iOS动画浅汇

2016-03-11 10:54 编辑：suiling 分类：iOS开发 来源：Love梅一生 博客 2 17542

UIView 动画 动画效果

在iOS开发中，制作动画效果是最让开发者享受的环节之一。一个设计严谨、精细的动画效果能给用户耳目一新的效果，吸引他们的眼光 —— 这对于app而言是非常重要的。我们总是追求更为酷炫的实现，如果足够仔细，我们不难发现一个好的动画通过步骤分解后本质上不过是一个个简单的动画实现。本文就个人搜集的一些动画相关的理论和实践知识做个小结，不足之处请勿见怪。

理论 UIview VS Ullayer

UIView只是CALyer之上的封装，更准确的来说，UIView是CALyer的简版封装，加上事件处理的集合类。 CALayer是QuartzCore库内的类，是iOS上最基本的绘制单元。其次，我们知道iOS平台的Cocoa Touch 是源于OS X平台的Cocoa)，是在Cocoa的基础上添加了适用于移动手机设备的手势识别、动画等特性；但从底层实现上来说，Cocoa Touch与Cocoa共用一套底层的库，其中就包括了QuartCore.framework；但QuartCore.framework一开始就是为OS X设计的，所以其中有部分特性是不适合做移动设备开发的，比如最重要的坐标系统。因此，我们也就不难理解为何UIView/NSView在CALayer上做了一层封装。

基于UIView实现的动画

简单的Block动画

```
[UIView animateWithDuration:0.2 animations:^{
    //
}];

[UIView animateWithDuration:0.5f animations:^{  } completion:^(BOOL finished)
 {
    //
 }];
```

热门资讯


源码：自己用Python写的iOS项目自动打包
点击量 37089


Runtime那些事儿（消息机制）
点击量 13740


iOS 高效添加圆角效果实战讲解
点击量 12976


每个Xcode开发者应该知道的七个使用技巧
点击量 12546


EXC_BAD_ACCESS的本质详解以及僵尸模式
点击量 7702

iOS开发中如何合理地制造BUG
点击量 7652


动画篇-layout动画初体验
点击量 6345


Apple Pay接入详细教程
点击量 5358


iOS 视图控制器转场详解
点击量 5099


一个减少App Store审核被拒的小技巧--善用
点击量 4627

综合评论

可以试试jenkins
http://jenkins-ci.org
flyfire 评论了 源码：自己用Python写的iOS项目自动打包脚本...

mark----
lqve521 评论了 在写一个iOS应用之前必须做的7件事(附相关资源)...

可以，感谢分享
yzhold 评论了 iOS 玩转推送通知

导航自定义跳转：https://github.com/Soul-Beats/UINavigationController-
348162291 评论了 源码推荐(03.16)：按钮间隔防连续点击，输入金...

很好的文章 点个赞
tangguoan 评论了 iOS 玩转推送通知

good thx your share . mark

UIView

```
UIView *headBgView = [[UIView alloc]init];
headBgView.frame = CGRectMake(0, 100, 50, 50);
[UIView beginAnimations:nil context:nil];
[UIView setAnimationDuration:1.0f];
headBgView.frame = CGRectMake(100, 200, 50, 50);
[UIView commitAnimations];
[self.view addSubview:headBgView];
```

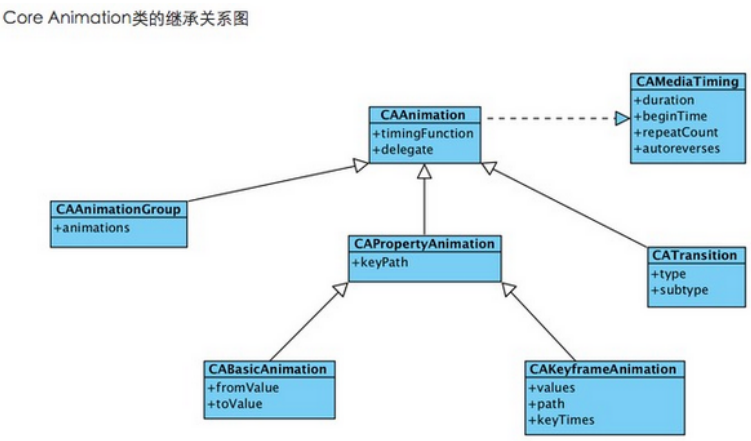
弹性动画

```
[UIView animateWithDuration:0.8
                      delay:0
     usingSpringWithDamping:0.5
      initialSpringVelocity:6
                    options:UIViewAnimationOptionCurveEaseOut
                 animations:^{
    //
} completion:^(BOOL finished) {  }]; //参数：持续时间，延迟时间，阻尼系数，初始速度
```

关键帧动画（中间可以添加合适多的帧来做不同的衔接动画）

```
[UIView animateKeyframesWithDuration:3.f
                               delay:0.0
                             options:UIViewKeyframeAnimationOptionCalculationModeLinear
                          animations:^{
[UIView addKeyframeWithRelativeStartTime:0.0
                        relativeDuration:0.8
                              animations:^{
                                //
                              }];
[UIView addKeyframeWithRelativeStartTime:0.8
                        relativeDuration:1.6
                              animations:^{
                                //
                              }];                                                    }
   completion:^(BOOL finished) {
                                //
    }];
```

CALayer动画



常用属性

duration : 动画的持续时间

beginTime : 动画的开始时间

repeatCount : 动画的重复次数

autoreverses : 执行的动画按照原动画返回执行

timingFunction : 控制动画的显示节奏，系统提供五种值选择， 分别是

- kCAMediaTimingFunctionLinear 线性动画
- kCAMediaTimingFunctionEaseIn 先慢后快（慢进快出）
- kCAMediaTimingFunctionEaseOut 先块后慢（快进慢出）
- kCAMediaTimingFunctionEaseInEaseOut 先慢后快再慢
- kCAMediaTimingFunctionDefault 默认，也属于中间比较快

path：关键帧动画中的执行路径

type：过渡动画的动画类型，系统提供了四种过渡动画：

- kCATransitionFade 渐变效果
- kCATransitionMoveIn 进入覆盖效果
- kCATransitionPush 推出效果
- kCATransitionReveal 揭露离开效果

subtype : 过渡动画的动画方向

- kCATransitionFromRight 从右侧进入
- kCATransitionFromLeft 从左侧进入
- kCATransitionFromTop 从顶部进入
- kCATransitionFromBottom 从底部进入

基础动画主要提供了对于CALayer对象中的可变属性进行简单动画的操作。比如：位移、透明度、缩放、旋转、背景色等等。 重要属性 fromValue： keyPath对应的初始值 toValue： keyPath对应的结束值。

- 基础动画（CABaseAnimation） 0:1 1:0 实现下拉剪头的展开和收起

```
CABasicAnimation* rotationAnimation = [CABasicAnimation animationWithKeyPath:@"transform.rotation.z"];
rotationAnimation.duration = 0.3f;
rotationAnimation.repeatCount = 1;
rotationAnimation.autoreverses = NO;
rotationAnimation.fromValue = [NSNumber numberWithFloat: beginValue*M_PI];
rotationAnimation.toValue = [NSNumber numberWithFloat:endValue*M_PI];
rotationAnimation.removedOnCompletion = NO;
rotationAnimation.fillMode = kCAFillModeForwards;
rotationAnimation.timingFunction = [CAMediaTimingFunction functionWithName:kCAMediaTimingFunctionEaseInEaseOut];
[pinLayer addAnimation:rotationAnimation forKey:@"revItUpAnimation"];
```

- 关键帧动画（CAKeyframeAnimation） CAKeyframeAnimation和CABaseAnimation都属于CAPropertyAnimatin的子类。CABaseAnimation只能从一个数值（fromValue）变换成另一个数值（toValue）,而CAKeyframeAnimation则会使用一个NSArray保存一组关键帧。 重要属性 values： 就是上述的NSArray对象。里面的元素称为”关键帧”(keyframe)。动画对象会在指定的时间(duration)内，依次显示values数组中的每一个关键帧 path： 可以设置一个CGPathRefCGMutablePathRef,让层跟着路径移动。path只对CALayer的anchorPoint和position起作用。如果你设置了path，那么values将被忽略。 keyTimes： 可以为对应的关键帧指定对应的时间点,其取值范围为0到1.0,keyTimes中的每一个时间值都对应values中的每一帧.当keyTimes没有设置的时候,各个关键帧的时间是平分的。
- 组合动画：

```
CAKeyframeAnimation *anima1 = [CAKeyframeAnimation animationWithKeyPath:@"position"];
NSValue *value0 = [NSValue valueWithCGPoint:CGPointMake(0, SCREEN_HEIGHT/2-50)];
NSValue *value1 = [NSValue valueWithCGPoint:CGPointMake(SCREEN_WIDTH/3, SCREEN_HEIGHT/2-50)];
NSValue *value2 = [NSValue valueWithCGPoint:CGPointMake(SCREEN_WIDTH/3, SCREEN_HEIGHT/2+50)];
NSValue *value3 = [NSValue valueWithCGPoint:CGPointMake(SCREEN_WIDTH*2/3, SCREEN_HEIGHT/2+50)];
NSValue *value4 = [NSValue valueWithCGPoint:CGPointMake(SCREEN_WIDTH*2/3, SCREEN_HEIGHT/2-50)];
NSValue *value5 = [NSValue valueWithCGPoint:CGPointMake(SCREEN_WIDTH, SCREEN_HEIGHT/2-50)];
anima1.values = [NSArray arrayWithObjects:value0,value1,value2,value3,value4,value5, nil];
//缩放动画
CABasicAnimation *anima2 = [CABasicAnimation animationWithKeyPath:@"transform.scale"];
anima2.fromValue = [NSNumber numberWithFloat:0.8f];
anima2.toValue = [NSNumber numberWithFloat:2.0f];
//旋转动画
CABasicAnimation *anima3 = [CABasicAnimation animationWithKeyPath:@"transform.rotation"];
anima3.toValue = [NSNumber numberWithFloat:M_PI*4];
//组动画
CAAnimationGroup *groupAnimation = [CAAnimationGroup animation];
groupAnimation.animations = [NSArray arrayWithObjects:anima1,anima2,anima3, nil];
groupAnimation.duration = 4.0f;
[_demoView.layer addAnimation:groupAnimation forKey:@"groupAnimation"];
```

- 过渡动画（CATransition） 多数为私有的API使用后无法上架app。 私有API提供了其他很多非常炫的过渡动画，比如@”cube”、@”suckEffect”、@”oglFlip”、 @”rippleEffect”、@”pageCurl”、@”pageUnCurl”、@”cameraIrisHollowOpen”、@”cameraIrisHollowClose”等。

粒子动画

transform动画

transform是一个非常重要的属性，它在矩阵变换的层面上改变视图的显示效果，完成旋转、形变、平移等等操作。在它被修改的同时，视图的frame也会被真实改变。有两个数据类型用来表示transform，分别是CGAffineTransform和CATransform3D。前者作用于UIView，后者为layer层次的变换类型。基于后者可以实现更加强大的功能。 对于想要

了解矩阵变换是如何作用实现的，可以参考这篇博客： CGAffineTransform 放射变换

```
/// 用来连接两个变换效果并返回。返回的t = t1 * t2
CGAffineTransformConcat(CGAffineTransform t1, CGAffineTransform t2)
/// 矩阵初始值。[ 1 0 0 1 0 0 ]
CGAffineTransformIdentity
/// 自定义矩阵变换，需要掌握矩阵变换的知识才知道怎么用。参照上面推荐的原理链接
CGAffineTransformMake(CGFloat a, CGFloat b, CGFloat c, CGFloat d, CGFloat tx, CGFloat ty)
/// 旋转视图。传入参数为 角度 * (M_PI / 180)。等同于 CGAffineTransformRotate(self.transform, angle)
CGAffineTransformMakeRotation(CGFloat angle)
CGAffineTransformRotate(CGAffineTransform t, CGFloat angle)
/// 缩放视图。等同于CGAffineTransformScale(self.transform, sx, sy)
CGAffineTransformMakeScale(CGFloat sx, CGFloat sy)
CGAffineTransformScale(CGAffineTransform t, CGFloat sx, CGFloat sy)
/// 缩放视图。等同于CGAffineTransformTranslate(self.transform, tx, ty)
CGAffineTransformMakeTranslation(CGFloat tx, CGFloat ty)
CGAffineTransformTranslate(CGAffineTransform t, CGFloat tx, CGFloat ty)
```

在开始使用transform实现你的动画之前，我先介绍几个常用的函数：


```
/// 用来连接两个变换效果并返回。返回的t = t1 * t2
CGAffineTransformConcat(CGAffineTransform t1, CGAffineTransform t2)
/// 矩阵初始值。[ 1 0 0 1 0 0 ]
CGAffineTransformIdentity
/// 自定义矩阵变换，需要掌握矩阵变换的知识才知道怎么用。参照上面推荐的原理链接
CGAffineTransformMake(CGFloat a, CGFloat b, CGFloat c, CGFloat d, CGFloat tx, CGFloat ty)
/// 旋转视图。传入参数为 角度 * (M_PI / 180)。等同于 CGAffineTransformRotate(self.transform, angle)
CGAffineTransformMakeRotation(CGFloat angle)
CGAffineTransformRotate(CGAffineTransform t, CGFloat angle)
/// 缩放视图。等同于CGAffineTransformScale(self.transform, sx, sy)
CGAffineTransformMakeScale(CGFloat sx, CGFloat sy)
CGAffineTransformScale(CGAffineTransform t, CGFloat sx, CGFloat sy)
/// 缩放视图。等同于CGAffineTransformTranslate(self.transform, tx, ty)
CGAffineTransformMakeTranslation(CGFloat tx, CGFloat ty)
CGAffineTransformTranslate(CGAffineTransform t, CGFloat tx, CGFloat ty)
```


transform严格的说不是一种动画，而是动画中的一部分操作，我拿出来说是因为它同时出现在了UIView 动画和CALayer动画中。

一些应用

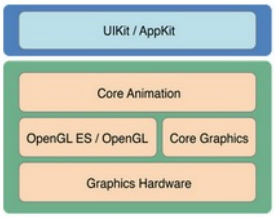
利用上面CALayer 基础动画的代码实现下拉剪头的展开和收起，还可以实现时钟指针的旋转

```
CALayer  *pinLayer = [CALayer layer];
[pinLayer setBounds:CGRectMake(0, 0, 14, 8)];
[pinLayer setContents:(id)[UIImage imageNamed:@"downArrow.png"].CGImage];
[pinLayer setPosition:CGPointMake(305,20)];
[pinLayer setAnchorPoint:CGPointMake(0.5, 0.5)];
[headBgView.layer addSublayer:pinLayer];
```

输入框在输入错误信息时的摇晃效果。

利用CAShapeLayer 和CABasicAnimation 可以实现加载动画。

iOS渲染视图的层级图：



更多内容可以点击 iOS开发UI篇--iOS动画（Core Animation）总结。

事实告诉我们动画是要靠设计的，你看我上面的动画抽的一笔，但事实上用同样的代码可以写出很漂亮的动画。

上一篇：源码推荐(03.11B)：LOL盒子上的皮肤展示效果轮播图，K线图

相关资讯

动画的微妙之处
iOS 视图控制器转场详解
源码推荐(02.29)：QQ最新版点击头像跳出资料卡片，咻
源码推荐(02.19B)：多个TableView关联，仿支付宝咻一咻
源码推荐(02.15)：模仿QQ弹出视图，tableView和
源码推荐(03.09B)：仿钉钉毛玻璃菜单动画，循环播放文
动画篇-layout动画初体验
源码推荐(02.22B)：模仿QQ弹出视图，tableView和
源码推荐(02.16B)：模仿简书、淘宝弹出效果动画，GIF文
iOS 开发之动画篇 - Transform和KeyFrame动画

我来说两句

您还没有登录！请 登录 或 注册

所有评论（2）

panlong 2016-03-14 13:56:45

Mark

0 0 回复

折尽尘沙 2016-03-12 14:37:09

非常有用~~~感觉思路清晰了很多.Mark

0 0 回复