



**FPT POLYTECHNIC**

LẬP TRÌNH ANDROID NÂNG CAO

## **Bài 5: Animation**

[www.poly.edu.vn](http://www.poly.edu.vn)

## Nội dung bài học

- Property Animation
- View Animation
- Drawable Animation



# Animation

- Android cung cấp một số API để hiển thị animation cho các phần tử UI và vẽ các đối tượng 2D và 3D
- Android framework cung cấp 3 loại hệ thống animation
  - Property Animation
  - View Animation
  - Drawable Animation



# Property Animation

- Giới thiệu trong Android 3.0 (API level 11)
- Được sử dụng nhiều hơn vì linh hoạt hơn và có một số chức năng mới hơn
- Cho phép animation thuộc tính của bất kỳ đối tượng UI nào



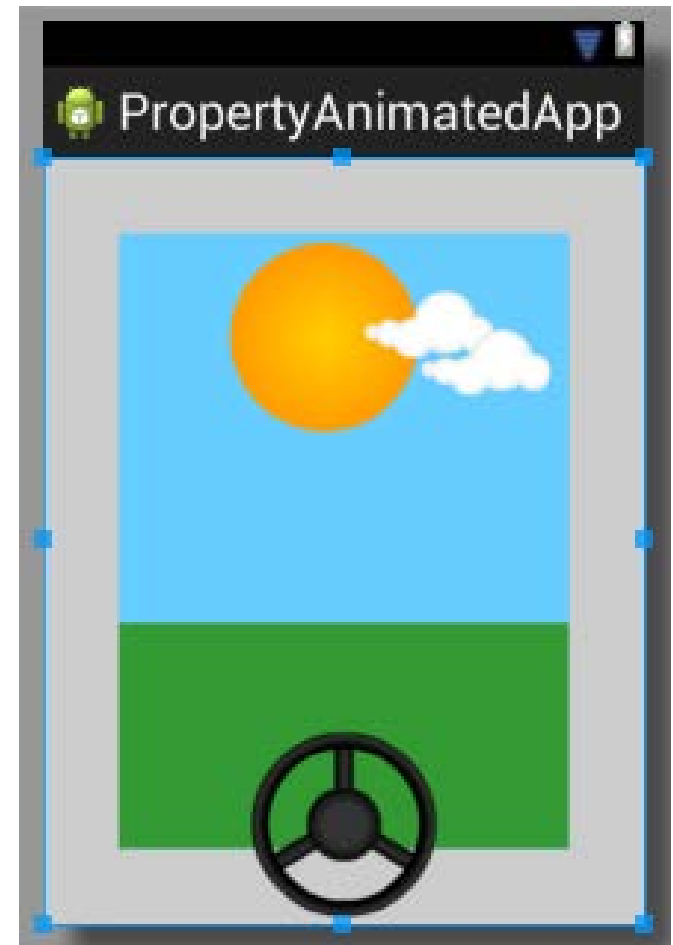
# Property Animation

- Property Animation là framework cho phép tạo hoạt hình từ nhiều nguồn khác nhau
- Có thể định nghĩa hoạt hình để thay đổi đối tượng theo thời gian, dù đối tượng có được vẽ trên màn hình hay không



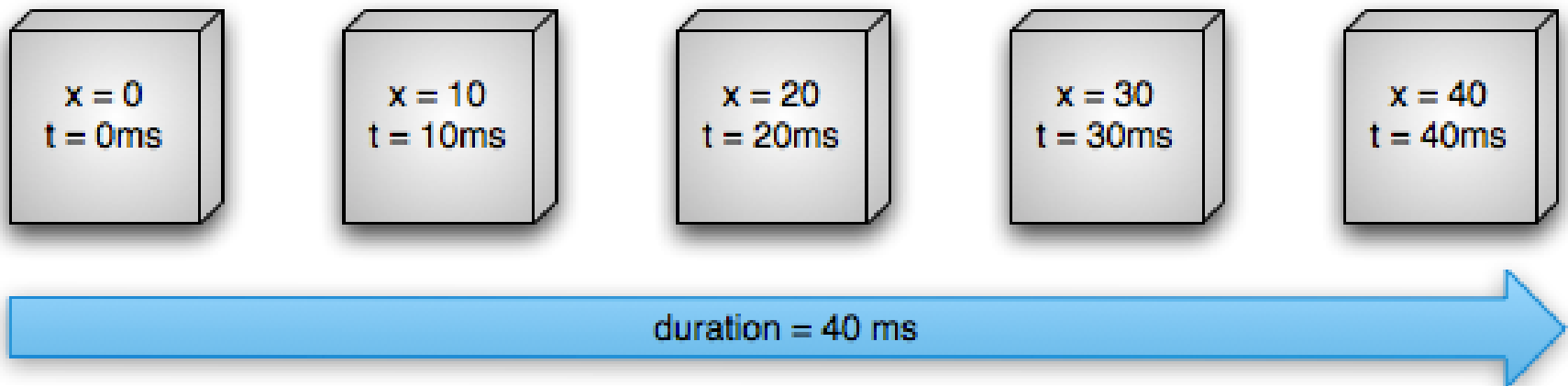
# Property Animation

- Hệ thống property animation giúp bạn định nghĩa các đặc tính của animation:
  - Duration
  - Time interpolation
  - Repeat count and behavior
  - Animator sets
  - Frame refresh delay



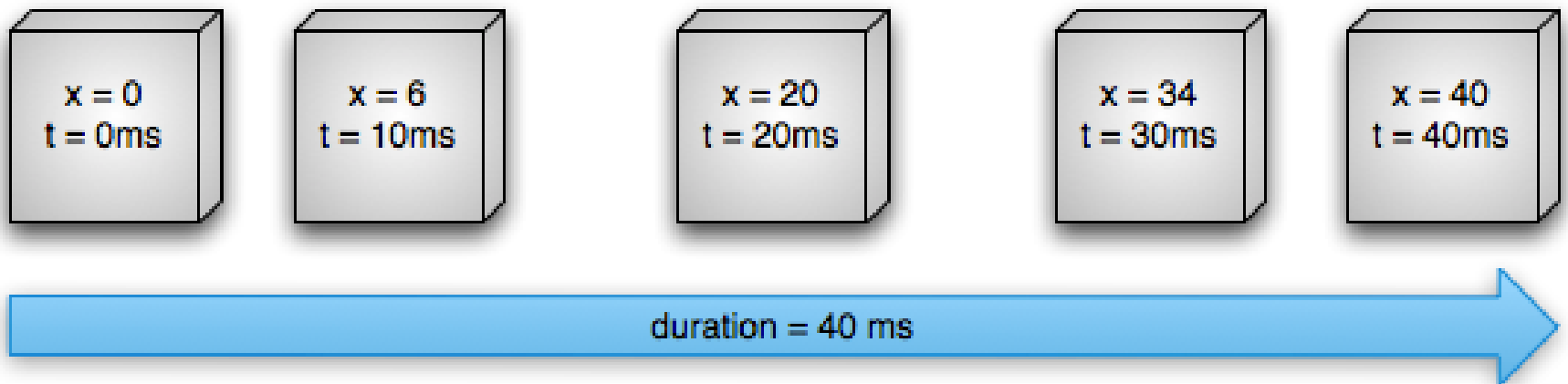
# Nội suy tuyến tính

- Giá trị các thuộc tính biến đổi mềm mại, liên tục trong suốt khoảng thời gian duration được thiết lập



# Nội suy không tuyến tính

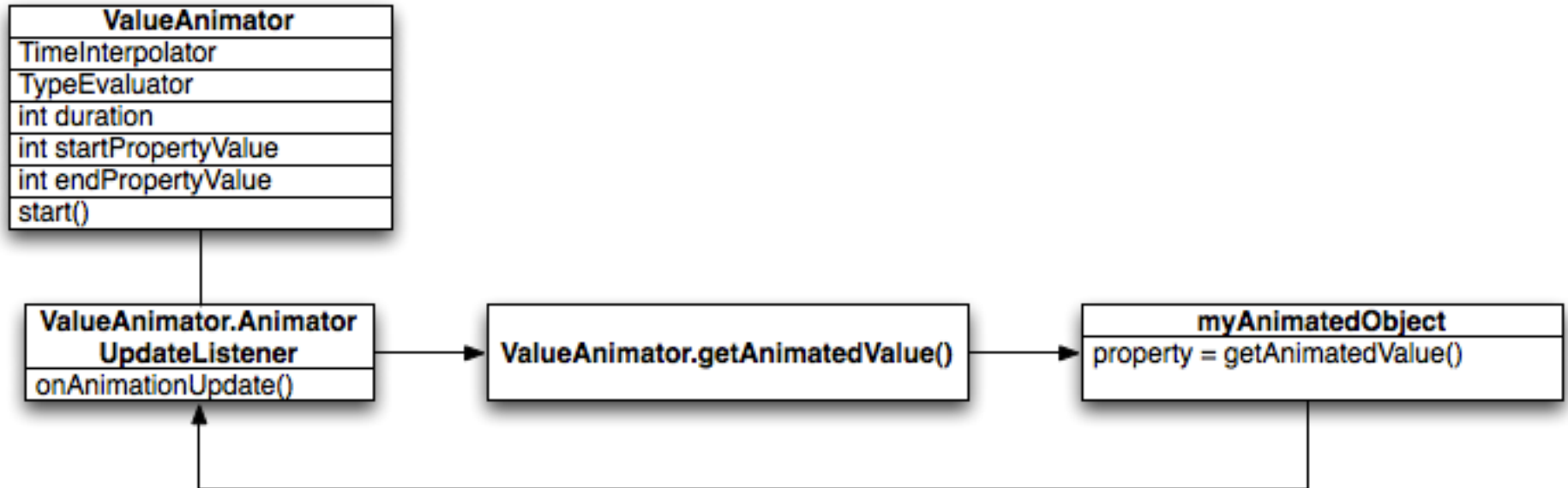
- Giá trị các thuộc tính thay đổi không ổn định theo thời gian





# Property Animation

- Cách hoạt động của các class quan trọng:



# Animate với ObjectAnimator

- ObjectAnimator là lớp con của ValueAnimator
- Giúp cho animate đối tượng bất kỳ dễ dàng hơn và không cần phải thực thi ValueAnimator.AnimatorUpdateListener
- Khởi tạo ObjectAnimator tương tự như ValueAnimator, nhưng phải xác định đối tượng và tên của thuộc tính đối tượng cùng với các giá trị để animate

```
ObjectAnimator anim = ObjectAnimator.ofFloat(foo, "alpha", 0f, 1f);  
anim.setDuration(1000);  
anim.start();
```

# Animation Listener

- **Animator.AnimatorListener**
  - onAnimationStart()
  - onAnimationEnd()
  - onAnimationRepeat()
  - onAnimationCancel()
  - onAnimationEnd()
- **ValueAnimator.AnimatorUpdateListener**
  - onAnimationUpdate()

# Animate ViewGroup

- Hệ thống property animation cung cấp khả năng tạo animation cho các đối tượng ViewGroup
- Thiết lập thuộc tính android: animateLayoutChanges bằng true

```
<LinearLayout  
    android:orientation="vertical"  
    android:layout_width="wrap_content"  
    android:layout_height="match_parent"  
    android:id="@+id/verticalContainer"  
    android:animateLayoutChanges="true" />
```

# Xác định keyframe

- Đối tượng keyframe bao gồm cặp gồm thời gian và giá trị để định nghĩa trạng thái xác định tại thời gian xác định của animation

```
Keyframe kf0 = Keyframe.ofFloat(0f, 0f);
Keyframe kf1 = Keyframe.ofFloat(.5f, 360f);
Keyframe kf2 = Keyframe.ofFloat(1f, 0f);
PropertyValuesHolder pvhRotation = PropertyValuesHolder.ofKeyframe("rotation", kf0, kf1);
ObjectAnimator rotationAnim = ObjectAnimator.ofPropertyValuesHolder(target, pvhRotation);
rotationAnim.setDuration(5000ms);
```

# Animate ViewGroup

- Hệ thống property animation cung cấp khả năng tạo animation cho các đối tượng ViewGroup
- Thiết lập thuộc tính android:animateLayoutChanges bằng true

```
<LinearLayout  
    android:orientation="vertical"  
    android:layout_width="wrap_content"  
    android:layout_height="match_parent"  
    android:id="@+id/verticalContainer"  
    android:animateLayoutChanges="true" />
```

# Khai báo animation trong XML

- Hệ thống property animation cho phép khai báo property animation trong XML
- Bằng cách định nghĩa animation trong XML, bạn có thể dễ dàng sử dụng animation trong nhiều activity khác nhau và có thể thay đổi trật tự của animation
- Các lớp property animation có các XML tag tương ứng như sau:
  - ValueAnimator - <animator>
  - ObjectAnimation - <objectAnimator>
  - AnimatorSet - <set>

# Khai báo animation trong XML

```
<set android:ordering="sequentially">
  <set>
    <objectAnimator
      android:propertyName="x"
      android:duration="500"
      android:valueTo="400"
      android:valueType="intType" />
    <objectAnimator
      android:propertyName="y"
      android:duration="500"
      android:valueTo="300"
      android:valueType="intType" />
  </set>
  <objectAnimator
    android:propertyName="alpha"
    android:duration="500"
    android:valueTo="1f" />
</set>
```



# View Animation

- Được các hệ thống cũ sử dụng và chỉ có thể sử dụng cho Views
- Có thể đáp ứng được các yêu cầu về animation của ứng dụng



# View Animation

```
<set android:shareInterpolator="false">
  <scale
    android:interpolator="@android:anim/accelerate_decelerate_interpolator"
    android:fromXScale="1.0"
    android:toXScale="1.4"
    android:fromYScale="1.0"
    android:toYScale="0.6"
    android:pivotX="50%"
    android:pivotY="50%"
    android:fillAfter="false"
    android:duration="700" />
  <set android:interpolator="@android:anim/decelerate_interpolator">
    <scale
      android:fromXScale="1.4"
      android:toXScale="0.0"
      android:fromYScale="0.6"
      android:toYScale="0.0"
      android:pivotX="50%"
      android:pivotY="50%"
      android:startOffset="700"
      android:duration="400"
      android:fillBefore="false" />
    <rotate
      android:fromDegrees="0"
      android:toDegrees="-45"
      android:toYScale="0.0"
      android:pivotX="50%"
      android:pivotY="50%"
      android:startOffset="700"
      android:duration="400" />
  </set>
</set>
```



# DEMO

Sử dụng View Animation



# Drawable Animation

- Liên quan đến hiển thị tài nguyên Drawable, sẽ tạo animation dưới dạng các frame cho các ảnh trong Drawable, giống như một cuộn phim
- File XML liệt kê danh sách frame tạo nên animation nằm trong thư mục **res/drawable**
- Có thể định nghĩa animation trong code, sử dụng lớp **AnimationDrawable**

```
<animation-list xmlns:android="http://schemas.android.com/apk/res/android"
    android:oneshot="true">
    <item android:drawable="@drawable/rocket_thrust1" android:duration="200" />
    <item android:drawable="@drawable/rocket_thrust2" android:duration="200" />
    <item android:drawable="@drawable/rocket_thrust3" android:duration="200" />
</animation-list>
```

# Drawable Animation

- Nếu thiết lập thuộc tính android:oneshot là true, animation sẽ được chạy một lần duy nhất và dừng lại ở frame cuối cùng. Nếu thuộc tính này được thiết lập là false, animation sẽ được lập đi lập lại

```
<animation-list xmlns:android="http://schemas.android.com/apk/res/android"
    android:oneshot="true">
    <item android:drawable="@drawable/rocket_thrust1" android:duration="200" />
    <item android:drawable="@drawable/rocket_thrust2" android:duration="200" />
    <item android:drawable="@drawable/rocket_thrust3" android:duration="200" />
</animation-list>
```

# Drawable Animation

- Ví dụ: trong Activity sau, animation được thêm vào một ImageView và sau đó được animation được khởi tạo khi chạm vào màn hình

```
AnimationDrawable rocketAnimation;

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

    ImageView rocketImage = (ImageView) findViewById(R.id.rocket_image);
    rocketImage.setBackgroundResource(R.drawable.rocket_thrust);
    rocketAnimation = (AnimationDrawable) rocketImage.getBackground();
}

public boolean onTouchEvent(MotionEvent event) {
    if (event.getAction() == MotionEvent.ACTION_DOWN) {
        rocketAnimation.start();
        return true;
    }
    return super.onTouchEvent(event);
}
```

# So sánh Property Animation và View Animation

- Bất lợi của view animation là chỉ có thể thay đổi View khi View đã được vẽ
- Ví dụ, khi tạo hoạt hình một button di chuyển xung quanh màn hình, button sẽ được vẽ chính xác, nhưng vị trí button thực tế khi bạn click vào button vẫn không thay đổi
- Với Property Animation, không bị ràng buộc này
- Đối với View Animation, đòi hỏi ít thời gian và ít code hơn



# DEMO

Demo sử dụng Draw animation





# Tổng kết nội dung bài học

- Property Animation
- View Animation
- Drawable Animation

