

Android アニメーション② XML で動きを定義する

Android アプリでは、文字列やレイアウトなど、リソース（ファイル）として XML に定義しました。これと同じく、アニメーションをリソースとして定義することができます。

拡大画像が右下から左上に向かって遠ざかっていく（縮小していく）サンプルを作成してみましょう。

●AnimationXML.java

```
package anim.xml;

import android.app.Activity;
import android.os.Bundle;
import android.view.animation.Animation;
import android.view.animation.AnimationUtils;
import android.widget.ImageView;

public class AnimationXML extends Activity {

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);

        ImageView iView = (ImageView) findViewById(R.id.image);
        iView.setImageResource(R.drawable.icon);

        // res/anim/ animation01.xmlファイルを読み込む
        Animation anim01 = AnimationUtils.loadAnimation(this,
            R.anim.animation01);
        iView.startAnimation(anim01);
    }
}
//class_end
```

●main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical" android:layout_width="fill_parent"
    android:layout_height="fill_parent">

    <TextView android:layout_width="fill_parent"
        android:layout_height="wrap_content" android:text="@string/hello"/>

    <ImageView android:id="@+id/image" android:layout_width="wrap_content"
        android:layout_height="wrap_content"></ImageView>

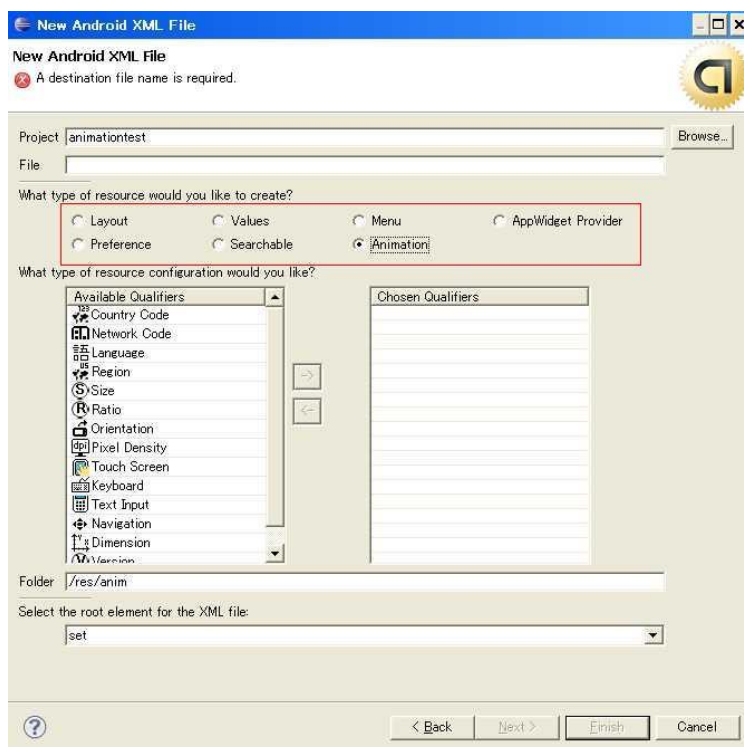
</LinearLayout>
```

Android アプリのアニメーションリソースの追加方法

まずは、アニメーションを定義する為の XML ファイルの追加の手順を説明します。

基本的に文字列やレイアウトの XML の追加方法と同じです。

但し、アニメーションを定義するための XML を追加する場合は以下の画面の赤枠部分のラジオボタンで「Animation」を選択してください。



ファイル名は、何のアニメーションかわかるような適当な名前でも OK です。

今回は、animation01.xml とでもしておきます。

そして、Finish ボタンを押すと、以下のように、Android プロジェクトの res フォルダの直下に「anim」というフォルダが生成され、追加した XML が生成されています。

これで XML ファイルの追加は完了です。

●animation01.xml

```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android"
    android:shareInterpolator="false">
    <scale
        android:fromXScale="10"
        android:toXScale="1"
        android:fromYScale="10"
        android:toYScale="1"
        android:pivotX="0%"
        android:pivotY="0%"
        android:duration="4000"/>
</set>
```

解説) 各アニメーションについて説明します。

■alpha

alpha タグで定義した場合、フェードするアニメーションになります。

以下の属性が設定可能です。

fromAlpha : アニメーション表示開始時のアルファ値で、0.0 から 1.0 で設定します。

toAlpha : アニメーション表示終了時のアルファ値で、0.0 から 1.0 で設定します。

なお、0 に近づくにつれて透明となります。

最初の表示は透明で徐々にはっきりと表示させる場合は、fromAlpha を 0、toAlpha を 1 とし、逆に最初のはっきりと表示させ、徐々に透明にする場合は、fromAlpha を 1、toAlpha を 0 とします。

alpha アニメーションの設定例

表示開始時は完全に表示させた状態で、10 秒間で完全に透明にする場合

```
<alpha xmlns:android="http://schemas.android.com/apk/res/android"
android:duration="10000" android:fromAlpha="1.0" android:toAlpha="0.0" />
```

■scale

scale タグで定義した場合、サイズが変わるアニメーションになります。

以下の属性が設定可能です。

fromXScale: アニメーション開始時の X 方向のサイズです。1.0 が元々のサイズを表します。

toXScale: アニメーション終了時の X 方向のサイズです。1.0 が元々のサイズを表します。

fromYScale: アニメーション開始時の Y 方向のサイズです。1.0 が元々のサイズを表します。

toYScale: アニメーション終了時の Y 方向のサイズです。1.0 が元々のサイズを表します。

pivotX: X 方向の変化の目標となる X 座標

pivotY: Y 方向の変化の目標となる Y 座標

pivotX、pivotY に関して、個人的にわかりやすいかなと思った表現にしました。

例えば、pivotX を「0」と指定したら画像の左端に向かって拡大・縮小します。

「100%」と指定したら、画像の右端に向かって拡大・縮小します。

pivotY は「0」と指定したら画像の上端に向かって拡大・縮小します。

「100%」と指定したら、画像の下端に向かって拡大・縮小します。

「%」を付ければ、その画像を表示させている View の相対座標を表し、「%」を付けなければ単位がよくわかりませんが、その View のエリアをはみ出してでも向かっていく感じです。

scale アニメーションの設定例

表示開始時は 100%のサイズで表示させた状態で、画像の左上端に向かって縮小しながら、10 秒間で 0%にする場合

```
<scale xmlns:android="http://schemas.android.com/apk/res/android"
android:duration="10000" android:fromXScale="1.0" android:toXScale="0.0"
android:fromYScale="1.0" android:toYScale="0.0" android:pivotX="0"
android:pivotY="0" />
```

pivotX、pivotYについては、ちょっと言葉では上手く伝わってないかもしれませんので、実際に試してみて、その Android アプリに適した数値を指定すればよいと思います。

■translate

translate タグで定義した場合、水平方向、垂直方向へ動くアニメーションになります。設定によっては斜めに動かすこともできます。

以下の属性が設定可能です。

fromXDelta : アニメーション開始時の X 方向の位置を示します。「-100%」から「100%」の範囲で指定します。

toXDelta : アニメーション終了時の X 方向の位置を示します。「-100%」から「100%」の範囲で指定します。

fromYDelta : アニメーション開始時の Y 方向の位置を示します。「-100%」から「100%」の範囲で指定します。

toYDelta : アニメーション終了時の Y 方向の位置を示します。「-100%」から「100%」の範囲で指定します。

単位には「%」か「%p」を指定します。

「%」は、そのアニメーションとなる View からみた相対位置

「%p」は、その親の View からみた相対位置となります。

translate アニメーションの設定例

画面の左端から右端へ 5 秒間掛けて移動し、次は右端から左端へ、その次はまた左から右へと動き回るアニメーション

```
<translate xmlns:android="http://schemas.android.com/apk/res/android"
  android:duration="5000" android:fromXDelta="0%p" android:toXDelta="100%p"
  android:fromYDelta="0%p" android:toYDelta="0%p"
  android:repeatMode="reverse" android:repeatCount="-1" >
</translate>
```

■rotate

rotate タグで定義した場合、回転アニメーションになります。

以下の属性が設定可能です。

fromDegrees : アニメーション開始時の角度です。

toDegrees : アニメーション終了時の角度です。

pivotX : 回転の中心点となる X 座標です。

pivotY : 回転の中心点となる Y 座標です。

この pivotX、pivotY も前述の scale と同じような設定の仕方です。

rotate アニメーションの設定例

画像の中心位置を中心に、5 秒間掛けて、1 回転を行い、その動作を繰り返すアニメーション

```
<rotate xmlns:android="http://schemas.android.com/apk/res/android"
    android:duration="5000" android:fromDegrees="0" android:toDegrees="360"
    android:pivotX="50%" android:pivotY="50%" android:repeatMode="restart"
    android:repeatCount="-1" >
</rotate>
```

■set

set は、上記で挙げた複数のアニメーションを組み合わせるために使用します。

これまで紹介した例を全て纏める事ができます。

たとえば、以下を満たすアニメーションを設定してみます。

- 画像の中心位置を中心に 1 回転をする
- 画面の左端から右端へ移動する。(ティッカーみたいな感じ。)
- アニメーション開始時は、完全に透過している状態で、徐々にはっきりと。
- アニメーション開始から終了に掛けて、徐々に大きく。

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

<rotate
  android:duration=" 5000"    android:fromDegrees=" 0"    android:toDegrees=" 360"
  android:pivotX=" 50%"    android:pivotY=" 50%"    android:repeatMode=" restart"
  android:repeatCount=" -1" />

<translate
  android:duration=" 5000"    android:fromXDelta=" 0%p"    android:toXDelta=" 100%p"
  android:fromYDelta=" 0%p"    android:toYDelta=" 0%p"
  android:repeatMode=" restart"    android:repeatCount=" -1" />

<scale
  android:duration=" 5000"    android:fromXScale=" 0"    android:toXScale=" 1"
  android:fromYScale=" 0"    android:toYScale=" 1"    android:pivotX=" 0"
  android:pivotY=" 0"    android:repeatMode=" restart"    android:repeatCount=" -1" />

<alpha
  android:duration=" 5000"    android:fromAlpha=" 0"    android:toAlpha=" 1"
  android:repeatMode=" restart"    android:repeatCount=" -1" />

</set>
```
