Создаем двойную анимацию восходящего солнца и идущих часов

|  |
| --- |
| [Главная](http://learn-android.ru/index.html) » Создаем двойную анимацию восходящего солнца и идущих часов |
| Представляю вашему вниманию еще один урок по анимации (до этого мы делали [переход](http://learn-android.ru/news/osnovy_raboty_s_perekhodami/2015-01-14-13.html)и простую[фреймовую анимацию](http://learn-android.ru/news/sozdanie_prostoj_frejmovoj_animacii_v_android_prilozhenii/2015-01-18-24.html)). Очертим примерно нашу цель и задачи. Мы создаем анимацию восходящего солнца, а также часов с вращающимися минутной и часовой стрелками. Каждый объект, который будет присутствовать на экране, нужно создать и задать его свойства. У нас этими объектами будут: небо, солнце, трава, часы с двумя стрелками и кнопка запуска анимации. Далее мы оживляем нужные объекты с помощью создания анимации. И последний, но и самый важный шаг, пишем код приложения, который будет запускать нужную анимацию при нажатии на кнопку в углу экрана. Для выполнения урока не потребуется задействовать никаких сторонних файлов - все, что нам нужно, есть в стандартном наборе инструментов ANdroid Studio.  Объем работы не так уж мал, но не бойтесь, все довольно просто. Будем выполнять поставленные задачи в таком порядке:       - сначала создаем все необходимое объекты;       - оформляем внешний вид программы (интерфейс в файле activity\_main.xml);       - создаем нужным объектам анимацию;       - кодим все необходимое в файле MainActivity.java.  Начнем. [Создаем](http://learn-android.ru/news/rabochaja_sreda_android_studio_i_hellow_development_world/2015-01-03-5.html) новый проект по имени Animation. Все названия оставляем стандартными, выбираем **Blank Activity**.  Начинаем с создания объектов. В папке **drawable**создаем файл **sky.xml**(выбрать папку, правый клик --- **New**---**Drawable resourse file**). Создаем здесь форму неба, заполняющего все пространство окна, настраиваем ему синенький градиент, угол градиента:  <shape xmlns:android="http://schemas.android.com/apk/res/android"  android:dither="true"  android:shape="rectangle">  <gradient  android:startColor="#ff0000ff"  android:endColor="#ff000033"  android:angle="90"/>  </shape>  Команда **dither**используется для надежности, чтобы градиент отобразился правильно.  В этой же папке **drawable**создаем файл для солнца **sun.xml**:  <shape xmlns:android="http://schemas.android.com/apk/res/android"  android:dither="true"  android:shape="oval">  <gradient  android:startColor="#ffffcc00"  android:endColor="#ffff6600"  android:gradientRadius="150"  android:type="radial"  android:useLevel="false"/>  <size  android:height="150dp"  android:width="150dp"/>  </shape>  Здесь мы создали круг с заданным размером 150dp\*150 dp.  Создаем тут же еще один файл для травы **grass.xml**:  <shape xmlns:android="http://schemas.android.com/apk/res/android"  android:dither="true"  android:shape="rectangle">  <gradient  android:startColor="#ff009900"  android:endColor="#ff003300"  android:angle="90"/>  </shape>  Он мало чем отличается от файла для создания неба, разве что цветом.  Теперь давайте создадим часики с минутной стрелкой. Здесь уже понадобится более объемная работа. В папке**drawable**создаем файл **clock.xml**со следующим содержанием:  <layer-list xmlns:android="http://schemas.android.com/apk/res/android" >  <item>  <shape  xmlns:android="http://schemas.android.com/apk/res/android"  android:dither="true"  android:shape="oval">  <gradient  android:startColor="#66ffffff"  android:endColor="#ffffffff"  android:gradientRadius="100"  android:type="radial"  android:useLevel="false"/>  <size  android:height="100dp"  android:width="100dp"/>  <stroke  android:width="2dp"  android:color="#99000000"/>  </shape>  </item>  <item  android:bottom="44dp"  android:left="48dp"  android:right="48dp"  android:top="5dp">  <shape  xmlns:android="http://schemas.android.com/apk/res/android"  android:shape="rectangle">  <solid  android:color="#99000000"/>  </shape>  </item>  </layer-list>  Мы используем**<layer-list>** для того, чтобы поместить в один файл сразу несколько объектов (до этого каждый файл создавал только один объект). Каждый новый объект в **<layer-list>**вложен в тег **<item></item>**. В файле создано 2 объекта: круг размера 100\*100 dp с серым градиентом и обрамлением тиснением по краю в 2dp, а также прямоугольник, который будет играть роль минутной стрелки.  Ну и последний штрих в рисовании все новых деталей анимации - создание часовой стрелки. Создаем в той же папке **drawable**файл **hour\_hand.xml**со следующим содержимым:  <layer-list xmlns:android="http://schemas.android.com/apk/res/android" >  <item>  <shape  xmlns:android="http://schemas.android.com/apk/res/android"  android:dither="true"  android:shape="oval" >  <solid android:color="#00000000" />  <size  android:height="100dp"  android:width="100dp" />  </shape>  </item>  <item  android:bottom="44dp"  android:left="48dp"  android:right="48dp"  android:top="15dp">  <shape  xmlns:android="http://schemas.android.com/apk/res/android"  android:shape="rectangle" >  <solid android:color="#99000000" />  </shape>  </item>  </layer-list>  Этот код фактически аналогичен коду для часов с минутной стрелкой, но здесь цвет круга (часов) сливается с фоном, а стрелка немного короче. На этом мы заканчиваем с рисованием.  Теперь у нас по плану оформление интерфейса программы. Открываем файл **activity\_main.xml** и добавляем туда для всех элементов, которые будут использованы (солнце, небо, трава, часы с минутной стрелкой, часовая стрелка и кнопка запуска анимации) соответствующие объекты. То есть, нам нужно создать 5 элементов **ImageView**и одну кнопку**Button**. Файл будет выглядеть так:  <**RelativeLayout** xmlns:android="http://schemas.android.com/apk/res/android"  xmlns:tools="http://schemas.android.com/tools"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  tools:context=".MainActivity">  <**ImageView**  android:layout\_width="fill\_parent"  android:layout\_height="fill\_parent"  android:id="@+id/sky"  android:contentDescription="@string/sky"  android:src="@drawable/sky"/>  <**ImageView**  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:id="@+id/sun"  android:contentDescription="@string/sun"  android:src="@drawable/sun"  android:layout\_centerHorizontal="true"/>  <**ImageView**  android:layout\_width="fill\_parent"  android:layout\_height="150dp"  android:id="@+id/grass"  android:contentDescription="@string/grass"  android:src="@drawable/grass"  android:layout\_alignParentBottom="true"/>  <**ImageView**  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:id="@+id/clock"  android:contentDescription="@string/clock"  android:src="@drawable/clock"  android:layout\_alignParentBottom="true"  android:layout\_alignParentRight="true"  android:padding="10dp"/>  <**ImageView**  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:id="@+id/hour"  android:layout\_alignParentRight="true"  android:layout\_alignParentBottom="true"  android:src="@drawable/hour\_hand"  android:padding="10dp"  android:contentDescription="@string/hour"/>  <**Button**  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:id="@+id/button"  android:text="Start"  android:onClick="Go"  android:layout\_alignParentBottom="true"  android:layout\_alignParentLeft="true"/>  </**RelativeLayout**>  Само окно программы должно выглядеть вот так вот:  http://developeroleg.ucoz.ru/Images/Studies/a1.jpg  Мы написали нашей кнопке Start метод **onClick**с именем **"Go"**(назначили ей нажатие). Нужно добавить этот метод в**MainActivity.java**. Ставим курсов на строчку **android:onClick="Go"**жмем **Alt+Enter** и еще раз **Enter**применяя команду создания метода (в других уроках это делается регулярно, поэтому подробно описывать в очередной раз это действие бессмысленно, если вы все же не могете, ничего, вы увидите нужные команды ниже, при редактировании файла MainActivity.java).  Теперь переходим к созданию анимации.  Создаем в файлах приложения папку по имени **anim**на одном уровне с **drawable**:  http://developeroleg.ucoz.ru/Images/Studies/anim1.jpg  У нас будет целых 3 одновременных анимации: одна - солнца, другая - минутной стрелки, третья - часовой стрелки. В этой папке создаем файл **sun\_rise.xml** для того, чтобы сделать анимацию солнцу:  <?xml version="1.0" encoding="utf-8"?>  <set xmlns:android="http://schemas.android.com/apk/res/android"  android:shareInterpolator="false"  android:duration="5000"  android:fillAfter="true"  android:interpolator="@android:anim/accelerate\_decelerate\_interpolator">  <scale  android:fromXScale="1.0"  android:toXScale="1.5"  android:fromYScale="1.0"  android:toYScale="1.5"  android:pivotX="50%"  android:pivotY="50%"/>  <translate  android:fromYDelta="80%p"  android:toYDelta="10%p"/>  <alpha  android:fromAlpha="0.3"  android:toAlpha="1.0"/>  </set>  Мы определили для анимации интерполятор **accelerate\_decelerate\_interpolator**, который сделает движение солнца не линейным, а медленным в начале и в конце, но ускоренным ближе к центру. Выставляем длительность анимации**duration**=5000 мс, 5 секунд, изменяем масштаб солнца от 1.0 размера в начале движения до 1.5 под конец анимации, в теге **translate**определяем движение только по вертикали (по Y) и указываем начать движение от 80% родительского окна (мутное объяснение, попробую лучше - вся высота окна 100%, отчет идет с верху вниз, указывая 80%p мы даем понять, что движение нужно начать с высоты в 80% и до высоты 10% от родительского окна). Также тег **alpha**дает нам возможность настроить прозрачность нашего солнца, увеличивая его от 0.3 до 1.0.  Теперь нужно создать анимацию для часов с минутной стрелкой. В папке **anim**создаем файл **clock\_turn.xml**:  <?xml version="1.0" encoding="utf-8"?>  <set xmlns:android="http://schemas.android.com/apk/res/android"  android:duration="5000"  android:fillAfter="true"  android:interpolator="@android:anim/linear\_interpolator"  android:shareInterpolator="false">  <rotate  android:fromDegrees="0"  android:pivotX="50%"  android:pivotY="50%"  android:toDegrees="720"/>  </set>  Здесь мы установили интерполятором линейное движение (то есть, скорость стрелки при обороте по кругу будет одинакова на всех участках движения), а также настроили вращение от 0 до 720 градусов (то есть, 2 полных оборота, каждый круг по 360 градусов). Точку вращения ставим и по X, и по Y на 50% (**pivotX**, **pivotY**), чтобы у нас получилась анимация относительно центра, а не болталась вся картинка.  Ну и создаем последнюю анимацию - движение часовой стрелки. Тут же в папке **anim**создаем файл **hour\_hand.xml**(да, название для файла самой стрелки и анимации совпали,  не растеряйтесь):  <?xml version="1.0" encoding="utf-8"?>  <set xmlns:android="http://schemas.android.com/apk/res/android"  android:duration="5000"  android:fillAfter="true"  android:interpolator="@android:anim/linear\_interpolator"  android:shareInterpolator="false">  <rotate  android:fromDegrees="150"  android:pivotX="50%"  android:pivotY="50%"  android:toDegrees="240"/>  </set>  Здесь то же, что и в предыдущем файле, за исключением того,  что мы сместили положение часовой стрелки от минутной, чтобы увидеть их раздельно, а также установили для часовой стрелки меньший угол поворота (ведь часовая стрелка вращается медленнее минутной:)).  На этом работа над файлами для анимации закончена. Теперь остается финальный штрих в нашей работе - написать код запуска анимации по щелчку на кнопку Start. Открываем файл **MainActivity.java**. **В методе** обработки щелчка**"Go"**(мы создали его выше при работе в **activity\_main.xml**) нам нужно объявить использование объектов **ImageView**и **Animation**, привязав их к созданным нами элементам в  **activity\_main.xml**и запустить все 3 анимации.  Код будет выглядеть так:  import **android.app.Activity**;  import **android.graphics.drawable.AnimationDrawable**;  import **android.os.Bundle**;  import **android.view.View**;  import **android.widget.ImageView**;  import **android.view.animation.Animation**;  import **android.view.animation.AnimationUtils**;  import **android.widget.ImageView**;  **public** **class** MainActivity **extends** Activity {  **@Override**  **protected** **void** onCreate(**Bundle** *savedInstanceState*) {  super.onCreate(savedInstanceState);  setContentView(**R**.layout.activity\_main);  }  **public** **void** Go(**View** *view*) {  **ImageView** **Sun** = (**ImageView**)findViewById(**R**.id.sun);  **Animation** **SunRise** = **AnimationUtils**.loadAnimation(this,**R**.anim.sun\_rise);  **Sun**.startAnimation(**SunRise**);  **ImageView** **Clock** = (**ImageView**)findViewById(**R**.id.clock);  **Animation** **ClockTurn** = **AnimationUtils**.loadAnimation(this,**R**.anim.clock\_turn);  **Clock**.startAnimation(**ClockTurn**);  **ImageView** **Hour** = (**ImageView**)findViewById(**R**.id.hour);  **Animation** **HourHand** = **AnimationUtils**.loadAnimation(this,**R**.anim.hour\_hand);  **Hour**.startAnimation(**HourHand**);  }  }  Вот и все, урок довольно объемный, но действия понятны и однотипные. Приложение готово для тестирования.**Тестируем**на устройствах/эмуляторах на **Android 4.2+**, на версии 2.3 работать не будет!  Сделано по мотивам [этого](http://learn-android.ru/news/sozdaem_dvojnuju_animaciju_voskhodjashhego_solnca_i_idushhikh_chasov/2015-01-20-27.html?p=http://code.tutsplus.com/tutorials/android-sdk-creating-a-simple-tween-animation--mobile-14898) урока. Но в нем нету кнопочки:), а она здесь реально нужна, постоянно перезапускать приложение, чтобы увидеть свою анимацию, не практично и не удобно. |