

Lập trình trên thiết bị di động

THIẾT KẾ GIAO DIỆN (PHẦN 1)

GV: Nguyễn Huy Cường

Email: nh.cuong@hutech.edu.vn

Nội dung

1. Thiết kế giao diện người dùng

2. View

 Widgets (SeekBar, WebView, SearchView...)

3. ViewGroup

 ConstraintLayout, RelativeLayout, LinearLayout

 FrameLayout, ScrollView...

 ListView, RecyclerView

Thiết kế giao diện người dùng

- Tìm hiểu nhu cầu của khách hàng về giao diện
- Càng đơn giản càng tốt
- Các đơn vị đo: **px**, **dp**, **dip**, **sp** và **dpi**

dpi: (số điểm ảnh **px** / inch)

-> ldpi (120), mdpi (160), hdpi (240), xhdpi (320), xxhdpi(480), xxxhdpi (640)

$$\text{px} = \text{dp} * (\text{dpi} / 160)$$

$$\text{dp} = 1/160 = 0.00625 \text{ inch}$$

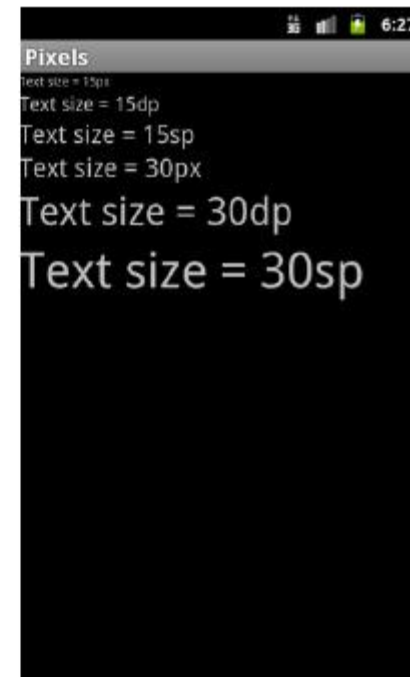
dp ~ sp (sử dụng cho kích thước văn bản – có thể resize)

- Các cách thiết kế giao diện:

☐ Palette

☐ XML

☐ Java



Thành phần giao diện trong Android

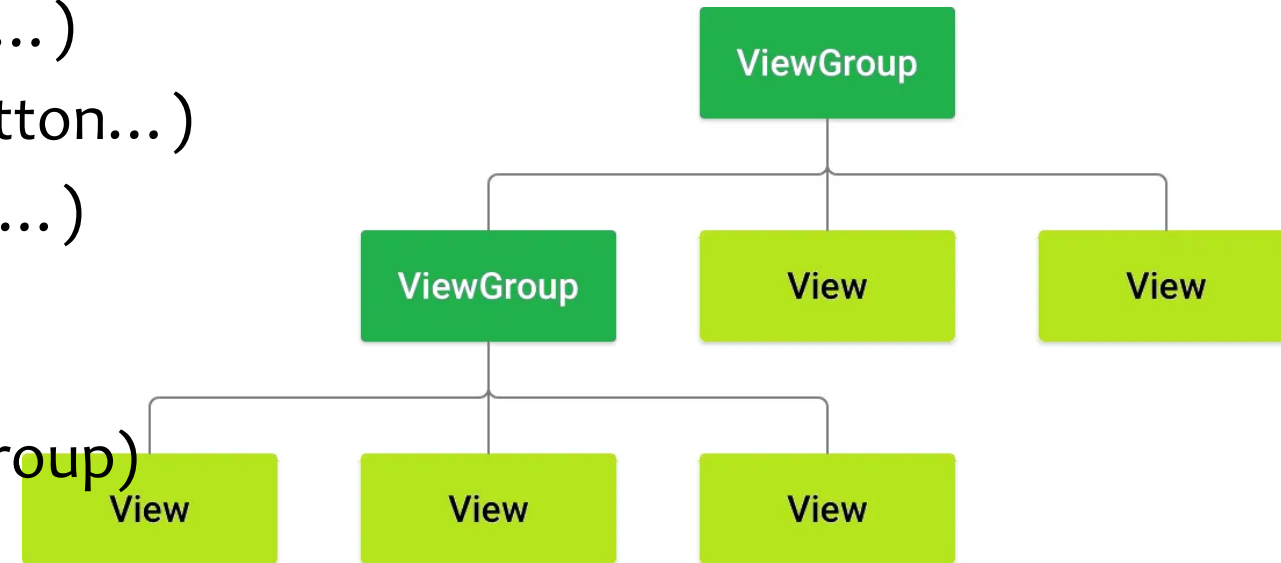
❖ Cấu trúc giao diện theo dạng cây phân cấp: View và ViewGroup

- View: Là lá của cây phân cấp, thành phần cơ bản của giao diện người dùng như:

- Text (TextView, EditText, Phone, Email...)
- Buttons (Button, CheckBox, ToggleButton...)
- Widgets (WebView, SeekBar, RatingBar...)

- ViewGroup: chứa các Views (hoặc View Group)

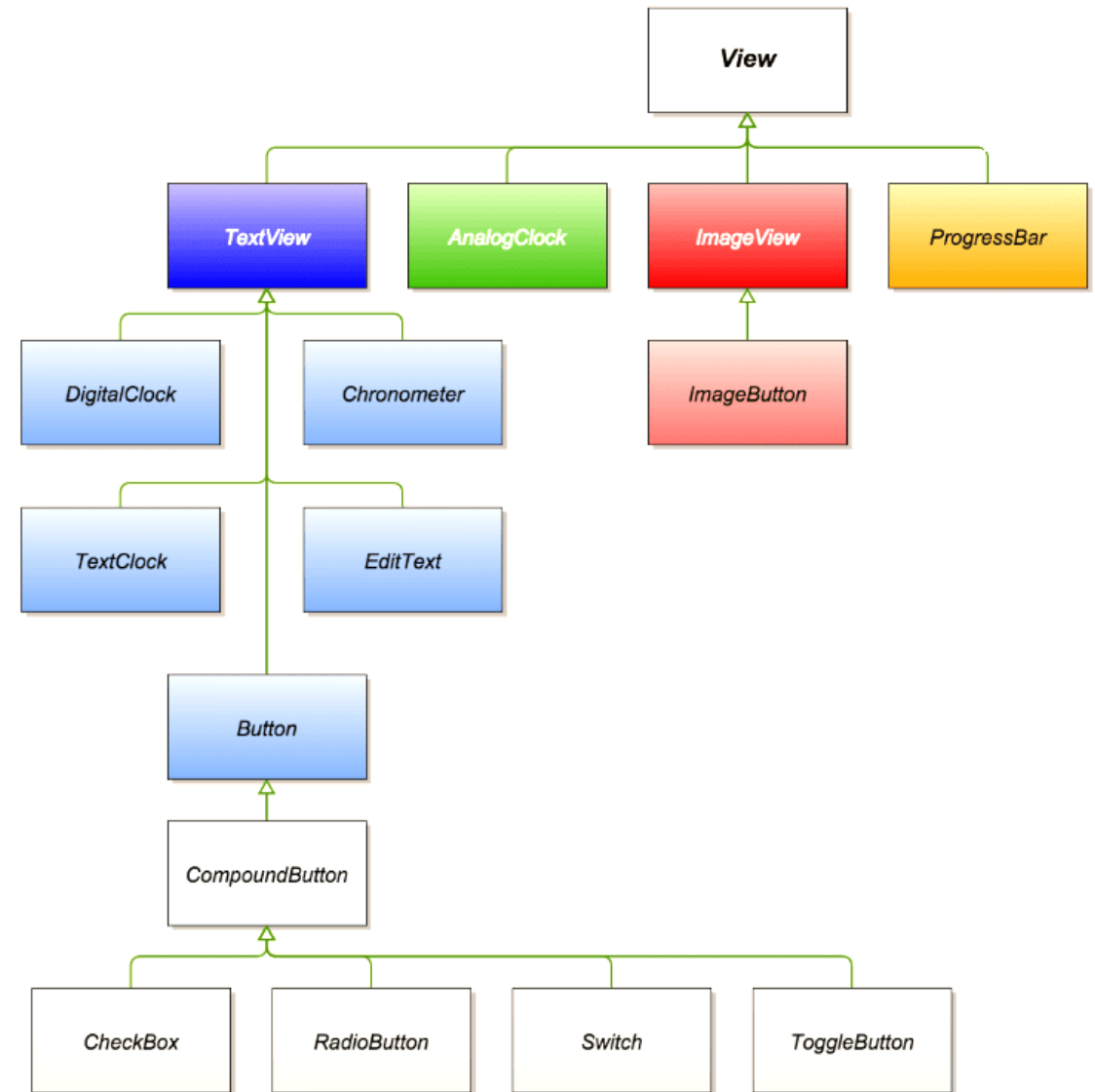
- Layouts (ConstraintLayout, LinearLayout, RelativeLayout, FrameLayout, TableLayout..)
- GridLayout, GridView, ScrollView...
- ListView, RecyclerView...
- ...



Thành phần giao diện

View

- View biểu diễn một hình chữ nhật, trong đó nó hiện thị thông tin nào đó cho người dùng, và người dùng có thể tương tác với View



View

- Các thuộc tính:

- ☐ **id**: Xác định id cho View ở giao diện XML, và sử dụng findViewById để ánh xạ
`TextView txtMSSV = findViewById(R.id.txtMSSV);`

- ☐ **margin**: Khoảng cách từ cạnh view tới các view khác

- ☐ **padding**: Khoảng cách từ cạnh view tới nội dung

- ☐ **layout_height**

- ☐ **layout_width**

có các giá trị

- **match_parent**: bằng của phần tử cha chứa nó.

- **wrap_content**: phụ thuộc vào content của nó.

- “*giá trị xác định*”: xác định theo 1 đơn vị (dp, px, in, mm, sp...)

- ☐ **background**: Xác định màu nền của view

- ☐ **gravity**

- ☐ **hint**: Hiển thị text khi empty

- ☐ **Visibility**:

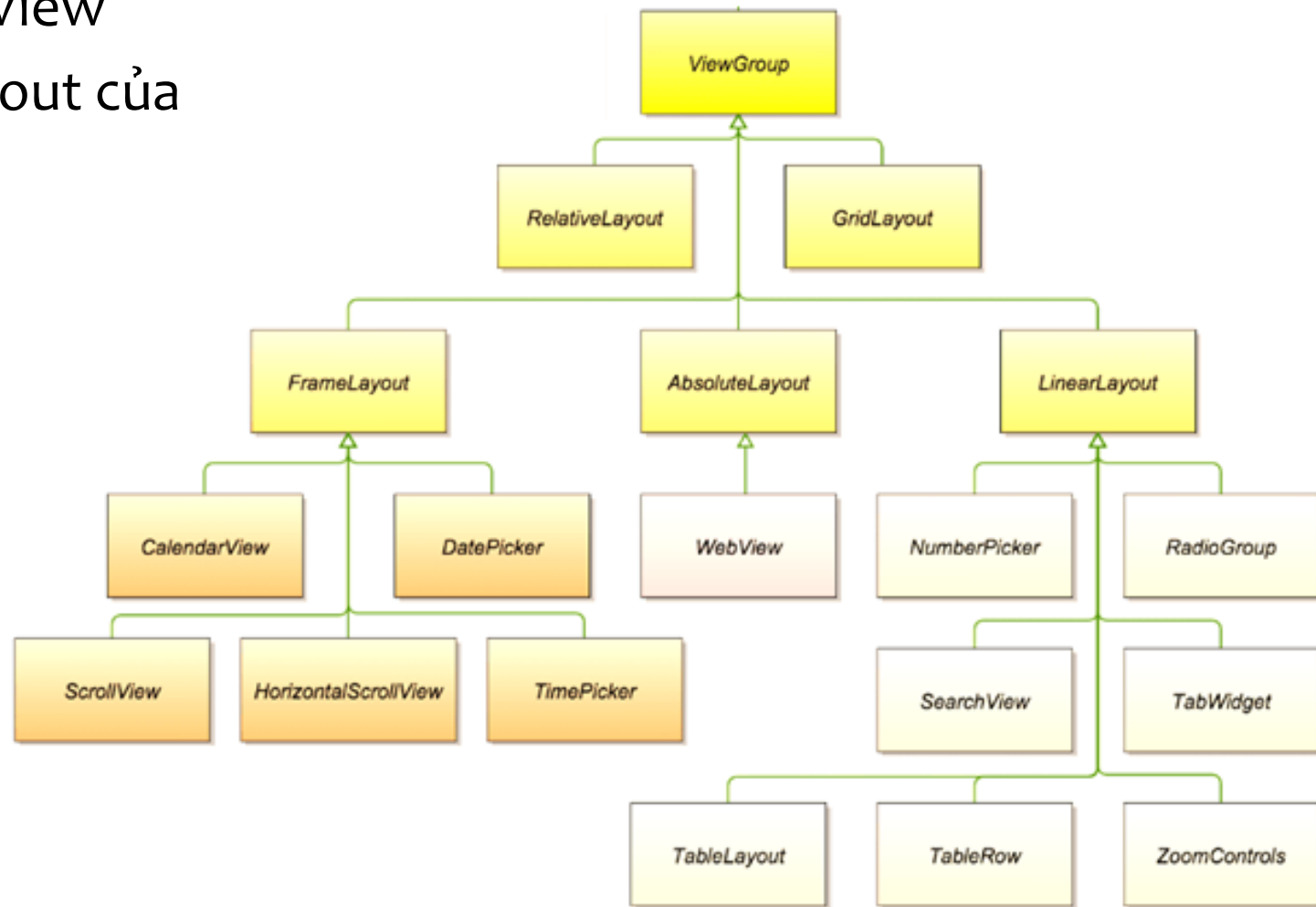
- ☐ **textColor**

- ☐ **textSize**

Thành phần giao diện

ViewGroup

- ViewGroup: chứa các Views hoặc View Group để tổ chức và kiểm soát layout của một màn hình



VIEW

 Text

 ImageView

 Button

 Spinner

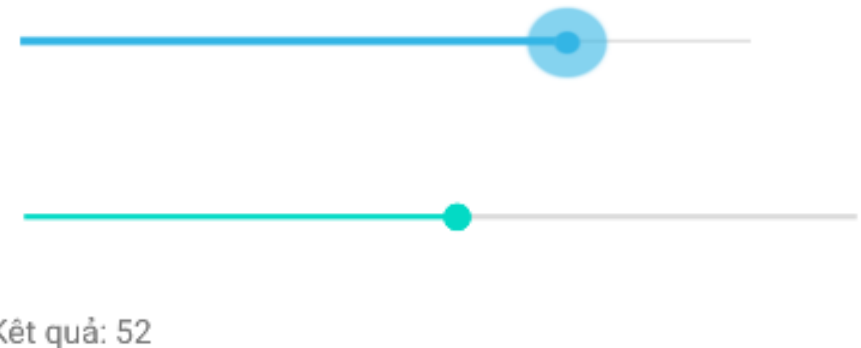
 Widgets (SeekBar, WebView, SearchView...)

SeekBar

- SeekBar mở rộng từ **ProgressBar** có thêm một cái cần gạt. Người dùng có thể chạm vào cần gạt và kéo sang trái hoặc phải để thiết lập giá trị của tiến trình (progress) hoặc sử dụng các phím mũi tên để di chuyển cần gạt.
- Sử dụng trong XML:

```
<SeekBar  
    android:id="@+id/seekBar"  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content" />
```

- Phương thức lấy giá trị: seekBar.**getProgress()**
- Xử lý sự kiện trên SeekBar:
seekBar.**setOnSeekBarChangeListener (...)**
 - ❑ **onProgressChanged**
 - ❑ **onStopTrackingTouch**



WebView

- WebView là một dạng xem hiển thị các trang web bên trong ứng dụng
- Sử dụng trong XML:

<WebView

android:id="@+id/webView"

android:layout_width="match_parent"

android:layout_height="575dp"

/>

- Xin phép truy cập internet cho ứng dụng ở **manifest.xml**:

<uses-permission **android:name="android.permission.INTERNET"** />

- Một số phương thức:

WebView browser = findViewById(R.id.**webView**);

browser.loadUrl("https://www.hutech.edu.vn");

☐ **canGoBack()**

☐ **canGoForward()**

☐ **getUrl()**



SearchView

- **SearchView** hỗ trợ cho người dùng những gợi ý liên quan khi nhập vào **EditText**. Những gợi ý đó sẽ được hiển thị trong một danh sách thả xuống từ đó người dùng có thể chọn một item để thay thế cho nội dung của mình vừa nhập vào
- Các phương thường sử dụng
 - ❑ **setQueryTextListener(OnQueryTextListenerlistener):** Interface này dùng để lắng nghe sự kiện QueryTextChange Ví dụ sau thiết lập sự kiện setOnQueryTextListener cho **SearchView**.



VIEWGROUP

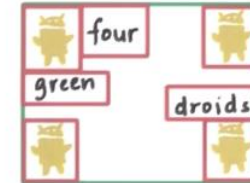
FrameLayout



LinearLayout








RelativeLayout



GridLayout



-  1- ConstraintLayout
-  2 -LinearLayout
-  3- RelativeLayout
-  4- FrameLayout
-  5- ScrollView

-  ListView
-  RecyclerView

1- ConstraintLayout

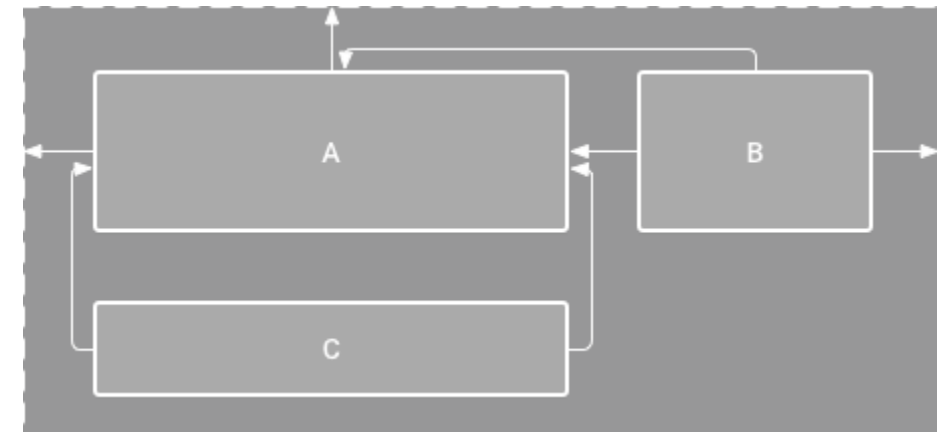
- Khai báo trong Android

implementation 'androidx.constraintlayout:constraintlayout:2.1.4'

- Mỗi view trong ConstraintLayout để định vị được chính xác cần tối thiểu 2 ràng buộc, một theo phương ngang (X) và một theo phương đứng (Y). các View không có ràng buộc sẽ định vị ở góc trái - trên (tọa độ 0,0).

- Một số thuộc tính ràng buộc

<code>layout_constraintLeft_toLeftOf</code>	Ràng buộc cạnh trái của phần tử tới phần tử chỉ ra trong giá trị (gán ID)
<code>layout_constraintLeft_toRightOf</code>	Bên trái với bên phải của phần tử chỉ ra
<code>layout_constraintRight_toLeftOf</code>	Bên phải với bên trái
<code>layout_constraintRight_toRightOf</code>	Phải với phải
<code>layout_constraintTop_toTopOf</code>	Cạnh trên với cạnh trên
<code>layout_constraintTop_toBottomOf</code>	Cạnh trên nối với cạnh dưới
<code>layout_constraintBottom_toTopOf</code>	Dưới với trên
<code>layout_constraintBottom_toBottomOf</code>	Dưới với dưới
<code>layout_constraintBaseline_toBaselineOf</code>	Trùng Baseline
<code>layout_constraintStart_toEndOf</code>	Bắt đầu - Kết thúc
<code>layout_constraintStart_toStartOf</code>	Bắt đầu - Bắt đầu
<code>layout_constraintEnd_toStartOf</code>	Cuối với bắt đầu
<code>layout_constraintEnd_toEndOf</code>	Cuối với cuối



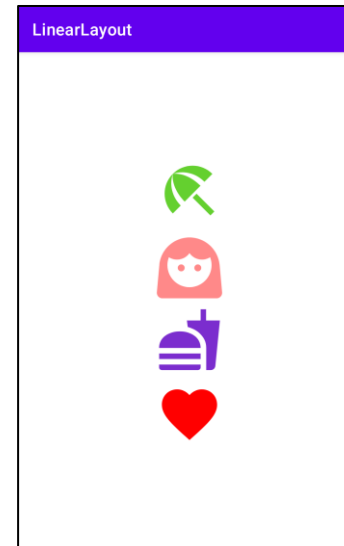
2- LinearLayout

- Sắp xếp các **View** con một chiều (ngang hoặc dọc) theo giá trị của thuộc tính **android:orientation**
- Khai báo trong Android

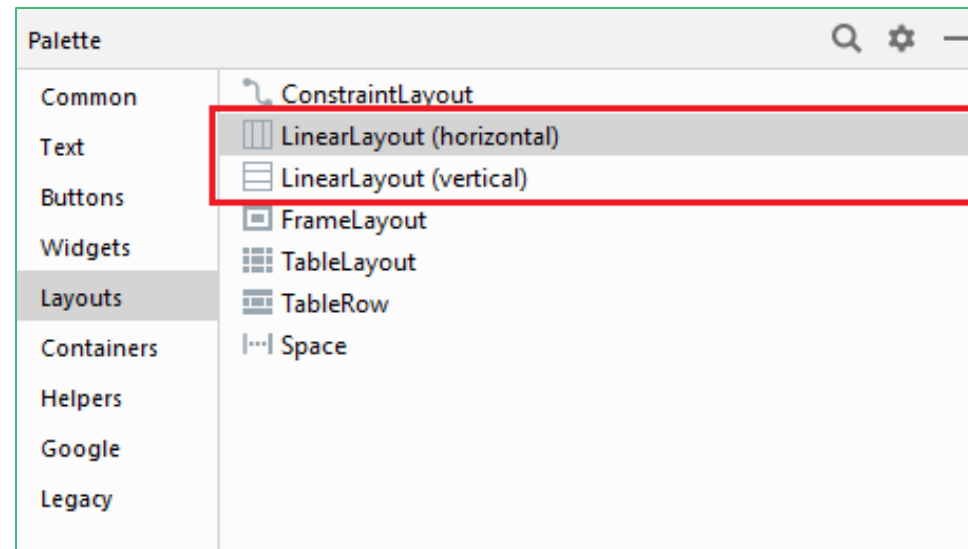
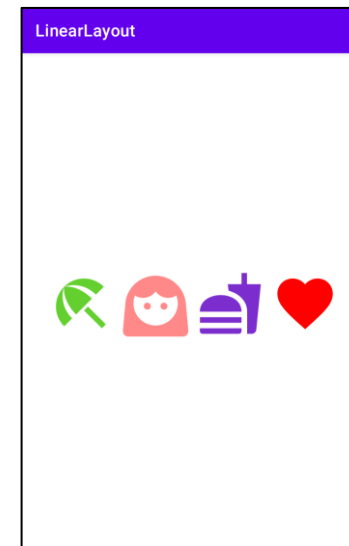
```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="horizontal">
    <!--View child-->
</LinearLayout>
```

- Qui tắc: Các view con không nằm đè lên nhau
 - Lồng nhau ≥ 5 làm chậm việc tải giao diện

Horizontal



Vertical

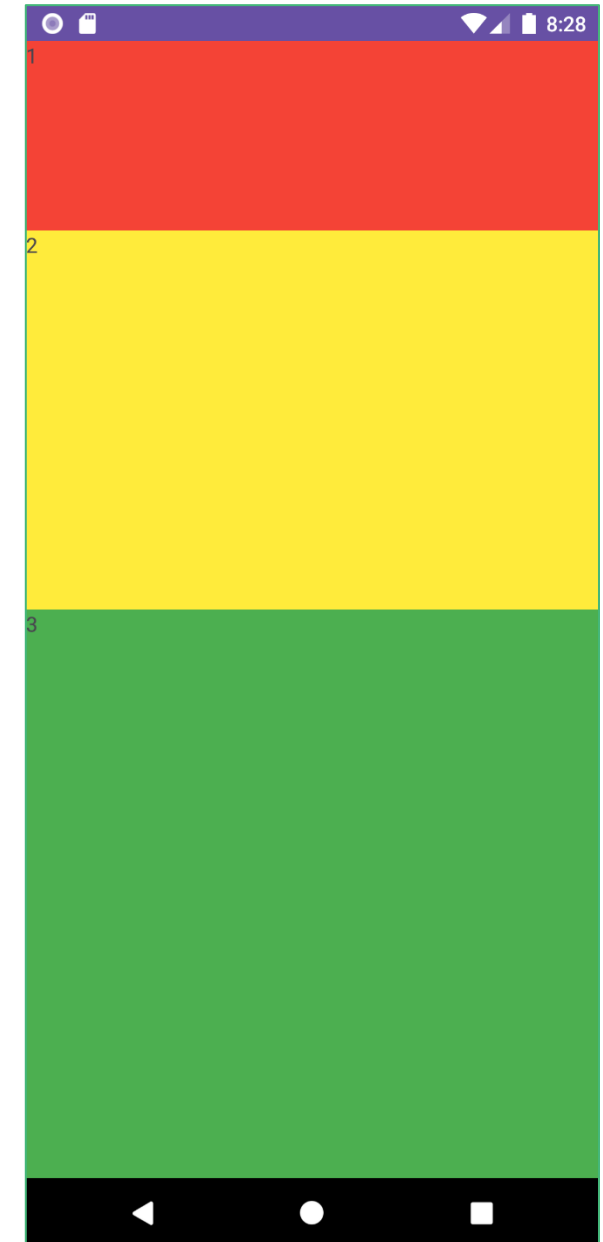


LinearLayout

Thuộc tính layoutweight

❏ **layout weight**: Trọng số view con chiếm.

```
<androidx.appcompat.widget.LinearLayoutCompat
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"
android:orientation="vertical"
tools:context=".MainActivity">
<TextView
    android:id="@+id/textView"
    android:layout_width="match_parent"
    android:layout_height="0dp"
    android:layout_weight="1"
    android:background="#F44336"
    android:text="1" />
<TextView
    android:id="@+id/textView2"
    android:layout_width="match_parent"
    android:layout_height="0dp"
    android:layout_weight="2"
    android:background="#FFEB3B"
    android:text="2" />
<TextView
    android:id="@+id/textView3"
    android:layout_width="match_parent"
    android:layout_height="0dp"
    android:layout_weight="3"
    android:background="#4CAF50"
    android:text="3" />
</androidx.appcompat.widget.LinearLayoutCompat>
```



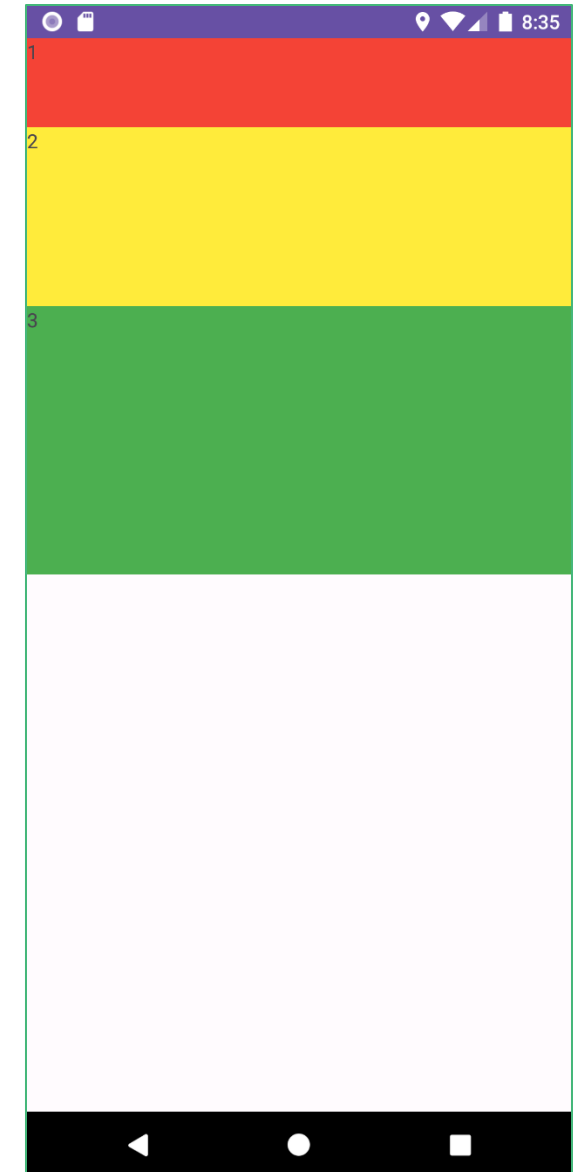
LinearLayout

Thuộc tính weightsum

□ weightSum:

Tổng trọng
số toàn bộ
LinearLayout

```
<androidx.appcompat.widget.LinearLayoutCompat
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"
android:orientation="vertical"
android:weightSum="12"
tools:context=".MainActivity">
  <TextView
    android:id="@+id/textView"
    android:layout_width="match_parent"
    android:layout_height="0dp"
    android:layout_weight="1"
    android:background="#F44336"
    android:text="1" />
  <TextView
    android:id="@+id/textView2"
    android:layout_width="match_parent"
    android:layout_height="0dp"
    android:layout_weight="2"
    android:background="#FFEB3B"
    android:text="2" />
  <TextView
    android:id="@+id/textView3"
    android:layout_width="match_parent"
    android:layout_height="0dp"
    android:layout_weight="3"
    android:background="#4CAF50"
    android:text="3" />
</androidx.appcompat.widget.LinearLayoutCompat>
```

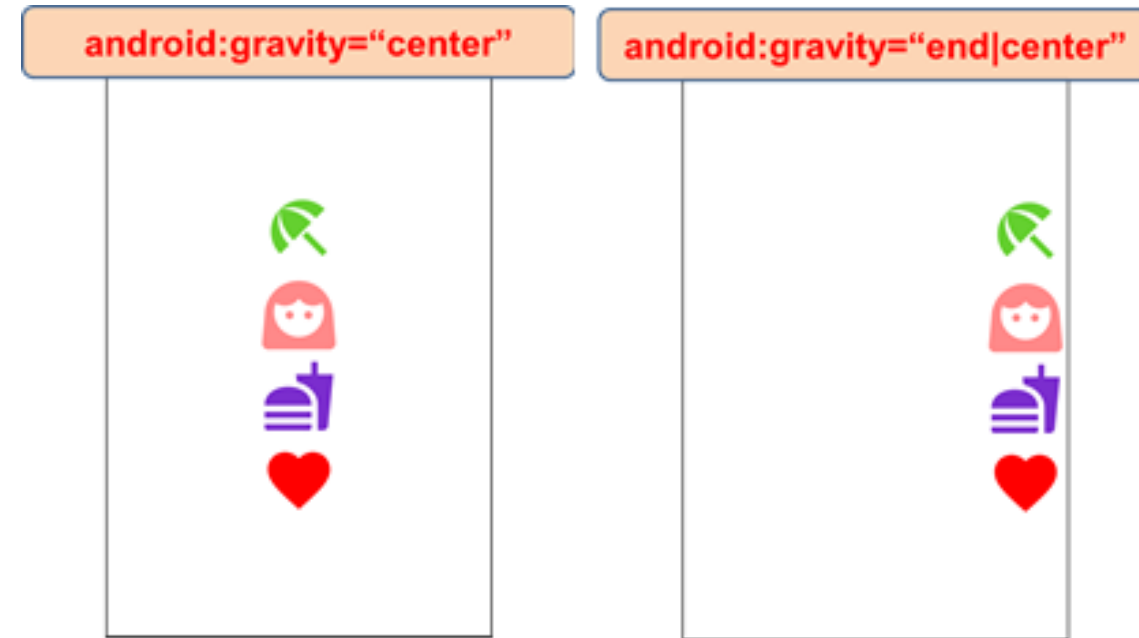


LinearLayout

Một số thuộc tính - gravity

- ❑ **gravity**: dùng để căn chỉnh các View nằm ở vị trí nào trong LinearLayout, nó nhận các giá trị bên dưới (có thể tổ hợp lại với ký hiệu |)

Giá trị	Ý nghĩa
center	Căn ở giữa
top	Căn ở phần trên
bottom	Căn ở phần dưới
center_horizontal	Căn ở giữa theo chiều ngang
center_vertical	Căn ở giữa theo chiều đứng
start (left)	Căn theo cạnh trái
end (right)	Căn theo cạnh phải

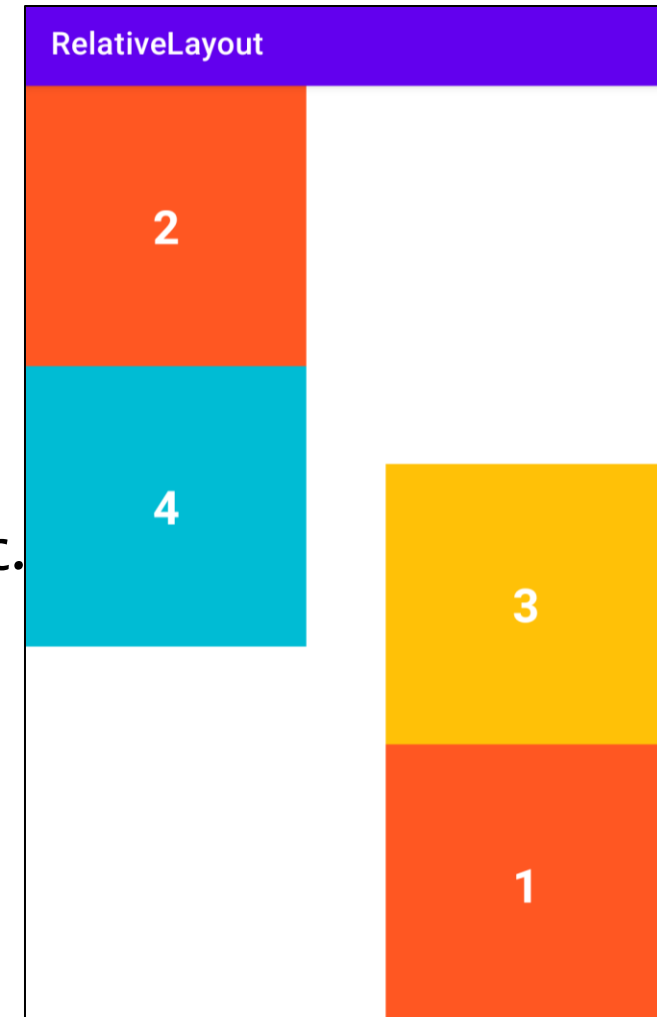


3- RelativeLayout

- Sắp xếp các **View** theo vị trí tương đối giữa các view khác trên giao diện (kể cả view chứa nó)
- Khai báo trong Android

```
<RelativeLayout  
    android:layout_width="match_parent"  
    android:layout_height="match_parent">  
    <!--View child-->  
  
</RelativeLayout>
```

- Quy tắc: Các view con nằm đè lên nhau, view sau đè view trước. Thường nó dựa vào **Id** của các widget khác để sắp xếp theo vị trí tương đối



Định vị từ view cha

❖ Định vị View con từ View cha

- ☐ layout_alignParentBottom
- ☐ layout_alignParentStart
- ☐ layout_alignParentEnd
- ☐ layout_alignParentTop
- ☐ layout_centerInParent
- ☐ layout_centerHorizontal
- ☐ layout_centerVertical

```
<RelativeLayout
```

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

```
    <View
```

```
        android:id="@+id/view1"
```

```
        android:layout_width="200dp"
```

```
        android:layout_height="50dp"
```

```
        android:layout_alignParentEnd="true"
```

```
        android:layout_centerInParent="true"
```

```
        android:background="#e8d33636" />
```

```
</RelativeLayout>
```

RelativeLayout



Định vị từ các view con

❖ Định vị từ liên hệ các view con tương ứng

- ☐ layout_above
- ☐ layout_below
- ☐ layout_toLeftOf
- ☐ layout_toRightOf
- ☐ layout_alignBottom
- ☐ layout_alignStart
- ☐ layout_alignEnd
- ☐ layout_alignTop

<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">
```

<TextView

```
android:id="@+id/txtText1"
android:layout_width="200dp"
android:layout_height="60dp"
android:background="#E91E63"
android:text="1" />
```

<TextView

```
android:id="@+id/txtText2"
android:layout_width="100dp"
android:layout_height="60dp"
android:layout_below="@id/txtText1"
android:layout_toEndOf="@id/txtText1"
android:background="#8BC34A"
android:text="2" />
```

</RelativeLayout>



RelativeLayout

Định vị layout – kết hợp

```

<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/btnButton1"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="40dp"
        android:text="Button 1"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />
    <Button
        android:id="@+id/btnButton2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@id/btnButton1"
        android:layout_alignParentLeft="true"
        android:layout_marginTop="40dp"
        android:text="Button 2" />
    <Button
        android:id="@+id/btnButton3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@id/btnButton1"
        android:layout_toRightOf="@id/btnButton2"
        android:layout_marginTop="40dp"
        android:layout_alignParentRight="true"
        android:layout_marginLeft="20dp"
        android:text="Button 3"
    />
</RelativeLayout>

```



4-FrameLayout

- FrameLayout được thiết kế để chứa các view con. Sắp xếp các **View** theo vị trí tương đối giữa các view khác trên giao diện (kể cả control chứa nó)
- Khai báo trong Android

```
<FrameLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <!-- View Child -->

</FrameLayout>
```

- Quy tắc: khi gắn các view lên layout này thì nó sẽ luôn giữ các view này ở phía góc trái màn hình và không cho thay đổi vị trí của chúng. Các view con nằm đè lên nhau, view sau đè view trước.
- Một số thuộc tính định vị
 - ❑ **android:layout_gravity**: vùng trọng tâm mặc định **left|top**

FrameLayout

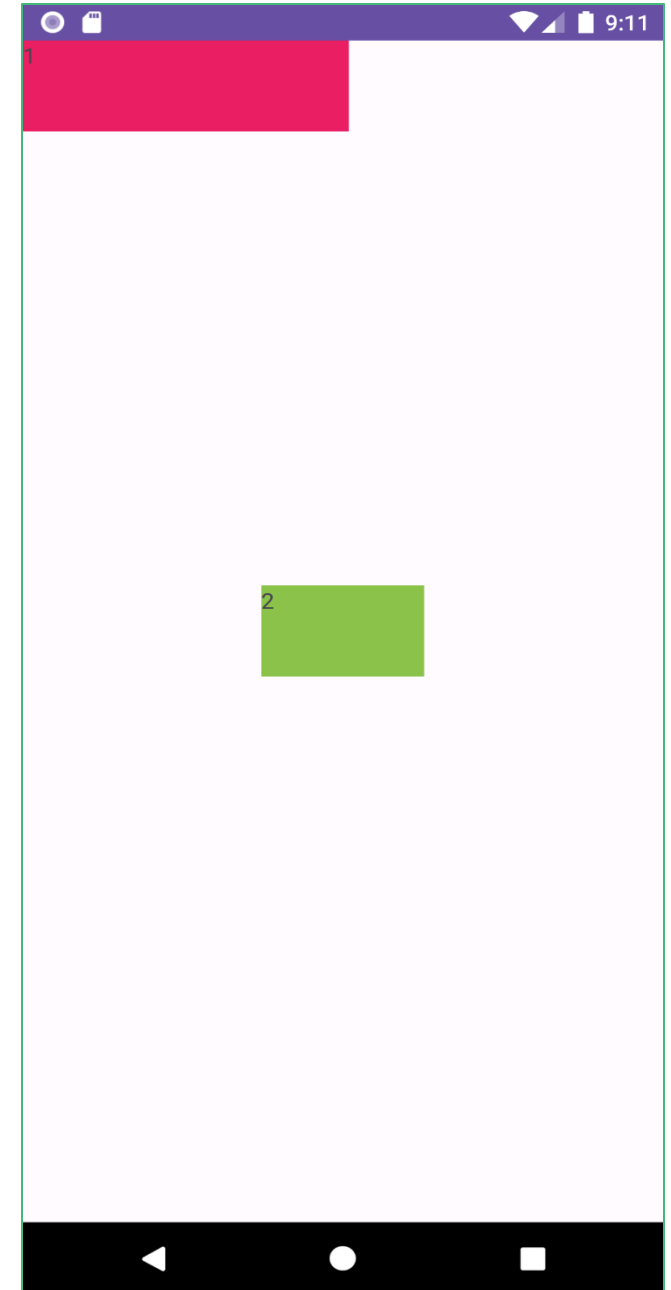
Demo-1

```
<FrameLayout 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/txtText1"
    android:layout_width="200dp"
    android:layout_height="60dp"
    android:background="#E91E63"
    android:text="1" />
```

```
<TextView
    android:id="@+id/txtText2"
    android:layout_width="100dp"
    android:layout_height="60dp"
    android:layout_gravity="center"
    android:background="#8BC34A"
    android:text="2" />
```

```
</FrameLayout>
```



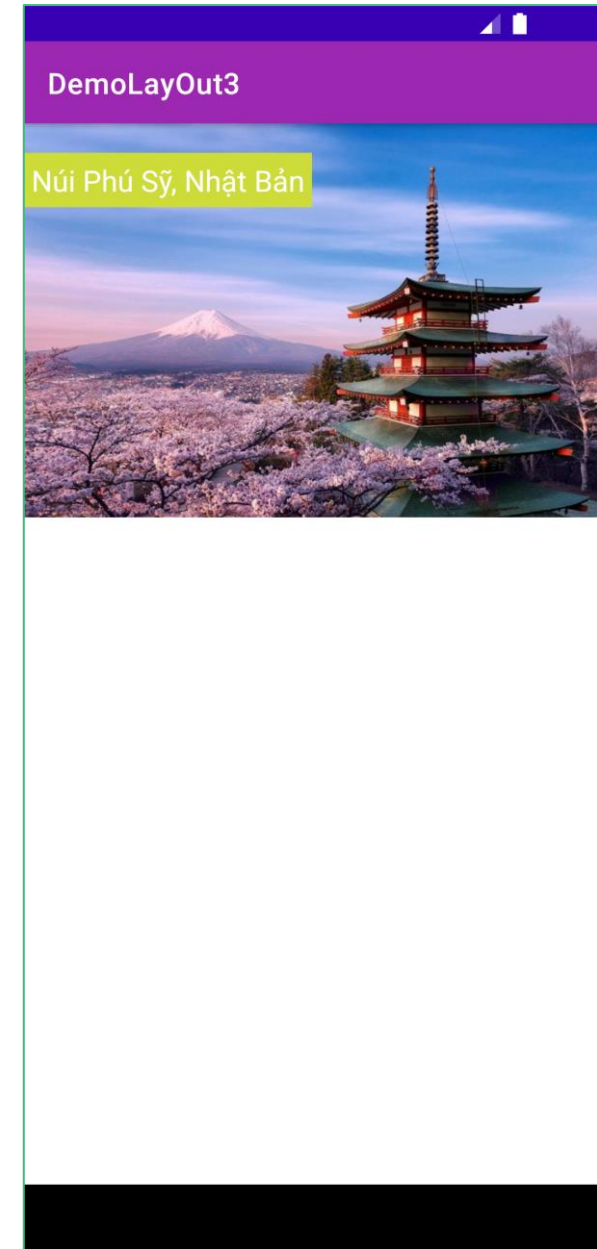
FrameLayout

Demo-2

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

    <ImageView
        android:id="@+id/imageView"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:scaleType="fitStart"
        android:src="@drawable/img_nhatban" />

    <TextView
        android:id="@+id/txtvw1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="20dp"
        android:background="#CDDC39"
        android:padding="5dp"
        android:text="Núi Phú Sĩ, Nhật Bản"
        android:textColor="#FFFFFF"
        android:textSize="20sp" />
</FrameLayout>
```



5- ScrollView, HorizontalScrollView

- **ScrollView** là một loại **FrameLayout** cho phép cuộn qua danh sách các khung nhìn chiếm nhiều không gian
- Là 1 viewGroup giúp cuộn khi nội dung theo chiều **dọc** (**ScrollView**) hoặc **ngang** (**HorizontalScrollView**)

```
<ScrollView
    android:layout_width="match_parent"
    android:layout_height="match_parent">

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

        <!-- things to scroll -->

    </LinearLayout>
</ScrollView>
```

ScrollView

Demo

<ScrollView

```
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"
android:padding="15dp"
tools:context=".MainActivity">
```

<LinearLayout

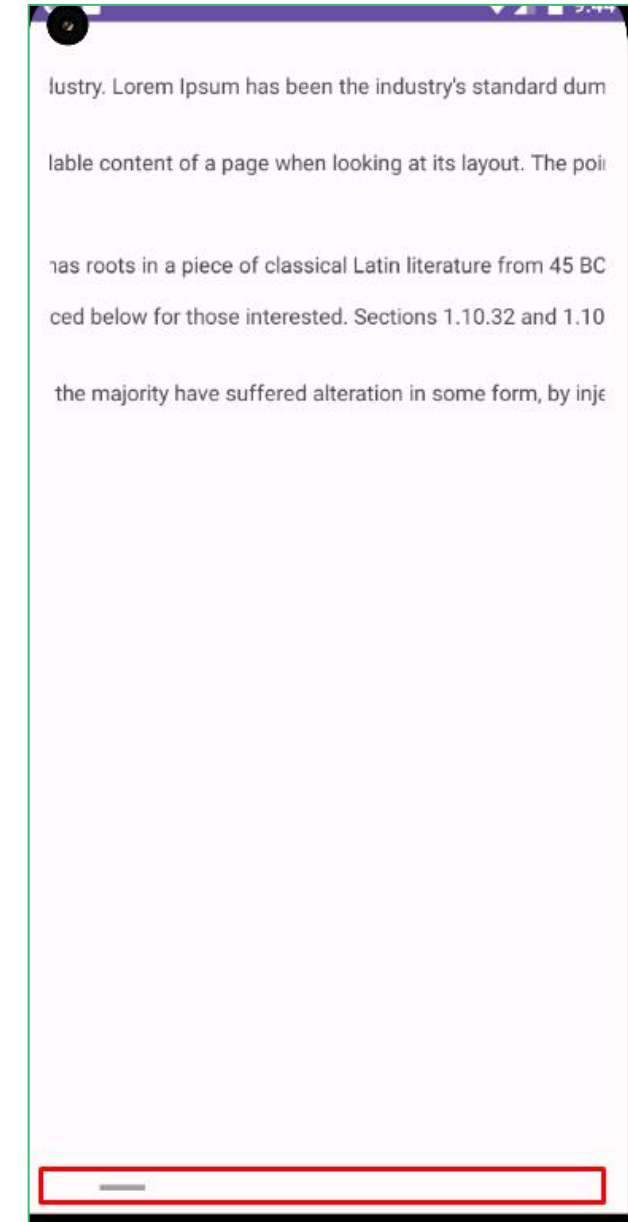
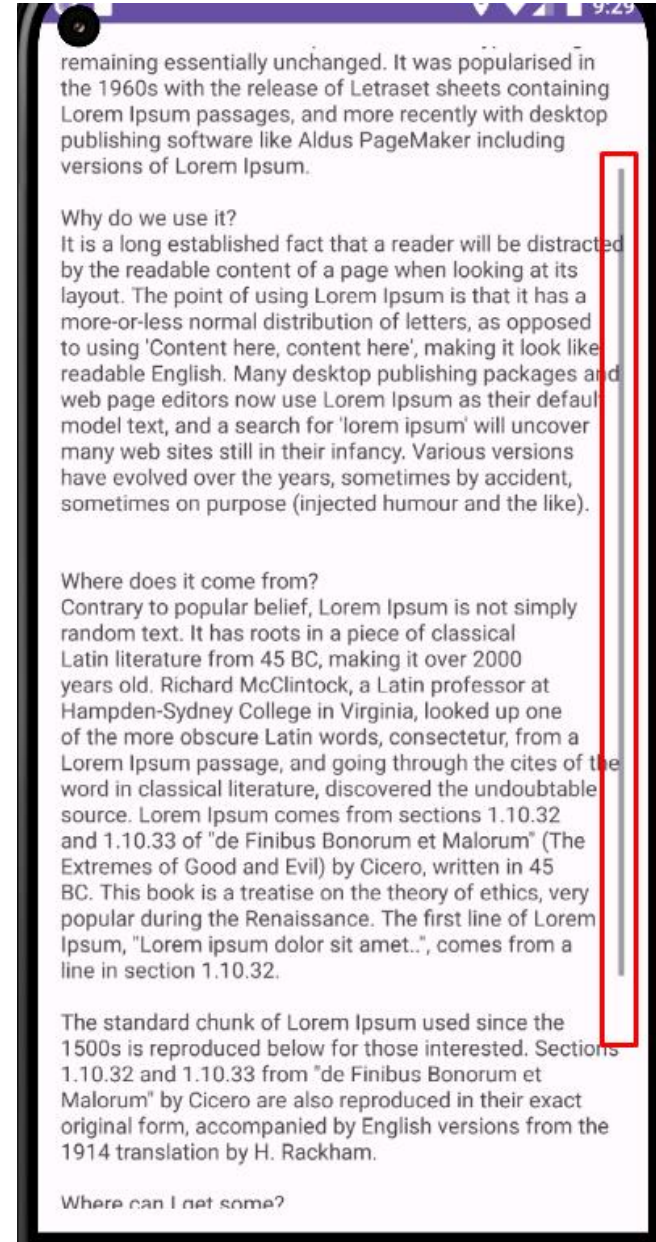
```
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:orientation="vertical" >
```

<TextView

```
android:layout_width="wrap_content"
android:layout_height="match_parent"
android:text="@string/KEY_TEXT" />
```

</LinearLayout>

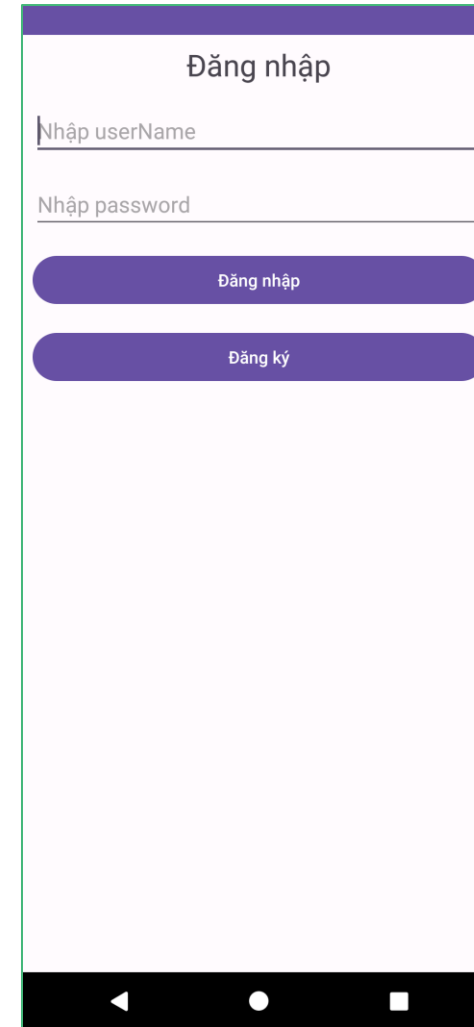
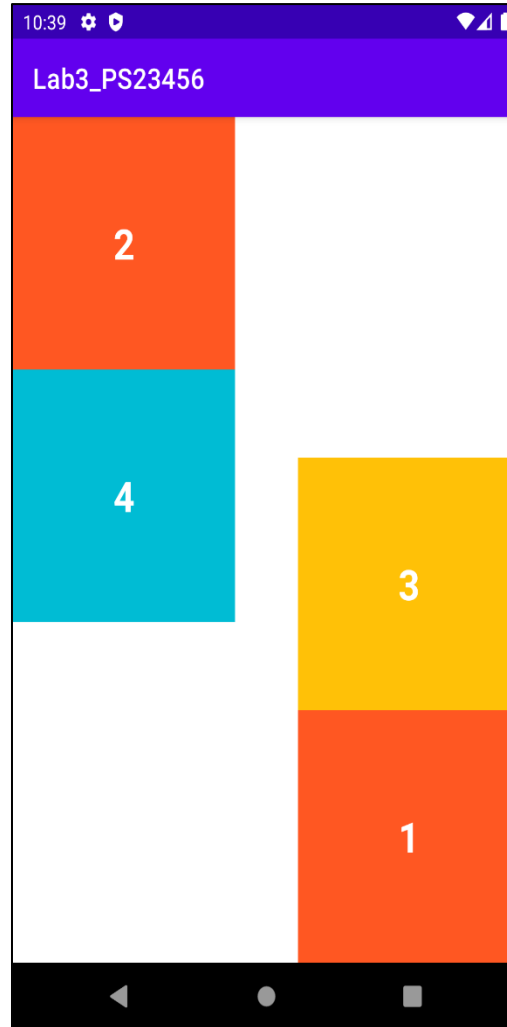
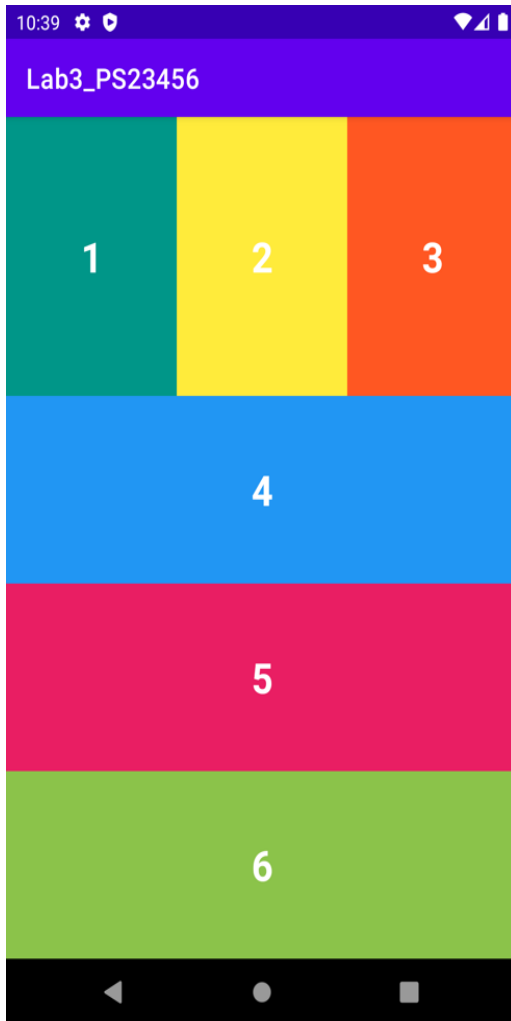
</ScrollView>



HorizontalScrollView



Thiết kế 1 số giao diện như sau: (không sử dụng ConstraintLayout, 3 activity)



ListView

- Hiển thị danh sách các item có thể scroll được từ **Adapter**
 - ❑ Adapter chịu trách nhiệm chuyển đổi từng data entry vào view, sau đó có thể được thêm vào AdapterView (ListView/ RecyclerView) để hiển thị
- Khai báo trong Android

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

- Quy tắc:
 - Được tạo từ ListItem
 - Mỗi dòng mặc định là TextView
 - Thông thường được load dữ liệu động



Listview cơ bản với ArrayAdapter

```
public class MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.demo_listview);

        List<String> lstMonHoc = new ArrayList<>();
        lstMonHoc.add("Kiểm thử phần mềm");
        lstMonHoc.add("Lập trình di động");
        lstMonHoc.add("Lập trình ứng dụng Java");
        lstMonHoc.add("Công cụ và môi trường phát triển phần mềm");
        ListView listView = findViewById(R.id.listView);

        ArrayAdapter arrayAdapter = new ArrayAdapter(getApplicationContext(), android.R.layout.simple_list_item_1, lstMonHoc);
        listView.setAdapter(arrayAdapter);
        listView.setOnItemClickListener(new AdapterView.OnItemClickListener() {
            @Override
            public void onItemClick(AdapterView<?> adapterView, View view, int i, long l) {
                if(i==0){
                    //clicked Kiểm thử phần mềm
                    Toast.makeText(context: MainActivity.this, text: "Đã Chọn Kiểm thử phần mềm!", Toast.LENGTH_SHORT).show();

                }else if(i==1){
                    //clicked Lập trình di động
                    Toast.makeText(context: MainActivity.this, text: "Đã Chọn Lập trình di động!", Toast.LENGTH_SHORT).show();
                }else{
                    /// ...
                }
            }
        });
    }
}
```

Demo ListView

Kiểm thử phần mềm

Lập trình di động

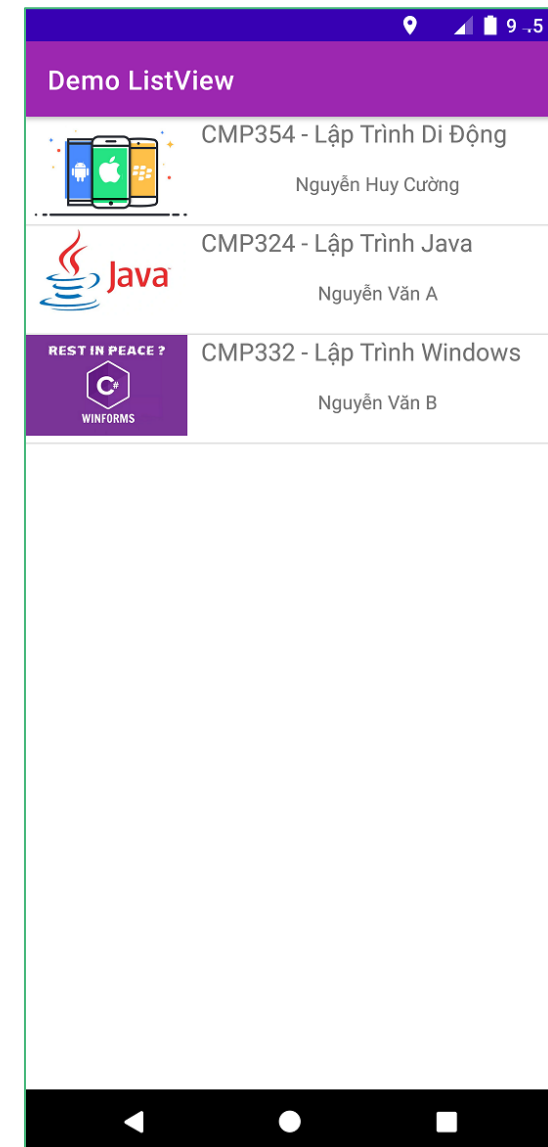
Lập trình ứng dụng Java

Công cụ và môi trường phát triển phần mềm



Listview với BaseAdapter

0. Chuẩn bị hình ảnh tương ứng copy vào **mipmap / drawable**
1. Thêm lớp **MonHoc** (TenHinh, MaHP, TenHP, TenGV)
2. Thêm **ListView** vào Activity layout
3. Tạo giao diện **row_item** cho listView
4. Thêm lớp **MonHocAdapter** *extends* từ **BaseAdapter**
 - Tạo lớp **ViewHolder** tương ứng với **row_item**
 - Implements các hàm: `getcount()`, `getitem()`, `getitemid()`, `getView()`
 - Có các thuộc tính: `Context`, `LayoutInflater`, `List<MonHoc>`
5. Sử dụng ở **MainActivity**



Hướng dẫn: Listview với BaseAdapter

- o. Chuẩn bị hình ảnh tương ứng copy vào **mipmap** hoặc **drawable**
1. Thêm lớp **MonHoc** (TenHinh, MaHP, TenHP, TenGV)
2. Thêm **ListView** vào Activity layout
3. Tạo giao diện **row_item** cho listView

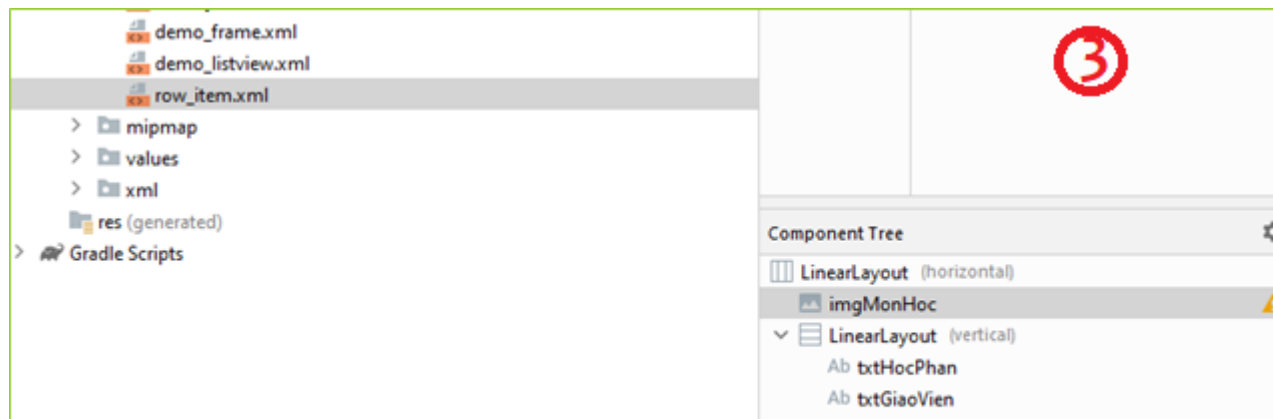
```
public class MonHoc{
    public String TenHinh;
    public String MaHP;
    public String TenHP;
    public String TenGV;
    public MonHoc(String th, String maHP, String tenHP, String tenGV) {
        TenHinh = th;
        MaHP = maHP;
        TenHP = tenHP;
        TenGV = tenGV;
    }
}
```

①

```
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <ListView
        android:id="@+id/listView"
        android:layout_width="match_parent"
        android:layout_height="match_parent" >
    </ListView>
</androidx.constraintlayout.widget.ConstraintLayout>
```

②



③

Hướng dẫn: Listview với BaseAdapter

4. Thêm MonHocAdapter extends từ BaseAdapter

MonHocAdapter có ViewHolder

- getCount()
- getItem()
- getItemId()
- getView()

5. Sử dụng

```
List<MonHoc> lstMonHoc= new ArrayList<>();
lstMonHoc.add(new MonHoc( th: "didong", maHP: "CMP354", tenHP: "Lập Trình Di Động", tenGV: "Nguyễn Huy Cường"));
lstMonHoc.add(new MonHoc( th: "java", maHP: "CMP324", tenHP: "Lập Trình Java", tenGV: "Nguyễn Văn A"));
lstMonHoc.add(new MonHoc( th: "window", maHP: "CMP332", tenHP: "Lập Trình Windows", tenGV: "Nguyễn Văn B"));

MonHocAdapter adapter= new MonHocAdapter(lstMonHoc);
lstView.setAdapter(adapter);
lstView.setOnItemClickListener(new AdapterView.OnItemClickListener() {
    @Override
    public void onItemClick(AdapterView<?> parent, View view, int position, long id) {

        MonHoc dataModel= lstMonHoc.get(position);
        Snackbar.make(view, String.format("%s - %s \n %s", dataModel.MaHP , dataModel.TenHP, dataModel.TenGV ), Snackbar.LENGTH_LONG)
            .setAction( text: "No action", listener: null).show();
    }
});
```

```
public class MonHocAdapter extends BaseAdapter {
    private List<MonHoc> listData;
    public MonHocAdapter( List<MonHoc> listMonHoc) {
        this.listData = listMonHoc;
    }

    @Override
    public int getCount() { return listData.size(); }
    @Override
    public Object getItem(int position) { return listData.get(position); }
    @Override
    public long getItemId(int position) { return position; }
    @Override
    public View getView(int position, View convertView, ViewGroup parent) {
        ViewHolder holder;
        if (convertView == null) {
            convertView = LayoutInflater.from(parent.getContext()).inflate(R.layout.row_item, root: null);
            holder = new ViewHolder();
            holder.imgView = (ImageView) convertView.findViewById(R.id.imgMonHoc);
            holder.txtMonHoc = (TextView) convertView.findViewById(R.id.txtHocPhan);
            holder.txtTenGV = (TextView) convertView.findViewById(R.id.txtGiaoVien);
            convertView.setTag(holder);
        } else {
            holder = (ViewHolder) convertView.getTag();
        }

        MonHoc temp = this.listData.get(position);
        holder.txtMonHoc.setText(temp.MaHP + " - " + temp.TenHP );
        holder.txtTenGV.setText(temp.TenGV);
        //đọc hình ảnh từ mipmap/ drawable
        int imageId = parent.getContext().getResources().getIdentifier(temp.TenHinh, defType: "mipmap", parent.getContext().getPackageName());
        if(imageId != 0)
            holder.imgView.setImageResource(imageId);
        return convertView;
    }

    static class ViewHolder {
        ImageView imgView;
        TextView txtMonHoc;
        TextView txtTenGV;
    }
}
```

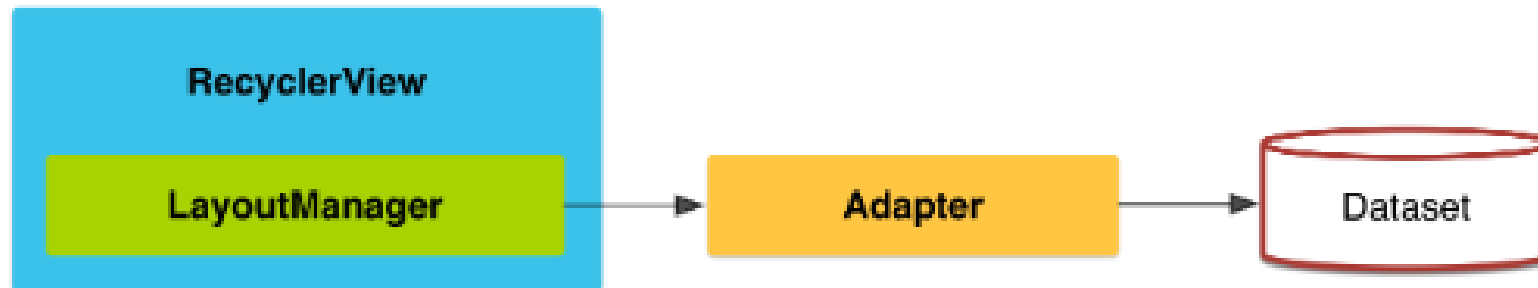

RecyclerView

- Linh hoạt và cung cấp nhiều tính năng hơn ListView
- Khai báo trong Android

```
<androidx.recyclerview.widget.RecyclerView
    android:id="@+id/rcvView"
    android:layout_width="match_parent"
    android:layout_height="match_parent" />
```

- So sánh với ListView:

- ☐ Yêu cầu **ViewHolder** ở **Adapter**
- ☐ Cho phép bố trí item layout **theo dọc** hoặc **ngang**
- ☐ Hiệu ứng **ItemAnimator**
- ☐ Dễ custom việc chia (**ItemDecoration**) các mục trong danh sách
- ☐ Có sự kiện **OnItemTouchListener**



RecyclerView

Các bước thực hiện

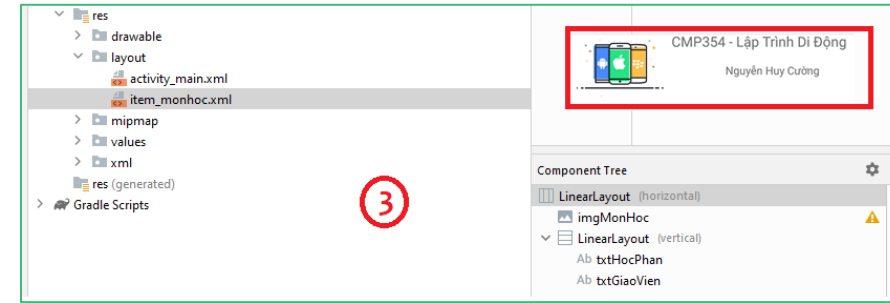
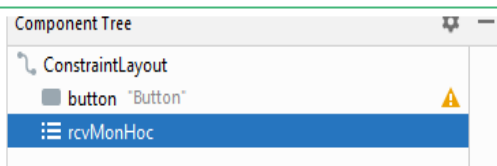
- (1) Định nghĩa model (2) Thêm RecyclerView vào Activity
- (3) Tạo item từ layout XML file
- (4) Tạo [ModelAdapter] extends `RecyclerView.Adapter<ModelAdapter.ViewHolder>` và implement các thành phần
- (5) Bind Adapter ở Activity Java (sử dụng)

```
public class MonHoc {
    public String TenHinh;
    public String MaHP;
    public String TenHP;
    public String TenGV;

    public MonHoc(String th, String maHP, String tenHP, String tenGV) {
        TenHinh = th;
        MaHP = maHP;
        TenHP = tenHP;
        TenGV = tenGV;
    }

    public static List<MonHoc> LayDSMonHoc() {
        List<MonHoc> lstMonHoc = new ArrayList<>();
        lstMonHoc.add(new MonHoc( th: "didong", maHP: "CMP354", tenHP: "Lập Trình Di Động", tenGV: "Nguyễn Huy Cường"));
        lstMonHoc.add(new MonHoc( th: "java", maHP: "CMP324", tenHP: "Lập Trình Java", tenGV: "Nguyễn Văn A"));
        lstMonHoc.add(new MonHoc( th: "window", maHP: "CMP332", tenHP: "Lập Trình Windows", tenGV: "Nguyễn Văn B"));
        return lstMonHoc;
    }
}
```

```
<androidx.recyclerview.widget.RecyclerView
    android:id="@+id/rcvMonHoc"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:layout_editor_absoluteX="0dp"
    tools:layout_editor_absoluteY="0dp" />
```



```
public class Adapter extends RecyclerView.Adapter<Adapter.ViewHolder> {
    List<MonHoc> lstMonHoc;
    public Adapter(List<MonHoc> lstMonHocs) { lstMonHoc = lstMonHocs; }

    @NonNull
    @Override
    public Adapter.ViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {
        // đọc xml layout file và chuyển đổi các thuộc tính của nó thành 1 View (LayoutInflater)
        return null;
    }

    @Override
    public void onBindViewHolder(@NonNull Adapter.ViewHolder holder, int position) {
        //Bind cho 1 ViewHolder ở vị trí position
    }

    @Override
    public int getItemCount() {
        return 0;
    }

    static class ViewHolder extends RecyclerView.ViewHolder {
        public ViewHolder(@NonNull View itemView) {
            super(itemView);
            //ánh xạ view sử dụng view.findViewById
        }
    }
}
```

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    // Ảnh xq RecyclerView
    RecyclerView rvMonHoc = (RecyclerView) findViewById(R.id.rcvMonHoc);
    // Tạo Adapter và truyền danh sách các môn học
    MonHocAdapter adapter = new MonHocAdapter(MonHoc.LayDSMonHoc());
    rvMonHoc.setAdapter(adapter);
    rvMonHoc.setLayoutManager(new LinearLayoutManager( context: this));
}
```



RecyclerView – Sử dụng và chuẩn bị model

(1) Định nghĩa lớp **MONHOC** (2) Đưa RecyclerView vào giao diện

```
public class MonHoc{
    public String TenHinh;
    public String MaHP;
    public String TenHP;
    public String TenGV;

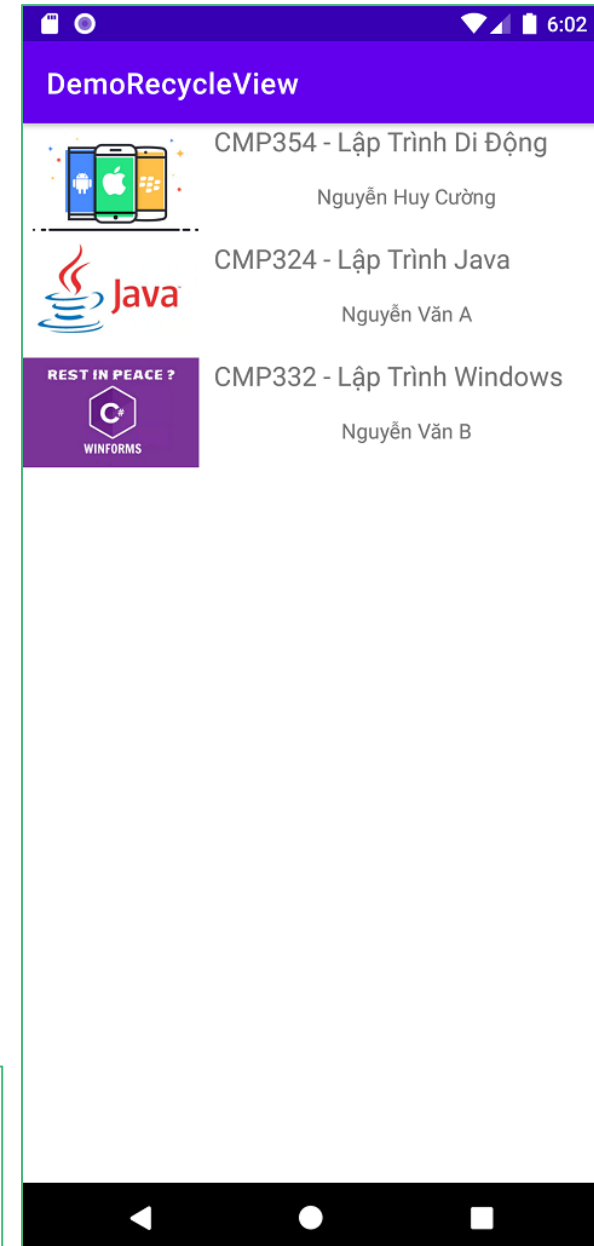
    public MonHoc(String th, String maHP, String tenHP, String tenGV) {
        TenHinh = th;
        MaHP = maHP;
        TenHP = tenHP;
        TenGV = tenGV;
    }

    public static List<MonHoc> LayDSMonHoc() {
        List<MonHoc> lstMonHoc= new ArrayList<>();
        lstMonHoc.add(new MonHoc( th: "didong", maHP: "CMP354", tenHP: "Lập Trình Di Động", tenGV: "Nguyễn Huy Cường"));
        lstMonHoc.add(new MonHoc( th: "java", maHP: "CMP324", tenHP: "Lập Trình Java", tenGV: "Nguyễn Văn A"));
        lstMonHoc.add(new MonHoc( th: "window", maHP: "CMP332", tenHP: "Lập Trình Windows", tenGV: "Nguyễn Văn B"));
        return lstMonHoc;
    }
}
```

```
<androidx.recyclerview.widget.RecyclerView
    android:id="@+id/rcvMonHoc"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:layout_editor_absoluteX="0dp"
    tools:layout_editor_absoluteY="0dp" />
```

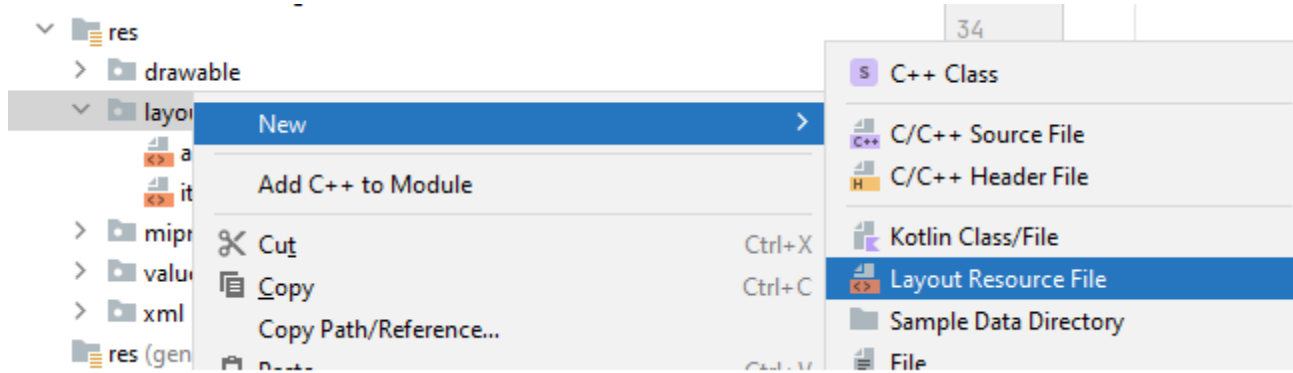
Component Tree

```
ConstraintLayout
├── button "Button"
└── rcvMonHoc
```

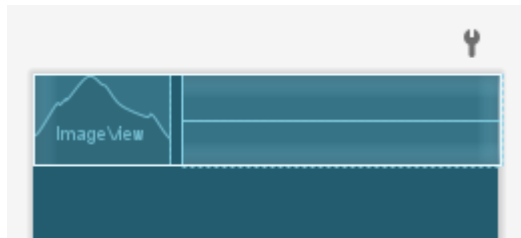


RecyclerView – thiết kế item layout

(3) Tạo item_monhoc từ layout XML file



Thiết kế giao diện tương ứng với 1 item



```
<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:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal"
    android:textAlignment="viewEnd">

    <ImageView
        android:id="@+id/imgMonHoc"
        android:layout_width="120dp"
        android:layout_height="80dp"
        android:scaleType="fitStart"
        android:layout_marginRight="10dp"
        android:src="@mipmap/ic_launcher" />

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:orientation="vertical"
        android:weightSum="2">

        <TextView
            android:id="@+id/txtHocPhan"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:textSize="18sp"
            android:layout_weight="1"
            />

        <TextView
            android:id="@+id/txtGiaoVien"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:layout_weight="1"
            android:textAlignment="center" />

    </LinearLayout>
</LinearLayout>
```



RecyclerView – Tạo Adapter và ViewHolder

(4) Tạo **MonHocAdapter** extends **RecyclerView.Adapter<ViewHolder>** để render item

- ☐ Tạo lớp **MonHocAdapter** kế thừa **RecyclerView.Adapter**
- ☐ Định nghĩa thành phần **ViewHolder** (tương ứng **item_monhoc**)
- ☐ Ánh xạ **view** tương ứng
- ☐ Tạo danh sách **List<MonHoc>** để chuẩn bị đưa dữ liệu vào Adapter
- ☐ Mỗi Adapter có 3 thành phần chính
onCreateViewHolder
onBindViewHolder
getItemCount

```
public class MonHocAdapter extends RecyclerView.Adapter<MonHocAdapter.ViewHolder> {

    private List<MonHoc> lstMonHoc;

    public MonHocAdapter(List<MonHoc> lstMonHocs) {
        lstMonHoc = lstMonHocs;
    }

    @NonNull
    @Override
    public ViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {
        return null;
    }

    @Override
    public void onBindViewHolder(@NonNull ViewHolder holder, int position) {

    }

    @Override
    public int getItemCount() {
        return 0;
    }

    public class ViewHolder extends RecyclerView.ViewHolder {
        ImageView imgView;
        TextView txtMonHoc;
        TextView txtTenGV;

        public ViewHolder(@NonNull View itemView) {
            super(itemView);

            imgView = (ImageView) itemView.findViewById(R.id.imgMonHoc);
            txtMonHoc = (TextView) itemView.findViewById(R.id.txtHocPhan);
            txtTenGV = (TextView) itemView.findViewById(R.id.txtGiaoVien);
        }
    }
}
```



RecyclerView

4. Implements các thành phần

chính trong MonHocAdapter

❑ **onCreateViewHolder**: cần chuyển layout

item_monhoc.xml sang view sử dụng **LayoutInflater**

❑ **OnBindViewHolder**

Render *item_monhoc* ở vị trí *position*

❑ **getItemCount**

Số lượng item hiển thị ở recyclerView

```
@NonNull
@Override
public ViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {
    Context context = parent.getContext();
    LayoutInflater inflater = LayoutInflater.from(context);
    // đọc xml layout file và chuyển đổi các thuộc tính của nó thành 1 View
    View monhocView = inflater.inflate(R.layout.item_monhoc, parent, attachToRoot: false);
    // Return a new holder instance
    ViewHolder viewHolder = new ViewHolder(monhocView);
    return viewHolder;
}
```

```
@Override
public void onBindViewHolder(@NonNull ViewHolder holder, int position) {
    // Get the data model based on position
    MonHoc temp = lstMonHoc.get(position);
    // Set item views
    holder.txtMonHoc.setText(temp.MaHP + " - " + temp.TenHP );
    holder.txtTenGV.setText(temp.TenGV);
    Context context = holder.imageView.getContext();
    int imageId = context.getResources().getIdentifier(temp.TenHinh, defType: "mipmap", context.getPackageName());
    if(imageId != 0)
        holder.imageView.setImageResource(imageId);
}
```

```
@Override
public int getItemCount() {
    return lstMonHoc.size();
}
```

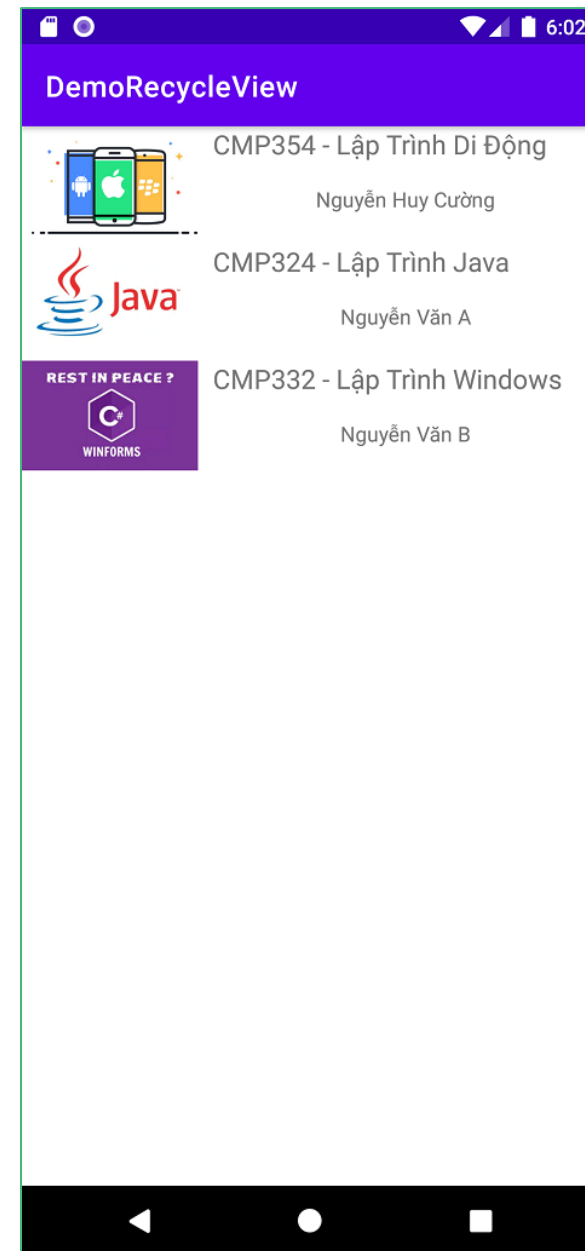



RecyclerView – Sử dụng setAdapter

5. Sử dụng trong Activity & Kết quả

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

    // Anh xạ RecyclerView
    RecyclerView rvMonHoc = (RecyclerView) findViewById(R.id.rcvMonHoc);
    // Tạo Adapter và truyền danh sách các môn học
    MonHocAdapter adapter = new MonHocAdapter(MonHoc.LayDSMonHoc());
    rvMonHoc.setAdapter(adapter);
    rvMonHoc.setLayoutManager(new LinearLayoutManager(context, this));
}
```



RecyclerView

Khác biệt với ListView

- ❑ Thêm trực tiếp vào RecyclerView

```
findViewById(R.id.button).setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        lstMonHoc.add(0, new MonHoc( th: "didong", maHP: "ZZZ", tenHP: "XXX", tenGV: "YYY"));
        adapter.notifyItemInserted( position: 0);
    }
});
```

- ❑ Có thể fixed Size để tăng performance (mặc định scrollable)

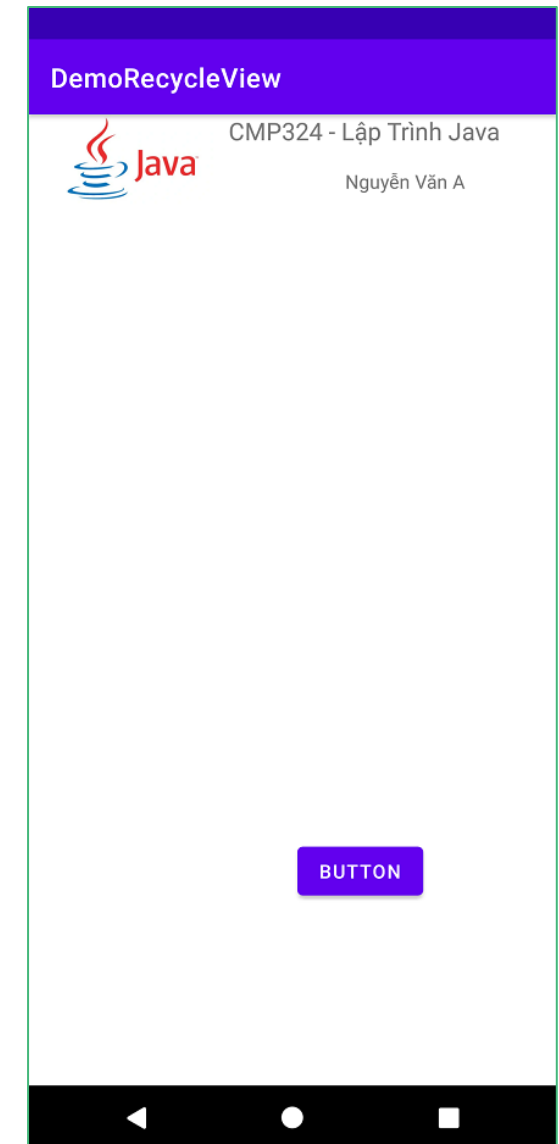
```
rvMonHoc.setHasFixedSize(true);
```

- ❑ Scroll ở vị trí bất kì

```
rvMonHoc.scrollToPosition(2);
```

- ❑ Dễ dàng điều chỉnh layout: chuyển sang chiều ngang và kết hợp custom nhiều loại

```
LinearLayoutManager layoutManager = new LinearLayoutManager(this,
    LinearLayoutManager.HORIZONTAL, false);
layoutManager.scrollToPosition(1);
rvMonHoc.setLayoutManager(layoutManager);
```



RecyclerView

Khác biệt với ListView

❑ Custom việc chia giữa các mục trong danh sách (**DividerItemDecoration**)

```
RecyclerView.ItemDecoration itemDecoration = new DividerItemDecoration(this,
    DividerItemDecoration.VERTICAL);
rvMonHoc.addItemDecoration(itemDecoration);
```

❑ Animators

1. Cài đặt thêm thư viện ở Gradle

```
implementation 'jp.wasabeef:recyclerview-animators:4.0.2'
```

2. Sử dụng

```
rvMonHoc.setItemAnimator(new SlideInUpAnimator());
```

❑ Touch Event: **addOnItemTouchListener**

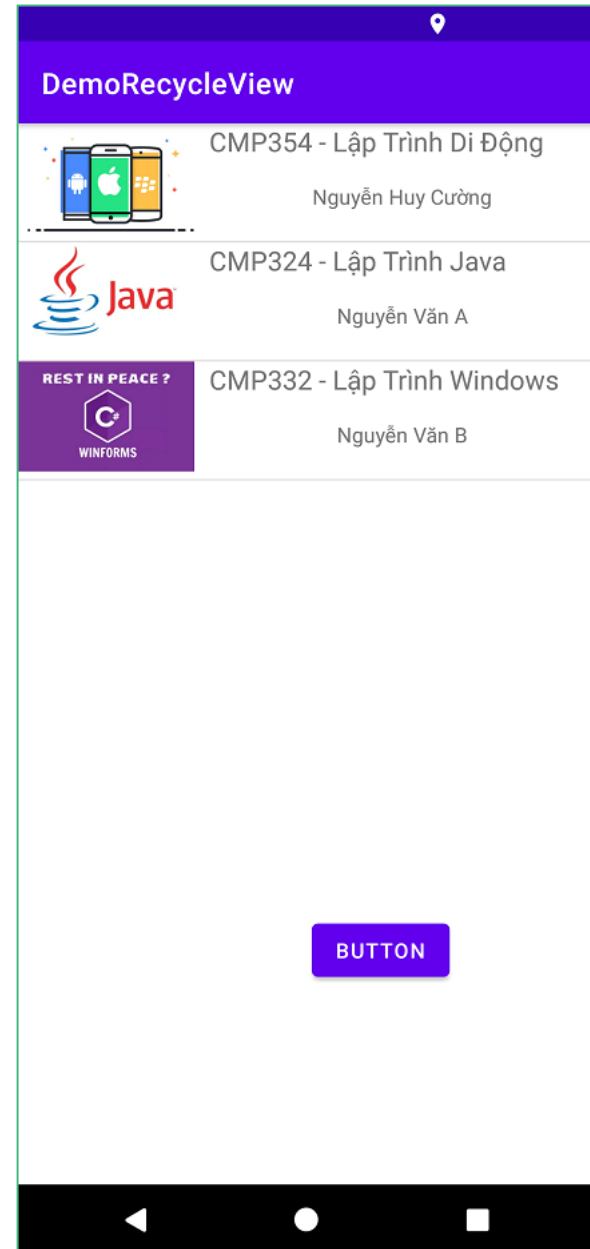
```
rvMonHoc.addOnItemTouchListener(new RecyclerView.OnItemTouchListener() {
    @Override
    public boolean onInterceptTouchEvent(@NonNull RecyclerView rv, @NonNull MotionEvent e) {
        return false;
    }

    @Override
    public void onTouchEvent(@NonNull RecyclerView rv, @NonNull MotionEvent e) {

    }

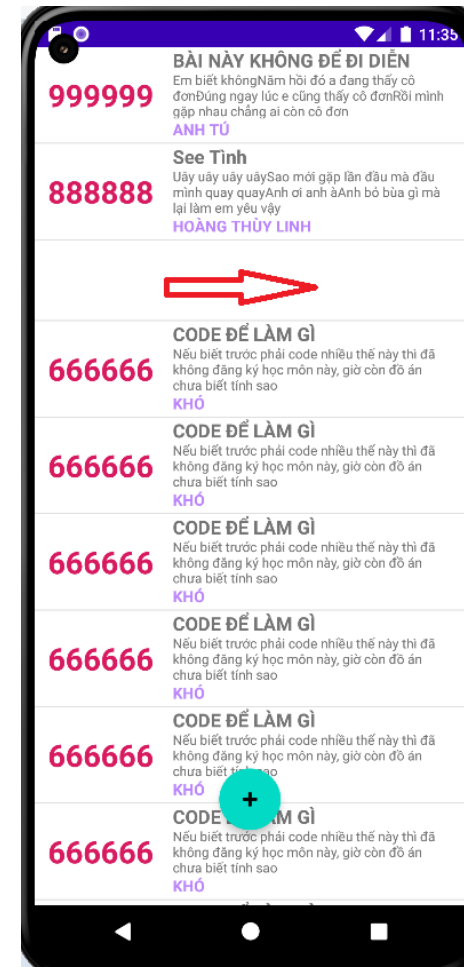
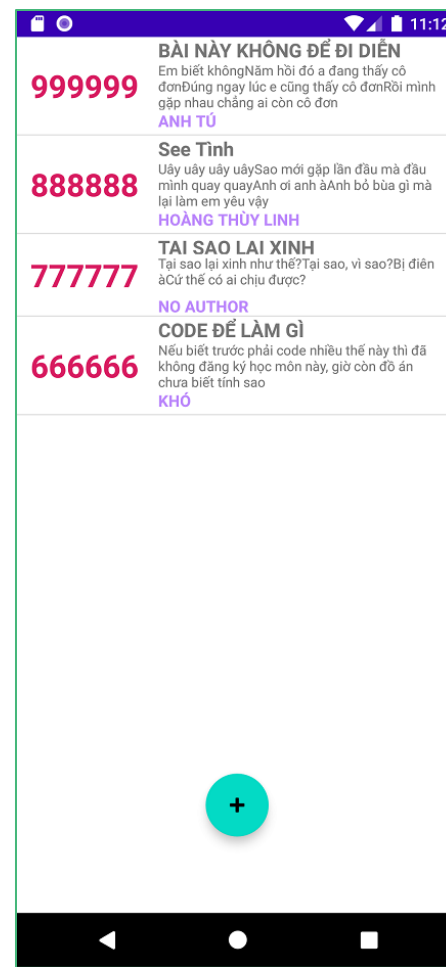
    @Override
    public void onRequestDisallowInterceptTouchEvent(boolean disallowIntercept) {

    }
});
```



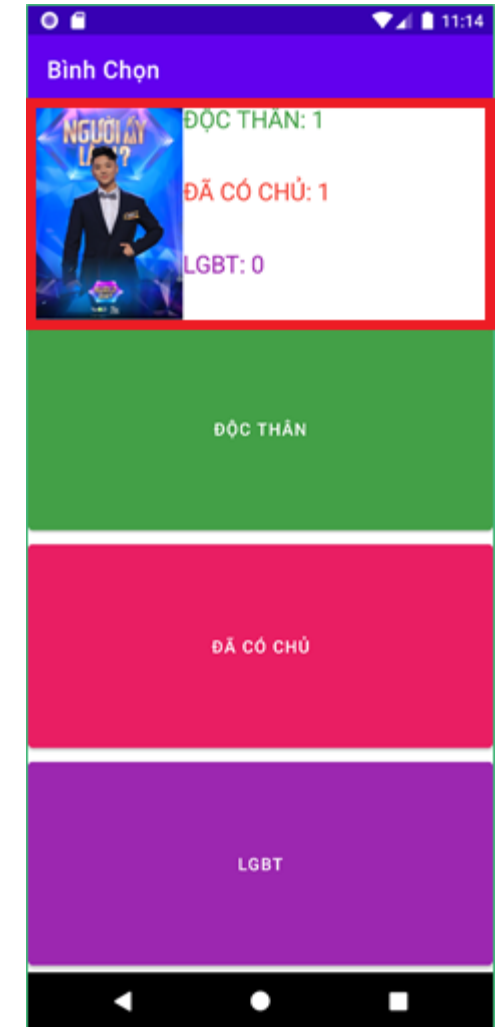
Exercise 3.1 Karaoke 6 số

1. Thiết kế giao diện chương trình “Karaoke 6 số” sử dụng **RecyclerView** (6đ)
2. Thêm **floatActionButton** cho phép thêm một bài hát mới (lấy bài hát cuối cùng) để thêm tiếp vào dưới **RecyclerView** hiện tại (2đ)
3. **Swipe To Delete** (2đ)
(khi vuốt từ phải qua trái hoặc từ trái qua phải sẽ delete item này)



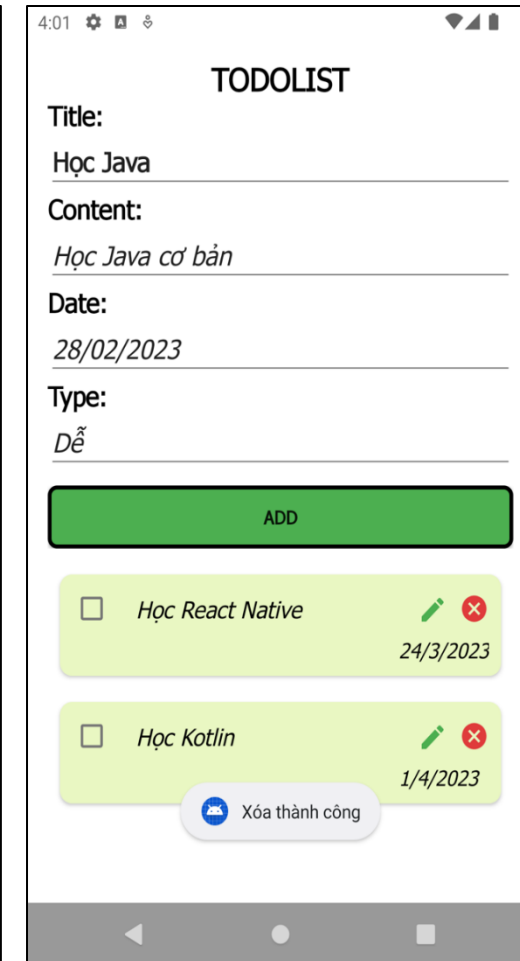
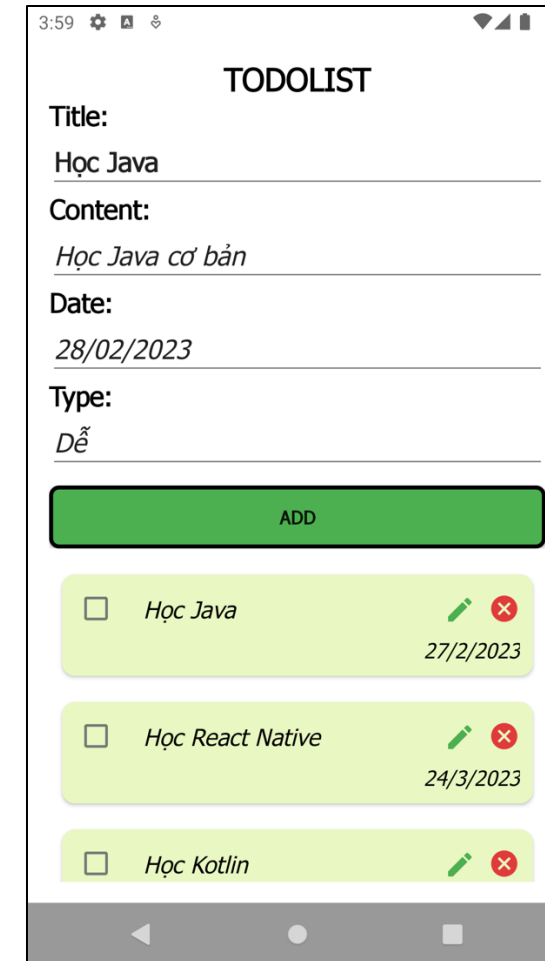
Exercise 3.2 Người ấy là ai ?

1. **Thiết kế giao diện** bình chọn như ở chương trình “Người ấy là ai” (5đ)
2. Xử lý sự kiện khi người dùng chọn “Độc Thân”, “Đã có Chủ” hoặc “LGBT”. Cập nhật lại giá trị sau khi bình chọn
khi click vào các bình chọn sẽ tăng số lượng bình chọn (ban đầu tất cả đều là 0) (3đ)
3. Sử dụng **RecyclerView** (cho phần Image và các kết quả bình chọn) để có thể bình chọn theo Horizontal cho 5 ứng viên. (2đ)



Exercise 3.3 ToDoList

1. Hiển thị môn học bằng RecyclerView sau Khi Add vào (title, content, Date, Type) (6đ)
2. Xử lý chức năng xóa một công việc có trong danh sách: khi click vào x (2đ)



Exercise 3.3 ToDoList

3. Xây dựng chức năng update trạng thái công việc (2đ)

(dung chung với nút ADD)

