Animaciones

-Video Introductorio:
https://www.youtube.com/watch?v=cQzien5H2Do

-Ejemplos de animaciones:
https://envato.com/blog/subtly-animated-material-design/

Animaciones

Digital House > Coding School

El SDK nos provee 2 frameworks para manejar animaciones 2D en android segun en la versión de SO que estemos trabajando.

View Animation	Property Animation
Animan el contenido de una vista sin modificar su propiedades. El sdk provee un stock de animaciones, pudiendo crear si lo deseamos animaciones propias.	Animan las propiedades de una vista. Nos permite crear nuestras propias vistas y animar sus propiedades fácilmente.

View Animation

Digital House > Coding School

DEPRECATED

Property Animations

Digital House > Coding School

Este framework nos permite animar casi cualquier atributo de una vista. Para esto utiliza reflection, con lo que solo debemos indicarle el nombre del atributo y tipo de dato a modificar.

Podemos trabajar con una clase:

- ObjectAnimator

Como crear una animación

Digital House > Coding School

1. Primero debemos crearnos un objeto ObjectAnimator, para esto usamos un método estático que varía según el tipo de atributo que queremos modificar(float o int) y le indicamos la vista a la cual aplicar la transformación.

```
// Para variables float
ObjectAnimator animation = ObjectAnimator.ofFloat(view, ...);

// Para variables int
ObjectAnimator animation = ObjectAnimator.ofInt(view, ...);
```

2. Luego tenemos que indicar que atributo queremos modificar. Las vistas ya poseen propiedades básicas con las que podemos trabajar.

Transparencia	ObjectAnimator.ofFloat(view, “**alpha**”, ...);
Rotacion	ObjectAnimator.ofFloat(view, “**rotation**”, ...);
Escala eje x	ObjectAnimator.ofFloat(view, “**scaleX**”, ...);
Escala eje y	ObjectAnimator.ofFloat(view, “**scaleY**”, ...);
Traslación eje x	ObjectAnimator.ofFloat(view, “**translationX**”, ...);
Traslación eje y	ObjectAnimator.ofFloat(view, “**translationY**”, ...);
Posicion eje x	ObjectAnimator.ofFloat(view, “**x**”, ...);
Posicion eje y	ObjectAnimator.ofFloat(view, “**y**”, ...);

3. Luego indicamos los valores que tendrá el atributo que queremos animar al principio y en el final

```
// Animamos la propiedad “alpha” hasta hacer invisible la vista, en este caso se inicia con el valor que ya contenía el atributo.
ObjectAnimator animation = ObjectAnimator.ofFloat(view, “alpha”, 0f);

// Animamos la propiedad “alpha” hasta hacer invisible la vista, en este caso indicamos el valor de inicio del atributo alpha.
ObjectAnimator animation = ObjectAnimator.ofFloat(view, “alpha”, 0.5f, 0f);
```

4. Por último indicaremos el tiempo e iniciaremos la animación.

```
ObjectAnimator animation = ObjectAnimator.ofFloat(view, “alpha”, 0.5f, 0f);
animation.setDuration(1000); // indicamos el tiempo en milisegundos
animation.start();
```

Ejemplo


```
public class PostHoneycombAnimationActivity extends Activity implements View.OnClickListener{
    private Button button;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        ...
        button = (Button) findViewById(R.id.button);
        button.setOnClickListener(this);
    }
    @Override
    public void onClick(View view) {
        Toast.makeText(this, "Click!", Toast.LENGTH_SHORT).show();
        ObjectAnimator animation = ObjectAnimator.ofFloat(button, View.TRANSLATION_X, 300);
        animation.setDuration(1000);
        animation.start();
    }
}
```

¿Cómo saber cuándo terminó una animación?

Digital House > Coding School

Tengo que agregar un listener:

```
ObjectAnimator anim = ObjectAnimator.ofFloat(v, "alpha", 0.2f);
anim.addListener(new AnimatorListenerAdapter() {
    @Override
    public void onAnimationEnd(Animator animation) {
        Toast.makeText(MainActivity.this, "End!",
Toast.LENGTH_SHORT).show();
    }
});
anim.start();
```

Concatenar animaciones

Digital House > Coding School

1) Para poder concatenar varias animaciones primero tenemos que definirlas.

```
ObjectAnimator expandirAncho = ObjectAnimator.ofFloat(view, "scaleX", 1.0f, 2.0f)
        .setDuration(2000);
ObjectAnimator expandirAlto = ObjectAnimator.ofFloat(view, "scaleY", 1.0f, 2.0f)
        .setDuration(2000);
```

2) Luego agregarlas a un set de animaciones

```
AnimatorSet set = new AnimatorSet();
set.playTogether(expandirAlto,expandirAncho);
set.start();
```

3) También podemos concatenar sets de animaciones

```
AnimatorSet setDeAnimacion1 = new AnimatorSet();
setDeAnimacion1.playTogether(expandirAlto,expandirAncho);

AnimatorSet setDeAnimacion2 = new AnimatorSet();
setDeAnimacion2.playTogether(expandirAlto,expandirAncho);

AnimatorSet setDeAnimacionCompleta = new AnimatorSet();
setDeAnimacionCompleta.playTogether(setDeAnimacion1,setDeAnimacion2);
setDeAnimacionCompleta.start();
```

XML Animations

Digital House > Coding School

Android nos da la posibilidad de guardar y reusar las animaciones para esto hay que:

1) Definirlas en XML
(res/animator/fade_out.xml)

```
<objectAnimator
xmlns:android="http://schemas.android.com/apk/res
/android"
    android:propertyName="alpha"
    android:duration="1000"
    android:valueTo="0" />
```

2) La instanciamos y se la asignamos al objeto que queremos animar.

```
Animator anim = AnimatorInflater.loadAnimator(this,
R.animator.fade_out);
anim.setTarget(btnExample);
anim.setDuration(1000);
anim.setStartDelay(10);

anim.start();
```

XML set Animations

Digital House > Coding School

```
<set
xmlns:android="http://schemas.android.
com/apk/res/android"
    android:ordering="together" >
    <objectAnimator
        android:propertyName="alpha"
        android:valueTo="0.5" >
    </objectAnimator>
    <objectAnimator

android:propertyName="translationY"
        android:valueTo="100.0" >
    </objectAnimator>
</set>
```

Transition Animations

Digital House > Coding School

Para crear animaciónes de Transiciones hay que declararlas en XML, para eso se crea un Android Resource File de tipo Animation (ojo, animation, no animator), por ejemplo (Fade In):

```
<alpha xmlns:android="http://schemas.android.com/apk/res/android"
   android:interpolator="@android:anim/accelerate_interpolator"
   android:fromAlpha="0.0" android:toAlpha="1.0"
   android:duration="500" />
```

Transition Animations

Digital House > Coding School

1) Activities: Después de hacer el startActivity le sobreescribimos la animación que queremos que haga:

```
Intent i = new Intent(MainActivity.this,
SecondActivity.class);
startActivity(i);
overridePendingTransition(R.anim.in, R.anim.out);
```

Transition Animations

Digital House > Coding School

2) Fragments: En la transacción tenemos que setear la animación que queremos que ocurra antes de hacer el commit.

```
FragmentTransaction fragTrans =
getSupportFragmentManager().beginTransaction();
fragTrans.setCustomAnimations(R.anim.in, R.anim.out);
fragTrans.replace(R.id.fragment_container, newFragment);
fragTrans.commit();
```