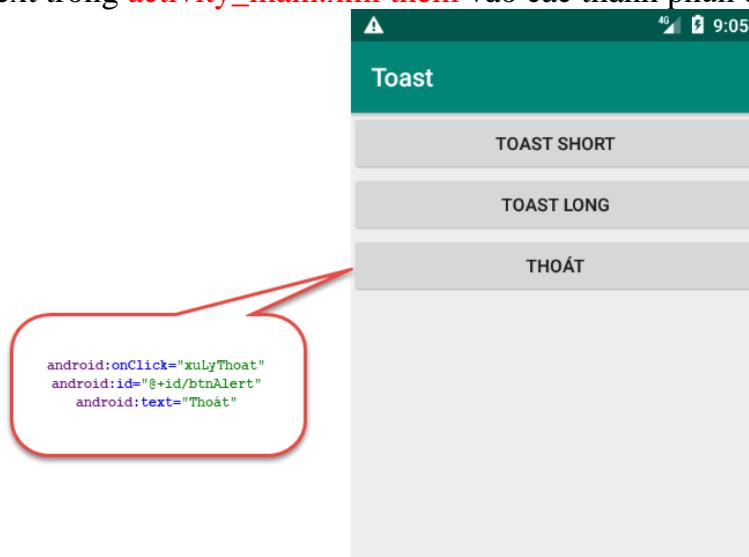
```

Ta chú ý lúc coding nhớ tận dụng tổ hợp phím Ctrl+Space để Android Studio tự động thêm các thư viện vào phần mềm để không bị báo lỗi.

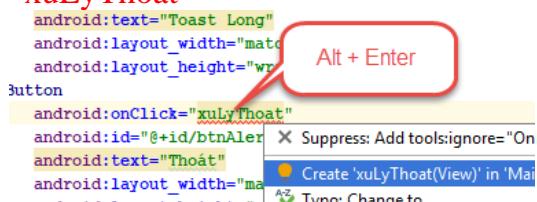
Ví dụ: Tiếp tục sử dụng Project **ToastAlertDialog** trong **Bài 1** để thực hiện.

- **Yêu cầu:** Khi bấm vào nút Thoát → xuất hiện 1 hộp thoại xác nhận có muốn thoát hay không.

B1 - Sử dụng text trong **activity_main.xml** thêm vào các thành phần của giao diện



B2 - Trong `activity_main.xml`, tại nút Thoát thêm sự kiện On Click XML “xuLyThoat”

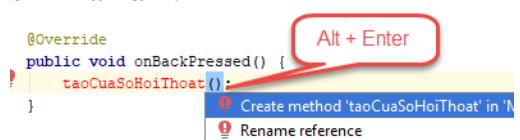


B4 - Trong `MainActivity.java`, tại hàm `taoCuaSoThoat()`, cập nhật:

```

private void taoCuaSoHoiThoat () {
    AlertDialog.Builder builder = new AlertDialog.Builder(MainActivity.this);
    builder.setTitle("Xác nhận Thoát");
    builder.setIcon(android.R.drawable.ic_dialog_info);
    builder.setMessage("Bạn muốn thoát hẳn?");
    //thiết lập các nút lệnh để người dùng tương tác
    builder.setPositiveButton("Có chứ", new DialogInterface.OnClickListener() {
        @Override
        public void onClick(DialogInterface dialogInterface, int i) {
            finish();
        }
    });
    builder.setNegativeButton("Không", new DialogInterface.OnClickListener() {
        @Override
        public void onClick(DialogInterface dialogInterface, int i) {
    
```

B3 - Trong `MainActivity.java`, tại hàm `xuLyThoat();`, cập nhật thêm 1 hàm `taoCuaSoHoiThoat()` → Alt + Enter để phát sinh ra hàm.



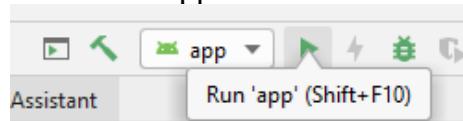
B5 - Trong `MainActivity.java`, bổ sung thêm hàm `onBackPressed()`; → để không cho người dùng bấm nút Back thoát khỏi chương trình khi có bảng thông báo hiện lên.

```

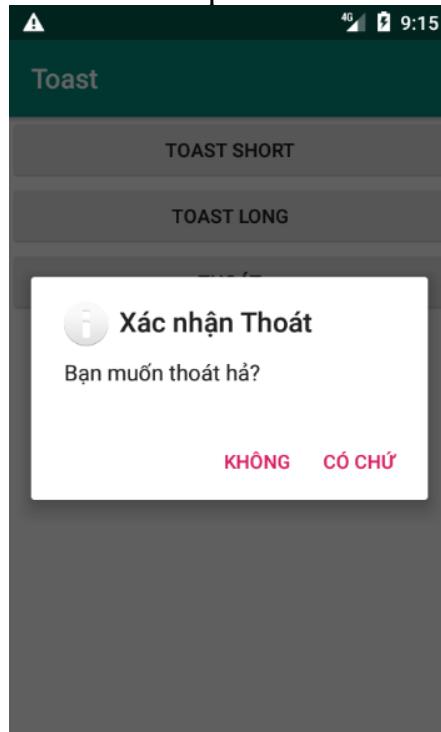
public void onBackPressed () {
    taoCuaSoHoiThoat();
}

```

B6 – Run ‘app’



B7 – Kiểm tra kết quả:



```

        dialogInterface.dismiss();
    }
})
//tạo cửa sổ dialog:
AlertDialog
dialog=builder.create();
//không cho người dùng
nhấn ở ngoài để đóng cửa sổ

dialog.setCanceledOnTouchOut-
side(false);
//hiển thị cửa sổ này
lên:
dialog.show();
}

```

2.3. Notification

Notification là cửa sổ thông báo ở phía trên điện thoại, cho phép người dùng tương tác. Notification thường được dùng khi phần mềm ở trạng thái background (chạy nền). Ta cần tạo đối tượng Builder từ NotificationCompat.Builder, các phương thức thường dùng:

- setSmallIcon: tạo icon
- setContentTitle: tạo tiêu đề
- setContentText: tạo nội dung
- setSound: tạo âm thanh.

Ta cần lấy đối tượng NotificationManager ra để tương tác.

```

NotificationManager
notificationManager=getSystemService(NOTIFICATION_SERVICE
);

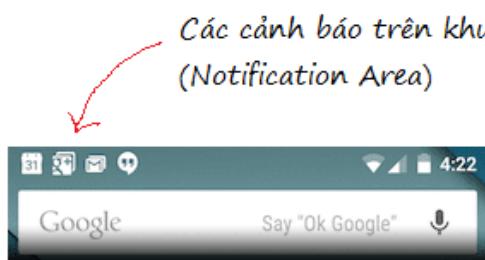
```

Hàm notify dùng để mở cửa sổ Notification

```
notificationManager.notify(123,builder.build());
```

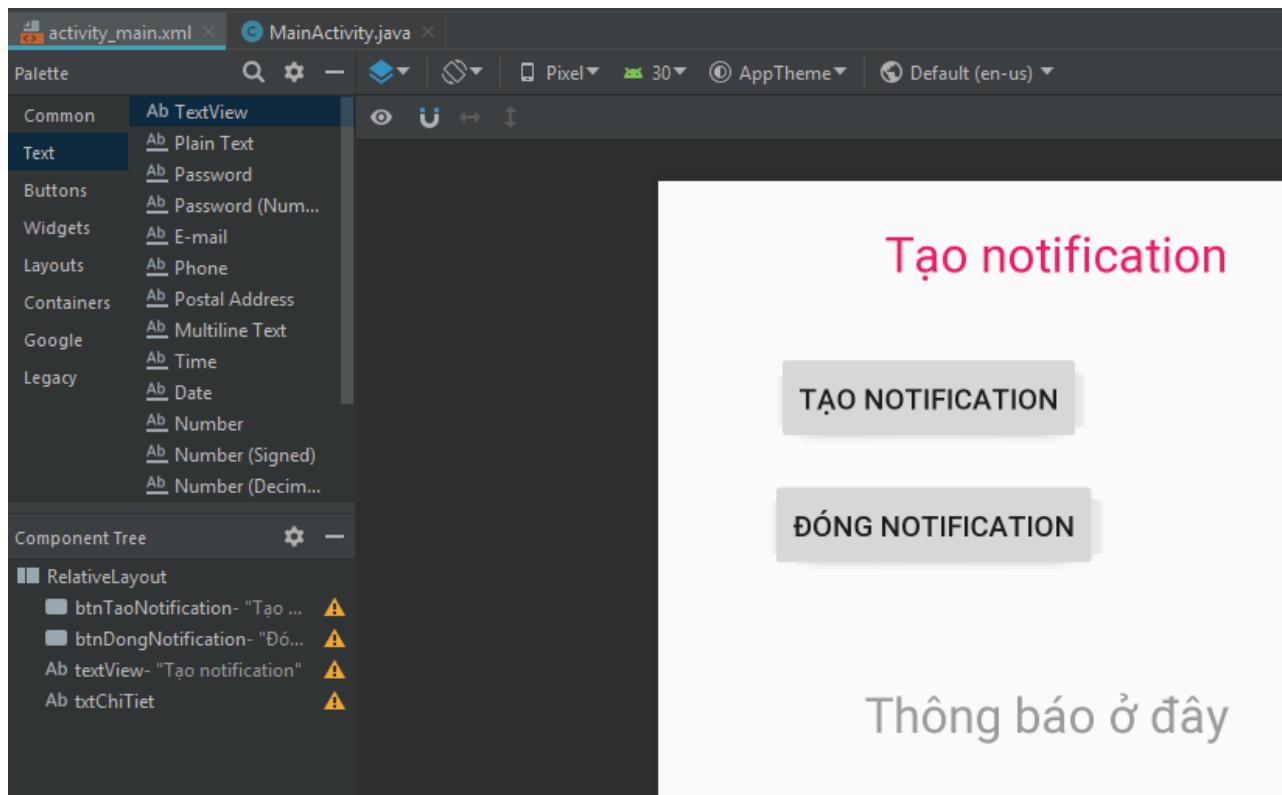
Hàm cancel dùng để đóng cửa sổ Notification

```
notificationManager.cancel(123);
```



Ta có thể hiển thị hoặc tắt Notification thông qua Id.

Ví dụ: viết chương trình tạo notification đơn giản. Ta tiến hành kéo thả một số View như hình minh họa dưới đây:



Cấu trúc XML của màn hình này như sau:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity" >

    <Button
        android:id="@+id/btnTaoNotification"
        android:onClick="xuLyTaoNotification"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentStart="true"
        android:layout_alignParentTop="true"
        android:layout_marginStart="57dp"
        android:layout_marginTop="82dp"
        android:text="Tạo notification" />
```

```
<Button  
    android:id="@+id/btnDongNotification"  
    android:onClick="xuLyDongNotification"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:layout_alignParentStart="true"  
    android:layout_alignParentTop="true"  
    android:layout_marginStart="54dp"  
    android:layout_marginTop="144dp"  
    android:text="Đóng notification" />  
  
<TextView  
    android:id="@+id/textView"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:layout_alignParentStart="true"  
    android:layout_alignParentTop="true"  
    android:layout_centerHorizontal="true"  
    android:layout_marginStart="111dp"  
    android:layout_marginTop="19dp"  
    android:text="Tạo notification"  
    android:textAlignment="center"  
    android:textColor="#E91E63"  
    android:textSize="24sp" />  
  
<TextView  
    android:id="@+id/txtChiTiet"  
    android:hint="Thông báo ở đây"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:layout_alignParentStart="true"  
    android:layout_alignParentTop="true"  
    android:layout_marginStart="101dp"  
    android:layout_marginTop="244dp"  
    android:textSize="24sp" />  
</RelativeLayout>
```

Sau đó tiến hành coding:

```
package com.example.notification;  
  
import androidx.appcompat.app.AppCompatActivity;  
import androidx.core.app.NotificationCompat;  
import androidx.core.app.TaskStackBuilder;  
  
import android.app.NotificationManager;  
import android.app.PendingIntent;  
import android.content.Intent;
```

```
import android.media.RingtoneManager;
import android.os.Bundle;
import android.view.View;
import android.widget.TextView;

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        Intent i=getIntent();
        String s=i.getStringExtra("CHITIET");
        TextView
txtChiTiet=findViewById(R.id.txtChiTiet);
        txtChiTiet.setText(s);
    }

    public void xuLyTaoNotification(View view) {
        NotificationCompat.Builder builder=new
NotificationCompat.Builder(this);

        builder.setSmallIcon(android.R.drawable.ic_dialog_info);
        builder.setContentInfo("Có thông báo");
        builder.setContentText("Nhấn vào đây để xem chi
tiết");

        builder.setSound(RingtoneManager.getDefaultUri(RingtoneMa
nager.TYPE_NOTIFICATION));
        Intent resultIntent=new
Intent(this,MainActivity.class);
        String s="Xin chào Xiao Chin";
        resultIntent.putExtra("CHITIET",s);
        TaskStackBuilder
stackBuilder=TaskStackBuilder.create(this);
        stackBuilder.addParentStack(MainActivity.class);
        stackBuilder.addNextIntent(resultIntent);
        PendingIntent
resultPendingIntent=stackBuilder.getPendingIntent(0,Pendi
ngIntent.FLAG_UPDATE_CURRENT);
        builder.setContentIntent(resultPendingIntent);
        NotificationManager notificationManager=
(NotificationManager)
getSystemService(NOTIFICATION_SERVICE);
        notificationManager.notify(123,builder.build());
    }
}
```

```

        }

    public void xuLyDongNotification(View view) {
        NotificationManager notificationManager=
        (NotificationManager) getSystemService(NOTIFICATION_SERVICE);
        notificationManager.cancel(123);
    }
}

```

3. BÀI TẬP:

Bài 1: Hãy so sánh sự khác biệt khi dùng LENGTH_SHORT và LENGTH_LONG trong cửa sổ Toast.

Bài 2: Những trường hợp nào sử dụng Toast là phù hợp nhất?

Bài 3: Những trường hợp nào sử dụng AlertDialog, Notification là phù hợp nhất?

Bài 4: Viết phần mềm xử lý thông tin với màn hình như sau:

THÔNG TIN CÁ NHÂN	
Họ và tên:	<u>Họ và tên ở đây</u>
CMND:	<u>Cmnd ở đây</u>
BẰNG CẤP	
<input type="radio"/> Trung cấp <input type="radio"/> Cao đẳng <input checked="" type="radio"/> Đại học	
BẰNG CẤP	
<input checked="" type="checkbox"/> Đọc báo <input checked="" type="checkbox"/> Đọc sách <input type="checkbox"/> Viết code	
THÔNG TIN BỔ SUNG	
<u>Muốn tìm người yêu</u>	
<input type="button" value="CẬP NHẬT"/>	<input type="button" value="LÀM LẠI"/>

- Tên người dùng không được để trống và phải có ít nhất 03 ký tự.
- Cmnd chỉ được nhập kiểu số và phải có đúng 09 chữ số.
- Bằng cấp mặc định sẽ chọn là đại học.
- Sở thích phải chọn ít nhất 01 lựa chọn.
- Thông tin bổ sung có thể để trống.

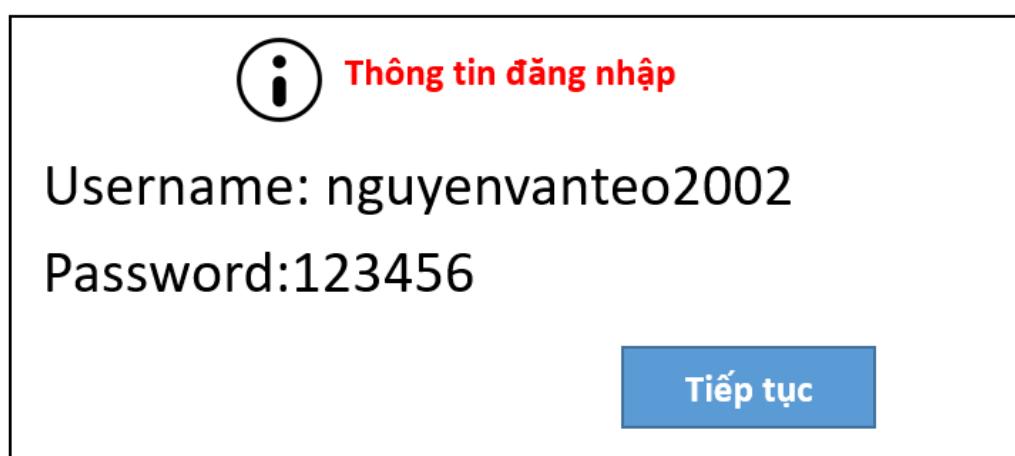
- Khi bấm gửi thông tin, chương trình sẽ hiển thị toàn bộ thông tin cá nhân cho người sử dụng biết (dùng Alert dialog).



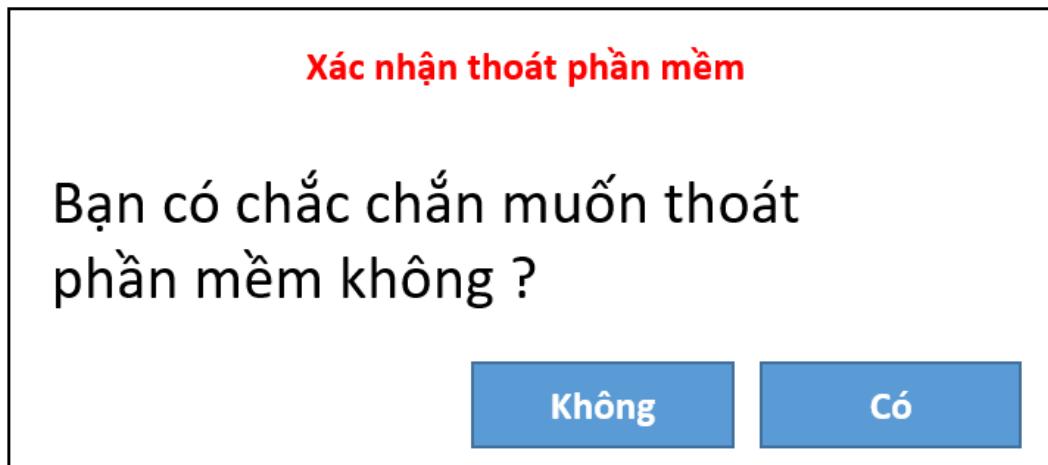
Bài 5: Viết phần mềm kiểm tra thông tin đăng nhập như yêu cầu dưới đây:

The screenshot shows a login form with a blue header bar containing the text 'ĐĂNG NHẬP HỆ THỐNG' in red. Below the header, there are two input fields: 'Username: Nhập username' and 'Password: Nhập password'. At the bottom are two blue buttons: 'Đăng nhập' on the left and 'Thoát' on the right.

- Khi bấm nút đăng nhập sẽ hiển thị thông tin đăng nhập vào Alertdialog:

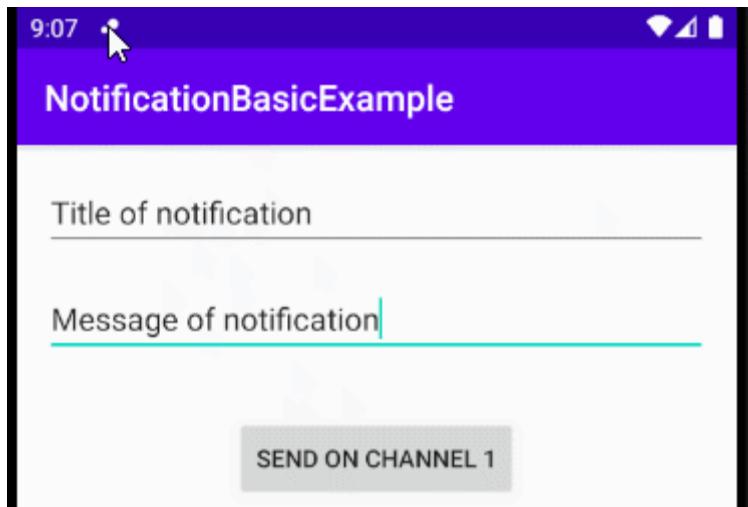


- Bấm tiếp tục thì xoá trống toàn bộ thông tin đăng nhập cũ và cho focus tới ô username.
 - Khi bấm nút Thoát thì xuất hiện cửa sổ xác nhận thoát phần mềm:



- Chọn có thì đóng phần mềm, chọn không sẽ quay về màn hình đăng nhập.

Bài 3: Viết chương trình cho phép nhập tiêu đề và nội dung notification sau đó hiển thị lên notification. Chương trình cho phép xem nội dung thông điệp khi nhấn vào Notification:



LAB 7. ADAPTER, LIST VIEW

1. MỤC TIÊU:

Đối với thiết bị di động, rất nhiều dữ liệu cần phải hiển thị dạng danh sách như danh bạ, tin nhắn, hiển thị dạng lưới như hình ảnh, hay các điều khiển về ngày giờ, phân trang... Đây là những view nâng cao được android hỗ trợ.

2. LÝ THUYẾT

2.1. Adapter trong android

Adapter giữ vai trò như cầu nối giữa các đối tượng điều khiển dạng danh sách (AdapterView) và các dữ liệu bên dưới, cung cấp các phương thức truy xuất dữ liệu. cho phép thực hiện quản lý giao diện, số lượng chỉ mục trên AdapterView và thực hiện truy vấn dữ liệu, sắp xếp dữ liệu. Adapter cũng chịu trách nhiệm tạo ra các view con trong tập các dữ liệu.

AdapterView là các đối tượng điều khiển dạng tập hợp, cho phép hiển thị thông tin cơ bản theo dạng danh sách và thực hiện quản lý theo từng mục riêng biệt.

2.2. ListView

ListView là view dùng để hiển thị danh sách dữ liệu, ListView thường dùng Adapter để vẽ dữ liệu lên giao diện. Ta thường đặt id cho ListView bắt đầu bằng lv.

Ta có 4 bước để hiển thị dữ liệu lên ListView như sau:

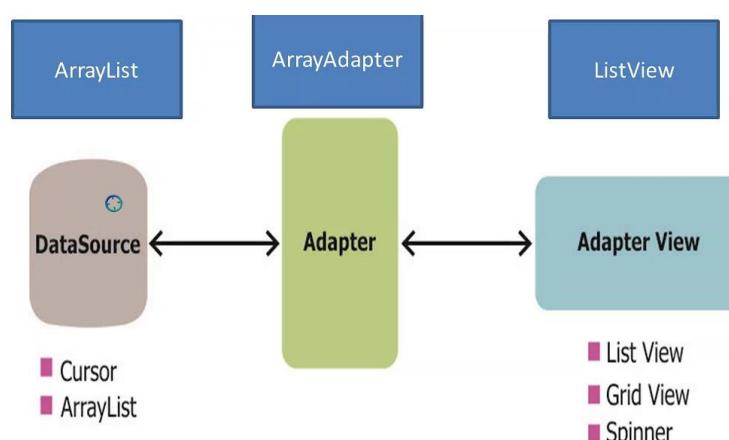
Bước 1: Đưa dữ liệu cần hiển thị lên ListView vào một mảng hoặc danh sách (ArrayList, mảng thông thường, mảng đối tượng...)

Bước 2: Tạo một ListView trên giao diện.

Bước 3: Tạo một đối tượng ArrayAdapter để liên kết giữa ListView và mảng hoặc danh sách dữ liệu.

Bước 4: Gán ArrayAdapter cho ListView.

Dưới đây là tổng quan cơ chế hoạt động của ListView:

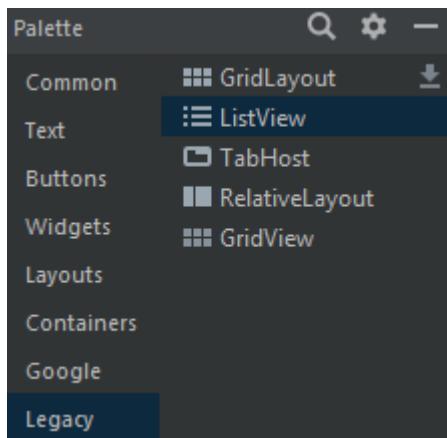


Ta phải xác định nguồn dữ liệu trước, nguồn dữ liệu có kiểu gì thì khai báo ArrayAdapter có kiểu đó rồi đổ dữ liệu vào, kế tới gán ArrayAdapter lên ListView.

Dưới đây là hướng dẫn chi tiết về 4 bước hiển thị dữ liệu lên Listview:

Bước 1: Giả sử ta có nguồn dữ liệu: String []arrData={"Hà Nội","Huế","Đà Nẵng"}, ta có thể đưa tập dữ liệu này vào ArrayList.

Bước 2: Kéo ListView trong nhóm Legacy của công cụ Palette ra (lvData1)



Cấu trúc xml của ListView như sau:

```
<ListView
    android:id="@+id/lvData1"
    android:layout_width="match_parent"
    android:layout_height="match_parent" />
```

Bảng 7.1. Các thuộc tính, sự kiện quan trọng của ListView

Các thuộc tính/sự kiện	Mô tả
android:id	Id của ListView
android:layout_width	Độ rộng
android:layout_height	Độ cao
android:listSelector	Thiết lập màu cho dòng đang chọn
setAdapter	Phương thức gán Adapter cho ListView
setHeaderView	Phương thức thêm Header cho ListView
setFooterView	Phương thức thêm Footer cho ListView
setSelection	Phương thức chọn dòng theo vị trí trên ListView
setOnItemClickListener	Phương thức gán sự kiện chọn 1 dòng

Bước 3: Tạo đối tượng ArrayAdapter, nguồn dữ liệu có kiểu nào thì Adapter phải khai báo kiểu đó (thường ta dùng ArrayAdapter có 3 đối số):

```
adapter =new ArrayAdapter<>(ListView1
.this, android.R.layout.simple_list_item_1, arrData);
//gán adapter cho listview
lvData1.setAdapter(adapter);
```

Đối số 1: là màn hình sử dụng ArrayAdapter này, màn hình nào sử dụng thì màn hình đó .this. Vậy MainActivity.this tức là màn hình MainActivity sử dụng.

Đối số 2: android.R.layout.simple_list_item_1 là layout hiển thị dữ liệu lên giao diện, được định nghĩa sẵn bởi Android.

Đối số 3: arrData là nguồn dữ liệu ta gán vào cho Adapter (có thể thay thế mảng 1 chiều bằng ArrayList)

Bước 4: gán ArrayAdapter cho ListView: lvData1.setAdapter(adapter);

Ta tiến hành chạy phần mềm và có kết quả sau:



Hình 7.1. Màn hình kết quả hiển thị ListView

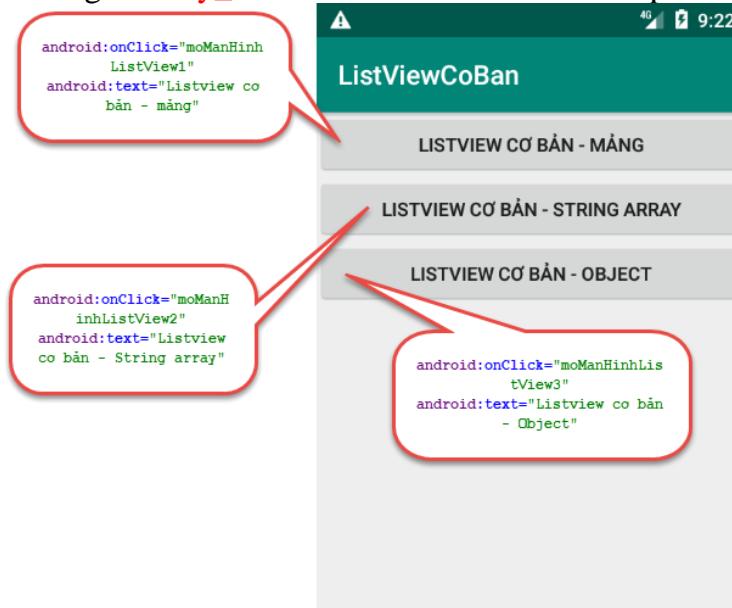
Để gán sự kiện lắng nghe người sử dụng chọn dòng dữ liệu trên ListView ta thường dùng cách sau (đối số thứ 3 có tên là i → vị trí dòng được chọn trên ListView):

```
lvData1.setOnItemClickListener(new
AdapterView.OnItemClickListener() {
    @Override
    public void onItemClick(AdapterView<?> adapterView,
View view, int i, long l) {
        Toast.makeText(ListView1.this,"Bạn chọn
"+arrData[i],Toast.LENGTH_LONG).show();
    }
});
```

Ví dụ 1: viết chương trình tạo ListView cơ bản sử dụng mảng

B1 - Khởi động Android Studio → Start a new android project → Empty Activity → Name: **ListViewCoBan**

B2 - Sử dụng text trong **activity_main.xml** thêm vào các thành phần của giao diện



B3 - Trong **activity_main.xml**, tại nút **ListView Cơ Bản – Mảng** thêm sự kiện On Click XML “**moManHinhListView1**”

```
<Button
    android:onClick="moManHinhListView1"
    android:text="Listview cơ bản - mảng"
    android:layout_width="match_parent"
    android:layout_height="wrap_content" />
```

B4 - Trong **MainActivity.java**, tại hàm **moManHinhListView1();**, cập nhật:

```
public void moManHinhListView1(View
view) {
    Intent intent = new
Intent(MainActivity.this, ListView1.c
lass);
    startActivity(intent);
}
```

B5 – Nhấp chuột phải lên Java → New → Activity → Empty Activity → Đặt tên **ListView1**



B6 - Trong **activity_list_view1.xml** bổ sung **ListView** như sau:

```
<ListView
    android:id="@+id/lvData1"

    android:layout_width="match_parent"

    android:layout_height="match_parent"

    android:listSelector="@android:color
/holo_blue_dark">
</ListView>
```

B7 – Trong **ListView1.java**, khai báo các biến và mảng sau:

```
ListView lvData1;
String [] arrData =
{"Trường Vũ", "Như
```

B8 - Trong **ListView1.java**, tại hàm **onCreate**, bổ sung 2 hàm là **addControls();** và **addEvents();**

<pre> Quỳnh", "Tâm Đoan", "Duy Khánh", "Băng Tâm", "Tuấn Vũ", "Nhật Trường", "Mai Thiên Vân", "Giao Linh", "Đan Nguyên"} ; ArrayAdapter<String> adapter; </pre>	<p>B9 – Trong hàm <code>addControls()</code>; khai báo các điều khiển sau:</p> <pre> //lấy listview ra lvData1 = findViewById(R.id.lvData1); //tạo adapter adapter = new ArrayAdapter<>(ListView1.this, android.R.layout.simple_list_item_1, arrData); //gán adapter cho listview lvData1.setAdapter(adapter); </pre>
<p>B10 – Trong hàm <code>addEvents()</code>; cập nhật vào đây đoạn code xử lí sự kiện khi chọn item trong ListView</p> <pre> private void addEvents() { lvData1.setOnItemClickListener(new AdapterView.OnItemClickListener() { @Override public void onItemClick(AdapterView<?> adapterView, View view, int i, long l) { Toast.makeText(ListViewCoBan.this, "Bạn chọn "+arrData[i], Toast.LENGTH_LONG).show(); } }); } </pre>	<p>B11 – Run ‘app’</p>  <p>B12 – Kiểm tra:</p> 

Ví dụ 2: Xây dựng ListView sử dụng String Array

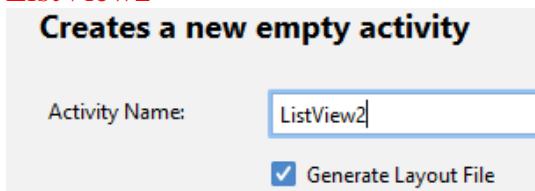
Tiếp tục sử dụng Project ListViewCoBan trong Bài 1 để thực hiện. Yêu cầu:

ListView cơ bản String Array →

B1 - Trong `activity_main.xml`, tại nút **ListView Cơ Bản – Mảng** thêm sự kiện On Click XML “`moManHinhListView2`”

```
<Button
    android:onClick="moManHinhListView2"
    android:text="Listview cơ bản - String array"
    android:layout_width="match_parent"
    android:layout_height="wrap_content" />
```

B3 – Nhấp chuột phải lên Java → New → Activity → Empty Activity → Đặt tên `ListView2`



B5 – Trong `strings.xml` tạo 1 mảng có dạng:

```
<string-array name="myArray">
    <item>Tả Lãnh Thiên</item>
    <item>Nhạc Bất Quần</item>
    <item>Quách Khiêu</item>
    <item>Thiên</item>
</string-array>
```

```
private void addControls() {
    lvData2 = findViewById(R.id.lvData2);
    arrData2 = getResources().getStringArray(R.array.myArray);
    adapter2 = new ArrayAdapter<String>(context: ListView2.this, andr
    lvData2.setAdapter(adapter2);
```

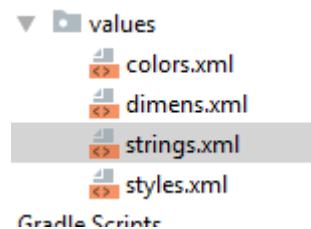
B7 – Trong `ListView1.java`, khai báo các biến và mảng sau:

```
ListView lvData2;
String [] arrData2;
ArrayAdapter<String>adapter2;
```

B2 - Trong `MainActivity.java`, tại hàm `moManHinhListView2()`; cập nhật:

```
public void moManHinhListView2(View view) {
    Intent intent = new Intent(MainActivity.this, ListView2.class);
    startActivity(intent);
}
```

B4 – Values → `strings.xml`



B6 - Trong `activity_list_view2.xml` bổ sung `ListView` như sau:

```
<ListView
    android:id="@+id/lvData2"
    android:layout_width="match_parent"
    android:layout_height="match_parent"></ListView>
```

B8 - Trong `ListView2.java`, tại hàm `onCreate`, bổ sung 2 hàm là `addControls();` `addEvents();`

B10 – Trong hàm `addEvents();` cập nhật vào đây đoạn code xử lý sự kiện khi chọn item trong `ListView`

```
private void addEvents() {
    lvData2.setOnItemClickListener(new
```

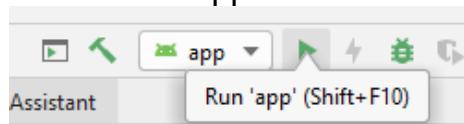
B9 – Trong hàm addControls(); khai báo các điều khiển sau:

```
private void addControls() {
    lvData2 = findViewById(R.id.lvData2);
    arrData2 = getResources().getStringArray(
        R.array.myArray);
    adapter2 = new ArrayAdapter<String>(ListView2
        .this, android.R.layout.simple_
        list_item_1, arrData2);

    lvData2.setAdapter(adapter2);
}
```

```
AdapterView.OnItemClickListener) {
    @Override
    public void onItemClick(AdapterView<?>
        adapterView, View view, int i,
        long l) {
        Toast.makeText(ListView2.t
            his, "Bạn chọn
            "+arrData2[i], Toast.LENGTH_
            SHORT).show();
    }
}) ;
```

B11 – Run ‘app’

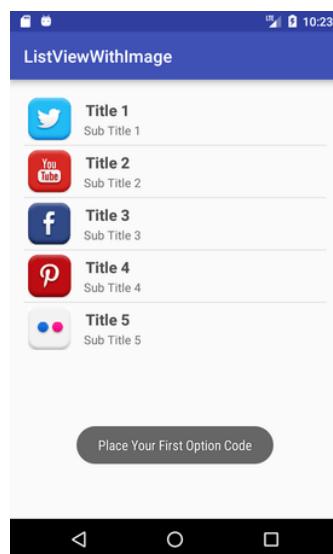


B12 – Kiểm tra



2.3. Custom ListView

ListView còn cho phép ta chỉnh sửa, tạo ra các dòng hiển thị phức tạp đáp ứng nhu cầu người dùng. Giả sử ta cần phải thiết kế màn hình ListView phức tạp như sau:

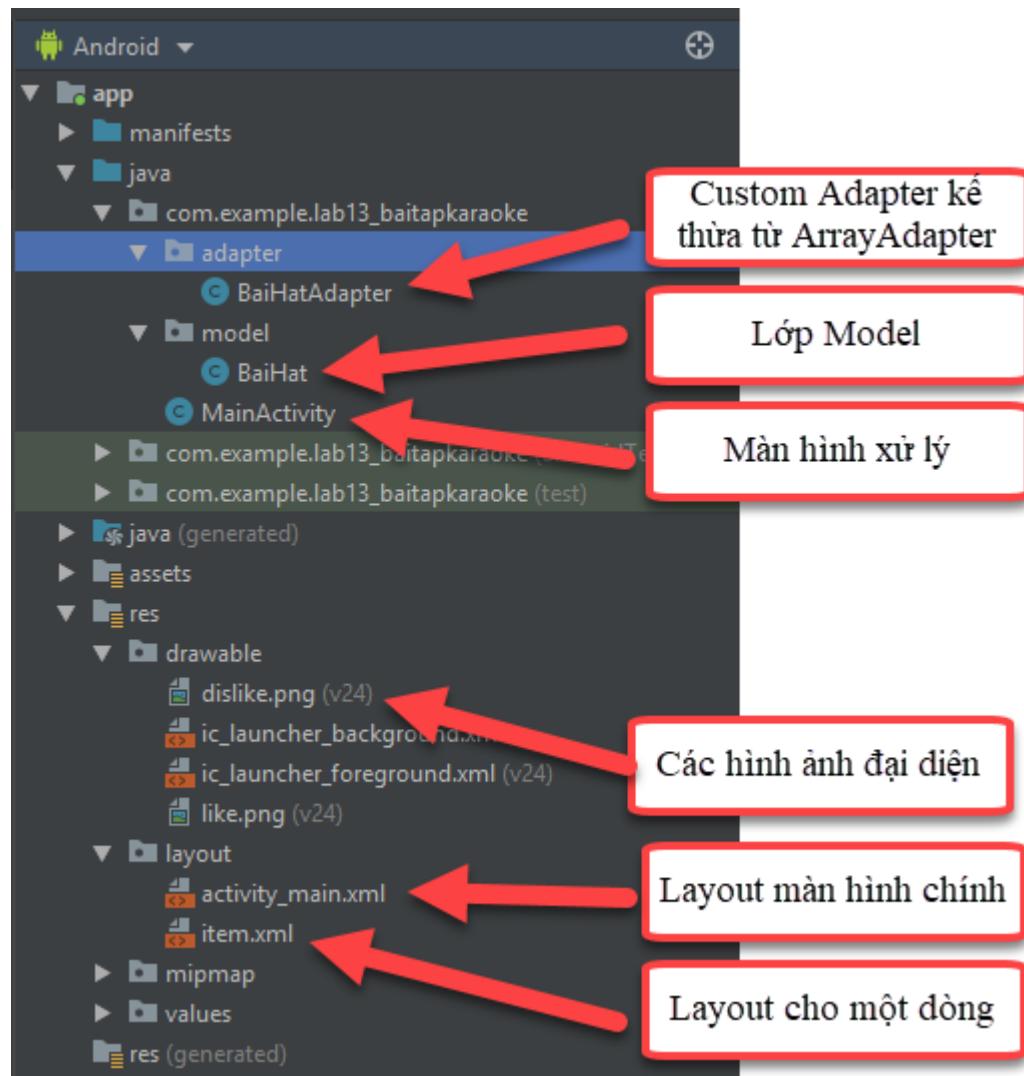


Hình 7.2. Màn hình Custom ListView

Để làm được yêu cầu trên ta phải:

- Mô hình hoá đối tượng.
- Thiết kế layout cho 1 dòng có cấu trúc giống yêu cầu
- Kế thừa ArrayAdapter, hiệu chỉnh hàm getView để hiển thị dữ liệu như yêu cầu
- Gán ArrayAdapter kế thừa cho ListView.

Dưới đây là cấu trúc tổng quan của Project mà ta phải làm cho yêu cầu trên:



Hình 7.3. Màn hình tổng quan cấu trúc Project của Custom ListView

Ở trên ta phân loại các lớp vào các package khác nhau, BaiHat là class mô hình đối tượng, BaiHatAdapter là lớp chỉnh sửa Adapter hiển thị dữ liệu. Các hình ảnh minh họa cho bài hát được lưu vào drawable, chi tiết từng bước:

Bước 1: mô hình hoá đối tượng

Bước 2: Thiết kế layout cho một dòng, ta cần quan sát để tạo ra cấu trúc XML đúng. Đây là cấu trúc xml của dòng custom layout:

Bước 3: kế thừa ArrayAdapter, hiệu chỉnh getView để hiển thị dữ liệu như yêu cầu, trong package adapter, ta tạo lớp BaiHatAdapter kế thừa từ ArrayAdapter

2.4. Recycle View

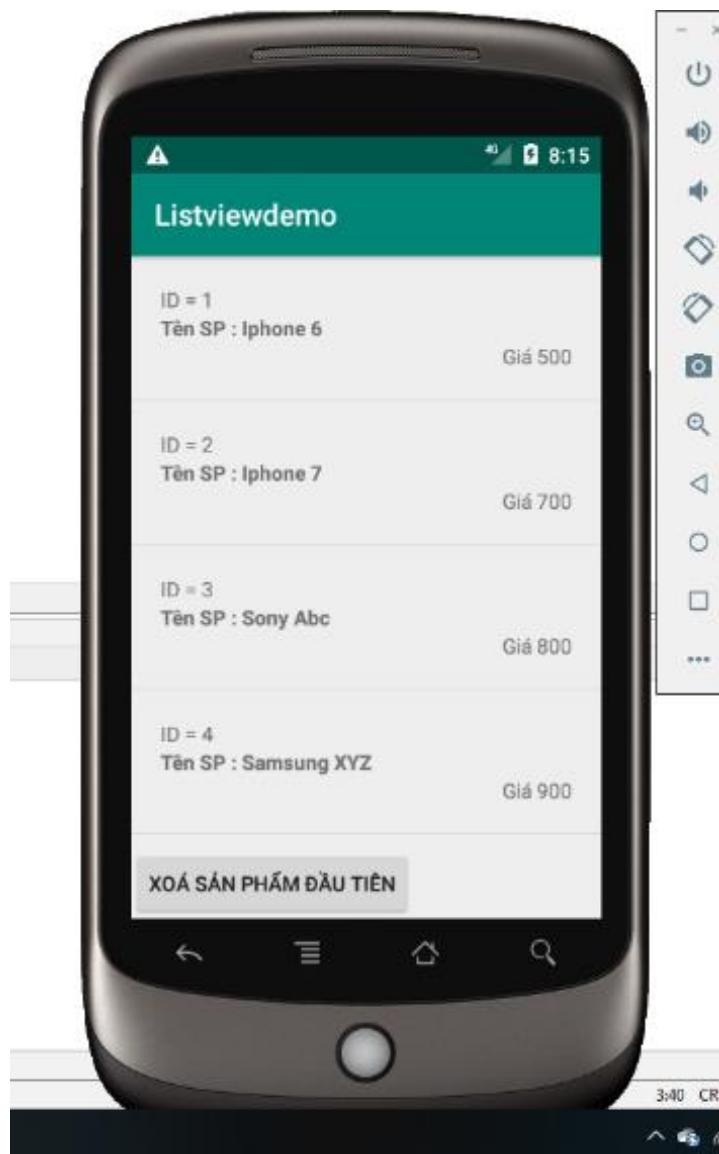
-đang cập nhật-

3. BÀI TẬP

Bài 1: Adapter đóng vai trò gì trong việc hiển thị dữ liệu lên các List Control

Bài 2: Để hiển thị custom layout đáp ứng nhu cầu khách hàng ta làm như thế nào

Bài 3: Xây dựng ứng dụng hiển thị một danh sách các sản phẩm đơn giản, sản phẩm được trình bày trong một layout có các thông tin như ID, tên sản phẩm, giá. Khi bấm vào chọn một sản phẩm, Pop up lên thông báo tên sản phẩm đó. Có chức năng cho phép xoá phần tử đầu tiên của danh sách (sau khi xoá ListView cập nhật lại)



Bài 4: Dùng mảng hoặc ArrayList để hiển thị dữ liệu lên ListView với giao diện như dưới đây:

...

Xử lý sự kiện khi người dùng chọn một dòng bất kỳ trên ListView thì tô màu xanh và hiển thị vị trí cuối cùng giá trị của dòng đó lên TextView ở phía trên.

Bài 5: Dùng Custom Layout để thiết kế và giả lập dữ liệu cho ListView với giao diện như sau:

...

Bài 6: Dùng Custom Layout để thiết kế và giải lập dữ liệu cho ListView với giao diện như sau (mỗi dòng có tên, số phone, nút gọi, nút nhắn tin...)

...

Bài 7: Giả lập phần mềm karaoke với giao diện custom layout dưới đây:

...

- Mô hình hoá và giả lập danh sách bài hát karaoke (mỗi bài hát có mã, tên, ca sĩ).
- Tab all: hiển thị toàn bộ bài hát.
- Nhấn vào like: bài hát này sẽ được hiển thị trong tab love.
- Nhấn dislike: gỡ bỏ bài hát khỏi tab love.

LAB 8. SPINNER

1. MỤC TIÊU:

...

2. THỰC HIỆN:

2.1. Bài 1: Spinner

B1 - Khởi động Android Studio → Start a new android project → Empty Activity → Name: **Spinner**

B2 - Sử dụng text trong **activity_main.xml** thêm vào các thành phần như sau:

```
<Spinner
    android:id="@+id/spinnerSanPham"
    android:layout_width="match_parent"
    android:layout_height="wrap_content">
</Spinner>
<TextView
    android:id="@+id/txtSanPham"
    android:layout_width="match_parent"
    android:layout_height="wrap_content" />
```

B3 – Trong **MainActivity.java**, khai báo các biến sau:

```
Spinner           spinnerSanPham;
TextView         txtSanPham;
ArrayAdapter<String> adapter;
```

B4 - Trong **MainActivity.java**, tại hàm **onCreate()**; thêm vào 2 hàm: **addControls()**; và **addEvents()** → Alt + Enter để phát sinh hàm mới.

```
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    addControls();
    addEvents();
}
```

B5 – Trong **MainActivity.java**, tại hàm **addControls ()**; → bổ sung các điều khiển và thêm 1 số sản phẩm trong adapter.

```
private          void          addControls()          {
    spinnerSanPham = findViewById(R.id.spinnerSanPham);
    adapter=new ArrayAdapter<String>(MainActivity.this,
        android.R.layout.simple_spinner_item);
    adapter.add("Cocacola");
    adapter.add("Pepsi");
    adapter.add("Redbull");
    adapter.add("Aquafina");

    adapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);

//adapter.setDropDownViewResource(android.R.layout.simple
```

```

e_list_item_checked);

//adapter.setDropDownViewResource(android.R.layout.simple
e_list_item_multiple_choice);
    spinnerSanPham.setAdapter(adapter);
    txtSanPham=findViewById(R.id.txtSanPham);
}
}

```

B6 - Trong **MainActivity.java**, tại hàm **addEvents ()**; → bổ sung hàm gán text cho TextView txtSanPham khi sản phẩm được chọn từ trong spinner.

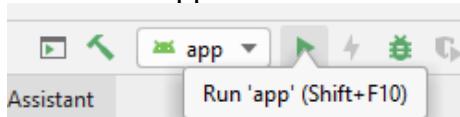
```

private void addEvents() {
    spinnerSanPham.setOnItemSelectedListener(new
AdapterView.OnItemSelectedListener() {
    @Override
    public void onItemSelected(AdapterView<?>
adapterView, View view, int i, long l) {
        String s = adapter.getItem(i);
        txtSanPham.setText(s);
    }

    @Override
    public void onNothingSelected(AdapterView<?>
adapterView) {
    }
}) ;
}
}

```

B7 – Run ‘app’

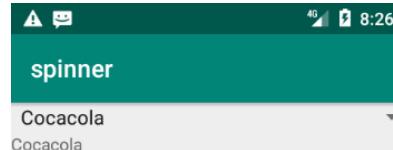


B10 – Thực hiện lại **B5, trong phương thức**

setDropDownViewResource

Thay đổi một số kiểu dropdown list khác nhau.

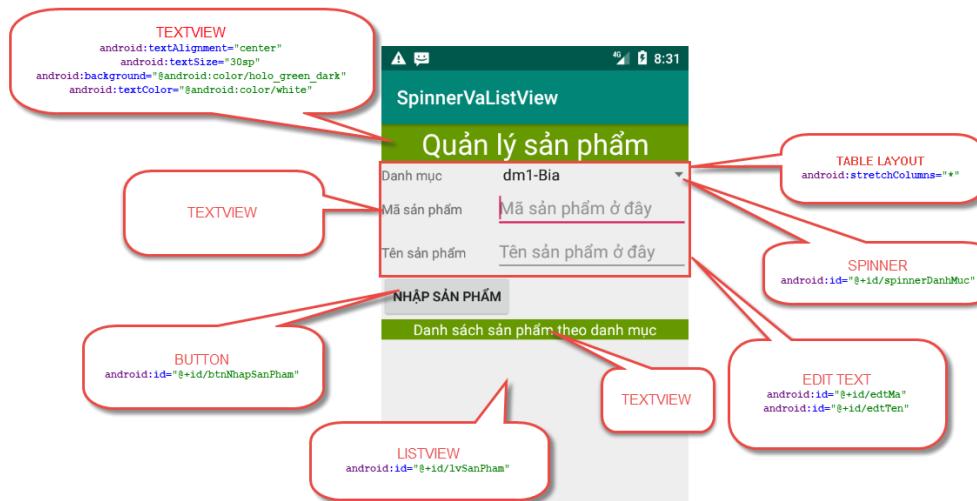
B8 – Kiểm tra:



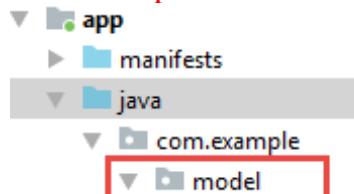
2.2. Bài 2: Kết hợp Spinner và ListView: Phần mềm quản lý sản phẩm

B1 - Khởi động Android Studio → Start a new android project → Empty Activity → Name: SpinnerVaListView

B2 - Sử dụng text trong `activity_main.xml` thêm vào các thành phần, đặt id đúng như mô tả:



B3 – Thêm package Model: chuột phải lên folder java → new → package → main → com.example.model



B5 – Tại class `DanhMuc`, khai báo các thuộc tính mã, tên và mảng các sản phẩm.

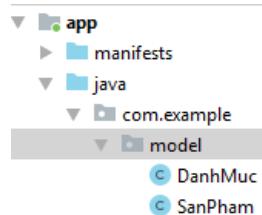
```
private String ma;
private String ten;
private ArrayList<SanPham> sanPhams=new ArrayList<>();
```

B7 – Tại class `DanhMuc`, thêm hàm `toString()` để nối chuỗi mã-tên.

```
public String toString() {
    return this.ma+"-"+this.ten;
}
```

B8 – Tại class `SanPham.java`, tiếp tục tạo các getter and setter và constructor tương tự B6.

B4 – Trong package Model, thêm 2 class `DanhMuc` và `SanPham`: chuột phải lên package Model → new → java class → Name: `DanhMuc`. Làm tương tự với class `SanPham`.



B6 – Tại class `DanhMuc.java`, thêm các `getter and setter` và `constructor` cho các thuộc tính ở B5.

B6.1 – Tạo `getter and setter`: tại `DanhMuc.java` → nhấp chuột phải chọn Generate → Getter & Setter → Ctrl + A → OK.

B6.2 – Tạo `Constructor`: tại `DanhMuc.java` → nhấp chuột phải chọn Generate → Constructor → **chọn** thuộc tính mã và tên, không chọn sản phẩm → OK

<p>B9 – Tại class SanPham, thêm hàm toString():</p> <pre>public String toString() { return this.ma + "-" + this.ten; }</pre> <p>B11 – Trong MainActivity.java, tại hàm OnCreate, bổ sung thêm 2 hàm addControls(); và addEvents(); → Alt + Enter để tự động phát sinh ra hàm mới.</p>	<p>B10 – Trong MainActivity.java, khai báo các biến sau:</p> <pre>Spinner spinnerDanhMuc; ArrayAdapter<DanhMuc> danhMucAdapter; ListView lvSanPham; ArrayAdapter<SanPham> sanPhamAdapter; EditText edtMa, edtTen; Button btnNhaphSanPham; DanhMuc selectedDanhMuc = null;</pre>
-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

B12 - Trong **MainActivity.java**, tại hàm **addControls()**; cập nhật:

```
private void addControls() {
    spinnerDanhMuc = findViewById(R.id.spinnerDanhMuc);
    danhMucAdapter = new
    ArrayAdapter<DanhMuc>(MainActivity.this, android.R.layout.simple_spinner_item);

    danhMucAdapter.setDropDownViewResource(android.R.layout.simple_list_item_checked);
    spinnerDanhMuc.setAdapter(danhMucAdapter);
    lvSanPham=findViewById(R.id.lvSanPham);
    sanPhamAdapter=new
    ArrayAdapter<SanPham>(MainActivity.this, android.R.layout.simple_list_item_1);
    lvSanPham.setAdapter(sanPhamAdapter);
    edtMa = findViewById(R.id.edtMa);
    edtTen = findViewById(R.id.edtTen);
    btnNhaphSanPham = findViewById(R.id.btnNhaphSanPham);
    danhMucAdapter.add(new
    danhMucAdapter.add(new
    danhMucAdapter.add(new
    danhMucAdapter.add(new
        DanhMuc("dm1", "Bia"));
        DanhMuc("dm2", "Rượu"));
        DanhMuc("dm3", "Thuốc lá"));
        DanhMuc("dm4", "Nước ngọt"));
    }
}
```

B13 - Trong **MainActivity.java**, tại hàm **addEvents()**; cập nhật: thêm hàm **xuLyNhaphSanPham()**:

```
private void addEvents() {
    btnNhaphSanPham.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            xuLyNhaphSanPham();
        }
    });
}
```

```

spinnerDanhMuc.setOnItemSelectedListener(new
AdapterView.OnItemSelectedListener()
{
    @Override
    public void onItemSelected(AdapterView<?> adapterView, View
view,      int             i,      long           l)
    {
        selectedDanhMuc=danhMucAdapter.getItem(i);
        sanPhamAdapter.clear();
        sanPhamAdapter.addAll(selectedDanhMuc.getSanPhams());
    }

    @Override
    public void onNothingSelected(AdapterView<?> adapterView) { }

})
);
}
}

```

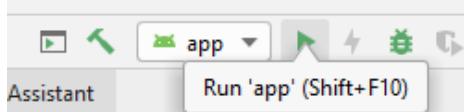
B13 - Trong **MainActivity.java**, tại hàm **xuLyNhapSanPham()**; cập nhật đoạn code:

```

private          void          xuLyNhapSanPham()
{
    SanPham           sp           = new
    SanPham(edtMa.getText().toString(),edtTen.getText().toString());
    selectedDanhMuc.getSanPhams().add(sp);
    sanPhamAdapter.clear();
    sanPhamAdapter.addAll(selectedDanhMuc.getSanPhams());
}

```

B14 – Run ‘app’



B15 – Kiểm tra:

Chọn Danh mục sản phẩm nhập vào mã và tên sản phẩm → bấm nút Nhập sản phẩm. Chuyển sang Danh mục khác, tiếp tục nhập thêm sản phẩm → Quay lại Danh mục trước đó, thấy danh sách các sản phẩm vừa nhập vẫn tồn tại.



3. BÀI TẬP:

...

LAB 9. AUTOCOMPLETE TEXTVIEW, DATE VÀ TIME PICKER, TAB SELECTOR

1. MỤC ĐÍCH

...

2. THỰC HIỆN

2.1. Bài 1: Autocomplete textview

...

2.2. Bài 2: Date và time picker

...

2.3. Bài 3: Tab selector

...

3. BÀI TẬP:

Bài 1: Viết chương trình quản lý mục tiêu hoàn thành công việc hàng tuần. Thiết kế giao diện như hình bên dưới và thực hiện một số chức năng:

...

- Khi chọn nút Date sẽ hiển thị DatePickerDialog: cập nhật ngày hoàn thành.
- Khi chọn nút Time sẽ hiển thị TimePickerDialog: cập nhật giờ hoàn thành
- Khi chọn nút Thêm công việc, chương trình sẽ cập nhật và ListView bên dưới màn hình.

Bài 2: Sử dụng AutoComplete TextView để hiển thị danh sách các tỉnh thành, thiết lập ki nhập một ký tự thì phần mềm tự động gợi ý dữ liệu.

...

Bài 3: Dùng tab selector thiết kế phần mềm có 02 tab như dưới đây:

...

Bài 4: Dùng tab selector thiết kế phần mềm toán học dưới đây:

...

- Tab đầu tiên “calculator” là giao diện cho phép tính cộng, trừ, nhân, chia. Mỗi thao tác sẽ được lưu vào bộ nhớ.
- Tab thứ hai “History” dùng để hiển thị danh sách các phép toán đã thực hiện trong tab thứ nhất.

LAB 10. ACTIVITY VÀ INTENT

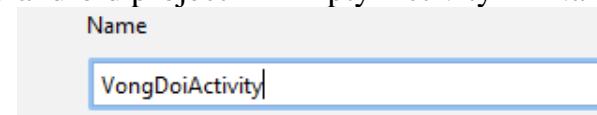
1. MỤC TIÊU:

- Tìm hiểu chi tiết về vòng đời hoạt động của một Activity từ khi kích hoạt đến khi kết thúc.
- cung cấp kiến thức về truyền và nhận dữ liệu giữa các Activity với nhau bằng phương thức.

2. THỰC HIỆN:

2.1. Bài 1: Vòng đời Activity và Intent

B1 - Khởi động Android Studio → Start a new android project → Empty Activity → Name: **VongDoiActivity**



B2 - Sử dụng text trong **activity_main.xml** thêm vào các thành phần của giao diện



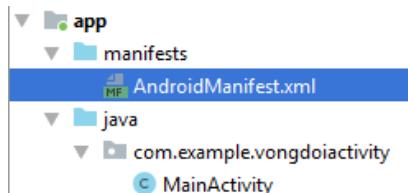
B3 - Trong **activity_main.xml**, tại nút **Mở màn hình che khuất toàn bộ** thêm sự kiện On Click XML “**moManHinhCheKhuatToanBo**”

```
<Button
    android:onClick="moManHinhCheKhuatToanBo"
    android:text="Mở màn hình che khuất toàn bộ"
    android:layout_width="match_parent"
    android:layout_height="wrap_content" />
```

B4 - Trong **activity_main.xml**, tại nút **Mở màn hình che khuất một phần** thêm sự kiện “**moManHinhCheKhuatMotPhan**”

```
<Button
    android:onClick="moManHinhCheKhuatMotPhan"
    android:text="Mở màn hình che khuất một phần"
    android:layout_width="match_parent"
    android:layout_height="wrap_content" />
```

B7 – Mở AndroidManifest.xml

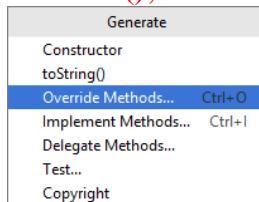


B9 – Ở bước này chúng ta sẽ thêm các phương thức báo trạng thái của chương trình. Trong **MainActivity.java, bấm Ctrl + O →**

B8 – Trong **AndroidManifest.xml, ch**

```
android:theme="@style/Theme.AppCompat.DayNight.Dialog"
<activity android:name=".Sub2Activity"
    android:theme="dial"
    @style/Theme.AppCompat.DayNight.Dialog
<ac @style/Theme.AppCompat.DayNight.Dialog.Alert
<ac @style/Theme.AppCompat.DayNight.Dialog.MinWidth
@style/Theme.AppCompat.DayNight.DialogWhenLarge
@style/Theme.AppCompat.Dialog
@style/Theme.AppCompat.Dialog.Alert
@style/Theme.AppCompat.Dialog.MinWidth
```

override các phương thức: `onStart(); onStop(); onDestroy(); onPause(); onResume(); onRestart();`



B12 - Trong `MainActivity.java`, tại hàm `moManHinhCheKhuatMotPhan()`; truyền vào Intent để khởi động màn hình Sub2Activity:

```
public void moManHinhCheKhuatMotPhan (View
view)
    Intent intent= new
Intent (MainActivity.this,Sub2Activity.class);
    startActivity(intent);
}
```

B14 – Quan sát các phương thức hiện lên trong `Toast`, thử bấm các nút xét.



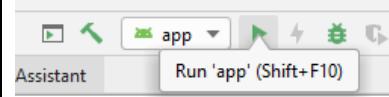
chức năng, nhận

B10 – Trong mỗi phương thức đã Overide, in ra màn hình trạng thái của ứng dụng. Ví dụ: ở phần `onCreate()` in ra `Toast.makeText(getApplicationContext(), "Lần lượt thêm cho tất cả các phương thức này", Toast.LENGTH_SHORT).show();`

B11 - Trong `MainActivity.java`, tại hàm `moManHinhCheKhuatToanBo()`; truyền vào Intent để khởi động màn hình Sub1Activity:

```
public void moManHinhCheKhuatToanBo (View
view)
    Intent intent= new
Intent (MainActivity.this,Sub1Activity.class);
    startActivity(intent);
}
```

B13 – Run ‘app’

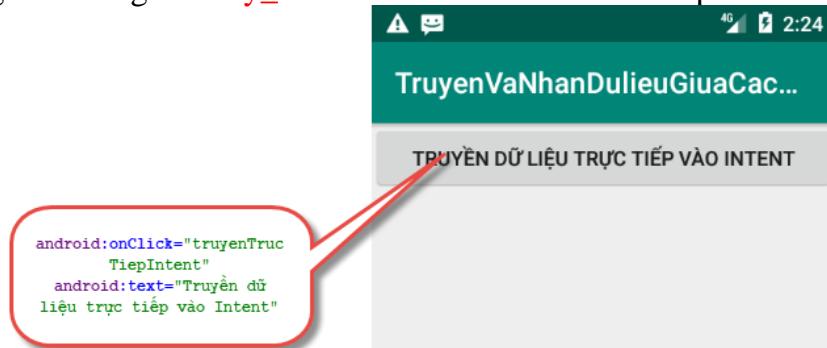


2.2. Bài 2: Truyền và nhận dữ liệu giữa các Activity

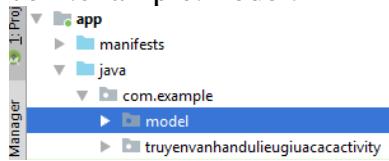
- Mục đích:** bài học này cung cấp kiến thức về truyền và nhận dữ liệu giữa các Activity với nhau bằng phương thức

B1 - Khởi động Android Studio → Start a new android project → Empty Activity → Name: **TruyenVaNhanDuLieuGiuaCacActivity**

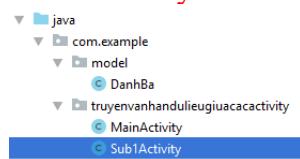
B2 - Sử dụng text trong **activity_main.xml** thêm vào các thành phần của giao diện



B3 – Tạo package **model**: nhấp chuột phải lên folder java → new → package → main → com.example.model.

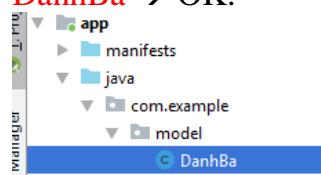


B5 – Tạo **Sub1Activity**: nhấp chuột phải lên folder truyenvanhndluiugiuacacactivit y → new → Activity → Empty Activity → Activity Name: **Sub1Activity**.



B9 – Trong **DanhBa.java**, bổ sung hàm **toString()**:

B4 – Tạo class **Danh bạ**: nhấp chuột phải lên package model → new → Java Class → Name: **DanhBa** → OK.



B6 – Trong **activity_sub1.xml**, bổ sung 1 **TextView** có id là **txtData**:

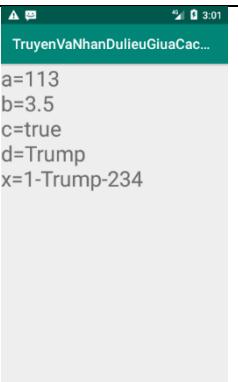
```
<TextView
    android:textSize="30sp"
    android:id="@+id/txtData"

    android:layout_width="match_parent"

    android:layout_height="wrap_content" />
```

B7 – Trong class **DanhBa.java**, cập nhật các phương thức:

```
public class DanhBa implements Serializable {
    private int ma;
    private String ten;
    private String phone;
}
```

<pre>public String toString() { return this.ma + "-" +this.ten + "-" +this.phone; }</pre>	<p>B8 – Trong class DanhBa.java, tạo các getter and setter và constructor cho các thuộc tính. (chi tiết cách tạo xem ở lab 8)</p>
<p>B11 – Trong Sub1Activity.java, tại hàm onCreate, bổ sung hàm addControl() để nhận dữ liệu từ MainActivity. Tại hàm addControls(); cập nhật:</p> <pre>private void addControl() { txtData = findViewById(R.id.txtDa ta); Intent intent = getIntent(); int a = intent.getIntExtra("a", 0); double b = intent.getDoubleExtra("b",0.0); boolean c = intent.getBooleanExtra("c",false); String d = intent.getStringExtra("d"); DanhBa x = (DanhBa) intent.getSerializableEx tra("x"); txtData.setText(""); txtData.append("a=" +a+ "\n"); txtData.append("b=" +b+ "\n"); txtData.append("c=" +c+ "\n"); txtData.append("d=" +d+ "\n"); }</pre>	<p>B10 – Trong MainActivity.java, cập nhật hàm truyenTrucTiepIntent để truyền dữ liệu sang Sub1Activity</p> <pre>public void truyenTrucTiepIntent(View view) { Intent intent = new Intent(MainActivity.this, Sub1Ac tivity.class); intent.putExtra("a", 113); intent.putExtra("b", 3.5); intent.putExtra("c", true); intent.putExtra("d", "Trump"); DanhBa Trump = new DanhBa(1, "Trump", "234"); intent.putExtra("x", Trump); startActivity(intent); }</pre> <p>B12 – Run ‘app’</p>  <p>B13 – Nhận xét: bấm vào nút mở Sub1Activity thấy có dữ liệu truyền sang từ MainActivity.</p> 

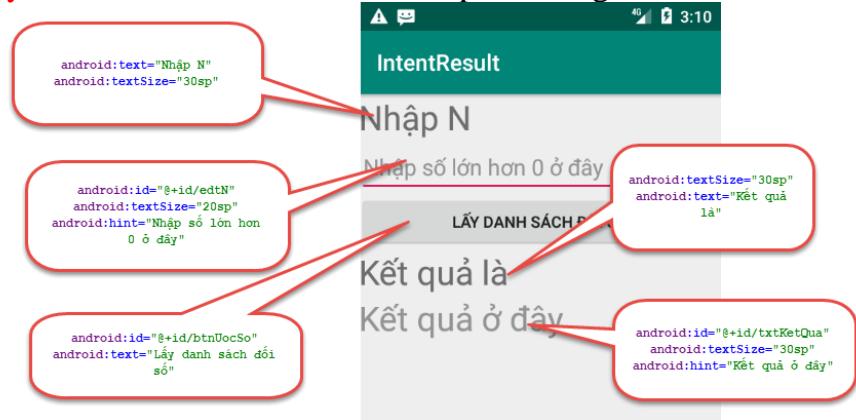
```
txtData.append ("x="+x+
\n);
}
```

2.3. BÀI 3: INTENT RESULT

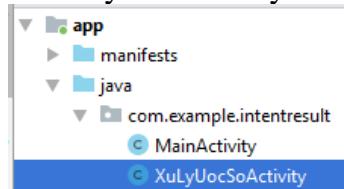
Viết chương trình lấy danh sách ước số gồm 2 Activity, MainActivity

B1 - Khởi động Android Studio → Start a new android project → Empty Activity → Name: IntentResult

B2 - Sử dụng text trong `activity_main.xml` thêm vào các thành phần của giao diện

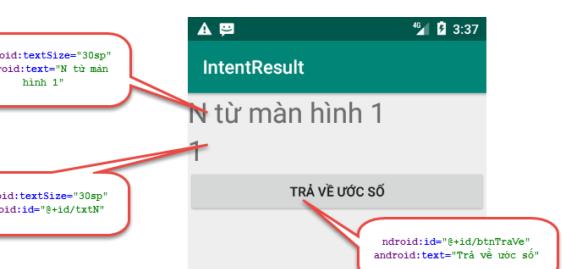


B3 - Tạo `XuLyUocSoActivity`: nhập chuột phải lên folder `intentresult` → new → Activity → Empty Activity → Activity Name: `XuLyUocSoActivity`.



B5 – Trong `MainActivity.java`, khai báo các biến:
EditText `edtN;`
Button `btnLayUocSo;`
TextView `txtKetQua;`

B4 – Trong `activity_xu_ly_uoc_so.xml`, cập nhật g



B6 – Trong `MainActivity.java`, tại hàm `onCreate`,
`addControl();`
`addEvents();`
➔ Alt + Enter để phát sinh ra hàm mới.

B7 - Trong `MainActivity.java`, tại hàm `addControls()`:

```
private void addControl() {
    edtN
    findViewById(R.id.edtN);
    btnLayUocSo
    findViewById(R.id.btnUocSo);
    txtKetQua
    findViewById(R.id.txtKetQua);
}
```

B8 - Trong `MainActivity.java`, tại hàm `addEvents()`; bổ sung thêm hàm `moManHinhXuLyUocSo()`:

```
private void addEvents() {
    btnLayUocSo.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            moManHinhXuLyUocSo();
        }
    });
}
```

B11 – Trong `XuLyUocSoActivity.java`

Khai báo các biến:

```
TextView txtN;
Button btnTrave;
```

```
Intent intent = null;
```

B13 – Trong `XuLyUocSoActivity.java` tại hàm `addControls()`; bổ sung

```
private void addControls() {
    intent = getIntent();
    int n
    intent.getIntExtra("N", 0);
    txtN
    findViewById(R.id.txtN);
    btnTrave
    findViewById(R.id.btnTrave);
    txtN.setText(n + "");
}
```

B9 - Trong `MainActivity.java`, tại hàm `moManHinhXuLyUocSo()`:

```
private void moManHinhXuLyUocSo(Intent intent) {
    Intent intent = Intent(MainActivity.this, XuLyUocSoActivity.class);
    intent.putExtra("N", Integer.parseInt(txtN.getText().toString()));
    startActivityForResult(intent, 113);
}
```

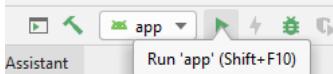
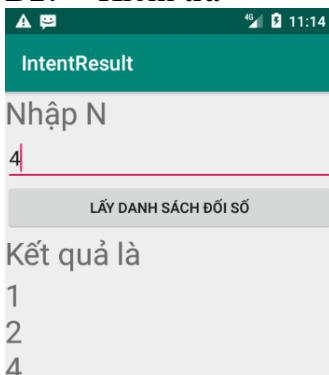
B10 - Trong `MainActivity.java`, bấm `Ctrl + C` vào `onActivityResult`:

```
protected void onActivityResult(int requestCode, @Nullable Intent data) {
    if (requestCode==113 && data!=null) {
        ArrayList<Integer> dsUS = data.getIntegerArrayListExtra("DSUS");
        txtKetQua.setText("");
        for (int us : dsUS) {
            txtKetQua.append(us + "\n");
        }
    }
}
```

B12 – Trong `XuLyUocSoActivity.java`, tại hàm `addControls()` và `addEvents()`:

B14 – Trong `XuLyUocSoActivity.java`, tại hàm `xuLyTrave()`:

```
private void xuLyTrave() {
    btnTrave.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            xuLyTrave();
        }
    });
}
```

<p>B16 – Run ‘app’</p>  <p>B17 – Kiểm tra</p> 	<p>B15 – Trong XuLyUocSoActivity.java, tại hàm nhật:</p> <pre> private void xuLyTr int n Integer.parseInt(txtN.getText().to ArrayList<Integer>dsUS=new for (int if dsUS.add(i); } intent.putExtra("DSUS",dsUS); setResult(114,intent); finish(); } </pre>
--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

3. BÀI TẬP

Bài 1: Viết chương trình tra cứu sản phẩm như hình dưới đây:

...

- Mặc định khi khởi động chương trình sẽ giả lập danh sách sản phẩm lên ListView/RecyclerView.
- Khi nhấn vào một sản phẩm bất kỳ thì dùng Intent để mở một Activity mới: hiển thị thông tin chi tiết của sản phẩm này:

...

Bài 2: Dùng Explicit và Implicit Intent để viết chương trình gọi điện thoại và nhắn tin SMS, giao diện chương trình như dưới đây:

...

Nhấn “save contact” để lưu thông tin vào ListView.

Nhấn vào danh bạ bất kỳ (ví dụ nhấn vào 5554), dùng Explicit intent để mở một Activity với ba chức năng: ...

- Call để gọi trực tiếp số này.
- Send SMS: để nhắn tin SMS tới số này, giao diện như hình dưới đây:
- ...
- Remove this contact: xoá danh bạ này và nhập nhật lại giao diện chính.

LAB 11. OPTION MENU – CONTEXT MENU – MENU ĐIỀU KHIỂN

1. MỤC TIÊU:

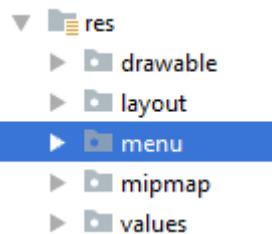
...

2. THỰC HIỆN

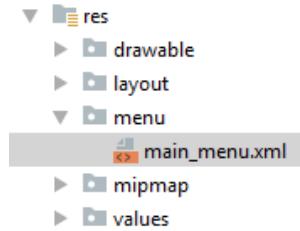
2.1. Bài 1: Option menu

B1 - Khởi động Android Studio → Start a new android project → Empty Activity → Name: **OptionMenu**

B2 – Tạo directory menu. Nhấp chuột phải lên folder **res** → new → directory → name: **menu**



B3 – Tạo **main_menu.xml**. Nhấp chuột phải lên folder **menu** → new → Menu resource file → name: **main_menu.xml**



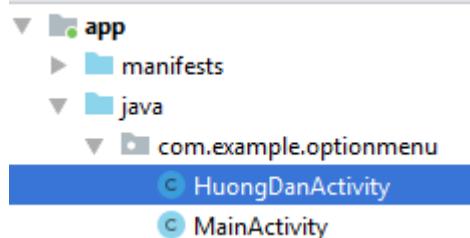
B4 – Trong **main_menu.xml**, thêm vào 2 menu item là Hướng dẫn và giới thiệu.

```
<item
    android:id="@+id/menuHuongDan"
    android:title="Hướng dẫn"
    ></item>
<item
    android:title="Giới thiệu"
    android:id="@+id/menuGioiThieu"
    ></item>
```

B6 – Thêm 1 **TextView** cho **activity_huong_dan.xml**. Trong **activity_huong_dan.xml** → Thêm **TextView** nội dung “Đây là màn hình hướng dẫn” Size 30sp.

```
android:text="Đây là màn hình
hướng
dẫn"
android:textSize="30sp"
```

B5 – Tạo thêm 1 Activity: **HuongDanActivity**. → Nhấp chuột phải lên com.example.optionmenu → new → activity → Empty activity → name: **HuongDanActivity**.



B7 – Trong **MainActivity.java**, cập nhật hàm khởi tạo menu **onCreateOptionsMenu**

B8 - Trong MainActivity.java , cập nhật hàm xử lý item trong menu khi được chọn: **onOptionsItemSelected**

```

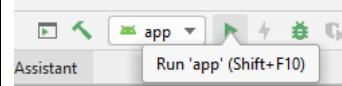
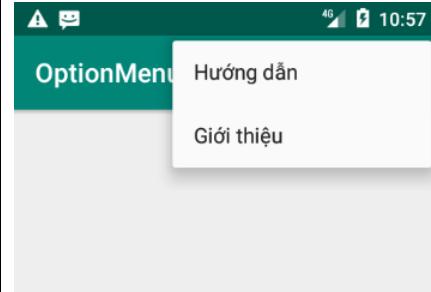
public boolean onOptionsItemSelected(@NonNull MenuItem item) {
    switch (item.getItemId()) {
        case R.id.menuHuongDan:
            Intent intent = new Intent(MainActivity.this, HuongDanActivity.class);
            startActivity(intent);
            break;
        case R.id.menuGioiThieu:
            break;
    }
    return super.onOptionsItemSelected(item);
}

```

```

public boolean onCreateOptionsMenu(Menu menu) {
    MenuInflater inflater = getMenuInflater();
    inflater.inflate(R.menu.main_menu, menu);
    return super.onCreateOptionsMenu(menu);
}

```

B9 – Run ‘app’**B10 – Kiểm tra:**

2.2. Bài 2: Context Menu

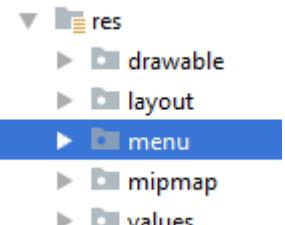
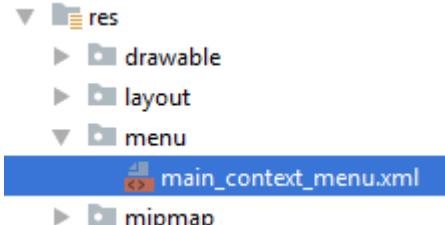
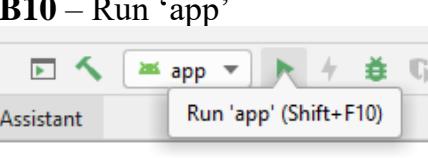
Sử dụng tiếp project trong bài 1 để thực hiện.

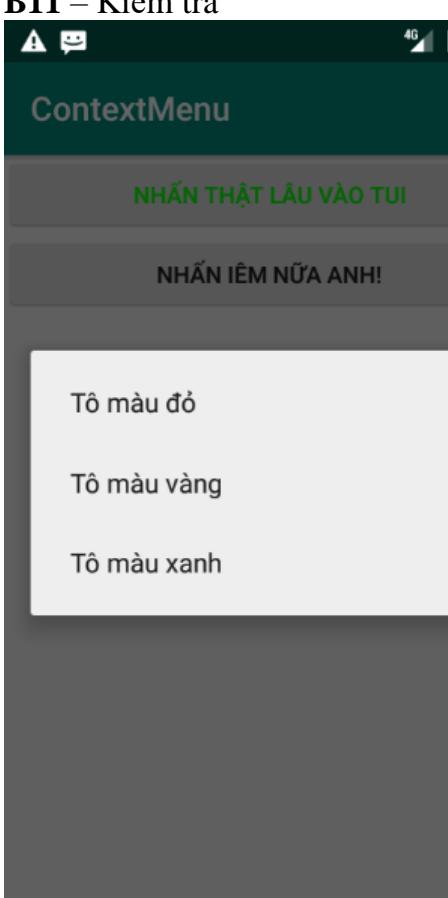
B1 – Trong **activity_main.xml** bổ sung thêm 1 nút bấm, nội dung: “**Bấm thật lâu vào tui**”, id là **btn1**.

```

<Button
    android:id="@+id	btn1"
    android:text="Nhấn thật lâu vào tui"
    android:layout_width="match_parent"
    android:layout_height="wrap_content" />
<Button

```

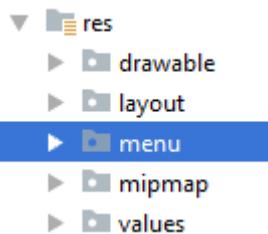
<p>B2 – Tạo directory menu. Nhấp chuột phải lên folder res → new → directory → name: menu</p> 	<p>B3 – Tạo main_context_menu.xml. Nhấp chuột phải lên folder menu → new → Menu resource file → name: main_context_menu.xml</p> 
<p>B4 – Trong main_context_menu.xml, bổ sung 3 item nội dung: Tô màu đỏ, tô màu vàng, tô màu xanh</p> <pre> <item android:id="@+id/menu_MauDo" android:title="Tô màu đỏ" ></item> <item android:id="@+id/menu_MauVang" android:title="Tô màu vàng" ></item> <item android:id="@+id/menu_MauXanh" android:title="Tô màu xanh" ></item> </pre>	<p>B5 – Trong MainActivity.java, khai báo Button btn1:</p> <pre>Button btn1;</pre> <p>B6 - Trong MainActivity.java, tại hàm onCreate(); thêm hàm addControls();</p> <pre>addControls();</pre> <p>B7 – Trong MainActivity.java, tại hàm addControls(); khai báo id của btn1 và đăng ký context menu cho btn1:</p> <pre> private void addControls() { btn1 findViewById(R.id.btn1); registerForContextMenu(btn1); }</pre>
<p>B10 – Run ‘app’</p> 	<p>B8 - Trong MainActivity.java, Override hàm onContextItemSelected, khai báo các case cho từng trường hợp chọn item của context menu:</p>

<p>B11 – Kiểm tra</p>  <p>NHẤN THẬT LÂU VÀO TUI NHẤN LÂM NỮA ANH!</p>	<pre>public boolean onContextItemSelected(@NotNull MenuItem item) { switch (item.getItemId()) { case R.id.menuMauDo: btn1.setTextColor(Color.RED); break; case R.id.menuMauVang: btn1.setTextColor(Color.YELLOW); break; case R.id.menuMauXanh: btn1.setTextColor(Color.GREEN); break; } return super.onContextItemSelected(item); }</pre>
	<p>B9 – Trong MainActivity.java, Override hàm onCreateContextMenu:</p> <pre>public void onCreateContextMenu(ContextMenu menu, View v, ContextMenu.ContextMenuItemInfo menuInfo) { MenuInflater inflater = getMenuInflater(); inflater.inflate(R.menu.main_context_menu, menu); super.onCreateContextMenu(menu, v, menuInfo); }</pre>

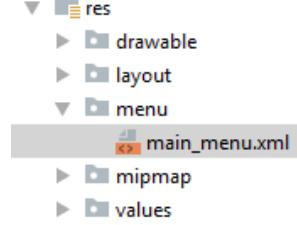
2.3. Bài 3: Menu tìm kiếm

B1 - Khởi động Android Studio → Start a new android project → Empty Activity → Name: **Option**

B2 – Tạo directory menu. Nhấp chuột phải lên folder res
→ new → directory → name: menu



B3 – Tạo main_menu.xml. Nhấp chuột
new → Menu resource file → name: ma



B4 - Trong main_context_menu.xml, bổ sung 3 item nội dung: Search, Help, About.

```
<item
    android:id="@+id/menuSearch"
    android:icon="@drawable/ic_search_black_24dp"
    android:title="Search"

    app:actionViewClass="android.widget.SearchView"
    app:showAsAction="always" />

<item
    android:title="Help"
    android:id="@+id/menuHelp" />

<item
    android:title="About"
    android:id="@+id/menuAbout"
    />
```

B5.1- Trong activity_main.xml, bổ sung một listview có id lvTinhThanh

```
<ListView
    android:id="@+id/lvTinhThanh"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
</ListView>
```

B9 – Trong MainActivity.java, override phương thức
onCreateOptionsMenu khởi tạo menuSearch. Hàm
onQueryTextChange có tác dụng giúp lọc các text khi người dùng
nhập vào ô tìm kiếm.

B5.2 – Sử dụng Google tìm
thêm vào mảng arrTinh'
strings.xml khai báo mảng

```
<string-array
    <item>An
    <item>Bà Rịa
    <item>Bắc
</string-array>
```

B6 - Trong MainActi
lvTinhThanh và một Array
tỉnh thành.

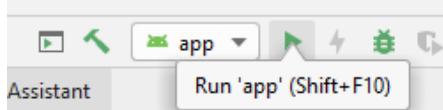
ListView
ArrayAdapter<Strin

B7 - Trong MainActivity.ja
addControls();

```

public boolean onCreateOptionsMenu(Menu menu) {
    MenuInflater inflater = getMenuInflater();
    inflater.inflate(R.menu.main_menu, menu);
    MenuItem menuSearch = menu.findItem(R.id.menuSearch);
    SearchView searchView= (SearchView)
    menuSearch.getActionView();
    searchView.setOnQueryTextListener(new
    SearchView.OnQueryTextListener() {
        @Override
        public boolean onQueryTextSubmit(String
s) {
            return false;
        }
        @Override
        public boolean onQueryTextChange(String
s) {
            adapterTinhThanh.getFilter().filter(s);
            return false;
        }
    });
    return super.onCreateOptionsMenu(menu);
}

```

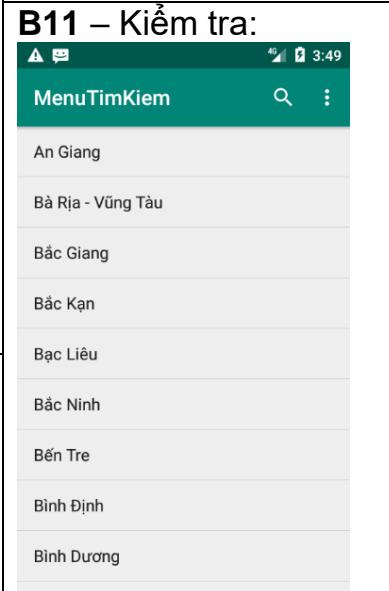
B10 – Run ‘app’

B8 - Trong MainActivity
mảng **ArrayAdapter** cho L

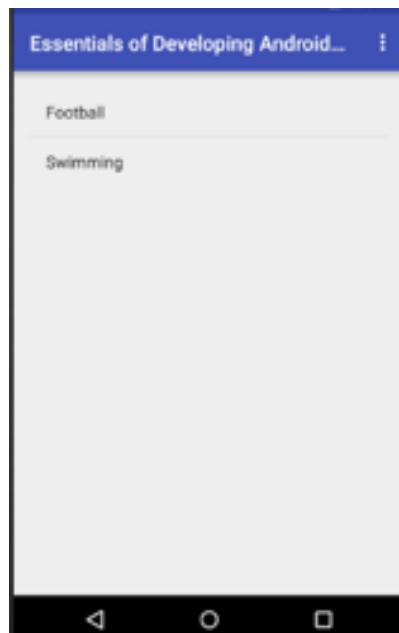
```

private void
    lvTinhThanh
    findViewById(R.id.
        adapterTinhThanh
        ArrayAdapter<String>
        MainAc
        android.R.layout.s
        adapterTinhThanh.a
        getStringArray(R.a
        lvTinhThanh.setAdapter
    }

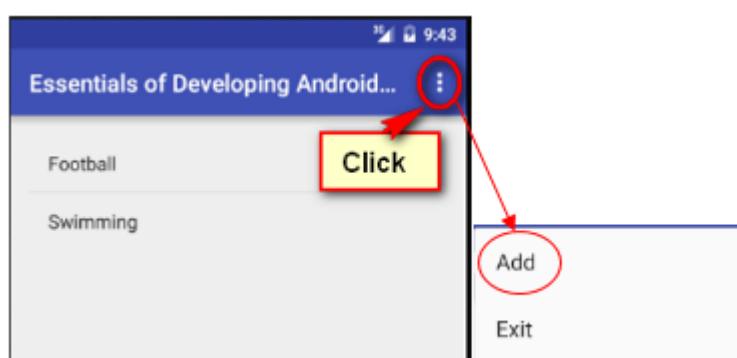
```

**3. BÀI TẬP**

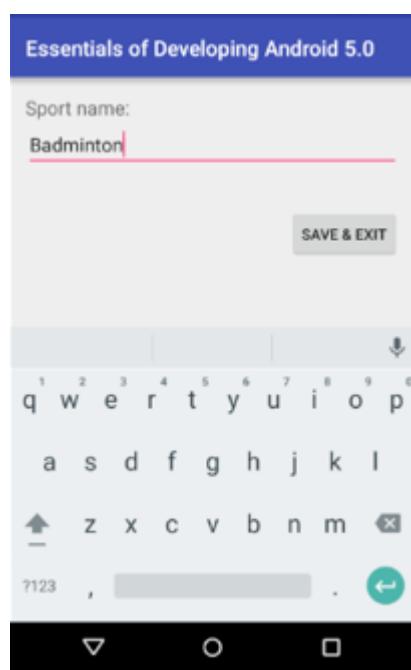
Bài 1: Sử dụng Option menu và context menu, tạo ứng dụng sau:



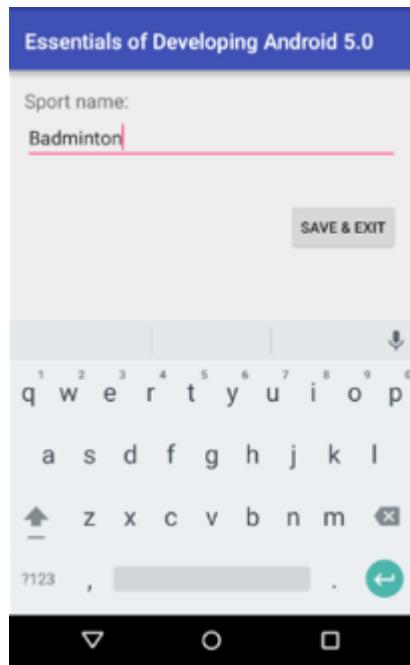
Thêm mới một môn thể thao



Màn hình nhập tên môn thể thao được hiển thị khi người dùng chọn Add



Xoá một môn thể thao (long click vào tên môn thể thao muốn xoá -> chọn Delete)



Thay đổi tên môn thể thao (long click vào tên môn thể thao muốn thay đổi -> chọn Edit -> Hiển thị màn hình cho phép thay đổi tên). Lưu ý tên môn thể thao mà người dùng đã chọn trước đó sẽ được hiển thị ở màn hình thay đổi.

Lưu ý:

- Tên môn thể thao phải được nhập trước khi chọn ‘Save & Exit’
- Hiển thị màn hình xác nhận khi người dùng chọn xử lý ‘Delete’. Nếu người dùng chọn ‘Yes’, xoá tên môn thể thao được chọn.
- Nâng cao: bổ sung thêm tính năng tìm kiếm (sử dụng menu tìm kiếm)

LAB 12. CƠ SỞ DỮ LIỆU SQLITE

1. MỤC TIÊU

- Làm quen với hệ quản trị CSDL SQLite
- Thực hiện các thao tác CRUD cơ bản
- Trích xuất CSDL ra thành file database
- Kết nối ứng dụng android với database SQLite.

2. THỰC HIỆN:

2.1. Bài 1: SQLite DB Browser.

B1 – Tải và cài đặt SQLite DB Browser → Truy cập link: <https://sqlitebrowser.org/dl/>
Chọn phiên bản Standard installer 32bit hoặc 64bit tùy vào hệ điều hành.

Downloads

Windows

Our latest release (3.11.2) for Windows:

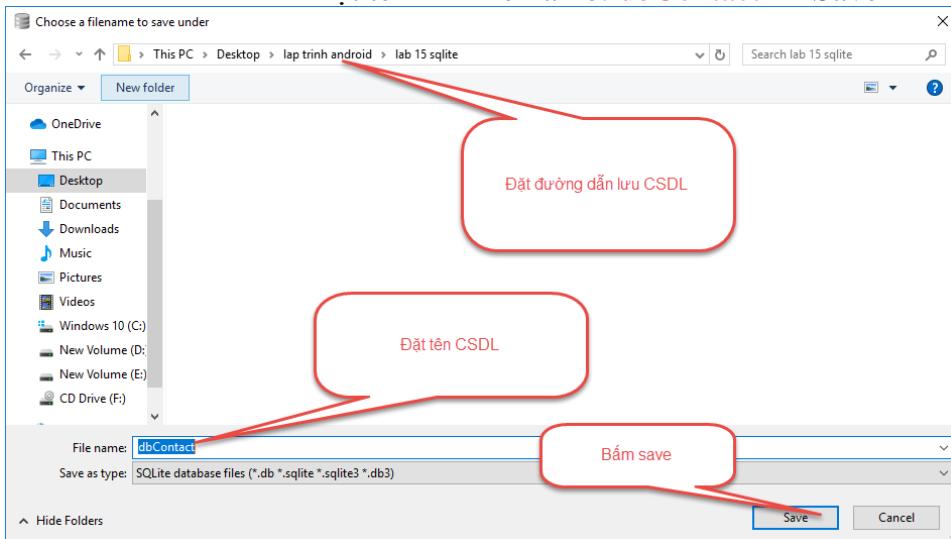
- DB Browser for SQLite – Standard installer for 32-bit Windows & Windows XP
- DB Browser for SQLite - .zip (no installer) for 32-bit Windows & Windows XP
- DB Browser for SQLite - Standard installer for 64-bit Windows
- DB Browser for SQLite - .zip (no installer) for 64-bit Windows
- DB Browser for SQLite - PortableApp

B2 – Tạo cơ sở dữ liệu.

Trong DB Browser for SQLite → bấm New Database

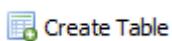


B3 – Chọn nơi lưu trữ CSDL và đặt tên → File name: dbContact → Save



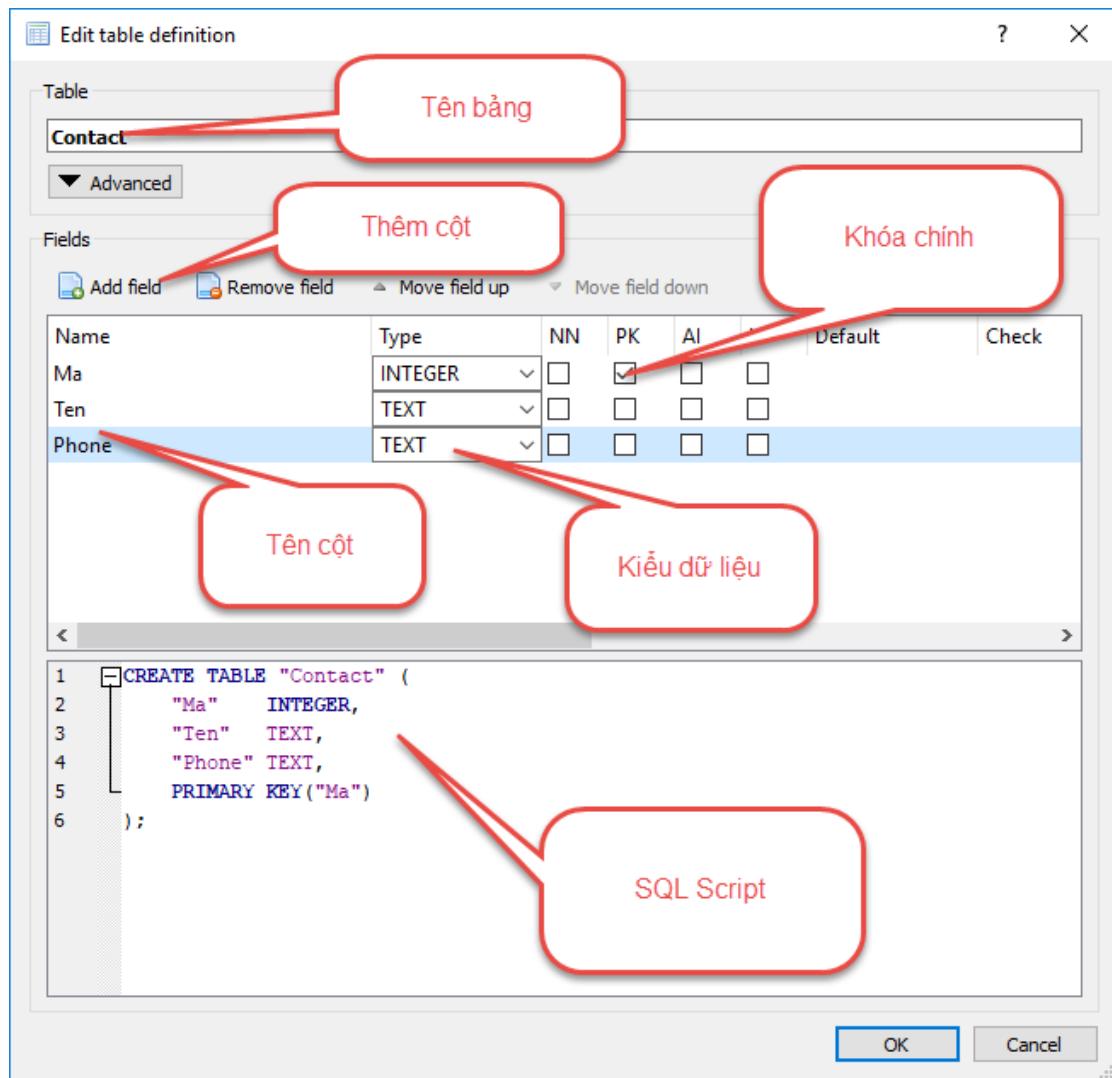
B4 – Tạo bảng.

Trong DB Browser for SQLite → bấm Create Table



B5 – Đặt tên bảng: Contact → Thêm các cột sau:

➔ **Add field:** Mã (khóa chính - kiểu int), Tên (kiểu text), Phone (kiểu text) → OK



B5 – Thêm các dòng dữ liệu mẫu

Trong DB Browser for SQLite → Browse Data → Table: Chọn bảng Contact → Thêm 6 dòng dữ liệu mẫu như hình.

	Ma	Ten	
	Filter	Filter	Filter
1	1	Putin	1234
2	2	Un	1234
3	3	Trump	1234
4	4	Bush	1234
5	5	Tony	1234
6	6	Captian	1234

MỘT SỐ THAO TÁC CRUD (CREATE, READ, UPDATE, DELETE)

- Xem toàn bộ dữ liệu có trong bảng.

Trong DB Browser for SQLite → Execute SQL
→ Gõ lệnh: `Select * From Contact`

The screenshot shows the DB Browser for SQLite interface. In the top menu, 'Execute SQL' is highlighted. Below it, the SQL editor contains the command `SELECT * FROM Contact`. A red box highlights the 'Execute SQL' button, with a callout 'Thực thi SQL Script'. The results pane shows a table with columns Ma, Ten, and Phone, containing five rows of data. A red box highlights the table, with a callout 'Kết quả trả về' (Result). At the bottom, the status bar shows 'Result: 6 rows returned in 6ms' and the executed query.

- Xóa 1 dòng dữ liệu có trong bảng

Trong DB Browser for SQLite → Execute SQL
→ Gõ lệnh: `Delete * From Contact Where ma = 3`

The screenshot shows the DB Browser for SQLite interface. In the top menu, 'Execute SQL' is highlighted. Below it, the SQL editor contains the command `DELETE FROM Contact WHERE ma = 3`. A red box highlights the entire query line. The results pane shows a message 'Result: query executed successfully. Took 1ms, 1 rows affected' and the executed query. A red box highlights this message, with a callout 'Kết quả thực hiện truy vấn' (Result of execution).

- Thêm một dòng dữ liệu vào bảng.

Trong DB Browser for SQLite → Execute SQL
→ Gõ lệnh: `Inset Into Contact ('Ma','Ten','Phone')Value(3,'Trump','1234')`

The screenshot shows the DB Browser for SQLite interface. In the top menu, 'Execute SQL' is highlighted. Below it, the SQL editor contains the command `INSERT INTO Contact ('Ma','Ten','Phone')VALUES(3,'Trump','1234')`. A red box highlights the entire query line. The results pane shows a message 'Result: query executed successfully. Took 0ms, 1 rows affected' and the executed query. A red box highlights this message, with a callout 'Kết quả thực hiện truy vấn' (Result of execution).

- Cập nhật một dòng dữ liệu trong bảng.

Trong DB Browser for SQLite → Execute SQL
→ Gõ lệnh: `Update Contact Set Ten='Donald Trump' Where Ma='3'`

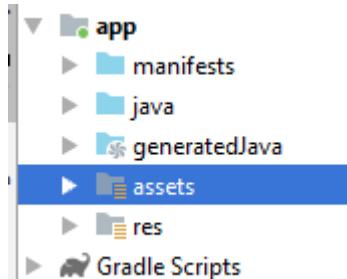
The screenshot shows the DB Browser for SQLite interface. In the top menu, 'Execute SQL' is highlighted. Below it, the SQL editor contains the command `UPDATE Contact SET Ten='Donald Trump' WHERE Ma='3'`. A red box highlights the entire query line. The results pane shows a message 'Result: query executed successfully. Took 0ms, 1 rows affected' and the executed query. A red box highlights this message, with a callout 'Kết quả thực hiện truy vấn' (Result of execution).

2.2. Bài 2: Sao chép CSDL từ Asset vào thiết bị Android.

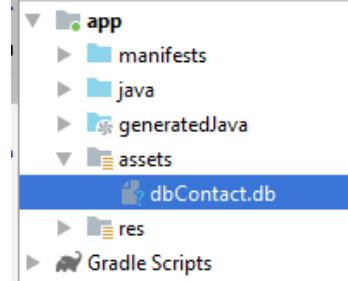
B1 - Khởi động Android Studio → Start a new android project → Empty Activity → Name: **OptionMenu**

B2 – Tạo thư mục assets.

Chuột phải vào app → new → folder → assets folder → finish



B3 – Dán CSDL vừa tạo trong Bài 1 vào thư mục assets



COPY CƠ SỞ DỮ LIỆU TỪ PROJECT → ĐIỆN THOẠI

B4 – Trong MainActivity.java, khai báo các biến sau:

```
public static String DATABASE_NAME="dbContact.db";
public static String DB_PATH_SUFFIX="/databases/";
```

B6 – Trong MainActivity.java, tại hàm processCopy bổ sung thêm hàm copyDatabaseFromAsset();

```
private void processCopy() {
    try {
        File dbFile=getDatabasePath(DATABASE_NAME);
        if (!dbFile.exists()) {
            copyDatabaseFromAsset();
            Toast.makeText(MainActivity.this,
                    "Sao chép CSDL vào Đt Thành công",
                    Toast.LENGTH_LONG).show();
        }
    }
```

B5 - Trong MainActivity.java, tại hàm onCreate(); bổ sung hàm sau:

```
processCopy();
```

B7 - Trong MainActivity.java, tại hàm copyDatabaseFromAsset(); bổ sung:

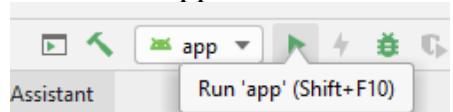
```
private String getDatabasePath() {
    return getApplicationInfo()
        .dataDir+DB_PATH_SUFFIX+DATABASE_NAME;
}

private void copyDatabaseFromAsset() {
    try {
        InputStream myInput=getAssets().open(DATABASE_NAME);
        String outFileName=getDatabasePath();
        File f=new File(getApplicationInfo().dataDir+DB_PATH_SUFFIX);
        if (!f.exists())
            f.mkdir();
        OutputStream myOutput = new FileOutputStream(outFileName);
        byte []buffer=new byte[1024];
        int length;
        while ((length=myInput.read(buffer))>0) {
            myOutput.write(buffer, 0, length);
        }
    }
```

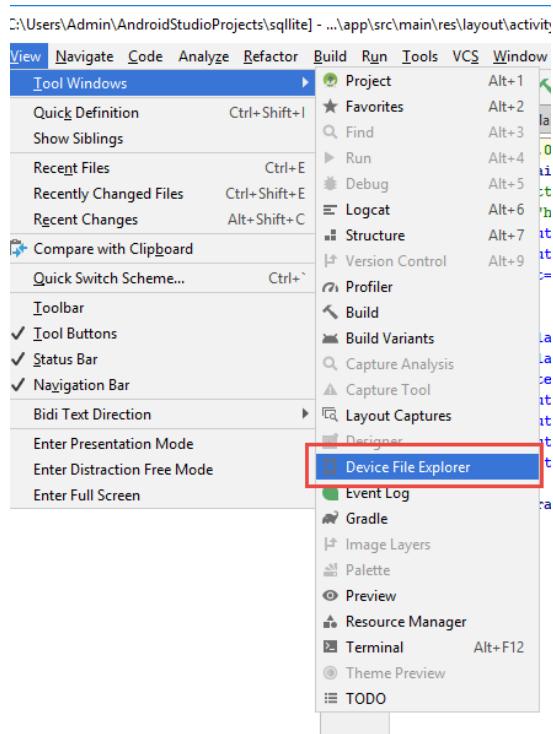
```
catch (Exception ex)
{
    Toast.makeText(MainActivity.this,
    ex.toString(), Toast.LENGTH_LONG).show();
    Log.e("Loi", ex.toString());
}

myOutput.flush();
myOutput.close();
myInput.close();
}

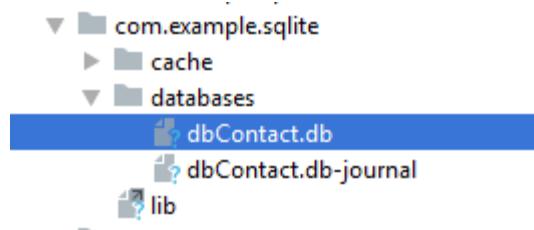
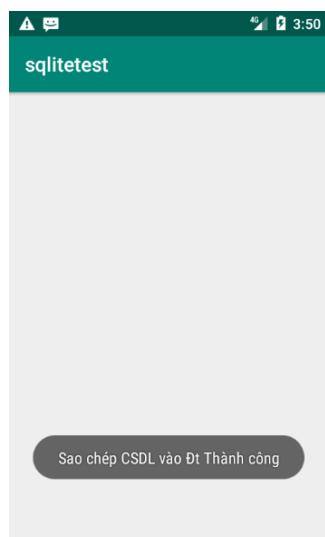
catch (Exception ex)
{
    Log.e("Loi", ex.toString());
}
```

B8 – Run ‘app’

B10 – Kiểm tra CSDL có tồn tại trên điện thoại bằng Device File Explore.
Trong Android Studio truy cập:
View → Tool Windows → Device File Explorer



B12 – Trong Device File Explorer tại com.example.sqlite → databases
Thấy có file dbContact.db

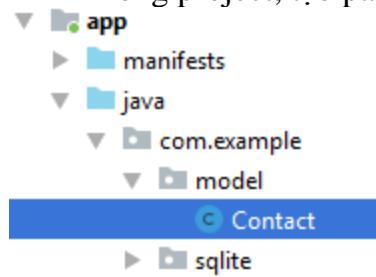
**B9 – Kiểm tra**

B11 – Trong Device File Explorer truy cập đường dẫn data → data → com.example.sqlite → databases
Thấy có file dbContact.db

Name	Permissions	Date
acct	drwxr-xr-x	2019-07-27 17
cache	drwxrwx---	2019-07-17 00
config	dr-x-----	2019-07-27 17
d	lrwxrwxrwx	2019-07-27 17
data		2019-07-02 10
adb	drwx-----	2019-07-02 10
app	drwxrwx--x	2019-07-27 17
app-asec	drwx-----	2019-07-02 10
app-lib	drwxrwx--x	2019-07-02 10
app-private	drwxrwx--x	2019-07-02 10
backup	drwx-----	2019-07-27 17
dalvik-cache	drwxrwx--x	2019-07-02 10
data		2019-07-27 17
com.android.backupconfirm	drwxr-x--x	2019-07-02 10
com.android.browser	drwxr-x--x	2019-07-15 13
com.android.calculator2	drwxr-x--x	2019-07-02 10
com.android.camera	drwxr-x--x	2019-07-02 10
com.android.captiveportallogin	drwxr-x--x	2019-07-02 10

2.3. Bài 3: Truy vấn SQLite trong Android.

B1 – Trong project, tạo package **model** và class **Contact**: com.example.model → Contact.java



B2 – Trong **Contact.java**, khai báo các biến bên dưới → Tạo **getter and setter** và **constructor** cho các biến vừa tạo.

```
private int ma;
private String ten;
private String phone;
```

B3 – Trong **Contact.java**, khai báo hàm **toString** để trả về các giá trị **ma, tên, phone**.

```
public String toString() {
    return this.ma+" - "+this.ten+"\n"+this.phone;
}
```

B4 – Trong **MainActivity.java**, khai báo một database = null, một ListView lvContact, một ArrayAdapter chứa Contact tên là adapter.

```
public static SQLiteDatabase database = null;
ListView lvContact;
ArrayAdapter<Contact> adapter;
```

B5 – Trong **MainActivity.java**, tại hàm **onCreate**, khai báo các hàm

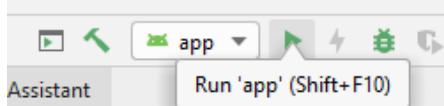
```
addControls();
hienThiToanBoSanPham();
```

B6 – Trong **MainActivity.java**, tại hàm **addControl**, khởi tạo ListView lvContact và adapter **Contact** → gán adapter Contact cho ListView lvContact

```
private void addControls() {
    lvContact=findViewById(R.id.lvContact);
    adapter=new ArrayAdapter<Contact>(MainActivity.this,
        android.R.layout.simple_list_item_1);
    lvContact.setAdapter(adapter);
}
```

B7 – Trong `ActivityMain.java`, tại hàm `hienThiToanBoSanPham`, khởi tạo `database` và `cursor` để đọc dữ liệu từ `database` → lấy dữ liệu này gán vào biến `contact` → lấy biến `contact` này đem gán cho `adapter`. Sau khi thực hiện xong dùng lệnh `cursor.close()`; để đóng cursor.

```
private void hienThiToanBoSanPham() {
    database=openOrCreateDatabase(DATABASE_NAME, MODE_PRIVATE, null);
    Cursor cursor=database.rawQuery("select * from Contact", null);
    adapter.clear();
    while (cursor.moveToNext()) {
        int ma=cursor.getInt(0);
        String ten=cursor.getString(1);
        phone=cursor.getString(2);
        Contact contact=new Contact(ma,ten,phone);
        adapter.add(contact);
    }
    cursor.close();
}
```

B7 – Run ‘app’**B8 – Kiểm tra:**

2.4. Bài 4: Thêm dữ liệu vào SQLite trong Android.

Thực hành bổ sung option menu “Thêm” contact cho chương trình danh bạ điện thoại.

B1 – Trong project, tạo Activity `ThemContactActivity.java`.

Com.example.sqlite → ThemContactActivity.java

B2 – Trong project, tại folder res → tạo thư mục menu → trong thư mục menu tạo Menu resource file đặt tên là menu_main.xml

B3 – Trong menu_main.xml, tạo một item có tiêu đề “Thêm Contact”, id là mnuThem.

```
<item
    android:id="@+id/mnuThem"
    android:title="Thêm Contact" />
```

ThemContactActivity.java

B4 – Trong ThemContactActivity.java, khai báo các biến sau:

EditText **edtMa, edtTen, edtPhone;**

B5 – Trong ThemContactActivity.java, tại hàm onCreate(), khai báo hàm

addControls();

B5 – Trong ThemContactActivity.java, tại hàm addControl(), cập nhật các EditText

```
private void addControls()
{
    edtMa=findViewById(R.id.edtMa);
    edtTen=findViewById(R.id.edtTen);
    edtPhone=findViewById(R.id.edtPhone);
}
```

B6 – Trong ThemContactActivity.java, tại hàm xuLyLuu(), sau khi thêm sẽ insert vào trong bảng và có hàm if để kiểm tra tình trạng thêm có thành công hay không.

```
public void xuLyLuu(View view) {
    int ma = Integer.parseInt(edtMa.getText().toString());
    String ten = edtTen.getText().toString();
    String phone = edtPhone.getText().toString();

    ContentValues values=new ContentValues();
    values.put("Ma",ma);
    values.put("Ten",ten);
    values.put("Phone",phone);

    //thêm dòng dữ liệu vào bảng Contact
    long kq=MainActivity.database.insert("Contact",null,values);
```

```

if (kq>0) {
    Toast.makeText(ThemContactActivity.this, "Thêm
thành công", Toast.LENGTH_LONG).show();
} else {
    Toast.makeText(ThemContactActivity.this, "Thêm thất
bại", Toast.LENGTH_LONG).show();
}
}

```

B6 – Trong **ThemContactActivity.java**, tại hàm **xuLyTiep()**, cập nhật xử lý cho nút **Tiếp**.

```

public void xuLyTiep(View
view) {
    edtMa.setText("");
    edtTen.setText("");
    edtPhone.setText("");
    edtMa.requestFocus();
}

```

B7 – Trong **ThemContactActivity.java**, tại hàm **xuLyDong()**, cập nhật xử lý cho nút **Đóng**.

```

public void xuLyDong(View
view) {
    finish();
}

```

MainActivity.java

B8 – Trong **MainActivity.java**, override hàm **onCreateOptionsMenu** để khởi tạo Option Menu.

```

public boolean onCreateOptionsMenu(Menu menu) {
    getMenuInflater().inflate(R.menu.menu_main,menu);
    return super.onCreateOptionsMenu(menu);
}

```

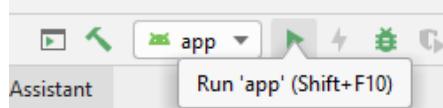
B9 – Trong `MainActivity.java`, override hàm `onOptionsItemSelected` để khởi tạo sự kiện khi bấm chọn vào item trong Option Menu có id `mnuThem`

```
public boolean onOptionsItemSelected(@NotNull MenuItem item) {
    if (item.getItemId() == R.id.mnuThem) {
        Intent intent = new Intent(MainActivity.this, ThemContactActivity.class);
        startActivity(intent);
    }
    return super.onOptionsItemSelected(item);
}
```

B10 – Trong `MainActivity.java`, override hàm `onResume` và chèn thêm hàm `hienThiToanBoSanPham` để khi resume lại chương trình tự động nạp lại ListView.

```
protected void onResume() {
    super.onResume();
    hienThiToanBoSanPham();
}
```

B11 – Run ‘app’



B12 – Kiểm tra



Bài 5: Cập nhật dữ liệu vào SQLite trong Android.

B1 - Trong project, tạo Activity `SuaContactActivity.java`.

Com.example.sqlite → SuaContactActivity.java

B2 - Trong folder menu, tạo `context_menu.xml`. Trong `context_menu.xml`, cập nhật 2 item là “Chỉnh sửa”, id `mnuEdit` và item Xóa id `mnuDelete`.

```
<item
    android:id="@+id/mnuEdit"
    android:title="Chỉnh sửa"
/>
<item
    android:id="@+id/mnuDelete"
    android:title="Xóa"
/>>
```

SuaContactActivity.java

B3 – Trong `SuaContactActivity.java`, khai báo các biến sau:

```
EditText
edtMa, edtTen, edtPhone;
```

B4 – Trong `SuaContactActivity.java`, tại hàm `onCreate`, bổ sung hàm

```
addControls();
```

B5 – Trong `MainActivity.java`, khai báo biến sau:

public static Contact selectedContacts;

B6 – Trong `SuaContactActivity.java`, tại hàm `addControls`, cập nhật:

```
private void addControls() {
    edtMa = findViewById(R.id.edtMa);
    edtTen = findViewById(R.id.edtTen);
    edtPhone = findViewById(R.id.edtPhone);
    edtMa.setText(MainActivity.selectedContacts.getMa() + "");
    edtTen.setText(MainActivity.selectedContacts.getTen() + "");
    edtPhone.setText(MainActivity.selectedContacts.getPhone() + "");
    edtMa.setEnabled(false);
}
```

B7 – Trong `SuaContactActivity.java`, bổ sung code xử lý cho hàm `xuLyLuu`:

```
public void xuLyLuu(View view) {
    //update thi không put khóa chính
    ContentValues values=new ContentValues();
    values.put("Ten", edtTen.getText().toString());
    values.put("Phone", edtPhone.getText().toString());
    int kq=MainActivity.database.update("Contact", values, "Ma=?", new String[]{edtMa.getText().toString()});
    if (kq>0) {
        Toast.makeText(SuaContactActivity.this, "Sửa thành
```

```

công", Toast.LENGTH_SHORT).show();
        finish(); }
else{
    Toast.makeText(SuaContactActivity.this, "Sửa thất bại",
Toast.LENGTH_LONG).show(); }
}

```

B8 – Trong **SuaContactActivity.java**, bổ sung code xử lý cho hàm **xuLyDong**;

```

public void xuLyDong(View view) {
    finish();
}

```

MainActivity.java

B9 – Trong **MainActivity.java**, tại hàm **onCreate**, bổ sung hàm **addEvents**.

```
addEvents();
```

B9 – Trong **MainActivity.java**, tại hàm **addEvents**, bổ sung đoạn code xử lý xác định item nào được bấm chọn.

```

private void addEvents () {
    lvContact.setOnItemClickListener(new AdapterView.OnItemClickListener() {
        @Override
        public void onItemClick(AdapterView<?> adapterView,
View view, int i, long l) {
            //lấy item thứ i
            selectedContacts=adapter.getItem(i);
        }
    });
    lvContact.setOnItemLongClickListener(new AdapterView.OnItemLongClickListener() {
        @Override
        public boolean onItemLongClick(AdapterView<?>
adapterView, View view, int i, long l) {
            selectedContacts=adapter.getItem(i);
            return false;
        }
    });
}

```

B10 – Trong **MainActivity.java**. tại hàm **addControl**, bổ sung phương thức gán Context menu cho ListView **lvContact**.

```
registerForContextMenu(lvContact);
```

B11 – Trong `MainActivity.java`, override hàm `onCreateContextMenu` để khởi tạo Context menu.

```
public void onCreateContextMenu(ContextMenu menu, View v,
ContextMenu.ContextMenuItemInfo menuInfo) {
    getMenuInflater().inflate(R.menu.context_menu, menu);
    super.onCreateContextMenu(menu, v, menuInfo);
}
```

B12 – Trong `MainActivity.java`, override hàm `onContextItemSelected` để xử lý cho sự kiện bấm vào item trong Context menu.

```
public boolean onContextItemSelected(@NonNull MenuItem item) {
    if (item.getItemId() == R.id.mnuEdit) {
        //nếu selectedContacts khác null thì gọi intent mở activity
        if (selectedContacts != null)
            {
                Intent intent = new Intent(MainActivity.this, SuaContactActivity.class);
                startActivity(intent);
            }
    } else if (item.getItemId() == R.id.mnuDelete) {
        if (selectedContacts != null)
            xuLyXoa();
    }
    return super.onContextItemSelected(item);
}
```

B13 – Run ‘app’



B14 – Kiểm tra:



Bài 6: Xóa dữ liệu SQLite trong Android.

B1 – Trong **MainActivity.java**, tại hàm **xuLyXoa**, cập nhật đoạn code xử lý xóa. Khi bấm xóa sẽ có hộp thoại hiện ra hỏi có muốn xóa hay không.

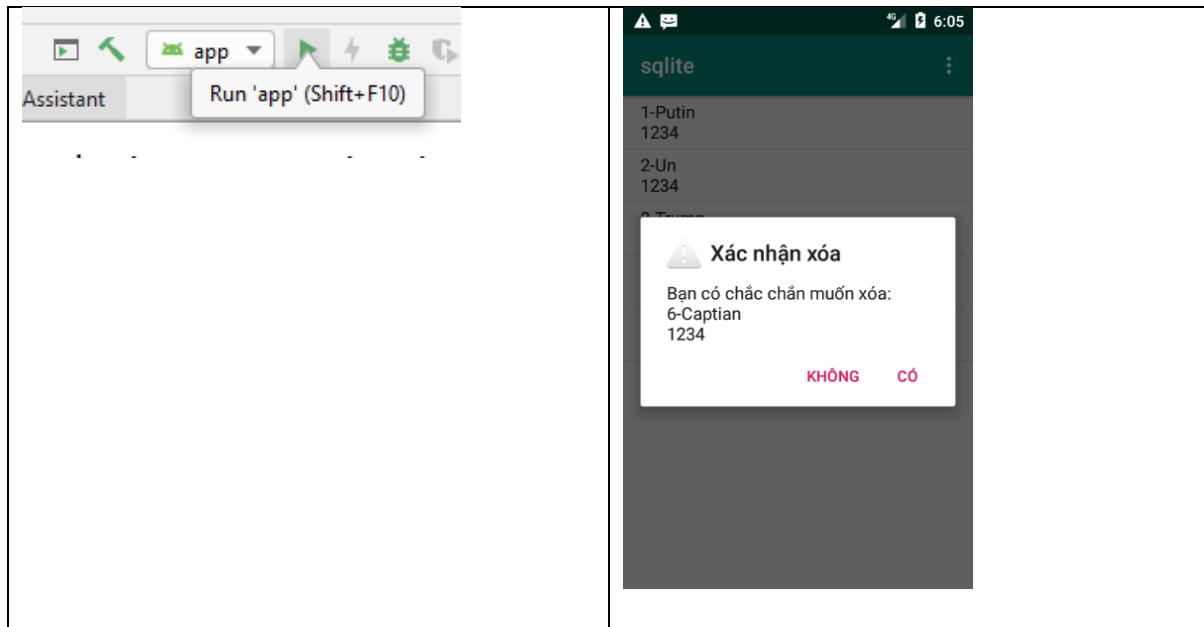
```

private void xuLyXoa() {
    AlertDialog.Builder builder = new
    AlertDialog.Builder(MainActivity.this);
    builder.setTitle("Xác nhận xóa");
    builder.setMessage("Bạn có chắc chắn muốn xóa: \n" + selectedContacts);
    builder.setIcon(android.R.drawable.ic_dialog_alert);
    //Xử lý hành động của nút có
    builder.setPositiveButton("Có",
        DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialogInterface,
                int i) {
                int kq = database.delete("Contact", "Ma=?", new
                String[] {selectedContacts.getMa() + ""});
                if (kq > 0) {
                    Toast.makeText(MainActivity.this, "Xóa thành công",
                        Toast.LENGTH_LONG).show();
                    hienThiToanBoSanPham();
                }
                else
                    Toast.makeText(MainActivity.this, "Xóa thất bại",
                        Toast.LENGTH_LONG).show();
            }
        });
    //Xử lý hành động của nút không
    builder.setNegativeButton("Không",
        DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialogInterface,
                int i) {
                    dialogInterface.dismiss();
                }
            });
    //Khởi tạo
    builder.create().show();
}

```

B2 – Run ‘app’

B3 – Kiểm tra



3. BÀI TẬP:

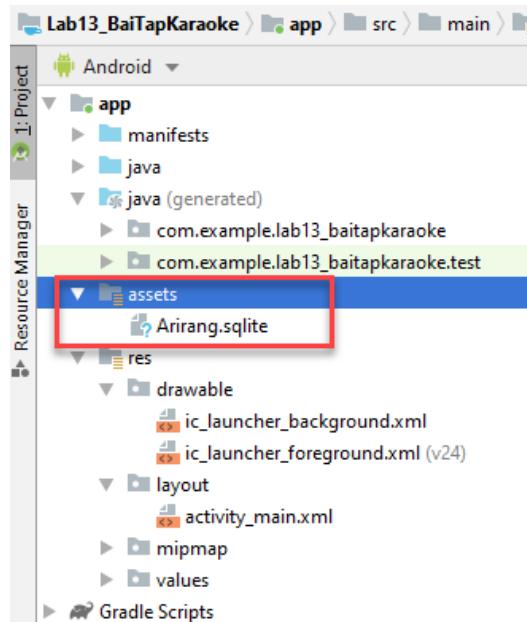
3.1. Viết phần mềm karaoke cho cơ sở dữ liệu karaoke bằng sqlite, viết chương trình hiển thị như hình dưới đây:

Sử dụng tab host để lưu trữ hai danh sách bài hát: toàn bộ, yêu thích. Mỗi dòng hiển thị ta có ImageView để thích hay không thích bài hát, nếu chọn thích thì bài hát được đưa vào tab Love, nếu chọn không thích thì bài hát quay trở lại tab tất cả.



3.2. Gợi ý cách làm:

B1-Tạo Assets Folder trong thư mục app và chép CSDL Arriang.sqlite vào thư mục assets.



B2-Tạo tab host tại activity_main.xml

```
<TabHost
    android:id="@+id/tabHost"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:orientation="vertical">

        <TabWidget
            android:id="@android:id/tabs"
            android:layout_width="match_parent"
            android:layout_height="wrap_content" />

        <FrameLayout
            android:id="@android:id/tabcontent"
            android:layout_width="match_parent"
            android:layout_height="match_parent">

            <LinearLayout
                android:id="@+id/tab1"
                android:layout_width="match_parent"
                android:layout_height="match_parent"
```

```
        android:orientation="vertical">

        <ListView
            android:id="@+id/lvAll"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
        />
    </LinearLayout>

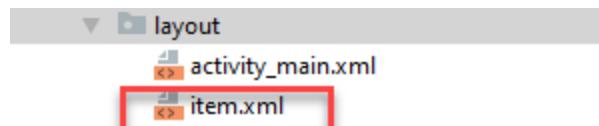
    <LinearLayout
        android:id="@+id/tab2"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:orientation="vertical">

        <ListView
            android:id="@+id/lvLove"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
        />
    </LinearLayout>

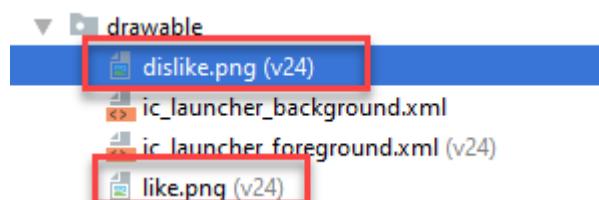
    </FrameLayout>
</LinearLayout>

</TabHost>
```

B3-Tạo item.xml trong folder layout (New → Layout Resource File)



B4-Thêm like.png và dislike.png vào thư mục drawable



B5-Điều chỉnh giao diện trong item.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
```

```
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="horizontal">

    <TextView
        android:id="@+id/txtMa"
        android:layout_width="90dp"
        android:layout_height="90dp"
        android:text="99999"
        android:textAlignment="center"
        android:textColor="#F44336"
        android:textSize="28sp"
        android:textStyle="bold" />

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:orientation="vertical">

        <TextView
            android:id="@+id/txtTen"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="TextView"
            android:textColor="#3F51B5"
            android:textSize="10pt"
            android:textStyle="bold" />

        <TextView
            android:id="@+id/txtCaSy"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="TextView"
            android:textSize="10pt" />

        <LinearLayout
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:orientation="horizontal">

            <ImageView
                android:id="@+id/imgLike"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_marginRight="15sp"
                app:srcCompat="@drawable/like" />

```

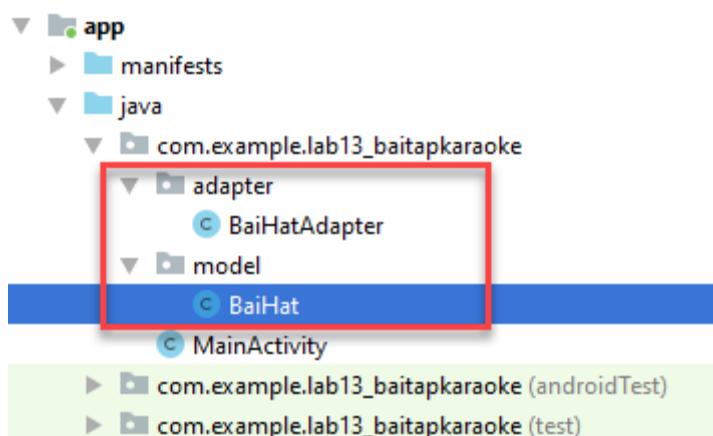
```

<ImageView
    android:id="@+id/imgDisLike"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    app:srcCompat="@drawable/dislike"      />
</LinearLayout>
</LinearLayout>

</LinearLayout>

```

B6-Tạo lớp BaiHatAdapter.class thuộc package adapter và lớp BaiHat.class thuộc package model.



B7-Mở MainActivity.java, thêm các hàm processCopy(), setupTabHost(),addControls(),hienThiToanBoBaiHat vào onCreate

B8- Viết các hàm sao chép CSDL vào điện thoại (processCopy, getDatabasePath, copyDataBaseFromAssets).

...

B9-Viết hàm cài đặt tab host

```

private void setupTabHost() {
    //cài đặt tabHost
    tabHost=findViewById(R.id.tabHost);
    tabHost.setup();

    TabHost.TabSpec tab1=tabHost.newTabSpec("tab1");
    tab1.setContent(R.id.tab1);
    tab1.setIndicator("Tất cả");
    tabHost.addTab(tab1);

    TabHost.TabSpec tab2=tabHost.newTabSpec("tab2");

```

```
tab2.setContent(R.id.tab2) ;  
tab2.setIndicator("Yêu  
tabHost.addTab(tab2) ;  
  
}  
}
```

B10-Viết hàm addControls();

...

B11-Viết hàm hiển thị toàn bộ bài hát (hienThiToanBoBaiHat)

...

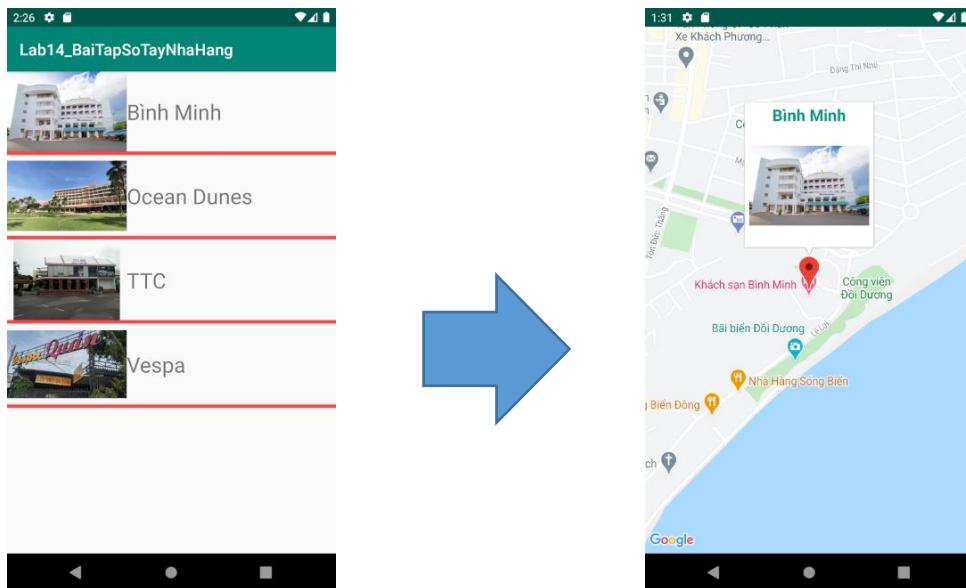
LAB 13. GOOGLE MAP

1. MỤC TIÊU

- Tạo google maps api key
- Viết phần mềm sốt tay nhà hàng.

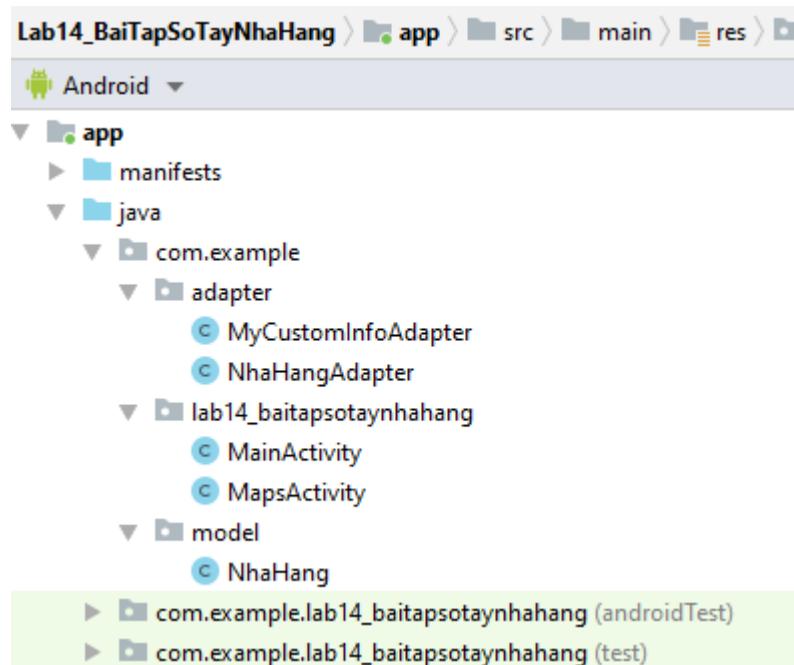
2. THỰC HÀNH

Ví dụ: phần mềm sốt tay nhà hàng



B1 – Trong Anroid Studio, tạo các package và các lớp như sau:

- Com.example.adapter: chứa class MycustomInfoAdapter và class NhaHangAdapter
- Com.example.lab14_baitapsotaynhahang: chứa MainActivity và MapsActivity (google maps activity)
- Com.example.model: chứa class NhaHang



B2 – Trong activity_main.xml, tạo ra một listView có id là lvNhaHang. Mã xml như sau:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <ListView
        android:id="@+id/lvNhaHang"
        android:layout_width="match_parent"
        android:layout_height="match_parent" />
</LinearLayout>
```

B3-Trong class NhaHang, khai báo các thuộc tính tên (String), hinh (int), viDo (float), kinhDo . Cụ thể như sau:

```
package com.example.model;

import java.io.Serializable;

public class NhaHang implements Serializable {
```

```

    private String ten;
    private int hinh;
    private float viDo;
    private float kinhDo;

}

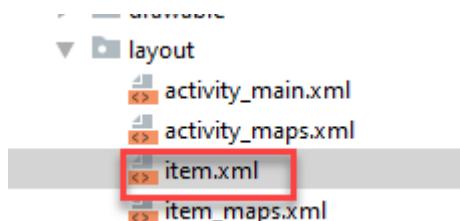
```

Bạn tiếp tục tạo ra các getter, setter, constructor đầy đủ đối số và không có đối số cho các thuộc tính kể trên.

B4-Tiếp theo, ta sẽ tạo custom layout để hiển thị danh sách nhà hàng.



Bạn bấm chuột phải vào folder layout → New Layout Resource File → đặt tên là item.



Mã xml như sau:

```

<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="horizontal">

        <ImageView
            android:id="@+id/imgHinh"
            android:layout_width="150dp"
            android:layout_height="100dp"

            tools:srcCompat="@tools:sample/avatars" />

```

```

<TextView
    android:id="@+id/txtTen"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_gravity="center"
    android:text="TextView"
    android:textSize="12pt" />
</LinearLayout>

<TextView
    android:id="@+id/textView2"
    android:background="@android:color/holo_red_light"
    android:layout_width="match_parent"
    android:layout_height="4dp" />
</LinearLayout>

```

B5-Trong lớp **NhaHangAdapter**, tạo ra ArrayAdapter để đầy các resource lên listView.

```

package com.example.adapter;

import android.app.Activity;
import android.content.Context;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ArrayAdapter;
import android.widget.ImageView;
import android.widget.TextView;

import androidx.annotation.NonNull;
import androidx.annotation.Nullable;

import com.example.lab14_baitapsotaynhahang.R;
import com.example.model.NhaHang;

public class NhaHangAdapter extends ArrayAdapter<NhaHang>
{
    Activity context;
    int resource;

    public NhaHangAdapter(@NonNull Activity context, int resource)
    {
        super(context, resource);
        this.context=context;
        this.resource=resource;
    }
}

```

```

@NonNull
@Override
public View getView(int position, @Nullable View convertView, @NonNull ViewGroup parent) {
    View view=context.getLayoutInflater().inflate(resource,null);
    ImageView imgHinh=view.findViewById(R.id.imgHinh);
    TextView txtTen=view.findViewById(R.id.txtTen);
    NhaHang nh=getItem(position);
    imgHinh.setImageResource(nh.getHinh());
    txtTen.setText(nh.getTen());
    return view;
}
}

```

B6-Trong MainActivity, khai báo:

ListView lvNhaHang, NhaHangAdapter

Hàm: addControls(), addEvents(),

Hàm giả lập dữ liệu: fakeData().

Lưu ý:

Tìm dữ liệu hình ảnh về nhà hàng và thêm vào thư mục drawable.

Dùng google maps copy tọa độ (kinh độ và vĩ độ) của địa điểm.

Cụ thể như sau:

```

package com.example.lab14_baitapsotaynhahang;

import androidx.appcompat.app.AppCompatActivity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.AdapterView;
import android.widget.ListView;

import com.example.adapter.NhaHangAdapter;
import com.example.model.NhaHang;

public class MainActivity extends AppCompatActivity {
    ListView lvNhaHang;
    NhaHangAdapter nhaHangAdapter;
}

```

```

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    addControls();
    addEvents();
    fakeData();
}

private void addEvents() {
    //hàm xử lý sự kiện
}

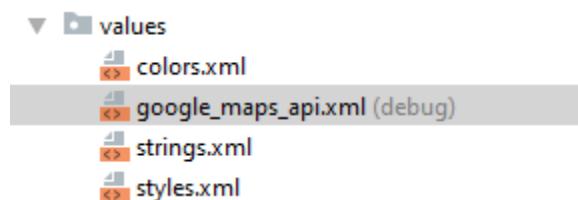
private void fakeData() {
    nhaHangAdapter.add(new NhaHang("Bình Minh", R.drawable.binhminh, 10.92545187659691f,
        108.11409712647445f));
    nhaHangAdapter.add(new NhaHang("Ocean Dunes", R.drawable.oceandune, 10.927746894932161f,
        108.11889765197587f));
    nhaHangAdapter.add(new NhaHang("TTC", R.drawable.ttc, 10.925328982436442f,
        108.11470590527449f));
    nhaHangAdapter.add(new NhaHang("Vespa", R.drawable.vespa, 10.938425784370038f,
        108.11587414308765f));
}

private void addControls() {
    lvNhaHang=findViewById(R.id.lvNhaHang);
    nhaHangAdapter=new NhaHangAdapter(MainActivity.this, R.layout.item);
    lvNhaHang.setAdapter(nhaHangAdapter);
}
}

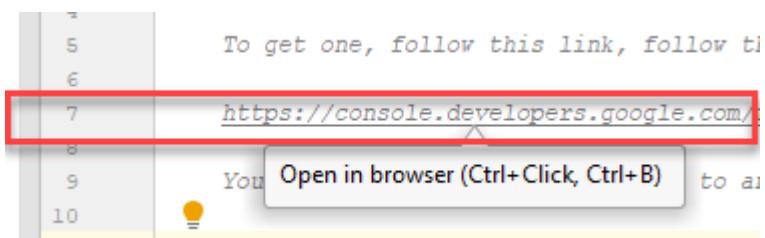
```

B7-Nhập chuột phải lên thư mục Java chọn New→Google→Google maps activity → để tên mặc định là MapsActivity, bấm finish.

Mở tiếp file google_maps_api.xml trong thư mục values lên.



Bấm vào đường link ở dòng số 7 trong file xml này để mở ra trang web tạo api key.



Trong trang web Google APIs, bạn tạo project mới hoặc chọn project có sẵn, sau đó bấm Continue

Google APIs Select a project ▾

Register your application for Maps SDK for Android in Google API Console

Google API Console allows you to manage your application and monitor API usage.

Select a project where your application will be registered
You can use one project to manage all of your applications, or you can create a different project for each application.

My Project 90189

Continue

Tiếp theo, bạn bấm vào Create API key để khởi tạo key

Google APIs Select a project ▾

The API is enabled

Maps SDK for Android has been enabled.

Next, you'll need to create an API key in order to call the API.

Create API key

Sau đó, bấm chuột vào biểu tượng copy để sao chép key.
API Keys

Name	Creation date	Restrictions	Key
API key 3	Dec 6, 2020	Android apps	AIzaSyDmEH...L6o88pehzI
API key 2	Nov 25, 2020	Android apps	AIzaSyBana...EXNHHjEeBM

B8-Quay lại file google_maps_api.xml bạn thay thế chữ YOUR_KEY_HERE bằng key đã copy ở bước 7.

Once you have your key (it starts with "AIza"), replace the "google_maps_key" string in this file.

```
--> <string name="google_maps_key" templateMergeStrategy="preserve" translatable="false">AIzaSyBanaMdVKcUqNatPc</string>
```

B9-Tại MainActivity, bổ sung code cho hàm addEvents(): khi bấm vào item sẽ tự động mở ra màn hình MapsActivity. Cụ thể như sau:

```

private void addEvents () {
    lvNhaHang.setOnItemClickListener(new
    AdapterView.OnItemClickListener() {
        @Override
        public void onItemClick(AdapterView<?> parent,
View view, int position, long id) {
            NhaHang
            nhaHang=nhaHangAdapter.getItem(position);
            Intent
                intent=new
                Intent(MainActivity.this,MapsActivity.class);
                intent.putExtra("NH",nhaHang);
                startActivity(intent);
        }
    });
}

```

B10-Trong MapsActivity, bổ sung trong hàm onMapReady: get vĩ độ và kinh độ từ lớp NhaHang.

```

public void onMapReady(GoogleMap googleMap) {
    mMap
        =
        googleMap;
    Intent
        nh=
        (NhaHang)
    intent.getSerializableExtra("NH");

    // Add a marker in Sydney and move the camera
    LatLng nhLoc = new LatLng(nh.getViDo(),
nh.getKinhDo());
    Marker
        marker=mMap.addMarker(new
    MarkerOptions().position(nhLoc).title(nh.getTen()));
    mMap.setInfoWindowAdapter(new
    MyCustomInfoAdapter(this,nh));

    mMap.moveCamera(CameraUpdateFactory.newLatLngZoom(nhLoc,1
6));
    marker.showInfoWindow();
}

```

B11-Trong MyCustomInfoAdapter (dùng thêm hình minh họa cho địa điểm trên bản đồ).



Bổ sung đoạn code sau:

```
package com.example.adapter;

import android.app.Activity;
import android.view.View;
import android.widget.ImageView;
import android.widget.TextView;

import com.example.lab14_baitapsotaynhahang.R;
import com.example.model.NhaHang;
import com.google.android.gms.maps.GoogleMap;
import com.google.android.gms.maps.model.Marker;

import org.w3c.dom.Text;

public class MyCustomInfoAdapter implements
GoogleMap.InfoWindowAdapter {

    Activity context;
    NhaHang nhaHang;
    public MyCustomInfoAdapter(Activity context, NhaHang nhaHang) {
        this.context=context;
        this.nhaHang=nhaHang;
    }
    public View getInfoWindow(Marker marker) {
        return null;
    }
}
```

```
@Override  
public View getInfoContents(Marker marker) {  
    View view=context.getLayoutInflater().inflate(R.layout.item_ma  
    ps,null);  
    ImageView imgHinh=view.findViewById(R.id.imgHinh);  
    TextView txtTen=view.findViewById(R.id.txtTen);  
    imgHinh.setImageResource(nhaHang.getHinh());  
    txtTen.setText(nhaHang.getTen());  
    return view;  
}  
}
```

B12-Chạy thử chương trình.

3. BÀI TẬP

Làm tiếp bài tập Phần mềm sô tay nhà hàng, bạn tìm hiểu và làm chức năng tìm đường đi từ địa điểm đang đứng tới nhà hàng.

LAB 14. SENSOR

1. MỤC TIÊU:

...

2. THỰC HIỆN:

2.1. Bài 1: ...

3. BÀI TẬP:

...