Animated Transitions

C.-Z. Yang
http://syslab.cse.yzu.edu.tw/~czyang

3.6 Basic Transitions between Activities

- The quality of an application can depend on several characteristics, such as
  - content
  - usability
  - design features
- Enhancements made to interaction design, through the use of animation, can make a fundamental difference in the usability of an application.
- User interfaces are not static designs, but rather engaging and dynamic design patterns.

Animated transitions

- Animated transitions primarily serve a functional purpose, but they may also improve the overall beauty of a user experience.
- Custom transition animations are resources that can be built by methods in XML code.
- The method overridePendingTransition() allows custom-built transitions to be entering and exiting activities.
  - overridePendingTransition() requires two arguments:
    - A resource ID of the animation resource to use for the incoming activity. Use 0 for no animation.
    - A resource ID of the animation resource to use for the outgoing activity. Use 0 for no animation.

Code example

- ActivityA.java

```
Intent intent = new Intent(ActivityA.this, ActivityB.class);
startActivity(intent);
```

- ActivitB.java

```
@Override
  protected void onStart(){
     super.onStart();
     overridePendingTransition(R.main.transition_in,
                                                    R.main.transition_out);
  }
```

Code example

- transition.xml for Property animation

```
<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>
```

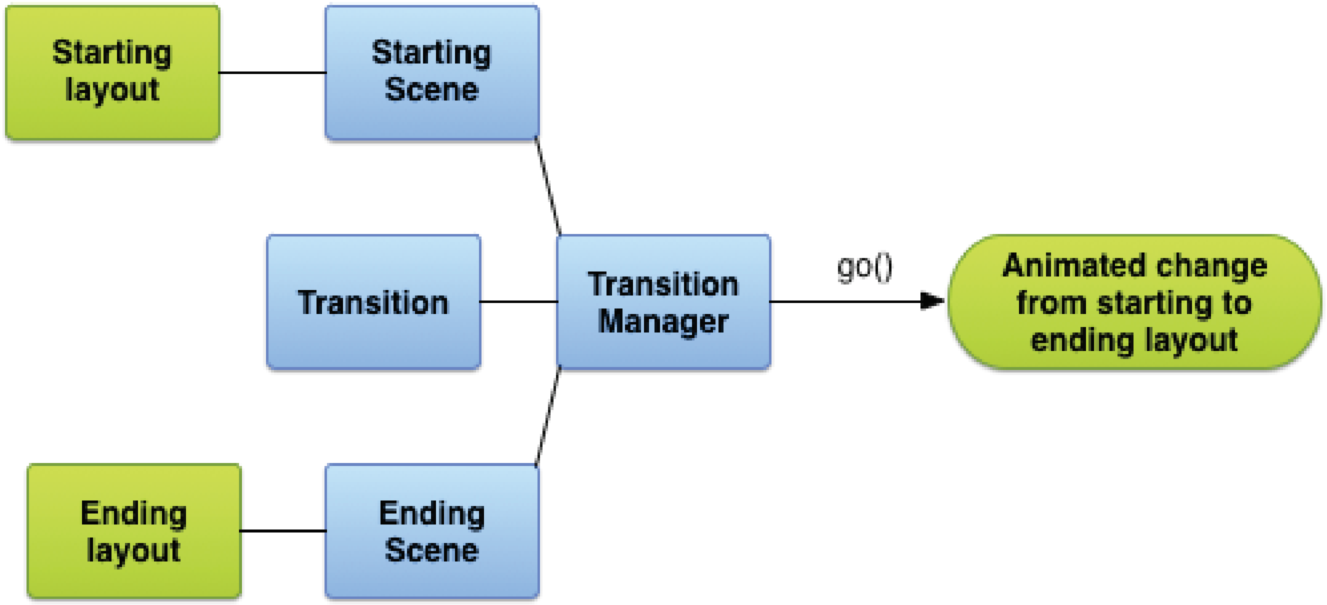
- An animated transition is implemented using an objectAnimator.
- The x and y properties are set to values 400 and 300.
- The animation will play over a duration of 500 ms.

3.7 Scene Transitions

- Android has a transition framework that supports the definition of scene transitions.
  - Android's transition framework allows you to animate all kinds of motion in your UI by simply providing the starting layout and the ending layout.
- Scenes are used specifically for transition animations.
  - Scenes can be created as a View hierarchy, much like a layout.
  - A scene contains values of various properties in the View hierarchy.
- As scenes enter or exit a View hierarchy, they will be animated based on these properties.

Transition Framework

- The relationship between your layouts, the scenes, the transition, and the final animation.



https://developer.android.com/training/transitions

Transitions and Scenes

- Transitions and scenes produce animations based on specific properties.
  - A Transition holds information about animations that are run on its targets during a Scene change.
- Every Transition object has two functions:
  - To capture property values
  - To Play animations based on changes to the captured property values

Code example

- Myactivity.java

```
setContentView(R.layout.activity_my);

paintingContainer = (ViewGroup) findViewById(R.id.painting_container);
transition = TransitionInflater.from(this).inflateTransition(
        R.anim.transition);

activeScene = Scene.getSceneForLayout(paintingContainer,
        R.layout.scene01, this);
passiveScene = Scene.getSceneForLayout(paintingContainer,
        R.layout.scene02, this);
activeScene.enter();
```

1. A View defined in the layout.

2. A scene layout resource.

3. The context used in the process of inflating a scene layout.

- Two scene layouts: scene01 and scene02
- paintingContainer is a ViewGroup to contain other view objects.
- A Transition object is used to hold the animation information.

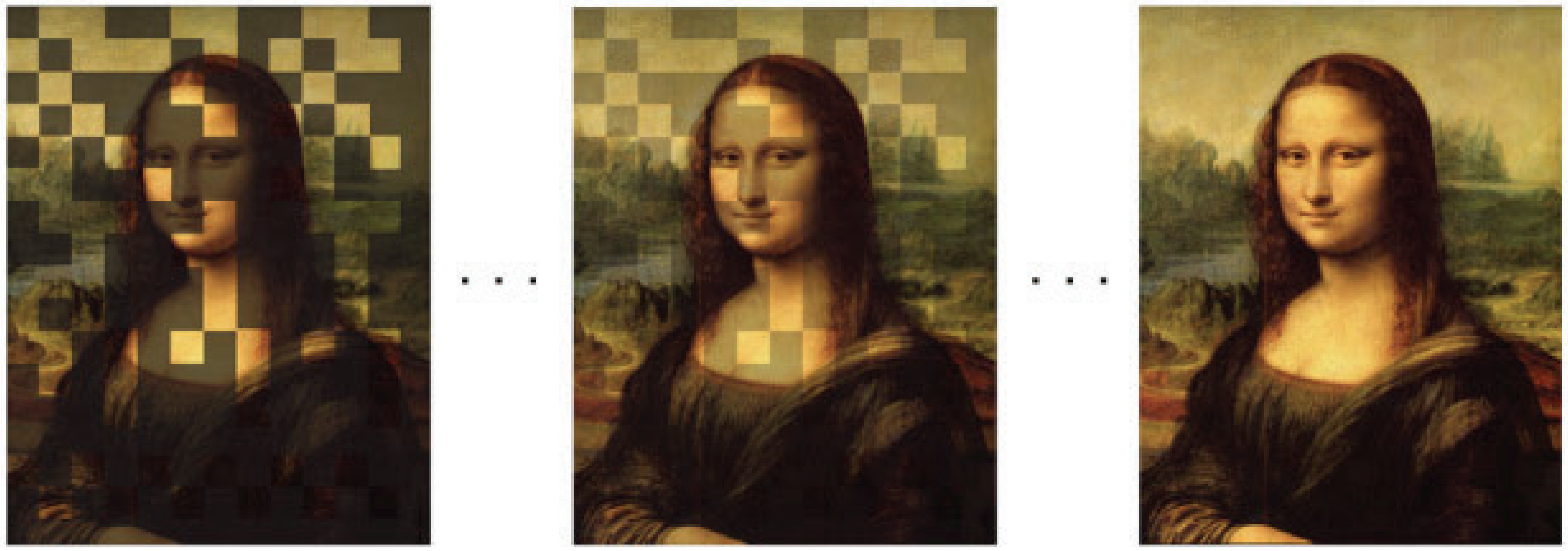
Code example

- Myactivity.java

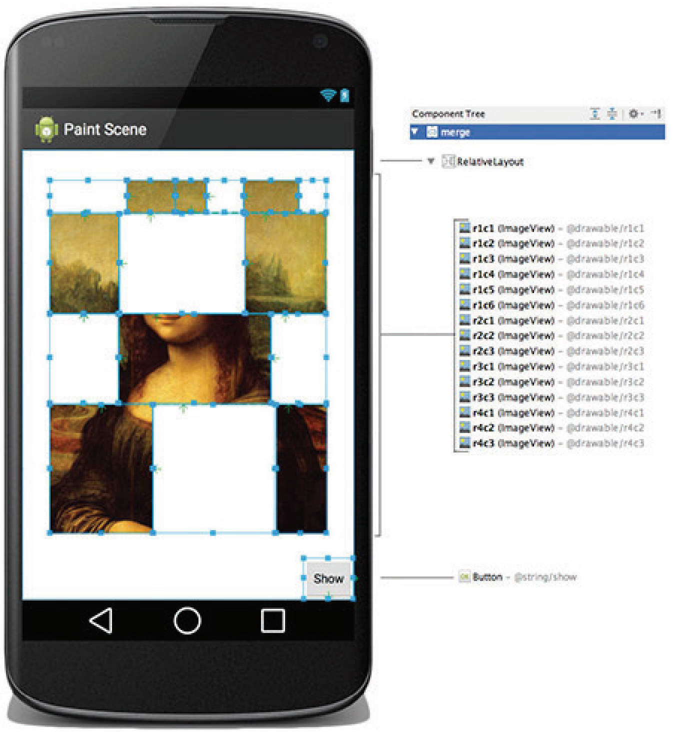
```
public void changeScenes(View view) {

    Scene temp = passiveScene;
    passiveScene = activeScene;
    activeScene = temp;

    TransitionManager.go(activeScene, transition);

  }
```

- TransitionManager.go() will change to the given Scene using the given Transition.

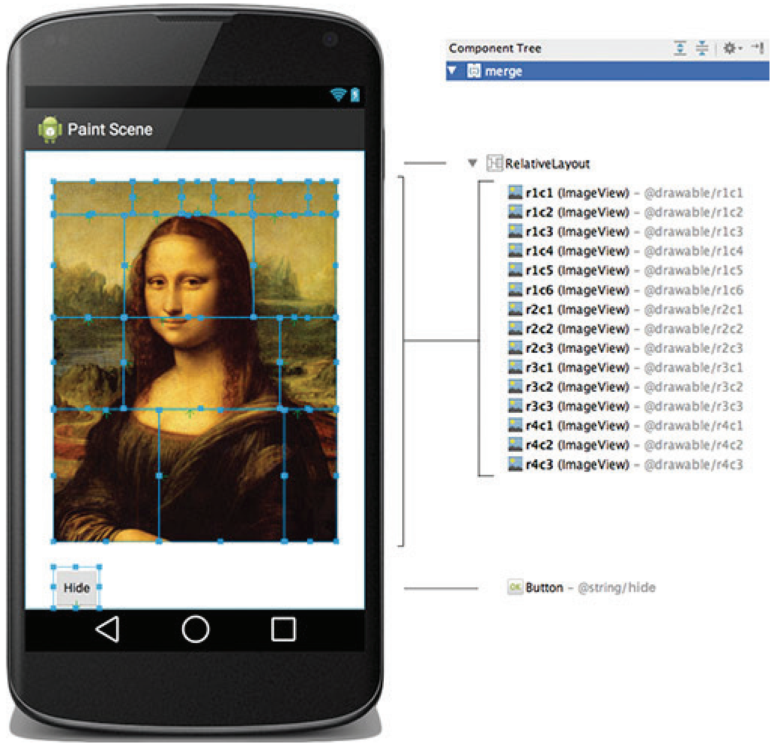
Animation example



Layout example



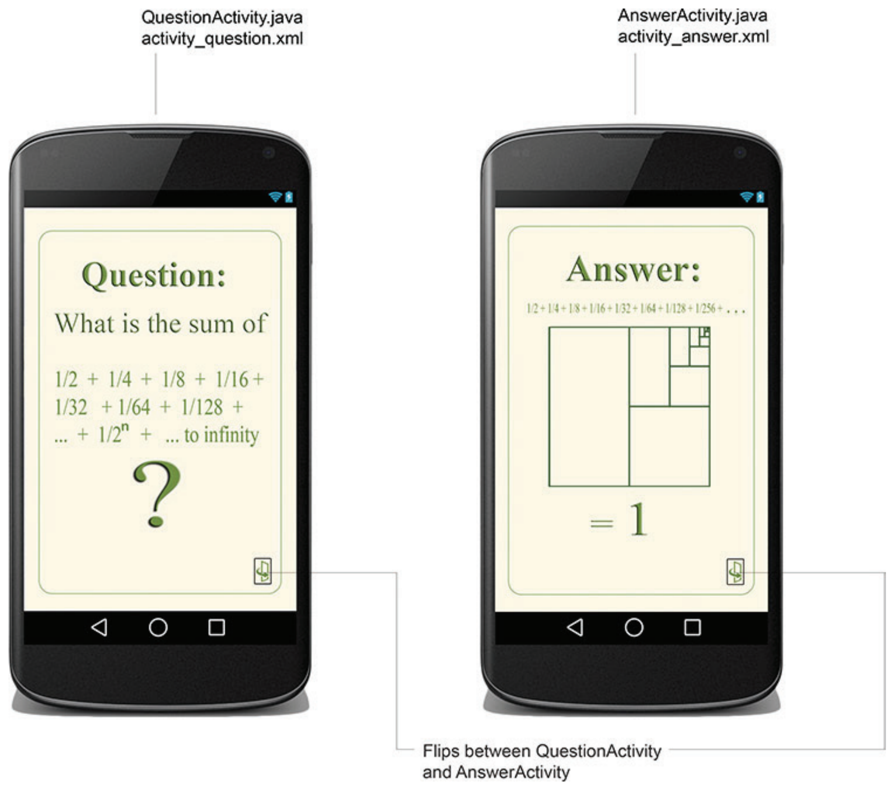
```
Component Tree
▼ merge
  ▼ RelativeLayout
      r1c1 (ImageView) - @drawable/r1c1
      r1c2 (ImageView) - @drawable/r1c2
      r1c3 (ImageView) - @drawable/r1c3
      r1c4 (ImageView) - @drawable/r1c4
      r1c5 (ImageView) - @drawable/r1c5
      r1c6 (ImageView) - @drawable/r1c6
      r2c1 (ImageView) - @drawable/r2c1
      r2c2 (ImageView) - @drawable/r2c2
      r2c3 (ImageView) - @drawable/r2c3
      r3c1 (ImageView) - @drawable/r3c1
      r3c2 (ImageView) - @drawable/r3c2
      r3c3 (ImageView) - @drawable/r3c3
      r4c1 (ImageView) - @drawable/r4c1
      r4c2 (ImageView) - @drawable/r4c2
      r4c3 (ImageView) - @drawable/r4c3
    Button - @string/show
```



```
Component Tree
▼ merge
  ▼ RelativeLayout
      r1c1 (ImageView) - @drawable/r1c1
      r1c2 (ImageView) - @drawable/r1c2
      r1c3 (ImageView) - @drawable/r1c3
      r1c4 (ImageView) - @drawable/r1c4
      r1c5 (ImageView) - @drawable/r1c5
      r1c6 (ImageView) - @drawable/r1c6
      r2c1 (ImageView) - @drawable/r2c1
      r2c2 (ImageView) - @drawable/r2c2
      r2c3 (ImageView) - @drawable/r2c3
      r3c1 (ImageView) - @drawable/r3c1
      r3c2 (ImageView) - @drawable/r3c2
      r3c3 (ImageView) - @drawable/r3c3
      r4c1 (ImageView) - @drawable/r4c1
      r4c2 (ImageView) - @drawable/r4c2
      r4c3 (ImageView) - @drawable/r4c3
    Button - @string/hide
```

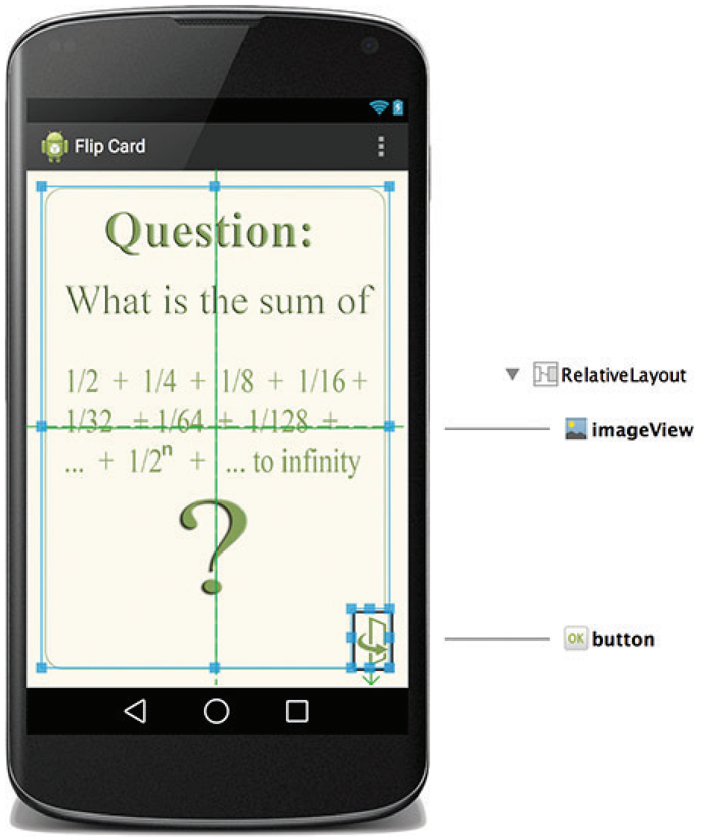
Lab example 3-4: Flip Cards

- This lab example is used to illustrate how an animation can be used when navigating between activities.



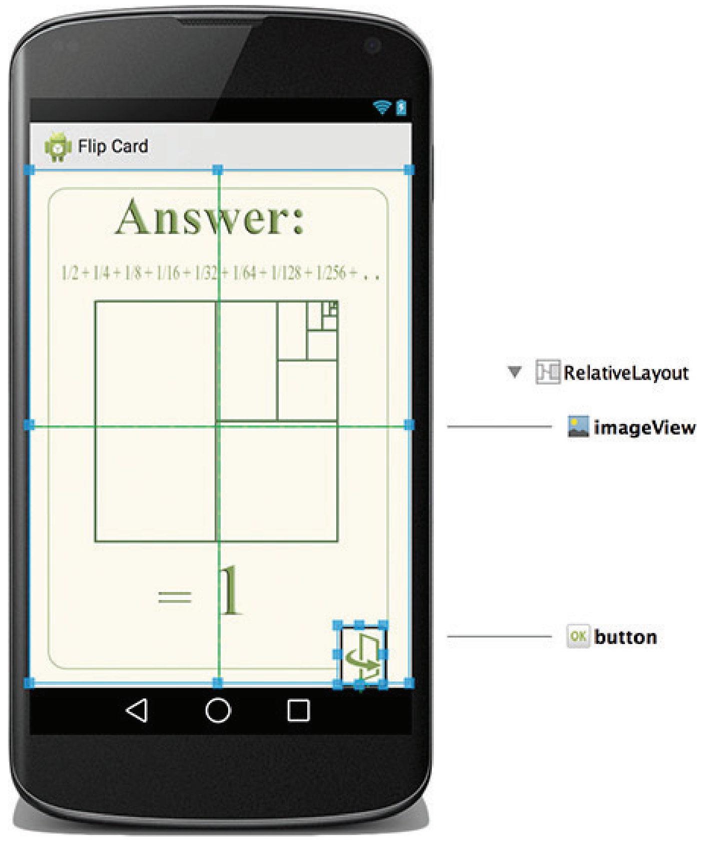
User interface

- activity_question.xml



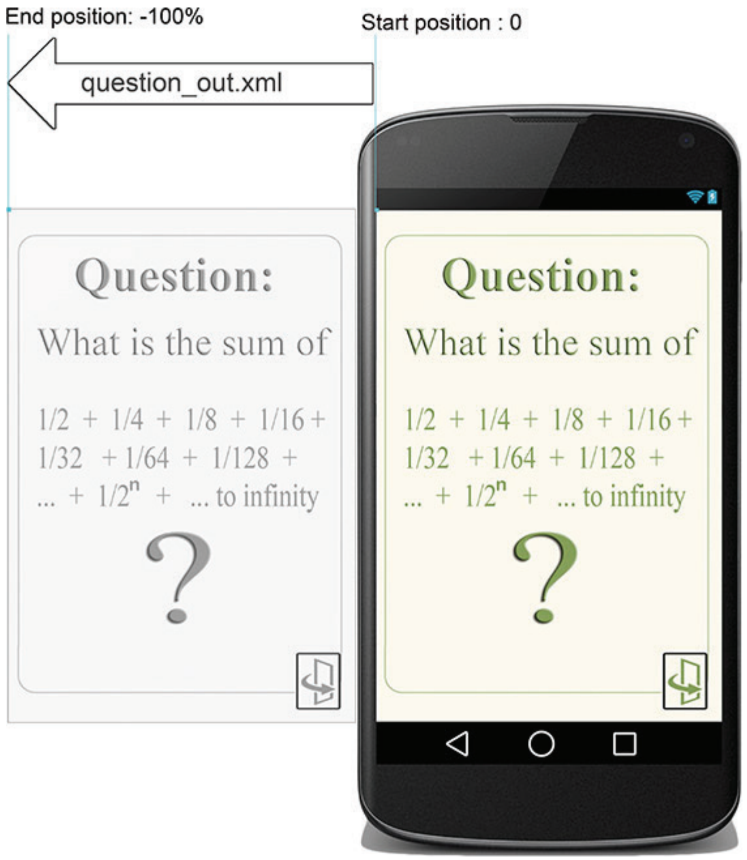
User interface

- activity_answer.xml



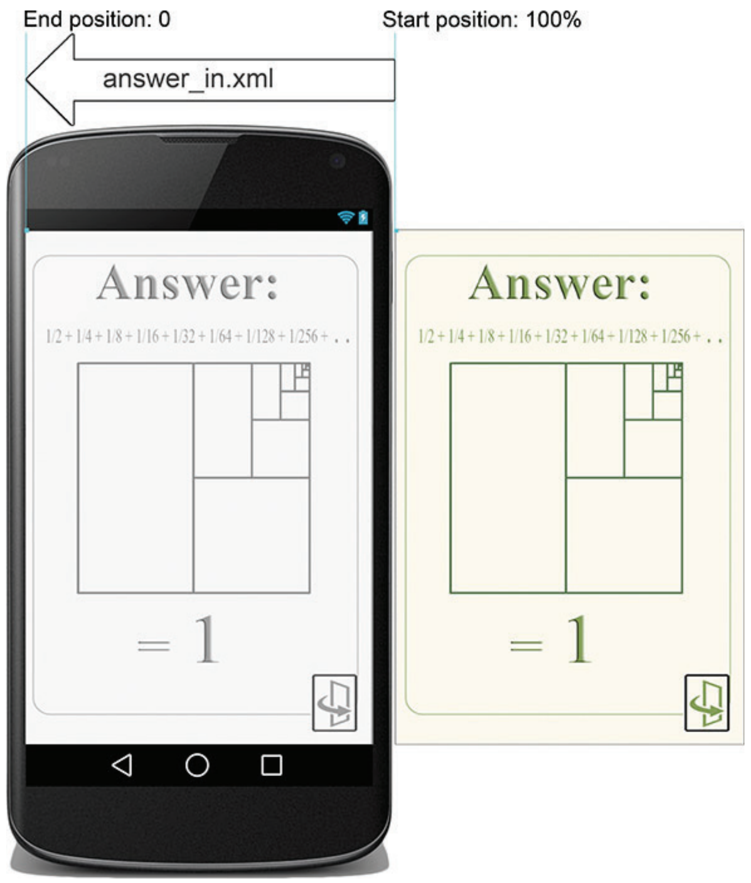
Animated transition

- A question exits the screen by moving to the right



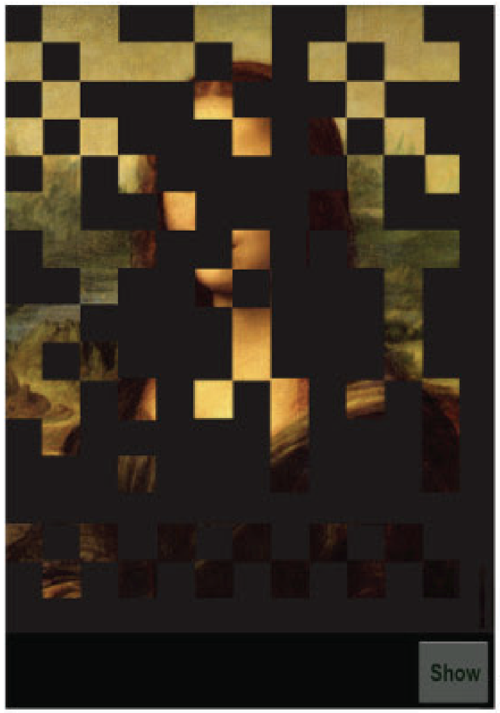
Animated transition

- An answer image enters the screen from the left



Lab example 3-5: Painting Scene

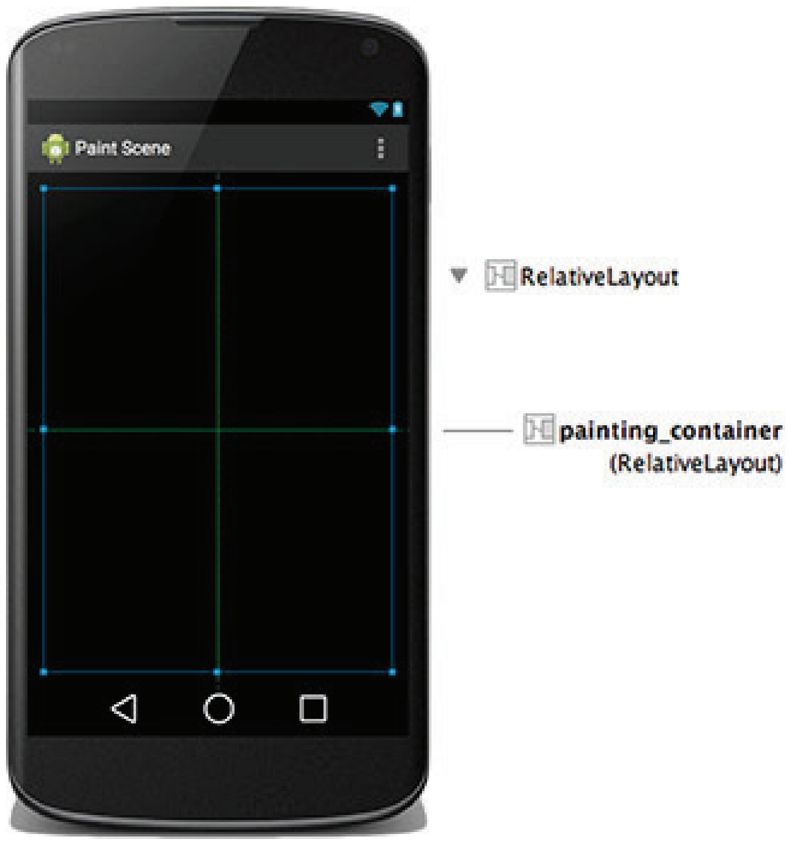
- This lab example is used to explore the use of the TransitionManager and the construction of Scenes and Transitions.



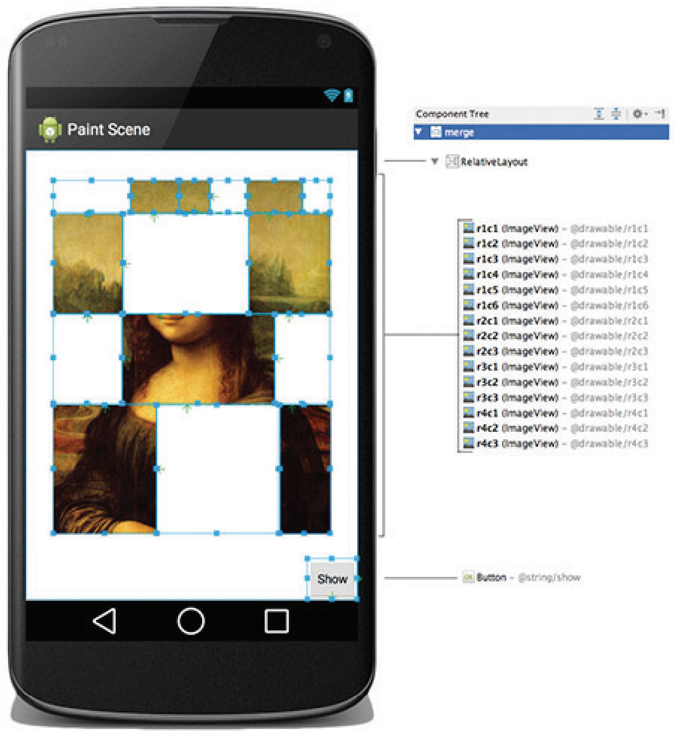
An animated transition is used to reveal the complete painting.

User interface

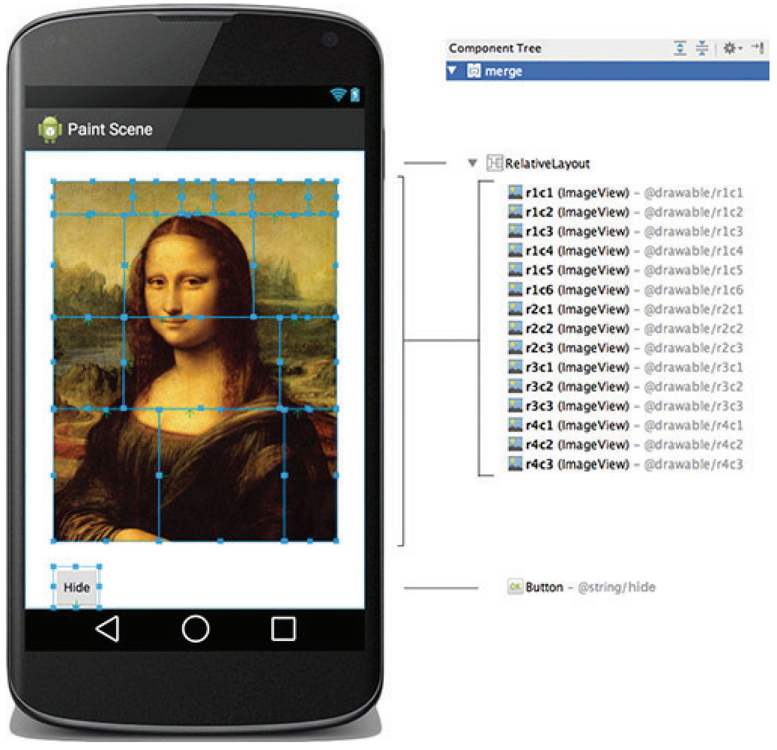
- The graphic design of the layout activity_my.xml



User interface



```
Component Tree
▼ merge
  ▼ RelativeLayout
      r1c1 (ImageView) - @drawable/r1c1
      r1c2 (ImageView) - @drawable/r1c2
      r1c3 (ImageView) - @drawable/r1c3
      r1c4 (ImageView) - @drawable/r1c4
      r1c5 (ImageView) - @drawable/r1c5
      r1c6 (ImageView) - @drawable/r1c6
      r2c1 (ImageView) - @drawable/r2c1
      r2c2 (ImageView) - @drawable/r2c2
      r2c3 (ImageView) - @drawable/r2c3
      r3c1 (ImageView) - @drawable/r3c1
      r3c2 (ImageView) - @drawable/r3c2
      r3c3 (ImageView) - @drawable/r3c3
      r4c1 (ImageView) - @drawable/r4c1
      r4c2 (ImageView) - @drawable/r4c2
      r4c3 (ImageView) - @drawable/r4c3
    Button - @string/show
```



```
Component Tree
▼ merge
  ▼ RelativeLayout
      r1c1 (ImageView) - @drawable/r1c1
      r1c2 (ImageView) - @drawable/r1c2
      r1c3 (ImageView) - @drawable/r1c3
      r1c4 (ImageView) - @drawable/r1c4
      r1c5 (ImageView) - @drawable/r1c5
      r1c6 (ImageView) - @drawable/r1c6
      r2c1 (ImageView) - @drawable/r2c1
      r2c2 (ImageView) - @drawable/r2c2
      r2c3 (ImageView) - @drawable/r2c3
      r3c1 (ImageView) - @drawable/r3c1
      r3c2 (ImageView) - @drawable/r3c2
      r3c3 (ImageView) - @drawable/r3c3
      r4c1 (ImageView) - @drawable/r4c1
      r4c2 (ImageView) - @drawable/r4c2
      r4c3 (ImageView) - @drawable/r4c3
    Button - @string/hide
```

Concluding Remarks