ENCS5150 · EXP 5 COMPANION

Animation on Android

Frame and tween animation, why a moved button still clicks in its old place, and what modern apps animate with instead.

Five questions

01 What kinds of animation does Android have?

02 How do frame and tween animation work?

03 Why does a moved button still click in its old place?

04 What should a new app use?

05 What makes motion feel right?

01 · KINDS OF ANIMATION

Three animation systems, from oldest to newest

DRAWABLE · THIS LAB

Change the picture

AnimationDrawable plays frames like a flip book. TransitionDrawable cross-fades two pictures.

The blinking and switching bulbs.

VIEW · THIS LAB

Tween the drawing

Move, rotate, scale or fade how a view is drawn, from XML in res/anim.

The rotating, scaling, touring image.

PROPERTY · MODERN

Animate any property

ValueAnimator, ObjectAnimator and view.animate() change real values over time.

What most Views apps use today.

Above all three sit higher-level tools: layout transitions, MotionLayout, and Compose’s animate*AsState.

02 · FRAME AND TWEEN

You draw every frame, or only the ends

FRAME ANIMATION



An <animation-list> of pictures, each shown for a set time. Anything you can draw, but smooth motion needs many frames, and many frames make a big app.

TWEEN ANIMATION



You give the start and the end; Android calculates every frame in between. Small files, smooth at any frame rate, but only move, turn, scale and fade.

A tween in XML, played from Java

```
<rotate
    android:fromDegrees="0"
    android:toDegrees="360"
    android:pivotX="50%"
    android:pivotY="50%"
    android:duration="1000" />
```

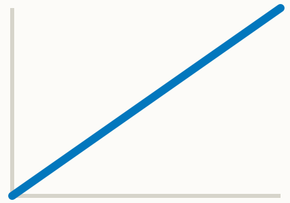
```
Animation spin = AnimationUtils
    .loadAnimation(this, R.anim.rotate);
image.startAnimation(spin);
```

Value	Means
100	100 pixels
50%	Half of the view itself: its centre as a pivot
30%p	30% of the parent’s size

The default pivot is the top-left corner, so a rotate without **50%** swings the view around its corner.

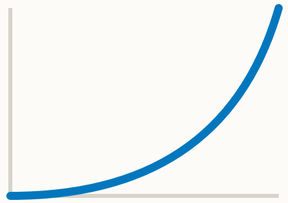
03 · TIMING

An interpolator shapes the speed



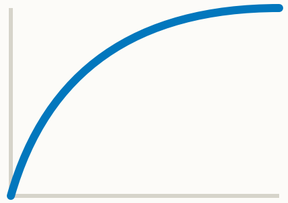
linear

Constant speed.
Spinners.



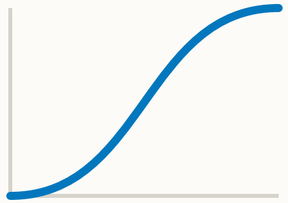
accelerate

Speeds up.
Things leaving.



decelerate

Slows down. Things arriving.



accelerate_ decelerate

The default.
Moving across.



overshoot

Passes the end, settles back. Playful.

```
android:interpolator="@android:anim/decelerate_interpolator"
```

03 · TIMING

A tween moves the drawing, not the view



Where it **is**: taps land here

Where it is **drawn**: taps do nothing

The fix: property animation

It changes the view’s real translationX, so the button and its touch area move together.

```
button.animate().translationX(600f).setDuration(500);
```

04 · WHAT TO USE TODAY

Property animation changes real values

```
// the short way, for common view properties
image.animate()
        .rotation(360f)
        .scaleX(1.5f).scaleY(1.5f)
        .alpha(0.5f)
        .setDuration(800);

// any property with a setter
ObjectAnimator spin =
    ObjectAnimator.ofFloat(image, "rotation", 0f, 360f);
spin.setDuration(1000);
spin.start();
```

No XML needed

Though res/animator XML works too.

Combine and chain

AnimatorSet plays animators together or one after another.

Not only views

ValueAnimator animates any number, such as a counter or a colour.

04 · WHAT TO USE TODAY

Pick the tool for the job

You want to…	Use
Move, fade or turn one view	view.animate()
Play several animations in order	AnimatorSet
Animate a layout as views appear or disappear	android:animateLayoutChanges, or TransitionManager
Morph an icon, such as play into pause	AnimatedVectorDrawable
Choreograph a whole screen, driven by a swipe	MotionLayout
Swap pictures, such as a blinking bulb	AnimationDrawable (this lab)
Animate anything in a Compose app	animate*AsState, AnimatedVisibility

05 · MOTION THAT FEELS RIGHT

Four rules for good motion

Keep it short

Most interface motion lasts a few hundred milliseconds. Longer feels slow after the second time.

Ease it

Decelerate what arrives, accelerate what leaves. Linear looks mechanical.

Give it a reason

Show where something came from or went. Decoration that repeats gets annoying.

Respect the setting

Users can turn animations off in Developer options or Accessibility. The app must still work, just without motion.

TAKEAWAYS

Four things to remember

01 Frame animation swaps pictures; tween animation calculates the frames between two ends.

02 Set the pivot to 50% unless you want to turn around a corner.

03 Tweens move the drawing only; view.animate() moves the view itself.

04 Short, eased, purposeful, and optional for users who turn motion off.