Exercise1：

使用函数setProgress即可：

animationView.setProgress(progress / 100f);



Exercise2：

首先在activity\_ch3ex2.xml中，将visibility赋值为“visible“

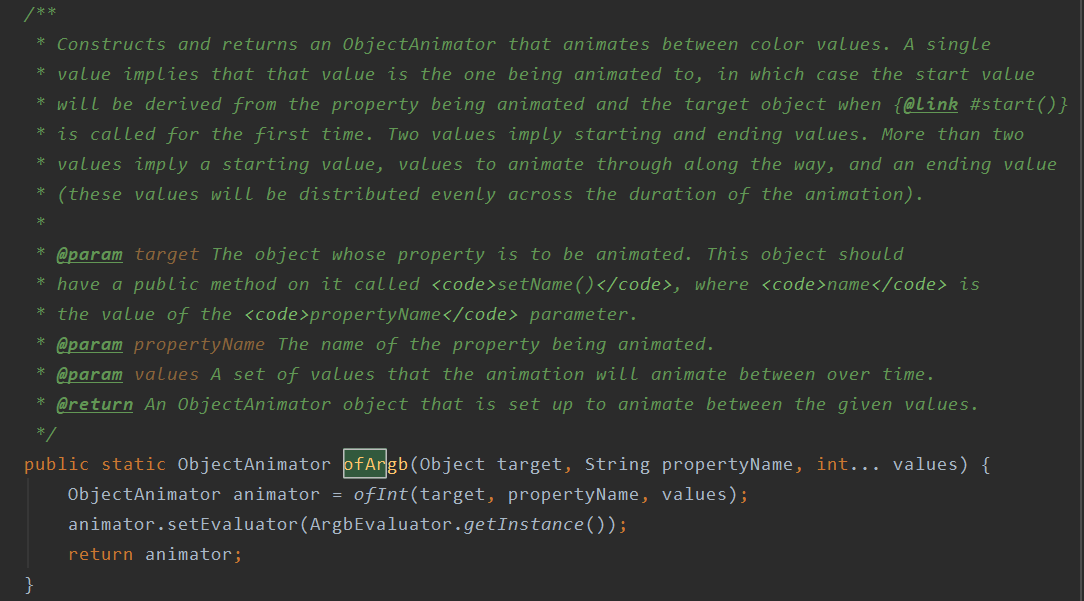
<!-- *TODO X: A fun rainbow view, try to set it to visible* -->  
<!--android:visibility="invisible"-->  
<com.example.chapter3.homework.RainbowTextView  
 android:id="@+id/rainbow"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="32dp"  
 android:textSize="22sp"  
 android:visibility="visible"  
 android:layout\_gravity="center\_horizontal"  
 android:text="@string/rainbow\_text" />

填充Ch3Ex2Activity.java，实现对target控件的大小缩放和透明度的修改，分别用到“scaleX“，”scaleY“，”alpha“三个属性。

// *TODO ex2-1：在这里实现另一个 ObjectAnimator，对 target 控件的大小进行缩放，从 1 到 2 循环*ObjectAnimator animatorX = ObjectAnimator.*ofFloat*(target,  
 "scaleX", 1f, 2f);  
animatorX.setRepeatCount(ValueAnimator.*INFINITE*);  
animatorX.setInterpolator(new LinearInterpolator());  
animatorX.setDuration(Integer.*parseInt*(durationSelector.getText().toString()));  
animatorX.setRepeatMode(ValueAnimator.*REVERSE*);  
  
ObjectAnimator animatorY = ObjectAnimator.*ofFloat*(target,  
 "scaleY", 1f, 2f);  
animatorY.setRepeatCount(ValueAnimator.*INFINITE*);  
animatorY.setInterpolator(new LinearInterpolator());  
animatorY.setDuration(Integer.*parseInt*(durationSelector.getText().toString()));  
animatorY.setRepeatMode(ValueAnimator.*REVERSE*);  
// *TODO ex2-2：在这里实现另一个 ObjectAnimator，对 target 控件的透明度进行修改，从 1 到 0.5f 循环*ObjectAnimator animator3 = ObjectAnimator.*ofFloat*(target,  
 "alpha", 1f, 0f);  
animator3.setRepeatMode(ValueAnimator.*REVERSE*);  
animator3.setRepeatCount(ValueAnimator.*INFINITE*);  
animator3.setDuration(Integer.*parseInt*(durationSelector.getText().toString()));  
// *TODO ex2-3: 将上面创建的其他 ObjectAnimator 都添加到 AnimatorSet 中*animatorSet = new AnimatorSet();  
animatorSet.playTogether(animator1, animatorX, animatorY, animator3);  
animatorSet.start();

\*\*可以思考下，这里为什么要使用 ofArgb，而不是 ofInt 呢？

ofAgrb是专门用来对颜色属性进行动画的函数，查看源代码，注释中有提到：Constructs and returns an ObjectAnimator that animates between Object values



ofInt：

/\*\*

\* Constructs and returns an ObjectAnimator that animates coordinates along a <code>Path</code>

\* using two properties. A <code>Path</code></> animation moves in two dimensions, animating

\* coordinates <code>(x, y)</code> together to follow the line. In this variation, the

\* coordinates are integers that are set to separate properties, <code>xProperty</code> and

\* <code>yProperty</code>.

\*/

此外还有ofObject用来对对象进行动画的，ofMultiInt,ofMultiFloat这种多参数的等等。

Exercise3:

// *TODO: ex3-1. 添加 ViewPager 和 Fragment 做可滑动界面*ViewPager pager = findViewById(R.id.*view\_pager*);  
pager.setAdapter(new FragmentPagerAdapter(getSupportFragmentManager()) {  
 @Override  
 public Fragment getItem(int i) {  
 return new PlaceholderFragment();  
 }  
  
 @Override  
 public int getCount() {  
 return *PAGE\_COUNT*;  
 }  
  
 @Override  
 public CharSequence getPageTitle(int position) {  
 if(position == 0)  
 return "好友列表";  
 else  
 return "我的好友";  
 }  
});  
  
// *TODO: ex3-2, 添加 TabLayout 支持 Tab*TabLayout tabLayout = findViewById(R.id.*tab\_layout*);  
tabLayout.setupWithViewPager(pager);

// TODO ex3-3: 修改 fragment\_placeholder，添加 loading 控件和列表视图控件

View view = inflater.inflate(R.layout.*fragment\_placeholder*, container, false);  
animationView = view.findViewById(R.id.*animation\_view*);  
listView = view.findViewById(R.id.*lv\_1*);  
listView.setAdapter(new ListAdapter());  
return view;

// TODO ex3-4：实现动画，将 lottie 控件淡出，列表数据淡入

采用Exercise2中一样的属性动画的方法即可：

ObjectAnimator animator = ObjectAnimator.*ofFloat*(animationView,  
 "alpha", 1f, 0f);  
ObjectAnimator animator1 = ObjectAnimator.*ofFloat*(listView,  
 "alpha", 0f, 1f);  
animatorSet = new AnimatorSet();  
animatorSet.playTogether(animator, animator1);  
animatorSet.start();