Lottie

What is Lottie?

Lottie เป็น Library ที่สามารถสร้างภาพ Animation จาก After effect แล้วส่งต่องานให้เป็นไฟล์ JSON และไฟล์รูปแบบอื่นๆ(เช่น .zip .lottie) ให้แก่ทีมพัฒนา เพื่อนำไปใช้งานต่อในแอปพลิเคชันทั้งใน IOS, Android และ React Native

01

Architectural Pattern

Layered Architecture

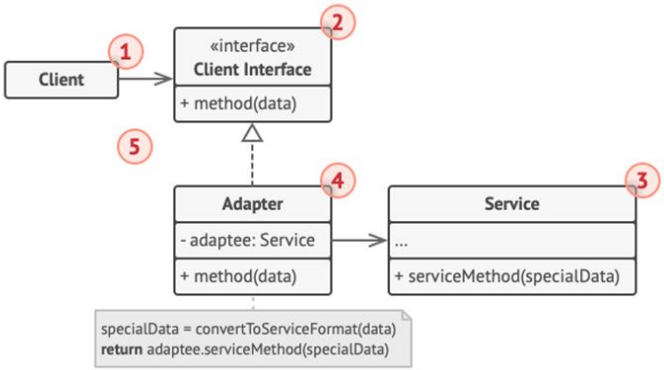
1. **Presentation Layer**
   - Jetpack compose ทำหน้าที่แสดง Animation ออกมาที่ UI
2. **Application Layer**
   - เป็น Java Class ที่ถูกเรียกนำไปใช้ ในการทำ Animation โดย Jetpack compose
3. **Persistence Layer**
   - Java Class จะเรียกใช้ ไฟล์ .JSON ที่เป็น format ของไฟล์ animation
4. **Database Layer**
   - ไฟล์ animation ที่เป็นไฟล์ .JSON

02

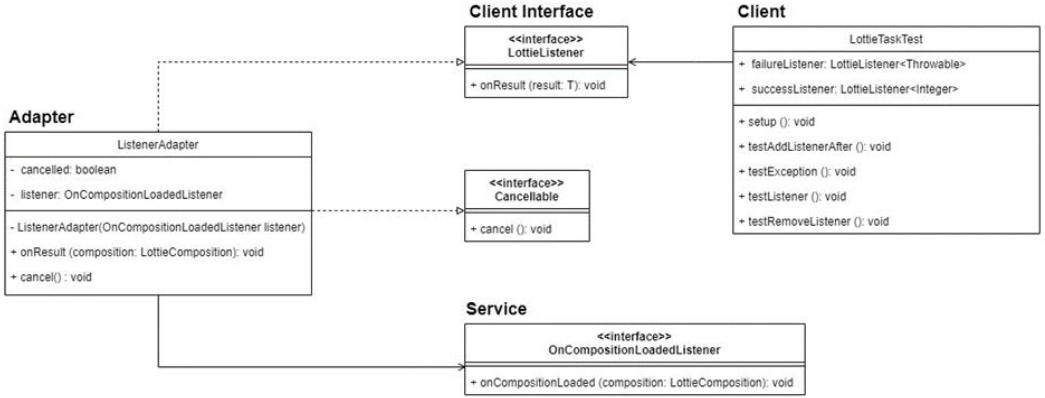
Design patterns

1. Adapter Design Pattern Diagram

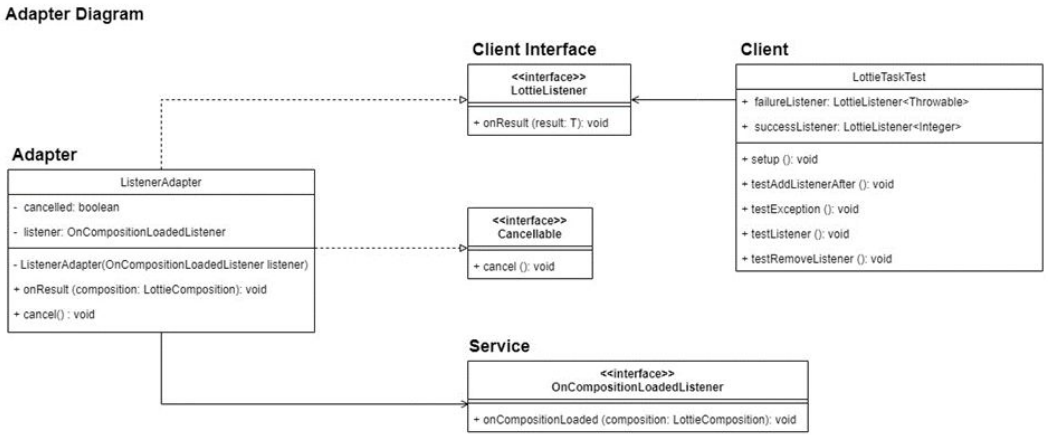
Structure



Adapter Diagram



1.1 Adapter Explanation



เป็นการแปลงจาก OnCompositionLoadedListener เป็น interface แบบเดียวกับ LottieListener<LottieComposition> โดยการใส่ OnCompositionLoadedListener ให้กับ ListenerAdapter ผ่านทาง Constructor ของ ListenerAdapter ซึ่งตรงกับ Adapter Design Pattern ในด้านของการแปลง interface ให้สามารถทำงานร่วมกันได้

1.2 Adapter Evidence

Source files: lottie-android/lottie/src/main/java/com/airbnb/lottie/LottieComposition.java

LottieComposition.java lottie/src/main/java/com/airbnb/lottie

```
360   @SuppressWarnings("deprecation")
361   private static final class ListenerAdapter implements LottieListener<LottieComposition>, Cancellable {
362     private final OnCompositionLoadedListener listener;
363     private boolean cancelled = false;
364
365     private ListenerAdapter(OnCompositionLoadedListener listener) {
366       this.listener = listener;
367     }
368
369     @Override public void onResult(LottieComposition composition) {
370       if (cancelled) {
371         return;
372       }
373       listener.onCompositionLoaded(composition);
374     }
375
376     @Override public void cancel() {
377       cancelled = true;
378     }
379   }
380 }
381 }
```

Line numbers: ListenerAdapter อยู่บรรทัดที่ 361

```
ListenerAdapter listener = new ListenerAdapter();   LottieComposition.java 246
ListenerAdapter listener = new ListenerAdapter();   LottieComposition.java 257
ListenerAdapter listener = new ListenerAdapter();   LottieComposition.java 268
ListenerAdapter listener = new ListenerAdapter();   LottieComposition.java 279
ListenerAdapter listener = new ListenerAdapter();   LottieComposition.java 290
```

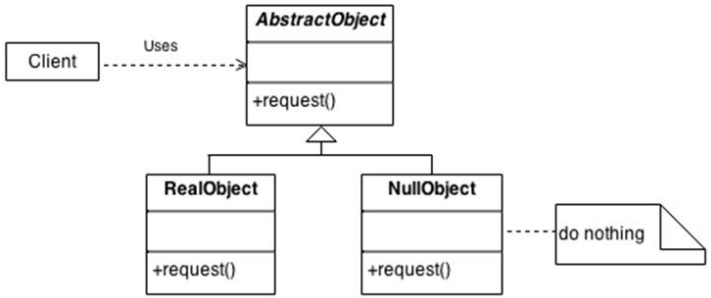
LottieComposition.java lottie/src/main/java/com/airbnb/lottie

```
243   @SuppressWarnings("deprecation")
244   @Deprecated
245   public static Cancellable fromAssetFileName(Context context, String fileName, OnCompositionLoadedListener l) {
246     ListenerAdapter listener = new ListenerAdapter(l);
247     LottieCompositionFactory.fromAsset(context, fileName).addListener(listener);
248     return listener;
249   }
```

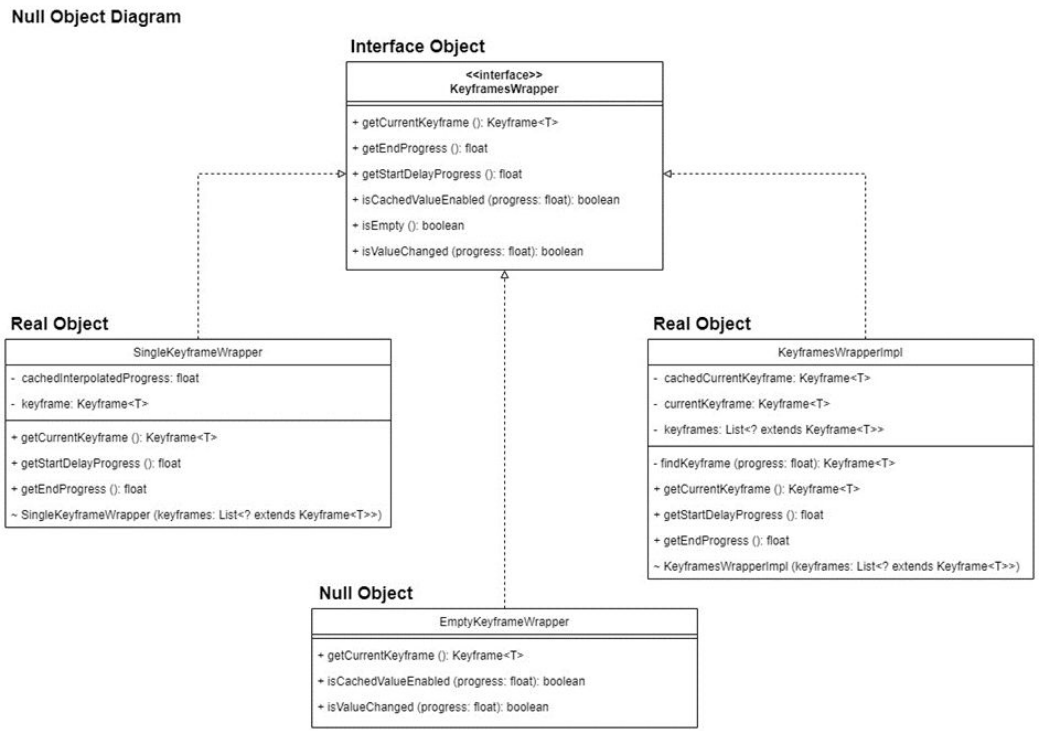
Line numbers: การเรียกใช้ ListenerAdapter
เรียกใช้บรรทัดที่ 246 ใน fromAssetFileName
เรียกใช้บรรทัดที่ 257 ใน fromRawFile
เรียกใช้บรรทัดที่ 268 ใน fromInputStream
เรียกใช้บรรทัดที่ 279 ใน fromJsonString
เรียกใช้บรรทัดที่ 290 ใน fromJsonReader

2. Null Object Design Pattern Diagram

Structure

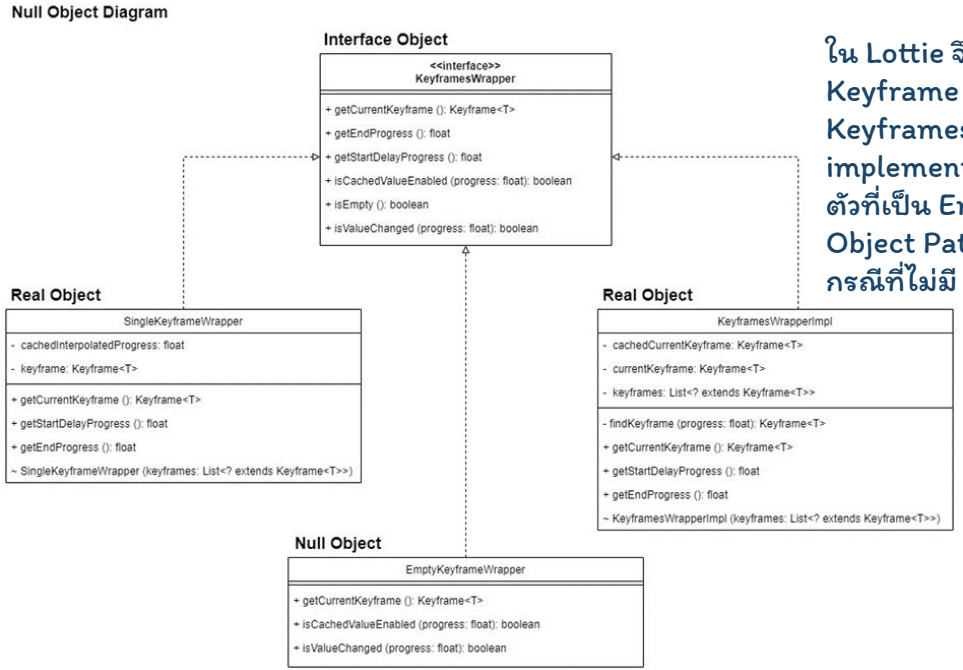


Null Object Diagram



2.1 Null Object Explanation

ใน Animation การทำภาพเคลื่อนไหว เป็นการขยับภาพเป็น frame อย่างต่อเนื่อง การที่จะเก็บข้อมูลทุก frame มาโหลดทั้ง frame ใหม่ตลอด เป็นวิธีที่สิ้นเปลือง และไม่มีประสิทธิภาพ ดังนั้นจึงอาจจะเก็บเต็ม frame แค่บาง frame แล้วที่เหลือก็ใช้การระบุส่วนที่เปลี่ยนแปลงจาก frame แบบเต็ม ๆ ซึ่ง frame ตัวที่เก็บแบบเต็ม ๆ ก็คือ Keyframe โดยที่ในแต่ละ Animation จะมีจำนวนไม่เท่ากัน



ใน Lottie จึงใช้ Null Object Design Pattern ในการตรวจเช็ค Keyframe โดยทั้ง SingleKeyframeWrapper, KeyframesWrapperImpi, และ EmptyKeyframeWrapper นั้น implement มาจาก interface เดียวกันคือ KeyframesWrapper ตัวที่เป็น EmptyKeyframeWrapper จะตรงตามแนวคิด Null Object Pattern คือการส่ง Null Object แทน Real Object ใน กรณีที่ไม่มี Keyframe

2.2 Null Object Evidence

Source files: lottie-android/lottie/src/main/java/com/airbnb/lottie/animation/keyframe/BaseKeyframeAnimation.java

BaseKeyframeAnimation.java lottie/src/main/java/com/airbnb/lottie/animation/keyframe

```
176   private static <T> KeyframesWrapper<T> wrap(List<? extends Keyframe<T>> keyframes) {
177     if (keyframes.isEmpty()) {
178       return new EmptyKeyframeWrapper<>();
179     }
180     if (keyframes.size() == 1) {
181       return new SingleKeyframeWrapper<>(keyframes);
182     }
183     return new KeyframesWrapperImpl<>(keyframes);
184   }
```

BaseKeyframeAnimation.java lottie/src/main/java/com/airbnb/lottie/animation/keyframe

```
186   private interface KeyframesWrapper<T> {
187     boolean isEmpty();
188
189     boolean isValueChanged(float progress);
190
191     Keyframe<T> getCurrentKeyframe();
192
193     @FloatRange(from = 0f, to = 1f)
194     float getStartDelayProgress();
195
196     @FloatRange(from = 0f, to = 1f)
197     float getEndProgress();
198
199     boolean isCachedValueEnabled(float progress);
200   }
```

Line numbers: KeyframesWrapper อยู่บรรทัดที่ 176 และ 186

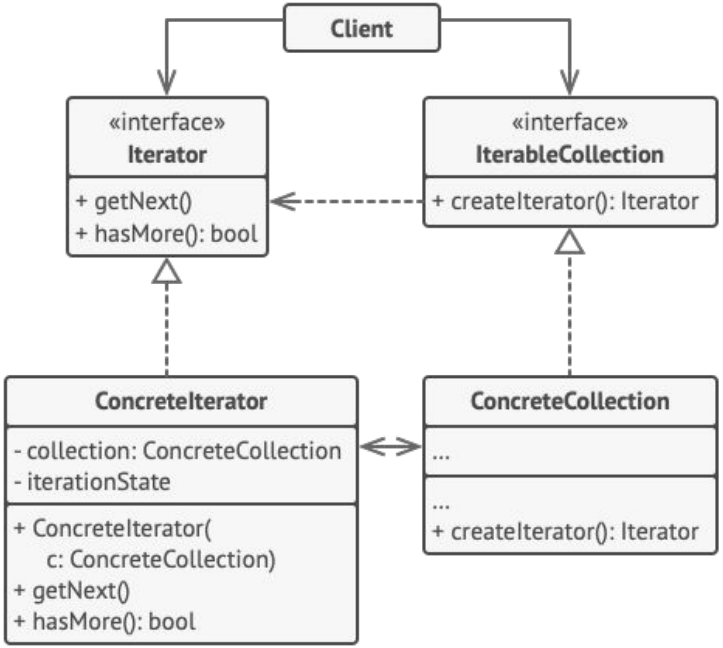
BaseKeyframeAnimation.java lottie/src/main/java/com/airbnb/lottie/animation/keyframe

```
73    protected Keyframe<K> getCurrentKeyframe() {
74      L.beginSection("BaseKeyframeAnimation#getCurrentKeyframe");
75      final Keyframe<K> keyframe = keyframesWrapper.getCurrentKeyframe();
76      L.endSection("BaseKeyframeAnimation#getCurrentKeyframe");
77      return keyframe;
78    }
```

Line numbers: เรียกใช้บรรทัดที่ 75 โดยการเรียก keyframe ปัจจุบัน

3. Iterator Design Pattern Structure

Structure



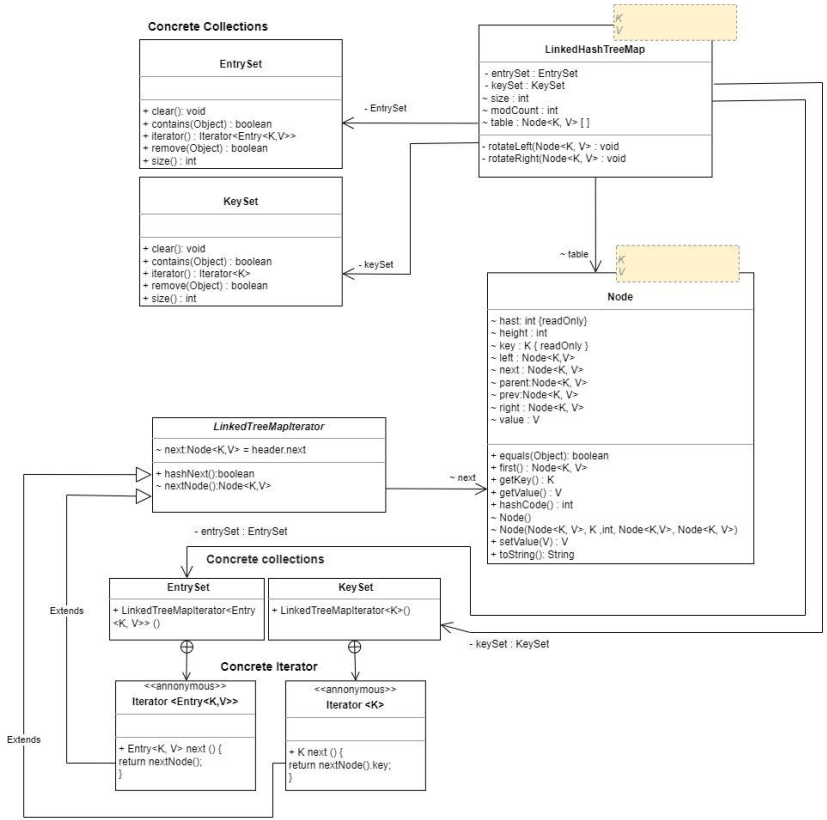
3.1 Iterator Design Pattern

Class Diagram

Generics type คือ K = Key , V = Value

Concrete Collections
การสร้าง Collections จะมี 2 ส่วน คือ
1.ใช้ EntrySet ที่เป็น Concrete Collection เพื่อเข้าถึงข้อมูลใน Collection
2.ใช้ KeySet ที่เป็น Concrete Collection เพื่อคัดเลือกว่าจะนำไปใช้กับ Collection ไหน

Concrete Iterator
ใช้ EntrySet และ KeySet และ inner คือ Anonymous ของทั้ง 2 class ซึ่ง extends จาก Class : LinkedTreeMapIterator



3.2 Iterator Explanation and Evidence

Abstract class : LinkedTreeMapIterator

```
abstract class LinkedTreeMapIterator<T> implements Iterator<T> {
  Node<K, V> next = header.next;
  Node<K, V> lastReturned = null;
  int expectedModCount = modCount;
```

มีการสร้าง Concrete Iterator จาก LinkedTreeMapIterator เป็นการเข้าถึง Collections

Abstract Class : LinkedTreeMapIterator

Source line : อยู่ในไฟล์ lottie-android /lottie/src/main/java/com/airbnb/lottie/parser/moshi/LinkedHashTreeMap.java // Line number : 767

3.3 Iterator Explanation and Evidence

Concrete Iterator จะคัดเลือกตัวใดตัวหนึ่งของ Class แล้ว return Iterator

```
@Override public Iterator<Entry<K, V>> iterator() {
  return new LinkedTreeMapIterator<Entry<K, V>>() {
    public Entry<K, V> next() {
      return nextNode();
    }
  };
}
```

Concrete Itrator<Entry<K,V>> ของ Abstraction

```
@Override public Iterator<K> iterator() {
  return new LinkedTreeMapIterator<K>() {
    public K next() {
      return nextNode().key;
    }
  };
}
```

Concrete Itrator<K> ของ Abstraction

สร้าง Concrete Iterator จาก LinkedTreeMapIterator
โดยมีการสร้าง anonymous เป็นของตนเอง

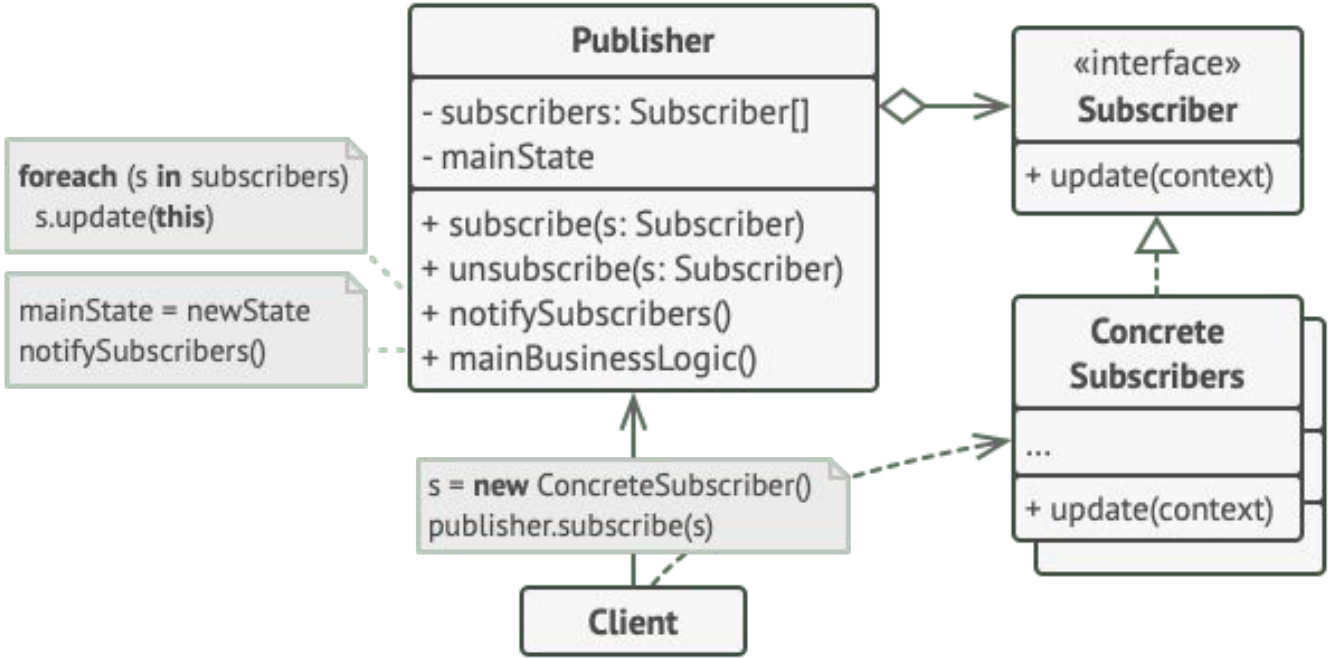
Concrete Class

Source line : อยู่ในไฟล์ lottie-android /lottie/src/main/java/com/airbnb/lottie/parser/moshi/LinkedHashTreeMap.java // Line number : 803

Source line : อยู่ในไฟล์ lottie-android /lottie/src/main/java/com/airbnb/lottie/parser/moshi/LinkedHashTreeMap.java // Line number : 839

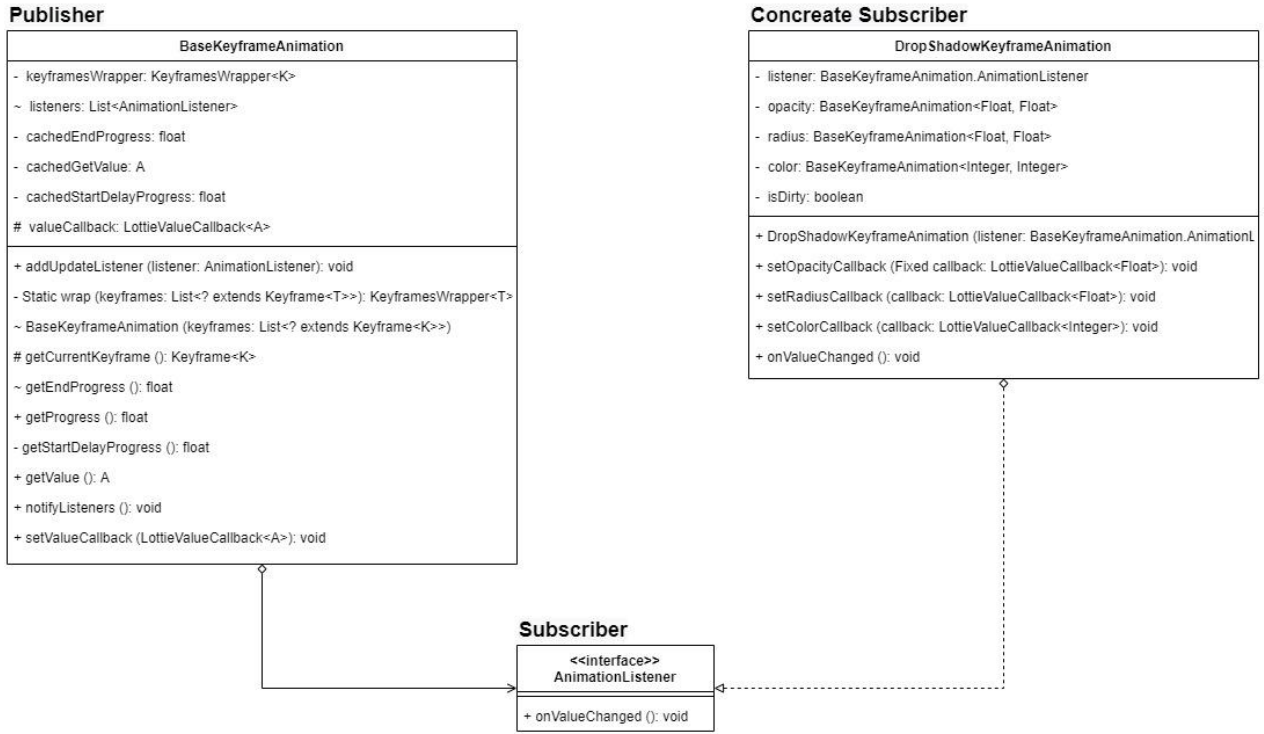
4. Observer Design Pattern

Structure



4.1 Observer Design Pattern

Observer Diagram



4.2 Observer Explanation and Evidence

```
40    public final void setValue(@Nullable T value) {
41      this.value = value;
42      if (animation != null) {
43        animation.notifyListeners();
44      }
```

BaseKeyframeAnimation ทำหน้าที่เป็น Publisher รองรับการ subscribe หรือ register AnimationListener และ BaseKeyframeAnimation ยังสามารถ notify AnimationListener ที่ subscribe ไว้ด้วย ในส่วนของ DorpShadowKeyframeAnimation จะเป็น Concrete Class ของ AnimationListener เวลา notify ไม่ต้องทำผ่าน AnimationListener แต่การ notify จะไปยัง Instance ของ DropShadowKeyframeAnimation เลย กล่าวคือ DropShadowKeyframeAnimation is a AnimtionListener ตอน register AnimationListener ให้ไป subscribe กับ BaseKeyframeAnimation แล้วตัว BaseKeyframeAnimation จะรับ AnimationListener มา จากนั้นก็ใส่ DropShadowKeyframeAnimation ไปได้เลย เพราะว่า DropShadowKeyframeAnimation ก็คือ AnimtionListener

Source line : lottie-android/lottie/src/main/java/com/airbnb/lottie/value/LottieValueCallback.java

Line number : 43

4.3 Observer Evidence

```
18  public abstract class BaseKeyframeAnimation<K, A> {
19    public interface AnimationListener {
20      void onValueChanged();
21    }
```

Interface AnimationListener
Source line : lottie-android/lottie/src/main/java/com/airbnb/lottie/animation/keyframe/BaseKeyframeAnimation.java
Line number : 19

```
13  public class DropShadowKeyframeAnimation implements BaseKeyframeAnimation.AnimationListener {
14    private static final double DEG_TO_RAD = Math.PI / 180.0;
```

การ implements BaseKeyframeAnimation ของ Class DropShadowKeyframeAnimation
Source line : lottie-android/lottie/src/main/java/com/airbnb/lottie/animation/keyframe/DropShadowKeyframeAnimation.java
Line number : 13

03

Quality attribute

Quality attribute (Pros)

Reusability

Lottie สามารถนำไปใช้งานได้ในแทบจะทุก Project ที่มีการใช้งาน Animation และนอกจากนี้แล้ว Animation ส่วนใหญ่ ก็มาจาก Adobe after effect

Ease of Development

เนื่องจาก Layered Architecture มีการแยก layer ในการทำงานอย่างชัดเจน จึงสามารถแยกสัดส่วนงานในการพัฒนาไปแต่ละ layer ได้อย่างชัดเจน

Portability

ตัว Lottie นั้น นอกจากจะรองรับการใช้งานสำหรับ Android แล้ว ยังสามารถใช้งานบน IOS และ Web ได้อีกด้วย

Quality attribute (Cons)

Performance

การทำงานส่วนใหญ่ใน Layer นั้น มักจะจำเป็นต้องส่ง request จาก layer บน

Scalability

เนื่องจาก Layer Architecture นั้นแยกการทำงานอย่างชัดเจน เมื่อต้องการ Scale จึงต้อง Scale แยกแต่ละ Layer

Thank You