Android Animation

droid girls #11 / Apr 20 / @yuichi_araki

Android Animation

今日話すこと

・アニメーションの前提知識
・アニメーションの基本機能
・Transition API
・Shared Element Transition
・おまけ: Dynamic Animation

アニメーションの前に

アニメーションの前に

Android の View の表示位置を決める要素

大きくわけて
- レイアウト
- 変形 (Transformation)

レイアウト

型は Int

指定する
- ViewGroup.LayoutParams / android:layout_なんとか

計算する
- ViewGroup.onLayout: 子 View を配置する

取得する
- View.getLeft/Top/Right/Bottom()

変形 (Transformation)

型は Float

- 平行移動: View.setTranslationX/Y()
- 回転: setRotation(), setPivotX/Y()
- 拡大縮小: setScaleX/Y()

取得する
- View.getX/Y()

レイアウトと変形の大切な違い

レイアウト

- 重い (60FPS ムリ)

変形

- 軽い (60FPS 余裕)

アニメーション中にレイアウトを起こしてはいけない

レイアウトを起こす動作

- View.requestLayout()
- View.setLayoutParams()
- View.setPadding()

- View の大きさが変わる操作一般

など多数

様々なアニメーション

様々なアニメーション

- Animation
- Animator (ValueAnimator, ObjectAnimator, TimeAnimator)
- ViewPropertyAnimator (View.animate)
- DynamicAnimation
- LayoutTransition (android:animateLayoutChanges)
- Transition
  - Shared Element Transition

基本的なアニメーション

- Animation
- Animator
- ViewPropertyAnimator

duration: 持続時間
interpolator: カーブ
リスナー

Animation

android.view.animation.Animation
忘れよう

ViewSwitcher, TextSwitcher
Activity.overridePendingTransition(),
Fragment.setCustomAnimations()

Animator

android.animation.Animator

ValueAnimator: 値をアニメーションする
ObjectAnimator: 値をアニメーションし、対象にセットする
TimeAnimator: 時間経過

ViewPropertyAnimator

View.animate()
- alpha(), translationX/Y(), scaleX/Y(), rotation()

Animator ではない

簡単なアニメーションには便利だが、応用度が低い

DynamicAnimation

android.support.animation.DynamicAnimation

物理法則に基づいたアニメーション

- FlingAnimation
- SpringAnimation

LayoutTransition

android.animation.LayoutTransition

android:animatedLayoutChanges="true"

addView(), removeView(), setVisibility()
などをアニメーション

Transition

画面の状態の差を何でもアニメーション

Animator に基づいた高度 API

アニメーション機能まとめ

ユーザーのドラッグ・スワイプに従ってアニメーション
→ DynamicAnimation

画面遷移をアニメーション
→ Transition
　カスタムの Transition が必要？
　→ ObjectAnimator

ObjectAnimator

ObjectAnimator

値をアニメーションし、対象に値をセットする

```
val box = view.findViewById<View>(R.id.box)

val animator = ObjectAnimator
  .ofFloat(box, View.TRANSLATION_X, 0f, 100f)
animator.start()
```

Transition の基本

Transition の基本

画面の状態の変化をアニメーション

- 通常の Transition
  - Activity/Fragment Transition
    - Shared Element Transition

```
val box: View = findViewById(...)
val container: ViewGroup = findViewById(...)

TransitionManager
  .beginDelayedTransition(container, Fade())
box.visibility = View.INVISIBLE
```

Transition の仕組み

ある ViewGroup 内の View それぞれに対して

1. 変化前の状態を記録する
2. 変化後の状態を記録する
3. 2つの状態の間をつなぐ Animator を生成する

→ Animator をまとめて実行

様々な Transition

様々な Transition

ChangeBounds, ChangeTransform

Fade, Slide, Explode

ChangeClipBounds

ChangeScroll

ChangeImageTransform

TransitionSet

AutoTransition

ChangeBounds, ChangeTransform

ChangeBounds
View のレイアウト位置
※ アニメーション自体は変形 (Transformation) で行う

ChangeTransform
View の変形 (Transformation) 位置

Fade, Slide, Explode

View の存在 / Visibility の変更

Fade: フェードイン・フェードアウト
Slide: スライドイン・スライドアウト
Explode: 中心点と画面外の移動

その他

ChangeClipBounds
clipBounds

ChangeScroll
スクロール位置

ChangeImageTransform
ImageView の scaleType

TransitionSet, AutoTransition

TransitionSet
複数の Transition をまとめて、同時または順番に実行

AutoTransition
Fade(OUT) → ChangeBounds → Fade(IN)

Transition を組み合わせる

カスタム Transition

Shared Element Transition