# ANIMATION MATERIAL DESIGN

## A. MỤC TIÊU

- Nắm vững lý thuyết, cách tạo và sử dụng animation.

- Hiểu biết cơ bản về lý thuyết thiết kế Material.

- Ứng dụng phong cách thiết kế, hiệu ứng vào các sản phẩm.

## B. NỘI DUNG

- Kiến thức cơ bản về Tween Animation.

- Các nguyên tắc thiết kế Material Design và các component của nó.

- Các kiểu chữ Typography

- Các Layout

**-** Components và Pattern

## C. KẾT QUẢ SAU KHI HOÀN THÀNH

Viết được các ứng dụng sử dụng của animation trong Android

Xây dựng được các ứng dụng có sử dụng animation và thiết kế theo nguyên tắc của Material design**.**

## D. YÊU CẦU PHẦN CỨNG - PHẦN MỀM

**-** Máy tính cài HĐH windows, RAM tối thiểu 1GB, có kết nối Internet.

**-** Phần mềm NETBEAN 8.0, JDK 1.8. MYSQL.

## E. HƯỚNG DẪN

### 1. Animation

- Trong Android,animation là quá trình tạo chuyển động và thay đổi hình dạng

- Animation trong android có thể được thực hiện bằng nhiều cách. Trong bài viết này chúng ta sẽ học một cách tạo animation trong Android được sử dụng rộng rãi và dễ dàng nhất gọi là Tween Animation.

**1.1. Tween Animation**

- Tween animation có một số tham số như start value , end value, size , time duration, rotation angle, … và thực hiện animation trên đối tượng. Nó có thể được áp dụng cho bất kỳ đối tượng nào. Để sử dụng Tween animation, Android cung cấp một lớp được gọi là Animation.

- Để thực hiện animation trong android, chúng ta sẽ gọi một phương thức tĩnh của lớp AnimationUtils đó là loadAnimation() theo cú pháp sau

***Animation animation = AnimationUtils.loadAnimation(getApplicationContext(),***

***R.anim.myanimation);***

Lưu ý tham số thứ hai là tên của tập tin xml được sử dụng để tạo animation. Chúng ta phải tạo một thư mục mới có tên gọi anim trong thư mục res và tạo một file xml trong thư mục anim này.

**- Các thuộc tính quan trọng của tập tin XML**

|  |  |
| --- | --- |
| **Thuộc tính** | **Miêu tả** |
| android:duration | Thời gian hoàn thành |
| android:startOffset | Thời gian chờ trước khi một animaiton bắt đầu và thường được sử dụng khi có nhiều animation |
| android:repeatMode | Thiết lập lặp lại animation |
| android:repeatCount | Xác định số lần lặp lại animation. Nếu bạn thiết lập giá trị này là **infinite** thì animation sẽ lặp lại lần vô hạn |
| android:interpolator | Tỷ lệ thay đổi animation |
| android:fillAfter | Xác định liệu có áp dụng việc chuyển đổi đối tượng về trạng thái ban đầu sau khi một animation đã hoàn thành hay không. |

**- Một số phương thức thường dùng của lớp Animation**

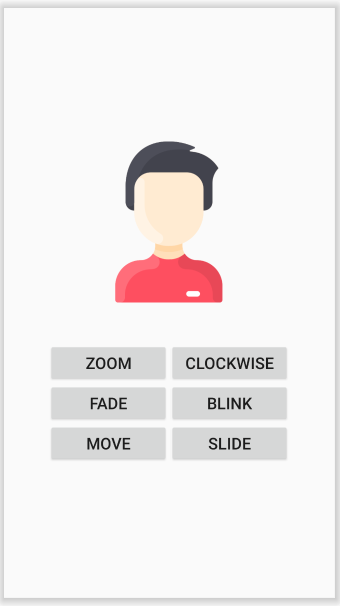
|  |  |
| --- | --- |
| **STT** | **Phương thức và miêu tả** |
| 1 | **start()**Bắt đầu animation. |
| 2 | **setDuration(long duration)**Thiết lập thời gian của một animation. |
| 3 | **end()**Kết thúc animation. |
| 4 | **cancel()**Huỷ animation. |

Để áp dụng animation cho một đối tượng, chúng ta sẽ gọi phương thức **startAnimation()** của đối tượng muốn áp dụng theo cú pháp sau:

***ImageView image1 = (ImageView)findViewById(R.id.imageView1);***

***image.startAnimation(animation);***

**1.2. Ví dụ 1:** Minh họa cách sử dụng của animation trong Android theo hình sau:

Ta có thể chọn các loại animation khác nhau trên màn hình và ImageView phía trên sẽ được áp dụng các loại animation ấy.

Các bước thực hiện:

|  |  |
| --- | --- |
| Bước | Mô tả |
| 1 | Tạo một project Android mới để thực hành |
| 2 | Sửa file MainActivity để thêm code animation |
| 3 | Sửa file layout XML để thêm các thành phần UI cần thiết |
| 4 | Tạo folder mới trong mục res đặt tên là anim |
| 5 | Chuột phải vào anim và chọn tạo file XML mới |
| 6 | Tạo các file myanimation.xml,clockwise.xml,fade.xml,move.xml ,blink.xml và slide.xml. |
| 7 | Chạy chương trình để thử nghiệm |

***Bước 1:*** Tạo project mới

***Bước 2:*** Chỉnh sửa file MainActivity

public class MainActivity extends AppCompatActivity {  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.activity\_main);  
 }  
 public void clockwise(View view){  
 ImageView image = (ImageView)findViewById(R.id.imageView);  
 Animation animation =  
 AnimationUtils.loadAnimation(getApplicationContext(),  
 R.anim.clockwise);  
 image.startAnimation(animation);  
 }  
 public void zoom(View view){  
 ImageView image = (ImageView)findViewById(R.id.imageView);  
 Animation animation1 =  
 AnimationUtils.loadAnimation(getApplicationContext(),  
 R.anim.myanimation);  
 image.startAnimation(animation1);  
 }  
 public void fade(View view){  
 ImageView image = (ImageView)findViewById(R.id.imageView);  
 Animation animation1 =  
 AnimationUtils.loadAnimation(getApplicationContext(),  
 R.anim.fade);  
 image.startAnimation(animation1);  
 }  
 public void blink(View view){  
 ImageView image = (ImageView)findViewById(R.id.imageView);  
 Animation animation1 =  
 AnimationUtils.loadAnimation(getApplicationContext(),  
 R.anim.blink);  
 image.startAnimation(animation1);  
 }  
 public void move(View view){  
 ImageView image = (ImageView)findViewById(R.id.imageView);  
 Animation animation1 =  
 AnimationUtils.loadAnimation(getApplicationContext(), R.anim.move);  
 image.startAnimation(animation1);  
 }  
 public void slide(View view){  
 ImageView image = (ImageView)findViewById(R.id.imageView);  
 Animation animation1 =  
 AnimationUtils.loadAnimation(getApplicationContext(), R.anim.slide);  
 image.startAnimation(animation1);  
 }  
}

***Bước 3***: Chỉnh sửa file XML

<LinearLayout android:layout\_height="match\_parent"

android:layout\_width="match\_parent"

xmlns:tools="http://schemas.android.com/tools"

android:orientation="vertical"

android:layout\_margin="16dp"

android:gravity="center"

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

<ImageView

android:id="@+id/imageView"

android:src="@drawable/person"

android:layout\_marginBottom="50dp"

android:layout\_width="200dp"

android:layout\_height="200dp"/>

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:gravity="center"

android:orientation="horizontal">

<Button

android:id="@+id/btnzoom"

android:text="Zoom"

android:onClick="zoom"

android:textSize="20sp"

android:layout\_width="150dp"

android:layout\_height="50dp" />

<Button

android:id="@+id/btnclock"

android:text="Clockwise"

android:textSize="20sp"

android:onClick="clockwise"

android:layout\_width="150dp"

android:layout\_height="50dp" />

</LinearLayout>

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:gravity="center"

android:orientation="horizontal">

<Button

android:id="@+id/btnfade"

android:text="Fade"

android:textSize="20sp"

android:onClick="fade"

android:layout\_width="150dp"

android:layout\_height="50dp" />

<Button

android:id="@+id/btnblink"

android:text="Blink"

android:textSize="20sp"

android:onClick="blink"

android:layout\_width="150dp"

android:layout\_height="50dp"/>

</LinearLayout>

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:gravity="center"

android:orientation="horizontal">

<Button

android:id="@+id/btnmove"

android:text="Move"

android:onClick="move"

android:textSize="20sp"

android:layout\_width="150dp"

android:layout\_height="50dp" />

<Button

android:id="@+id/btnslide"

android:text="Slide"

android:onClick="slide"

android:textSize="20sp"

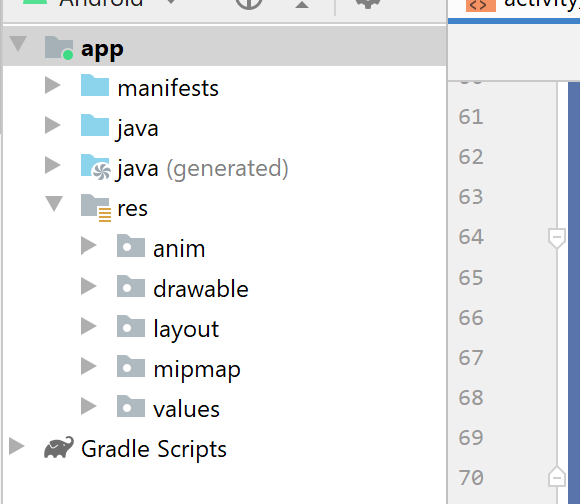
android:layout\_width="150dp"

android:layout\_height="50dp"/>

</LinearLayout>

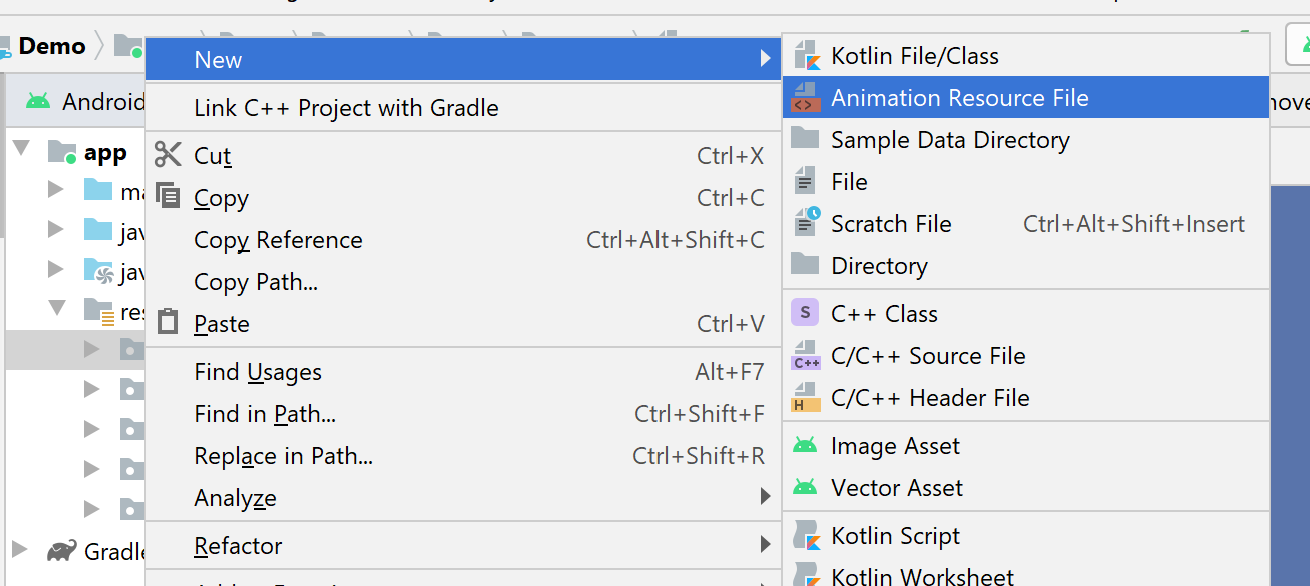
</LinearLayout>

***Bước 4:*** Thêm thư mục anim trong res



Hình 10.1. Thư mục anim trong res

***Bước 5:*** Chọn New -> Animation Resources File



Hình 10.2. Animation Resources File

***Bước 6:*** Tạo các file animation

* Code XML của myanimation

*<?*xml version="1.0" encoding="utf-8"*?>*

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

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

android:fromXScale="0.5"

android:toXScale="3.0"

android:fromYScale="0.5"

android:toYScale="3.0"

android:duration="5000"

android:pivotX="50%"

android:pivotY="50%" >

</scale>

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

android:startOffset="5000"

android:fromXScale="3.0"

android:toXScale="0.5"

android:fromYScale="3.0"

android:toYScale="0.5"

android:duration="5000"

android:pivotX="50%"

android:pivotY="50%" >

</scale>

</set>

* Code XML của clockwise

*<?*xml version="1.0" encoding="utf-8"*?>*

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

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

android:fromDegrees="0"

android:toDegrees="360"

android:pivotX="50%"

android:pivotY="50%"

android:duration="5000" >

</rotate>

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

android:startOffset="5000"

android:fromDegrees="360"

android:toDegrees="0"

android:pivotX="50%"

android:pivotY="50%"

android:duration="5000" >

</rotate>

</set>

Code XML của fade

*<?*xml version="1.0" encoding="utf-8"*?>*

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

android:interpolator="@android:anim/accelerate\_interpolator" >

<alpha

android:fromAlpha="0"

android:toAlpha="1"

android:duration="2000" >

</alpha>

<alpha

android:startOffset="2000"

android:fromAlpha="1"

android:toAlpha="0"

android:duration="2000" >

</alpha>

</set>

Code XML của blink

*<?*xml version="1.0" encoding="utf-8"*?>*

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

<alpha android:fromAlpha="0.0"

android:toAlpha="1.0"

android:interpolator="@android:anim/accelerate\_interpolator"

android:duration="600"

android:repeatMode="reverse"

android:repeatCount="infinite"/>

</set>

Code XML của move

*<?*xml version="1.0" encoding="utf-8"*?>*

<set

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

android:interpolator="@android:anim/linear\_interpolator"

android:fillAfter="true">

<translate

android:fromXDelta="0%p"

android:toXDelta="75%p"

android:duration="800" />

</set>

Code XML của slide

*<?*xml version="1.0" encoding="utf-8"*?>*

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

android:fillAfter="true" >

<scale

android:duration="500"

android:fromXScale="1.0"

android:fromYScale="1.0"

android:interpolator="@android:anim/linear\_interpolator"

android:toXScale="1.0"

android:toYScale="0.0" />

</set>

***Bước 7:*** Chạy thử nghiệm chương trình

|  |  |  |
| --- | --- | --- |
| Zoom | Clockwise | Fade |
| Blink (nhấp nháy) | Move | Slide (cuộn) |

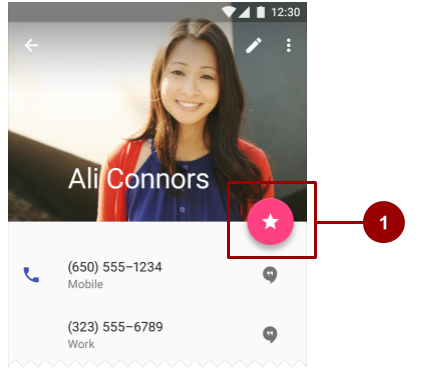
### 2. Material Design

Material Design để trải nghiệm người dùng thống nhất trên các nền tảng và kích thước thiết bị. Material Design bao gồm một danh sách các hướng dẫn về style, layout, motion và các khía cạnh khác của app design. Material Design dùng cho ứng dụng Web cũng như mobile. Nhưng trong phần này chỉ sẽ tập trung chủ yếu vào Material Design cho ứng dụng Android Mobile.

**2.1. Các nguyên tắc của Material Design**

Trong Material Design các thành phần trong ứng dụng Android giống như các thực thể trong thế giới thực: Chúng đổ bóng (giống như một vật nào có có ánh nắng mặt trời chiếu vào thì sẽ có bóng), chúng chiếm không gian, và tương tác với những những thành phần khác.

* *Bold, graphic, intentional*

Material Design liên quan đến màu sắc có chủ đích, hình ảnh góc cạnh, kiểu chữ quy mô lớn và khoảng trắng có mục đích tạo ra giao diện đậm và đồ họa. Giao diện được thiết kế để nhấn mạnh hành động của người dùng, hướng dẫn họ biết phải làm gì và làm như thế nào. Ví dụ như làm nổi bật những thứ mà người dùng có thể tương tác chẳng hạn như button, edittext, switch.

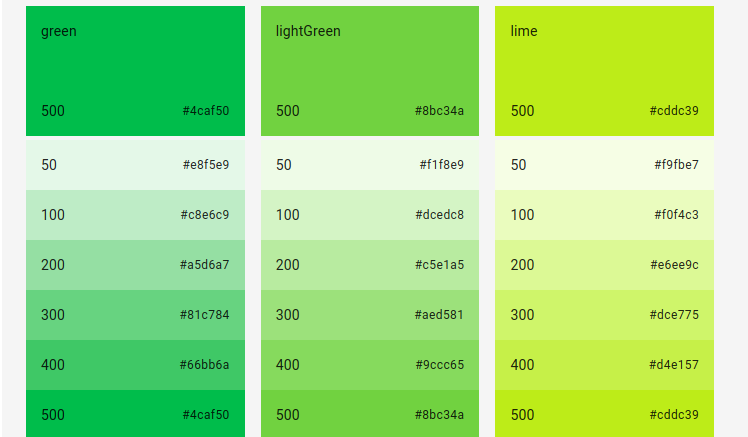
* *Meaningful motion (chuyển động có ý nghĩa)*

Làm cho hình ảnh động và chuyển động của ứng dụng chúng ta có ý nghĩa, tức không để chúng xảy ra một cách ngẫu nhiên. Sử dụng các chuyển động để cũng cố rằng người dùng đang làm chủ ứng dụng. Ví dụ như ta sẽ thiết kế ra ứng dụng mà hầu hết các chuyển động được bắt đầu bằng hành động của người dùng, không phải bởi các sự kiện nằm ngoài sự kiểm soát của người dùng. Ta cũng có thể dùng motion để tập trung sự chú ý của người dùng, cung cấp cho họ những phản hồi tinh tế hoặc làm nổi bật một vài yếu tố cần thiết của ứng dụng. Khi ứng dụng của ta đang trình bày một đối tượng, một sự kiện gì đó cho người dùng thì hãy đảm bảo chuyển động không phá vỡ tính liên tục của trải nghiệm người dùng. Ví dụ như người dùng không cần phải đợi animation hoặc quá trình chuyển đổi hoàn tất.

**2.2. Color**

Các nguyên tắc của Material Design bao gồm cả việc sử dụng màu sắc thế nào cho hợp lý.

* Bảng màu của Material Design



Hình 10.3. Bảng màu của Material Design

Khi chúng ta start một project bởi Android Studio thì trong file color.xml đã có 3 màu được định nghĩa:

**<resources>**

**<color name="colorPrimary">#FFEC86</color>**

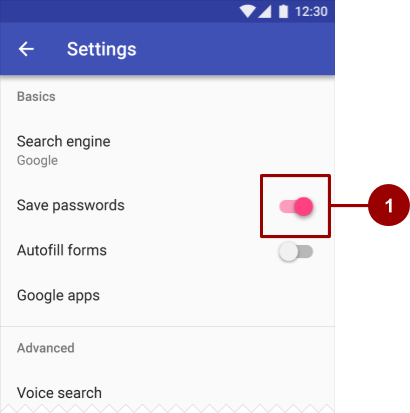
**<color name="colorPrimaryDark">#3700B3</color>**

**<color name="colorAccent">#DD2C00</color>**

**</resources>**

Và chúng được áp dụng cho Theme mặc định, và sẽ áp dụng cho các thành phần trong ứng dụng:

* colorPrimary được sử dụng cho một số View theo mặc định. Ví dụ trong AppTheme thì nó được dùng làm màu cho thanh action bar. Ta có thể thay giá trị này bằng màu 500 trong cột màu mà chúng ta chọn.
* colorPrimaryDark cần sử dụng ở những nơi cần tương phản nhẹ với primary color. Ví dụ thanh trạng thái phía trên ứng dụng. Màu này ta thường chọn tối hơn một chút màu 500.
* colorAccent được sử dụng làm màu tô sáng cho một số thành phần View. Nó cũng được sử dụng cho các công tắc ở ví trí tắt - bật, FloatingActionButton, …
* Như hình ảnh dưới thì hình của thanh action bar sử dụng colorPrimary (màu chàm), thanh status bar sử dụng colorPrimaryDark (một màu chàm tối hơn), công tắc sử dụng màu colorAccent (là một màu hồng).



* *Contrast (Độ tương phản):*

Bạn phải luôn đảm bảo rằng các văn bản trong UI phải tương phản với background của UI. Khi background tối thì văn bản có màu sáng và ngược lại.Nếu bạn sử dụng một theme như Theme.AppCompat, độ tương phản giữa văn bản và background sẽ được xử lý trước cho bạn.

* *Opacity (Độ mờ đục):*

Ứng dụng của ta có thể hiện thì độ mờ đục khác nhau của văn bản để truyền đạt tầm quan trọng khác nhau đến người dùng. Để set độ mờ đục ta có thể dụng format "#argb" hoặc "#aarrggbb" trong đó a và aa là độ alpha của color.

**2.3. Typography**

* Typeface: Roboto là kiểu chữ chuẩn của Material Design trong Android. Nó có sáu loại: Thin, Light, Regular, Medium, Bold, Black.

****

* Fontstyles:

Android cung cấp các kiểu và kích thước font chữ được xác định trước mà chúng ta có thể sử dụng, chúng được phát triển để cân bằng mật độ nội dung và sự thoải mái khi đọc trong các điều kiện khác nhau. Kích thước có đơn vị là sp (pixel có thể mở rộng). Không nên sử dụng nhiều style và size cùng nhau trong cùng một layout.

****

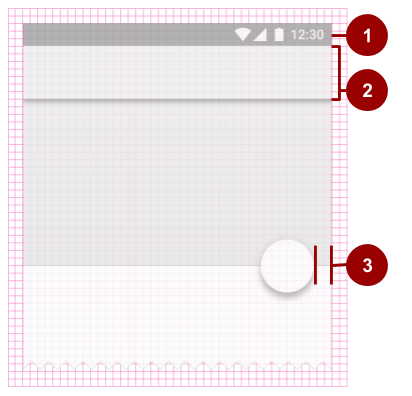
Ta có thể sử dụng thuộc tính android:textAppearance hoặc TextAppearance.AppCompat. Ví dụ để cho TextView hiển thị kiểu chữ loại 3 ta khai báo như đây:

***android:textAppearance="@style/TextAppearance.AppCompat.Display3"***

**2.4. Layout**

Layout được viết bằng XML và được liệt kê trong thư mục res của ứng dụng. Sau đây là một số thiết kế tốt cho layout:

* Metrics

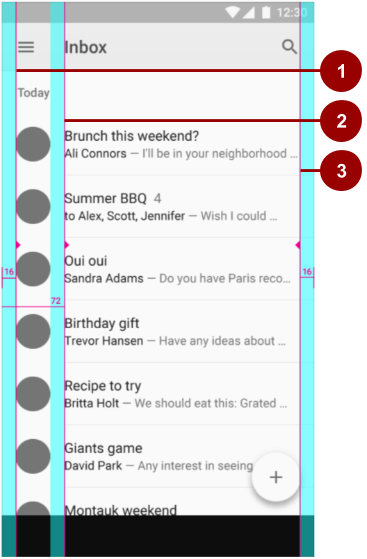


Trong hình trên:

1. Thanh status 3 cao 24dp, tức bằng 3 ô vuông.
2. Thanh Toolbar cao 56dp bằng 7 ô vuông.
3. Cách lề 16dp, bằng chiều rộng của 2 ô vuông  
   Các icon của toolbar được căn chỉnh với các ô vuông 4dp thay vì 8dp nên kích thước của chúng trên toolbar là bội số của 4.

- Keylines

Là những phác thảo trong xác định vị trí của văn bản và biểu tượng. Ví dụ các keylines đánh dấu các cạnh của lề trong bố cục.



Như hình trên thì:

1. Keyline hiển thị lề trái cho cạnh màn hình, trong trường hợp này là 16dp
2. Thể hiện avatar hoặc icon với các margin là 72dp
3. Thể hiện lề bên phải là 16dp.

Các kiểu chữ trong Material design được căn chỉnh theo lưới đơn vị là 4dp theo chiều ngang.

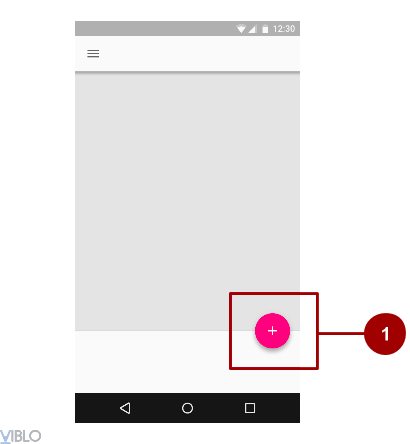
**2.5. Components và Pattern**

Các thành phần Button và nhiều View con sử dụng trong Android theo quy tắc Material Design. Material Design bao gồm các components (thành phần) và patterns (mẫu) để xây dựng nên những màn hình mà người dùng có thể sử dụng các tính năng dễ dàng cho dù họ chưa quen với màn hình hay ứng dụng.

Một số component thường dùng của Material Design

- Floating Action Button

Chúng ta sử dụng floating action buttons (FAB) cho các hành động mà muốn người dùng thực hiện. Một FAB là một icon tròn nổi phía trên UI. Khi focus thay đổi thì nó thay đổi màu sắc một chút và được nâng lên khi được chọn.

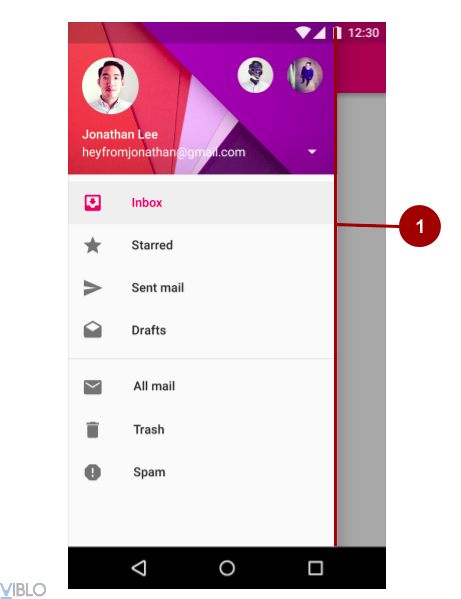


Để sử dụng thì chúng ta có thể khai báo một đoạn mã tương ứng dưới đây vào Activity hay Fragment mà mình muốn:

<com.google.android.material.floatingactionbutton.FloatingActionButton  
android:id="@+id/addNewItemFAB"  
android:layout\_width="150dp"  
android:layout\_height="150dp"  
android:src="@drawable/loop2"  
app:fabSize="normal"  
 android:scaleType="fitCenter"  
app:elevation="30dp" />

- Navigation Drawer

Một Navigation drawers là một bảng điều khiển trượt từ bên trái sang.

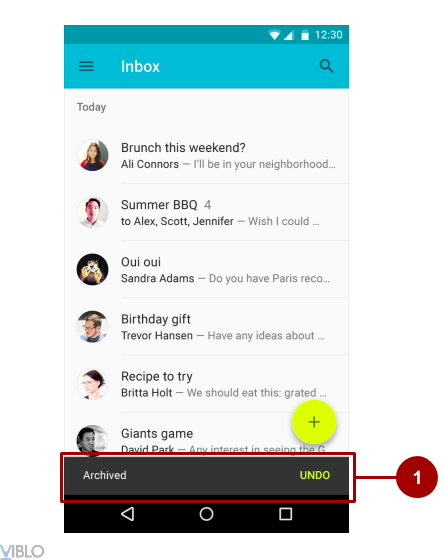


Trong XML, ta có thể dùng DrawerLayout là gốc của layout. Bên trong nó, một cho bố trí chính của layout là drawer và hai là nội dung của ngăn kéo.

<androidx.drawerlayout.widget.DrawerLayout  
xmlns:android="http://schemas.android.com/apk/res/android"  
android:id="@+id/drawer\_layout"  
android:layout\_width="match\_parent"  
android:layout\_height="match\_parent">  
*<!-- The main content view -->*<FrameLayout  
 android:id="@+id/content\_frame"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent" />  
*<!-- The navigation drawer -->*<ListView android:id="@+id/left\_drawer"  
 android:layout\_width="240dp"  
 android:layout\_height="match\_parent"  
 android:layout\_gravity="start"  
 android:choiceMode="singleChoice"  
 android:divider="@android:color/transparent"  
 android:dividerHeight="0dp"  
 android:background="#111"/>  
</androidx.drawerlayout.widget.DrawerLayout>

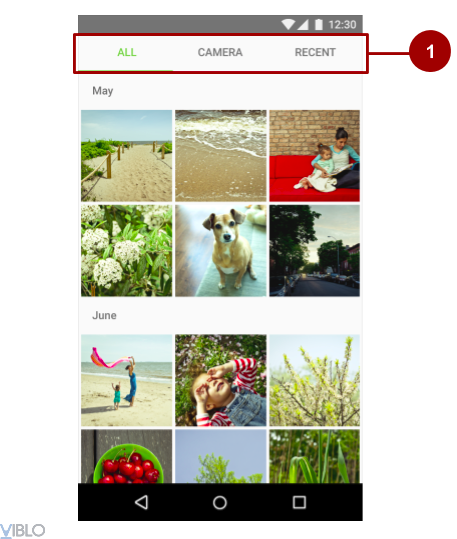
* Snackbar

Một snackbar cung cấp thông tin phản hồi ngắn gọn về một hoạt động thông qua một tin nhắn trong một thanh ngang trên màn hình.



* Tab

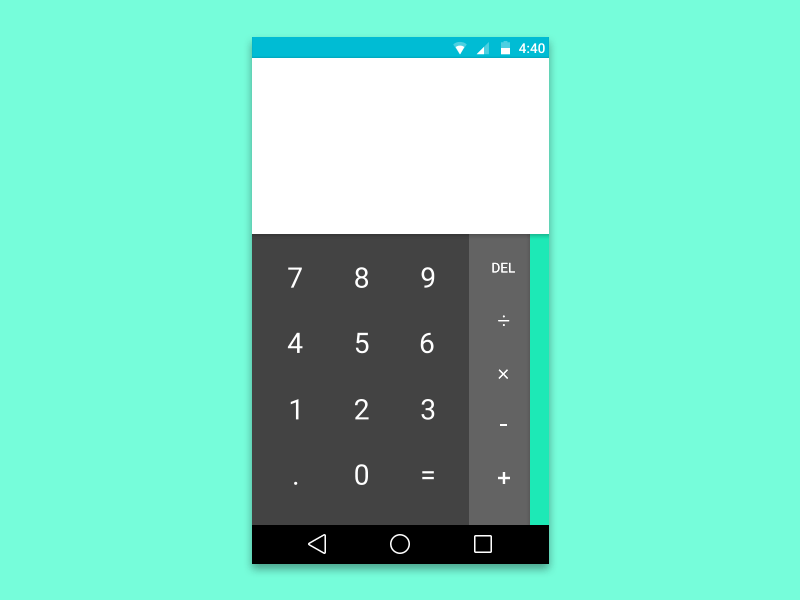
Thường sử dụng để người dùng có thể dùng tay vuốt sang trái hoặc sang phải để chuyển đổi giữa các màn hình.



## F. BÀI TẬP ÔN TẬP

***Lưu ý: Phải áp dụng animation và thiết kế theo các nguyên tắc của Material design***

**Bài 1**: Viết một demo nhỏ thực hiện các hiệu ứng và thiết kế theo Material design như sau:

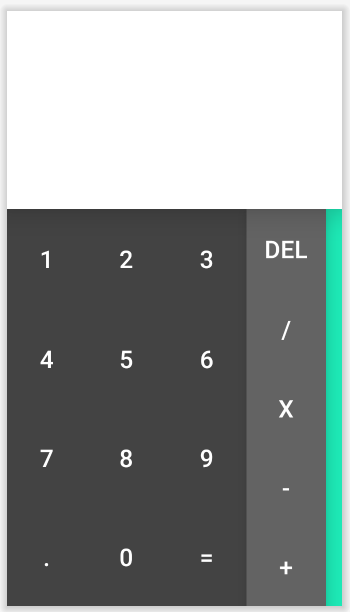


**Hướng dẫn:**

***Bước 1***: Tạo project

***Bước 2***: Xây dựng file XML

Kết hợp LinearLayout, RelativeLayout và FrameLayout thể thiết kế giao diện sau:



***Bước 3***: Ở đây chúng ta thấy có hành động trượt ra của giao diện mở rộng:

Đầu tiên,ta tạo 1 thư mục có tên “*anim*” trong thư mục *res*. Trong thư mục anim chúng ta sẽ thêm 1 file xml. Trong file xml ta sẽ sử dụng thẻ “*translate*” để thực hiện hành động trượt cho một view. Trong thẻ *translate* này ta sẽ cài đặt các thuộc tính như thời gian thực hiện hành động trượt (*android:duration*), vị trí bắt đầu hành động trượt (*android:fromXDelta*), vị trí kết thúc hành động trượt (*android:toXDelta*).

Hành động trượt từ trái sang phải và trượt từ phải sang trái:

**slide\_page\_left\_to\_right.xml (**trượt từ trái sang phải**)**

*<?*xml version="1.0" encoding="utf-8"*?>*<set xmlns:android="http://schemas.android.com/apk/res/android" >  
 <translate  
 android:duration="700"  
 android:fromXDelta="0%"  
 android:toXDelta="100%" />  
</set>

slide\_page\_right\_to\_left.xml (trượt từ phải sang trái)

*<?*xml version="1.0" encoding="utf-8"*?>*<set xmlns:android="http://schemas.android.com/apk/res/android">  
 > <translate  
 android:duration="700"  
 android:fromXDelta="100%"  
 android:toXDelta="0%" />  
</set>

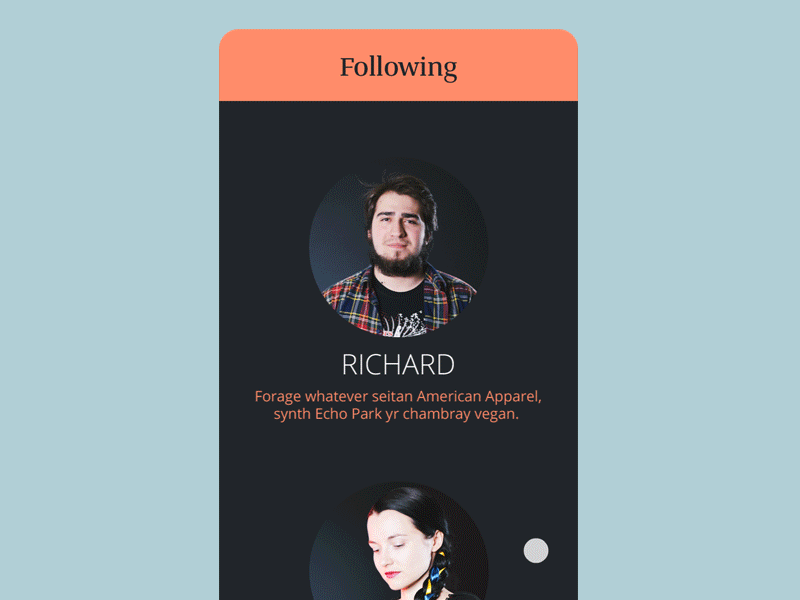
***Bước 4***: Tạo phương thức thực hiện cài đặt animation cho view trong MainActivity

public void animationView(View v, int animId) {  
 Animation anim = AnimationUtils.*loadAnimation*(this, animId);  
 v.startAnimation(anim);  
}

***Bước 5***: Xử lý sự kiện các nút bấm

import android.view.animation.Animation;  
import android.view.animation.AnimationUtils;  
  
*//import*public class MainActivity extends Activity {  
 public StringBuilder textEditText = new StringBuilder();  
 public static float *result* = 0; EditText mTxtShow;  
 RelativeLayout mLinear;  
 @Override  
 protected void onCreate(Bundle savedInstanceState) { super.onCreate(savedInstanceState); setContentView(R.layout.activity\_main); mTxtShow = (EditText) findViewById(R.id.screen);  
 mLinear = (RelativeLayout) findViewById(R.id.linearOption);  
 }  
 public void clickNumber(View v) {  
 switch (v.getId()) {  
 case R.id.number1:  
 textEditText.append("1");  
 mTxtShow.setText(textEditText.toString());  
 break;  
 case R.id.number2:  
 textEditText.append("2");  
 mTxtShow.setText(textEditText.toString());  
 break;  
 case R.id.number3:  
 textEditText.append("3");  
 mTxtShow.setText(textEditText.toString());  
 break;  
 case R.id.number4:  
 textEditText.append("4");  
 mTxtShow.setText(textEditText.toString());  
 break;  
 case R.id.number5:  
 textEditText.append("5");  
 mTxtShow.setText(textEditText.toString());  
 break;  
 case R.id.number6:  
 textEditText.append("6");  
 mTxtShow.setText(textEditText.toString());  
 break;  
 case R.id.number7:  
 textEditText.append("7");  
 mTxtShow.setText(textEditText.toString());  
 break;  
 case R.id.number8:  
 textEditText.append("8");  
 mTxtShow.setText(textEditText.toString());  
 break;  
 case R.id.number9:  
 textEditText.append("9");  
 mTxtShow.setText(textEditText.toString());  
 break;  
 case R.id.number0:  
 textEditText.append("0");  
 mTxtShow.setText(textEditText.toString());  
 break;  
 case R.id.cham:  
 textEditText.append(".");  
 mTxtShow.setText(textEditText.toString());  
 break;  
 case R.id.btnShowMenu:  
 if (mLinear != null) {  
 mLinear.setVisibility(View.VISIBLE);  
 animationView(mLinear,  
 R.anim.righttoleft);  
 }  
 break;  
 case R.id.closeThisView:  
 if (mLinear != null) {  
 mLinear.setVisibility(View.GONE);  
 animationView(mLinear,  
 R.anim.lefttoright);  
 }  
 break;  
 case R.id.s1:  
 textEditText.append("demo");  
 mTxtShow.setText(textEditText.toString());  
 break;  
 case R.id.s2:  
 textEditText.append("vidu");  
 mTxtShow.setText(textEditText.toString());  
 break;  
 case R.id.s3:  
 textEditText.append("test");  
 mTxtShow.setText(textEditText.toString());  
 break;  
 case R.id.s4:  
 textEditText.append("a");  
 mTxtShow.setText(textEditText.toString());  
 break;  
 case R.id.s5:  
 textEditText.append("b");  
 mTxtShow.setText(textEditText.toString());  
 break;  
 case R.id.s6:  
 textEditText.append("c");  
 mTxtShow.setText(textEditText.toString());  
 break;  
 case R.id.s7:  
 textEditText.append("d");  
 mTxtShow.setText(textEditText.toString());  
 break;  
 case R.id.s8:  
 textEditText.append("e");  
 mTxtShow.setText(textEditText.toString());  
 break;  
 case R.id.s9:  
 textEditText.append("f");  
 mTxtShow.setText(textEditText.toString());  
 break;  
 case R.id.s10:  
 textEditText.append("g");  
 mTxtShow.setText(textEditText.toString());  
 break;  
 case R.id.s11:  
 textEditText.append("h");  
 mTxtShow.setText(textEditText.toString());  
 break;  
 case R.id.s12:  
 textEditText.append("i");  
 mTxtShow.setText(textEditText.toString());  
 break;  
 default:  
 break;  
 }  
 }  
 public void animationView(View v, int animId) {  
 Animation anim = AnimationUtils.*loadAnimation*(this, animId);  
 v.startAnimation(anim);  
 }  
}

**Bài 2**: Viết một demo nhỏ thựchiệncác hiệu ứng và thiết kế theoMaterial design như sau:



**Bài 3:** Áp dụng hiểu biết về animation và Material Design để áp dụng và thiết kế app Quản Lý Chi Tiêu, action bar, button, switch, dialog, hiệu ứng hoạt cảnh... cần tuân thủ đúng đặc tả của Material Design

