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
  - Fragment Transition
  - Shared Element Fragment Transition

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

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

計算する
- ViewGroup.onMeasure: 子 View の大きさを測る
- ViewGroup.onLayout: 子 View を配置する

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

大切なこと

レイアウトは遅い (60 FPS に間に合わない)

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

offsetLeftAndRight/TopAndBottom

ViewCompat.offsetLeftAndRight(View, int)
ViewCompat.offsetTopAndBottom(View, int)

レイアウトを起こさずに、レイアウト位置をずらす

変形 (Transformation)

レイアウトの位置を基準にして変形する

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

取得する
- View.getX/Y()

大切なこと

アニメーション中にレイアウトを起こしては行けない

View を動かしたいときは offset〜 または 変形を使う

様々なアニメーション

様々なアニメーション

- Animation
- Animator
- ViewPropertyAnimator
- DynamicAnimation
- LayoutTransition
- Transition
  - Shared Element Transition

基本的なアニメーション

- Animation
- Animator
- ViewPropertyAnimator

duration: 持続時間
interpolator
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
AnimatorSet: 複数の Animator をまとめる
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
ViewPropertyAnimator でできることは
ObjectAnimator でもできる

DynamicAnimation

android.support.animation.DynamicAnimation

物理法則に基づいたアニメーション

- FlingAnimation
- SpringAnimation

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

LayoutTransition

android.animation.LayoutTransition

android:animateLayoutChanges="true"

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

Transition ではない
LayoutTransition でできることは Transition でもできる

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

値: Int, Float, etc
対象: 主に View
セットする: android.util.Property#set(), get()

```
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

- 普通の Transition
- Fragment Transition
  - Shared Element Transition
- Activity 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

```
TM.beginDelayedTransition(container, Fade())
if (box.visibility == View.VISIBLE) {
  box.visibility = View.INVISIBLE
} else {
  box.visibility = View.VISIBLE
}
```

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・大きさ

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)

```
val transition = TransitionSet().apply {
  duration = 300
  ordering = TransitionSet.ORDERING_TOGETHER
  addTransition(Explode().apply {
    interpolator = FastOutLinearInInterpolator()
  })
  addTransition(ChangeBounds()).apply {
    interpolator = FastOutSlowInInterpolator()
  }
}
```

その他の Transition 機能

その他の Transition 機能

- Transition Group
- Transition Name
- Target add/exclude
- PathMotion
- Propagation
- Epicenter
- Match Order

Transition Group

ViewGroupCompat.setTransitionGroup(container, true)

指定した ViewGroup を、ひとかたまりにする

中の View 一つ一つに Transition をかけるのではなく
全体でまとめてかける

Transition Name

ViewCompat.setTransitionName(view, "abc")

View に Transition 用の名前をつける
使いみちは後述

addTarget/excludeTarget

Transition#addTarget()/excludeTarget()

Transition の対象を
- View のインスタンス
- View の ID
- View のクラス
- View の Transition Name

でフィルタリング

PathMotion

Transition#setPathMotion()

二次元的な動きを Path に従ってアニメーション

ChangeBounds や ChangeTransform 用

Propagation

Transition#setPropagation()

その Transition の対象が複数あるとき
アニメーションの開始を少しずつずらす

Explode や Slide ではデフォルトで有効

Epicenter

Transition#setEpicenterCallback()

Transition の中心点を指定する
基本的にはタッチした場所
Explode や Slide ではデフォルトで有効

Match Order

変化前と変化後の View のマッチングをどう行うか

- View のインスタンス
- View の ID
- View の Transition name
- View のアイテム ID (ListView 用)

カスタム Transition

カスタム Transition

Transition を継承して実装する

実装するメソッド

- captureStartValues()
- captureEndValues()
- createAnimator()

captureStart/EndValues()

この Transition に必要な値を View から抜き出して TransitionValues に入れる

captureStartValues: 遷移前の状態を記録する
captureEndValues: 遷移後の状態を記録する

beginDelayedTransition で指定された ViewGroup 以下のすべての View それぞれに対して呼び出される

createAnimator()

TransitionValues に入れておいた値をもとに
Animator を作って返す

すべての View それぞれに対して呼び出される
関係のない View については null を返せばいい

Fragment Transition

Fragment Transition

Fragment の遷移で Transition を使う

- Fragment Transition
  共通しない要素に対して
  主に Fade, Slide, Explode など
- Shared Element Fragment Transition
  前後の Fragment 間で共通の要素に対して
  主に ChangeBounds, ChangeTransform など

Fragment Transition

Fragment#setExitTransition()
Fragment#setEnterTransition()
Fragment#setReturnTransition()
Fragment#setReenterTransition()

元 Fragment [exit] -------> 先 Fragment [enter]
元 Fragment [reenter] <---- 先 Fragment [return]

```
class ListFragment : Fragment() {
  override fun onCreate(state: Bundle?) {
    super.onCreate(state)
    exitTransition = Slide(GravityCompat.START)
  }
}
```

```
class DetailFragment : Fragment() {
  override fun onCreate(state: Bundle?) {
    super.onCreate(state)
    enterTransition = Fade()
  }
}
```

Shared Element Fragment Transition

Fragment#setSharedElementEnterTransition()
Fragment#setSharedElementReturnTransition()

遷移先の Fragment で指定

```
class DetailFragment : Fragment() {
  override fun onCreate(state: Bundle?) {
    super.onCreate(state)
    sharedElementEnterTransition = TransitionSet()
      .apply {
        addTransition(ChangeBounds())
        addTransition(ChangeTransform())
      }
  }
}
```

共通要素の指定

遷移元 遷移先 両方の View に Transition name が必要
画面内でユニークでなければならない

```
val view: View = ...
supportFragmentManager.beginTransaction()
    .replace(R.id.container, detailFragment)
    .addToBackStack(null)
    // 遷移元の View, 遷移先の Transition name
    .addSharedElement(view, "target")
     // ※
    .setReorderingAllowed(true)
    .commit()
```

Transition の延期

遷移元・遷移先 両方の Fragment

onViewCreated で
  延期: postponeEnterTransition()
データ読み込み・レイアウトが終わったら
  再開: startPostponedEnterTransition()

setReorderingAllowed(true) が必要

Fragment Transition まとめ

まずは Shared Element を使わない Transition から

ハンズオン

ハンズオン

Fragment Transition のサンプル・コードラボ
github.com/yaraki/CheeseMotion

- cheesemotion-start: 課題 - TODO を埋めていく
- cheesemotion-final: 完成形
- common: データ置き場
- deck: このスライド

普通の Transition

[易] LinearLayout の最初の項目を GONE にして
beginDelayedTransition
[易] beginDelayedTransition の第 2 引数を変えてみる
[中] ある ViewGroup から別の ViewGroup に View を動かす
[難] Expandable な FrameLayout
(layout_height 固定値 ←→ wrap_content 切り替え)

参考

記事

- https://chris.banes.me/2018/02/18/fragmented-transitions/

アプリ

- Tivi: github.com/chrisbanes/tivi
- Topeka github.com/googlesamples/android-topeka
- Plaid: github.com/nickbutcher/plaid

おわり