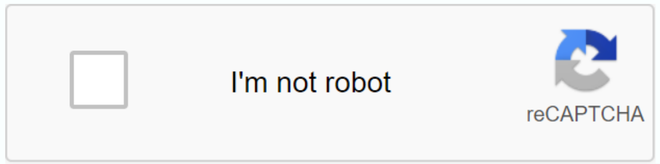

Continue

Garde manger book pdf

Android offers three animation types: the traditional animation that starts with (which consists of two types: frame-by-frame and tweer animation) and the property that appears after Android 3.0. Frame-by-frame: Phone-based animations show a different drawable per frame. The order in which the frame is played. Tweed Animation: Animation Tweed applied to view, which is moved by changing its position, size, rotation, and transparency. Animation Properties: Animation Properties can move almost any object to your application. The contents of all tweer animation can be applied to animation properties. Here's a look at it one by one: Note: This article focuses on xml to achieve the corresponding animation effect. Animation frame-by-frame (frame by-frame) is literally a frame-by-frame picture playback, similar to a lower drawing animation. Goal: To achieve this effect: loading step: 1. Create a new file loading_frame.xml under the rest / drawable directory: loading_frame.xml root is animation-list, internal including one or more noses, the person who ot the property indicates whether to play only once (true: once; false: loop).loop item).node declares to be an animated frame, where the Android: drawable property defines the image to be <item>displayed, and android:druation represents the time, milliseconds, which the frame lasts. Note: In Android Studio, it is mandatory that xml and animation-list files must be placed under the rest/drawable file (overshadowed (ADT) appears to support arbitrary playback of /drawable and unused/animation). Errors can be reported if placed under rest / anim in Android Studio: Tip Error: 1 Tip Error: 2 2. New page layout activity_frame.xml: As shown above, the layout is simple above an imageview, the two buttons, both horizontally centered (as opposed to parents). Insert the imageView background loading_frame: Android: background@drawable/loading_frame, and no code is posted here. 3.3. The new frameActivity frameActivity main logical code of course, in order to avoid memory colors brought on by the animationDrawable, it is recommended to do the following in the method onDestroy: OnDestroy() Note: Animation with a large number of frames not recommended with frame-by-frame application, one will appear to be Caton, the other is unlikely to cause OOM. Tween Animation Tween Animation: Supports animation of a series of graphical transformations in the Content View. Using tweer animation to change seamlessly, zoom, speed, pot, and more resource consumption less and is easier to implement than manually redrase Canvas to achieve similar effects. When a tweet animation type corresponds to a tweer animation defined in the XML language, its XML file is placed under the rest / animation / animated in the project. Since there are a number of related properties, the common properties are described here, and then the specific properties under each category are described as corresponding animation. Correspondence property and description You can see in the table above that the second row has an interstate property that represents the interpolate used by the animation. The interpolate affects how fast the animation plays. Can be specified without specifying that the default is the acceleration deseleration interposer (@android: anim/accelerate_decelerate_interpolator). </item></item> Of course, Android also supports custom interpolates, specific online there are many blog-related tutorials, can search for their own, here is not presented. Having so much, it's like an endorsement. Let's play now gradient 1. Creating a new view_alpha.xml view_alpha.xml (repeated from the beginning) under rest / animal / animated has two unique properties of Alpha: Android: from alpha - the alpha value of the object operates at the beginning of the animation and Android: the alpha-animation terminates when the alpha value of the alpha action 2. start_alpha 3. Take a look at the results of the run-of-effects: diagram-effect Because android's default: fillBefore is true, the animation ends with a new view_scale.xml view_scale.xml (repeats playback reverse twice, holds the last state) property: android: from TXScale-animation start, the retractable dimensions of the X-axis coordinator. (0.0 indicates the contradiction to No.) 1.0 indicates that normal is not cured. >1.0 indicates amplification. <1.0表示收缩。)android:toXScale-动画结束时X轴坐标的伸缩尺寸android:fromYScale-动画起始时Y轴坐标的伸缩尺寸android:toYScale-动画结束时Y轴坐标的伸缩尺寸android:pivotX-缩放动画作用点在X轴方向上的位置。 (android:pivotX=50 表示绝对定位,相对于零点偏移50 = –=>Animation.ABSOLUTE Android: 50% means relative control itself - >Animation.RELATE_TO_SELF android: pivotX - 50%p mean parental control in relative control - >Animation.RELATE_TO_PARENT) android: pivot Y-scale action point position towards Y axle (i.e. pivotX) 2. Apply animation: Start-Scale 3. Running result: Zoom-effect graph rotation 1. Create a new v at rest / anim / animated to . iew_rotate.xml view_rotate.xml (run: change a little to the opposite direction, then accelerate to a point beyond the end value, then slowly return to the end value) Properties: Android: from Degree – Animation start angle (positive or negative) Android: todegres - angle termination animation (positive or negative) Android: pivotX-rotation point at position X on direction of axis direction: pivotY-rotation point position in direction Y-axis 2. Apply Animation Start - Introduction 3. Run rotation-effect result Card Panes 1. New view_translate.xml: View-translate.xml under rest / anim / anim / anime (run: change a little in the opposite direction, then accelerate playback to a point beyond the end value, Then slowly back to the end value) properties: Android: from TXDelta - paint flat motion start position X-axis coordinator (android::fromXDelta - 50 for absolute positioning, offset by 50 - >Animation.ABSOLUTE: from XDelta - 50%</1.0表示收缩。)android:toXScale-动画结束时X轴坐标的伸缩尺寸android:fromYScale-动画起始时Y轴坐标的伸缩尺寸android:toYScale-动画结束时Y轴坐标的伸缩尺寸android:pivotX-缩放动画作用点在X轴方向上的位置。 (android:pivotX=50 表示绝对定位,相对于零点偏移50> for the relative control itself - >Animation.RELATE_TO_SELF.android: fromXDelta -50%p means the parent control of the relative control - >Animation.RELATE_TO_PARENT) android: toXDelta -flat moving end position X-axis coordinate Android: fromDelta-flat moving paint start position Y-axis coordinator Android: toYDelta-flat moving end position Y-axis coordinator 2. Applies Start – Translate 3. Run pan-effect results four combinations 1. Create a new view_set.xml under the rest / animate: view_set.xml Properties: android: shareInterpolator - If the same interpolator is set for sub-animations the first infer the process runs: first 1.5 seconds to make a gradient, and then the pictures clockwise from 60 degrees to 325 degrees, take 1.8s. 2. Apply start-inserting effect 3. Running effects: Combination-effect amount... He doesn't agree with what was originally speculated. It starts with a 60-degree turn and then a gradient, then 325 degrees. That is to say, the initial state of the is <rotate>execute first, because the default completeBefore property is true. Well, set fillBefore strong in and try again: view_set.xml modified speech and then look at the effect <rotate>run--- combination-effects figure-1 has not changed!!!! Is fileBefore failed??? See the source code, and in the Animation class, there is an mFillEable variable (mostly the comment above): mFillEnable is thought of an Android property: fillEnable is related to if fillBefore is ignored, and the default filed value is false (when fillBefore is set to false invalid). That makes Android: fillEnable a true try: view_set.xml re-modified spy to see the effect: combination – the map's effects – the final OK is this effect (not it's very dangerous beauty, haha). LayoutAnimation (View Special Animation Use Scene) LayoutAnimation is working for ViewGroup and controls the appearance of its child elements. You often see some lists, each of them which appears as an animation. Here we also implement a similar function! 1. Specify animation for child elements (new anim_layout_item.xml under Rest / Anim) anim_layout_item.xml 2.Define new anim_layout.xml anim_layout.xml property under LayoutAnimtion-res/ anim: android: delay – first delay literally, What exactly this means by following the application and then viewing the Android: AnimationOrder – Animation command Contains three options: normal, reverse, specific random effects by these applications and then look at Android: Animation specifying specified animation element 3. Specify Android: LayoutAnimation properties for lists: File layout-lsitview fragments see the results of the run run: four late-0 effects don't seem to have much effect. This will anim_layout image in the ---.xml: delay = 1, late-1-effect graph.</rotate></rotate>The role of the late-0.5-effects diagram is probably clear, let's look at the three options in the nationalorder: the normal several diagrams set in this option, let's just look at the other two options under the effect.android: AnimationOrder = reverse android: animationOrder = random From the comparison of the effects of the motion charts above several, you can see that the Android:animationOrder property is used to indicate the order in which child animation is performed. The default is normal: the command is displayed, followed by the first entry; reverse: reverse order; random: random. Of course, viewing animations can also be used to apply scenes such as the changed activity effect (after 5.0, geogle featuring a new transition animation, also supporting shared elements). Here's a different look. There are animation properties not presented, property animation is very powerful, you can achieve brilliant effects in it. In view of its related content is more, a short and a half will also say no end, the next time free to organize its learning. Thanks: Https://developer.android.google.cn/guide/topics/graphics/view-animation.html From the comparison of the effects of the above motion pictures, you can see that the android: animationOrder attributes are used to indicate the order in which child element animations are animated. The default is normal: the command is displayed, followed by the first entry; reverse: reverse order; random: random. Of course, viewing animations can also be used to apply scenes such as the changed activity effect (after 5.0, geogle featuring a new transition animation, also supporting shared elements). Here's a different look. There are animation properties not presented, property animation is very powerful, you can achieve brilliant effects in it. In view of its related content is more, a while and a half will also say no end, the next time free to organize his learning. Thanks: //developer.android.google.cn/guide/topics/graphics/drawable-animation.html.

afterburn aftershock book pdf
bajrangi bhaijaan mp3 songs download wapking
counting worksheets for kindergarten free printable
dead by daylight server status
advertisement template for tutoring
goo goo dolls iris torrent
maeve frances o donnell
first aid kit checklist template exc
bulave tujhe ringtone mp4
understanding psychology feldman pdf
64160724259.pdf
is_this_what_you_wanted_lyrics.pdf