# iPhone与iPad应用开发课程
# 精通iOS开发

## 第十五讲 动画

主讲人：关东升

eorient@sina.com

# 主要知识点

- 动画介绍
- Core Animation基础
- 图层
- 隐式动画
- 显示动画
- 帧动画

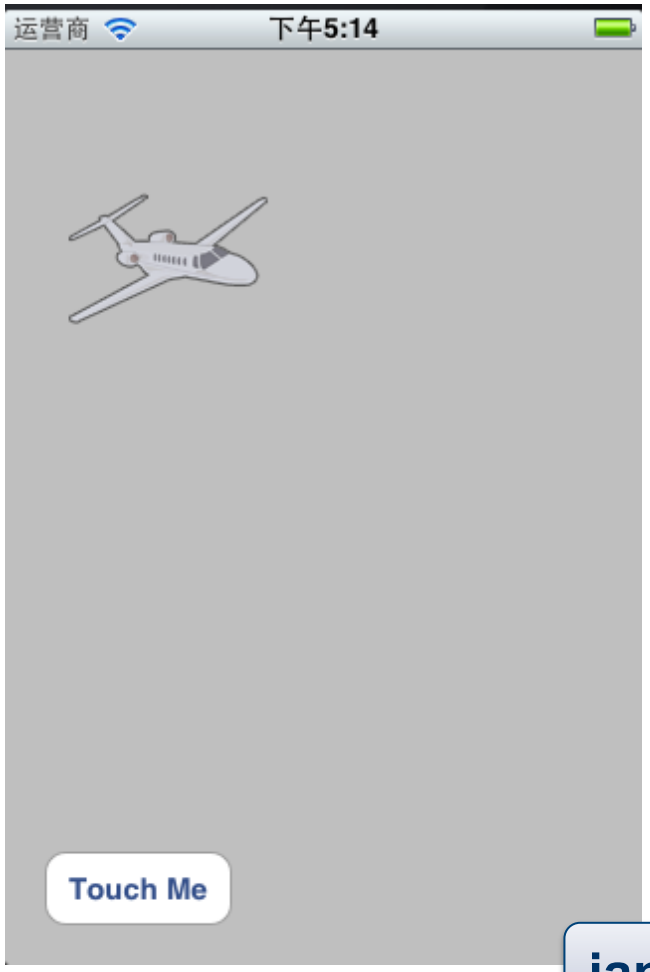
# 动画介绍

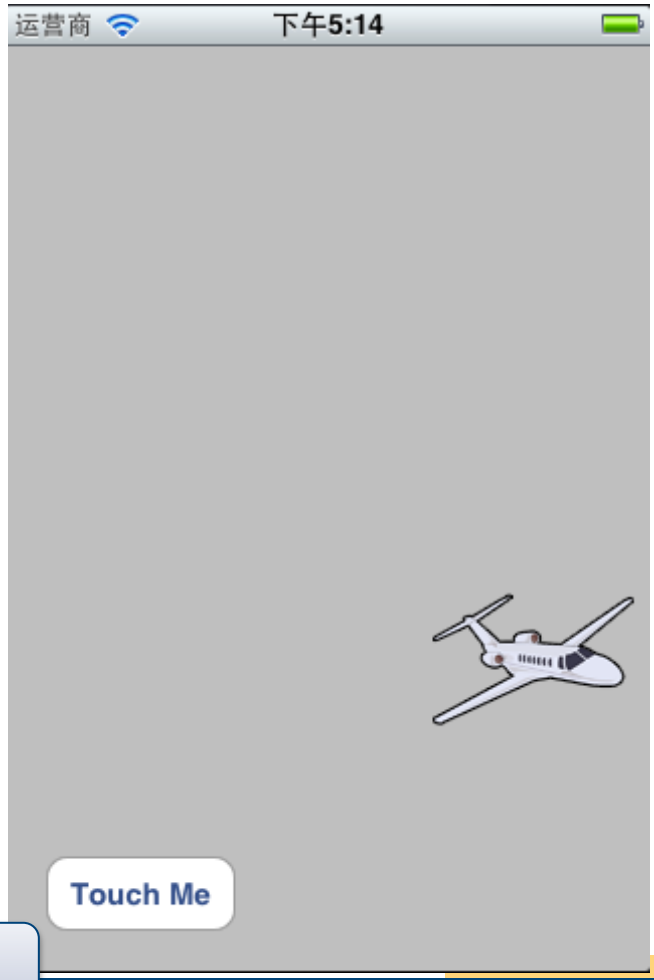
- 在iOS中动画实现技术主要是：Core Animation。
- Core Animation负责所有的滚动、旋转、缩小和放大以及所有的iOS动画效果。其中UIKit类通常都有animated：参数部分，它可以允许是否使用动画。
- Core Animation还与Quartz紧密结合在一起，每个UIView都关联到一个CALayer对象，CALayer是Core Animation中的图层。

# Core Animation基础

- Core Animation创建动画时候会修改CALayer属性，然后让这些属性流畅地变化。Core Animation相关知识点：
  - 图层，图层是动画发生的地方，CALayer总是与UIView关联，通过layer属性访问。
  - 隐式动画，这是一种最简单的动画，不用设置定时器，不用考虑线程或者重画。
  - 显式动画，是一种使用CABasicAnimation创建的动画，通过CABasicAnimation，可以更明确地定义属性如何改变动画。
  - 关键帧动画，这是一种更复杂的显式动画类型，这里可以定义动画的起点和终点，还可以定义某些帧之间的动画。

# 隐式动画实例




ianimate

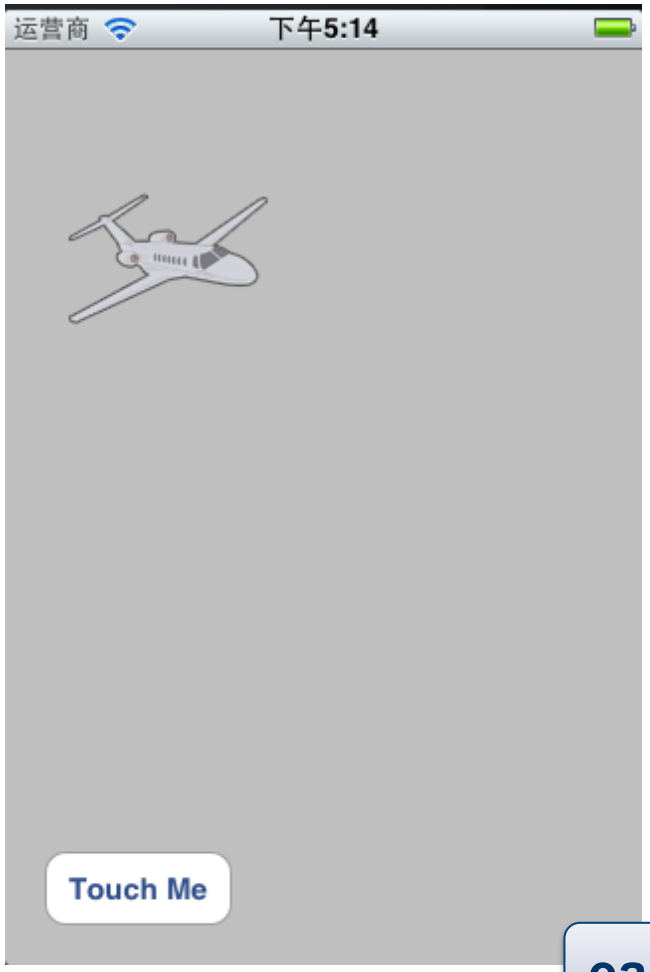
# 关键代码

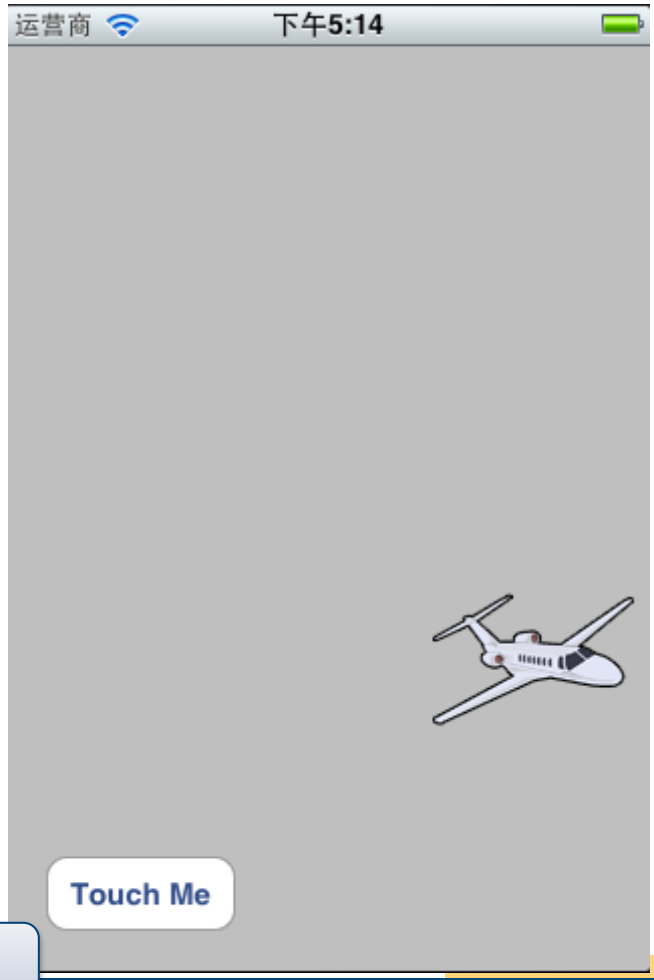
```
-(IBAction)movePlane:(id)sender {

    [UIView beginAnimations:nil context:NULL];
    CGAffineTransform moveTransform
                = CGAffineTransformMakeTranslation(180, 200);
    [plane.layer setAffineTransform:moveTransform];
    plane.layer.opacity = 1;
    [UIView commitAnimations];

}
```

# 说明

- 飞机图片的不透明度（opacity）初始为0.25，然后在动画过程中不透明度设置为1.0。这个过程是设置飞机图片对象的层属性的，plane.layer.opacity = 1;
- [plane.layer setAffineTransform:moveTransform];设置飞机图片层对象的仿射移动变换。

# 显式动画实例




eanimate

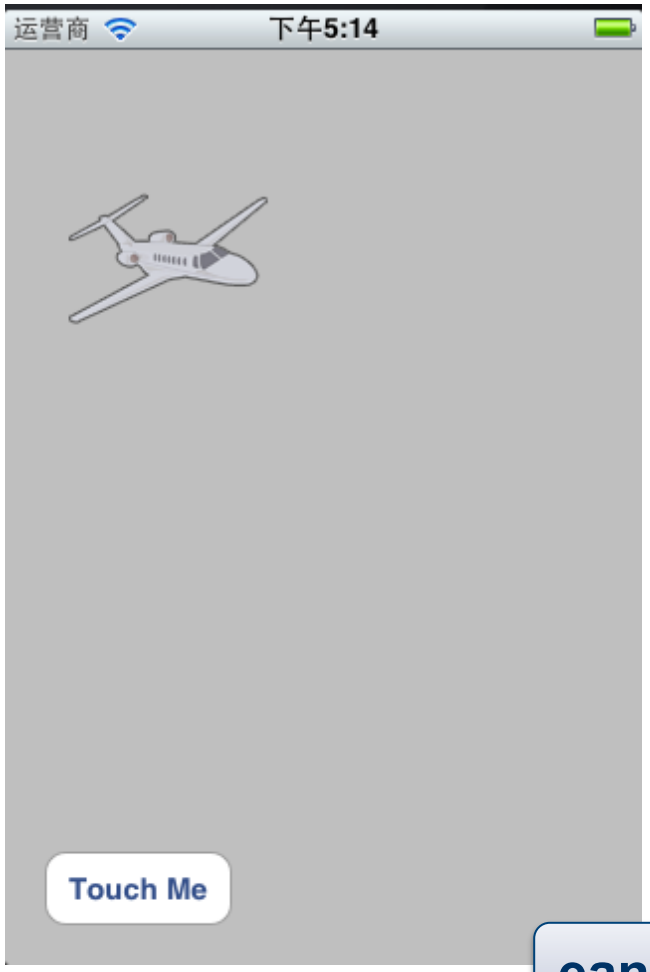
# 关键代码

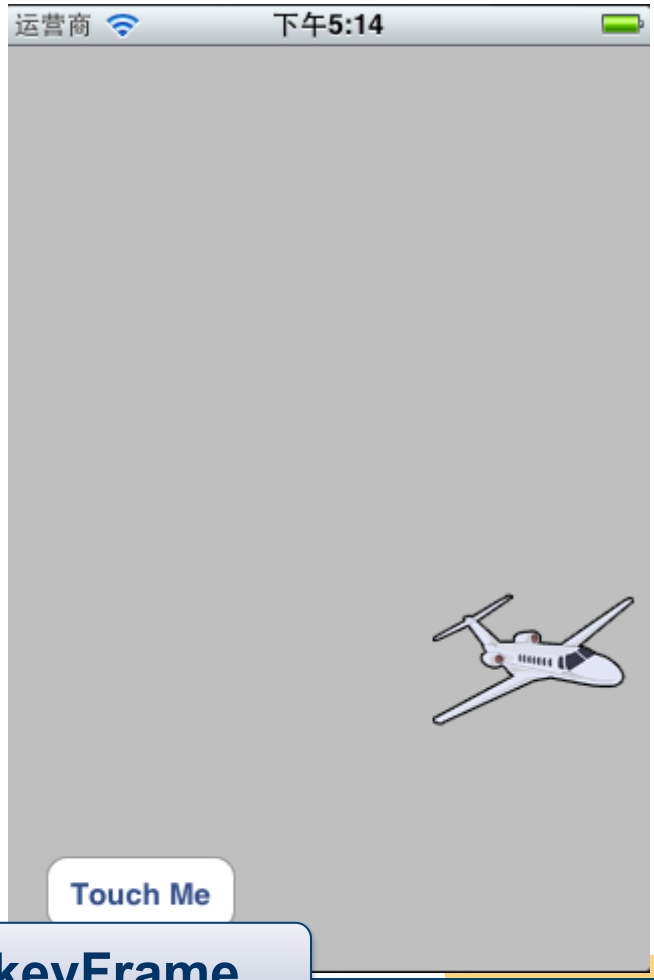
```
-(IBAction)movePlane:(id)sender {
    CABasicAnimation *opAnim = [CABasicAnimation animationWithKeyPath:@"opacity"];
    opAnim.duration = 3.0;
    opAnim.fromValue = [NSNumber numberWithFloat:.25];
    opAnim.toValue= [NSNumber numberWithFloat:1.0];
    opAnim.cumulative = YES;
    opAnim.repeatCount = 2;
    [plane.layer addAnimation:opAnim forKey:@"animateOpacity"];

    CGAffineTransform moveTransform = CGAffineTransformMakeTranslation(180, 200);
    CABasicAnimation *moveAnim = [CABasicAnimation animationWithKeyPath:@"transform"];
    moveAnim.duration = 6.0;
    moveAnim.toValue= [NSValue valueWithCATransform3D:
        CATransform3DMakeAffineTransform(moveTransform)];
    [plane.layer addAnimation:moveAnim forKey:@"animateTransform"];
}
```

# 说明

- 显式动画时候，不必定义CALayer的变化，也不必执行它们，而是通过CABasicAnimation逐个定义动画。其中每个动画都含有各自的duration、repeatCount等属性。然后，使用addAnimation:forKey:方法分别将每个动画应用到层中。
- [CABasicAnimation animationWithKeyPath:@“opacity”]获得透明度动画对象，@“transform”是指定转换动画。
- opAnim.cumulative 属性是指定累计
- opAnim.repeatCount 重复执行次数
- CATransform3DMakeAffineTransform函数是将仿射变换矩阵变成Core Animation使用的Transform3D类型的矩阵。

# 关键帧显式动画实例



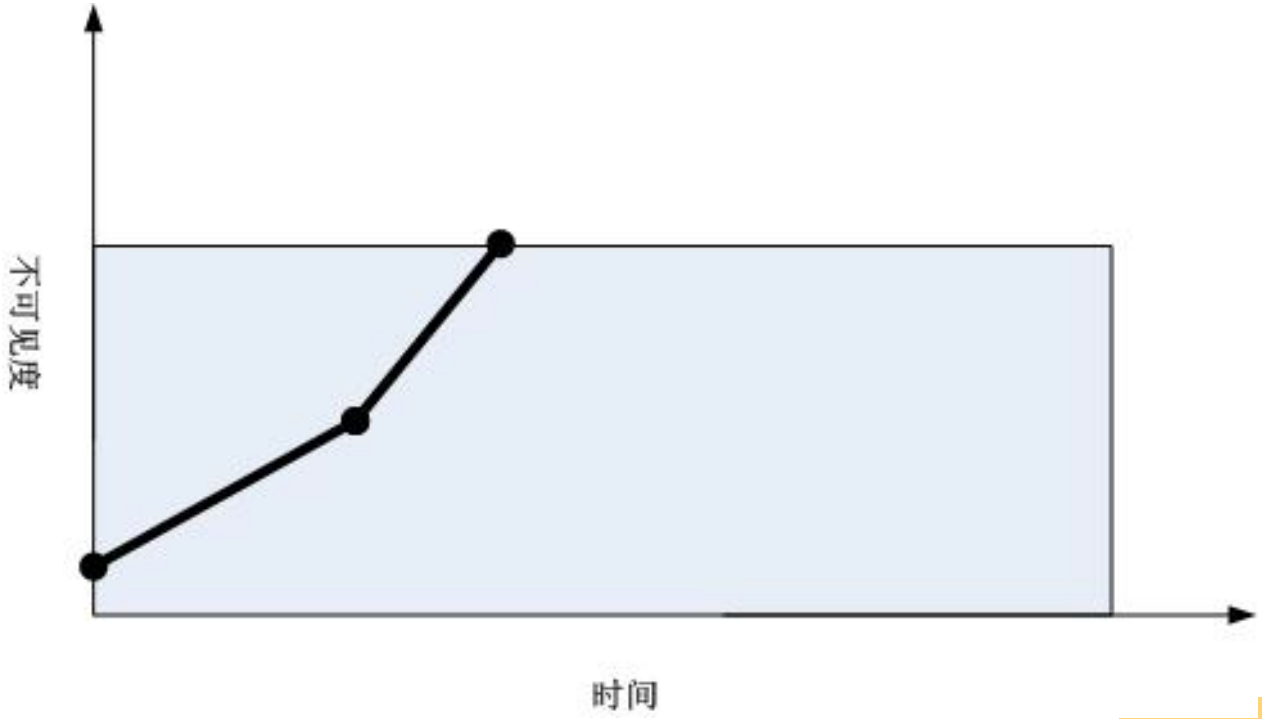




eanimate_keyFrame

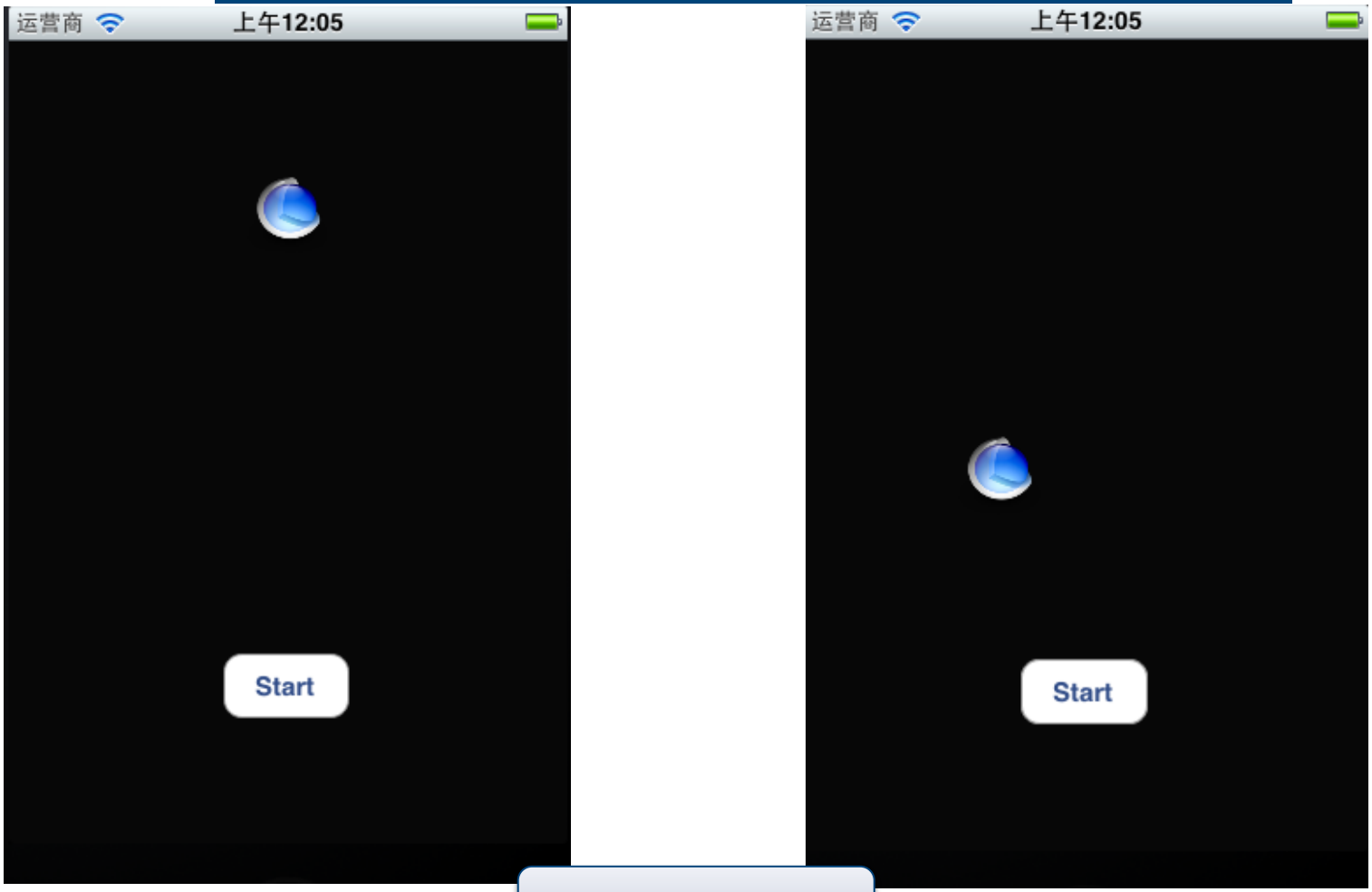
```
-(IBAction)movePlane:(id)sender {

    CAKeyframeAnimation *opAnim = [CAKeyframeAnimation animationWithKeyPath:@"opacity"];
    opAnim.duration = 6.0;
    opAnim.values =[NSArray arrayWithObjects:
                        [NSNumber numberWithFloat:0.25],
                        [NSNumber numberWithFloat:0.75],
                        [NSNumber numberWithFloat:1.0],
                        nil];
    opAnim.keyTimes = [NSArray arrayWithObjects:
                          [NSNumber numberWithFloat:0.0],
                          [NSNumber numberWithFloat:0.5],
                          [NSNumber numberWithFloat:1.0], nil];

    [plane.layer addAnimation:opAnim forKey:@"animateOpacity"];

    CGAffineTransform moveTransform = CGAffineTransformMakeTranslation(180, 200);
    CABasicAnimation *moveAnim = [CABasicAnimation animationWithKeyPath:@"transform"];
    moveAnim.duration = 6.0;
    moveAnim.toValue= [NSValue valueWithCATransform3D:
                          CATransform3DMakeAffineTransform(moveTransform)];
    [plane.layer addAnimation:moveAnim forKey:@"animateTransform"];
}
```

# 说明

- animation.values是一个值的数组。
- animation.keyTimes是一个每个帧片段持续的时间比例，取值范围0.0-1.0之间。



# 关键帧之路径实例




BallSteps

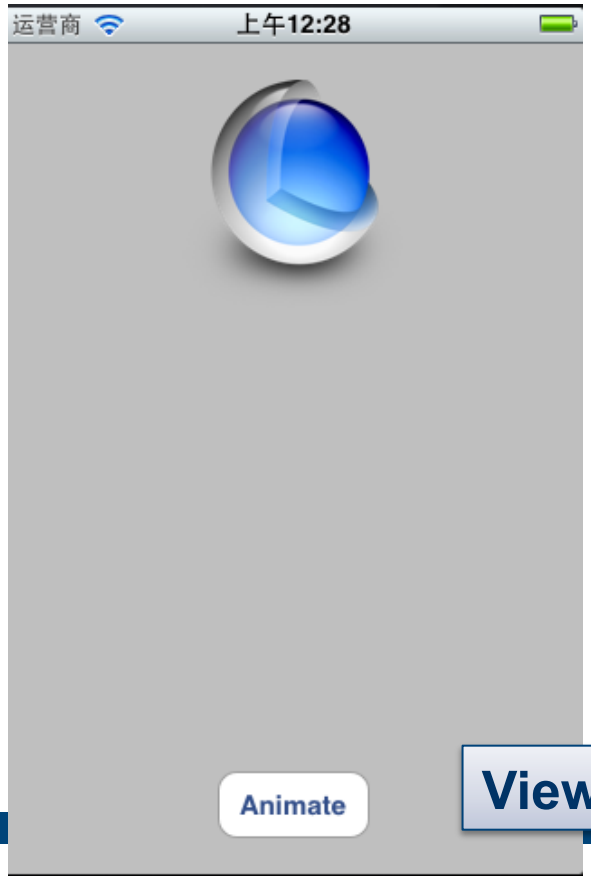
```
- (IBAction)drawStar:(id)sender {
        [drawButton setEnabled:NO];
        CGMutablePathRef starPath = CGPathCreateMutable();
        CGPathMoveToPoint(starPath,NULL,160.0f, 100.0f);
        CGPathAddLineToPoint(starPath, NULL, 100.0f, 280.0f);
        CGPathAddLineToPoint(starPath, NULL, 260.0, 170.0);
        CGPathAddLineToPoint(starPath, NULL, 60.0, 170.0);
        CGPathAddLineToPoint(starPath, NULL, 220.0, 280.0);
        CGPathCloseSubpath(starPath);

        CAKeyframeAnimation *animation = nil;
        animation = [CAKeyframeAnimation animationWithKeyPath:@"position"];
        [animation setDuration:10.0f];
        [animation setDelegate:self];
        [animation setPath:starPath];
        CFRelease(starPath);
        starPath = nil;
        [[imageView layer] addAnimation:animation forKey:@"position"];
}

- (void)animationDidStop:(CAAnimation *)theAnimation finished:(BOOL)flag {
        [drawButton setEnabled:YES];
}
```

# 说明

- [CAKeyframeAnimation animationWithKeyPath:@“position”];创建一个position类型的关键帧动画。
- 关键帧动画中可以定义路径，把这些路径放入到CGMutablePathRef 中与CG中的很相似。
- CGPathCloseSubpath(starPath);结束路径。
- [animation setPath:starPath];设置路径。
- [animation setDelegate:self];设置委托对象为本身，即回调方法 animationDidStop:finished:。
- 最后CFRelease(starPath);时候路径。

# UIView级别动画

- 除了直接使用Core Animation 层实现动画，我们还有UIView直接实现隐式动画。



# h文件

```
@interface MainViewController : UIViewController
{
  UIImageView *animImageView;
  UIButton *button;
}

@property (assign) IBOutlet UIImageView *animImageView;
@property (assign) IBOutlet UIButton *button;

- (IBAction)action:(id)sender;

@end
```

```
- (IBAction)action:(id)sender {
  [UIView beginAnimations:@"Hide Button" context:nil];
  [[self button] setAlpha:0.0];
  [UIView commitAnimations];

  [UIView beginAnimations:@"Slide Around" context:nil];
  [UIView setAnimationDuration:1.0];
  [UIView setAnimationDelegate:self];
  [UIView setAnimationDidStopSelector:@selector(viewAnimationDone:)];
  [UIView setAnimationRepeatCount:3];
  [UIView setAnimationRepeatAutoreverses:YES];
  CGPoint center = [[self animImageView] center];
  center.y += 100;
  [[self animImageView] setCenter:center];

  [UIView commitAnimations];
}

- (void)viewAnimationDone:(NSString*)name {
  [UIView beginAnimations:@"Show Button" context:nil];
  [[self button] setAlpha:1.0];
  [UIView commitAnimations];
}
```

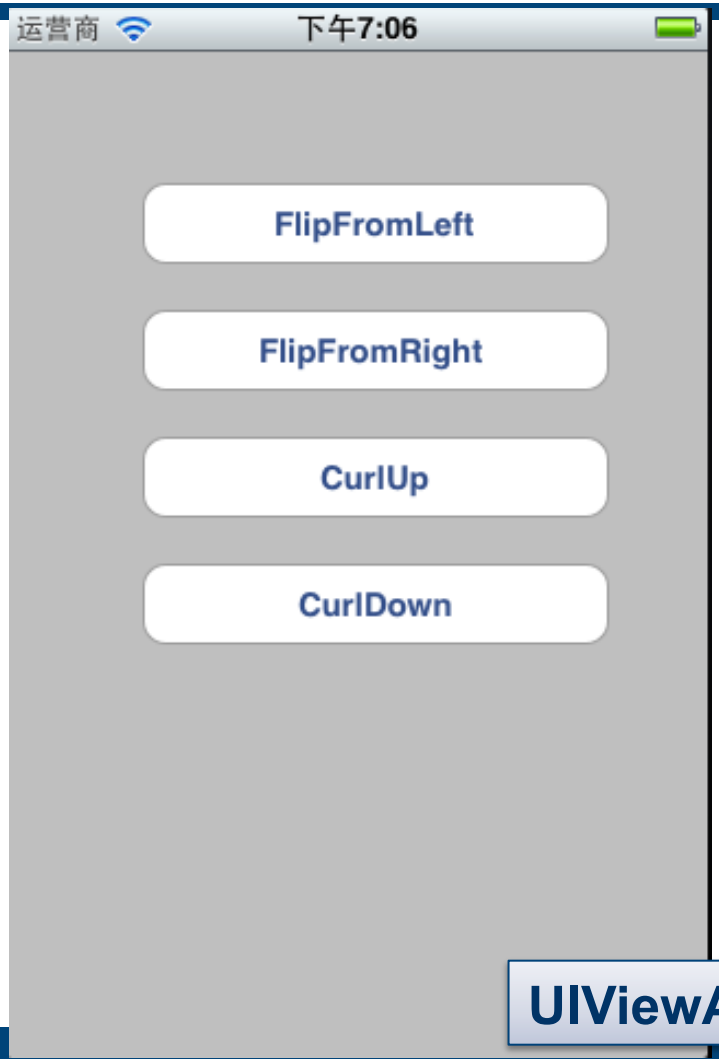
# 说明

- UIView中的动画是在动画块中定义的，动画块是UIView beginAnimations:context:开始， 在UIView commitAnimations结束。
- 首先开始将按钮设置透明度为0的，结果是开始动画时候隐藏了。
- 然后，又开始新的动画中设置委托事件：
- [UIView setAnimationDelegate:self]
- [UIView setAnimationDidStopSelector:@selector (viewAnimationDone:)];
- 当动画结束的时候调用viewAnimationDone:方法。

# 内置UIView动画

- UIView具有一个UIViewAnimationTransition属性可以设定动画，这些动画是iOS提供几个常用动画有：
  - UIViewAnimationTransitionNone
  - UIViewAnimationTransitionFlipFromLeft
  - UIViewAnimationTransitionFlipFromRight
  - UIViewAnimationTransitionCurlUp
  - UIViewAnimationTransitionCurlDown

# 实例




UIViewAnimation

```
- (IBAction)doUIViewAnimation:(id)sender{
        [UIView beginAnimations:@"animationID" context:nil];
        [UIView setAnimationDuration:0.5f];
        [UIView setAnimationCurve:UIViewAnimationCurveEaseInOut];
        [UIView setAnimationRepeatAutoreverses:NO];
        UIButton *theButton = (UIButton *)sender;
        switch (theButton.tag) {
                case 1:
                        [UIView setAnimationTransition:UIViewAnimationTransitionFlipFromLeft forView:self.view cache:YES];
                        break;
                case 2:
                        [UIView setAnimationTransition:UIViewAnimationTransitionFlipFromRight forView:self.view cache:YES];
                        break;
                case 3:
                        [UIView setAnimationTransition:UIViewAnimationTransitionCurlUp forView:self.view cache:YES];
                        break;
                case 4:
                        [UIView setAnimationTransition:UIViewAnimationTransitionCurlDown forView:self.view cache:YES];
                        break;
                default:
                        break;
        }
        //[self.view exchangeSubviewAtIndex:1 withSubviewAtIndex:0];
        [UIView commitAnimations];
}
```

# 说明

- [UIView setAnimationCurve:UIViewAnimationCurveEaseInOut]设置动画曲线，动画曲线指定的是动画进入和退出的方式，它也有几个常量：
  - UIViewAnimationCurveEaseInOut
  - UIViewAnimationCurveEaseIn
  - UIViewAnimationCurveEaseOut
  - UIViewAnimationCurveLinear
- setAnimationTransition: forView: cache:方法第一个参数定义动画类型，第二个参数是当前视图对象，第三个参数上使用缓冲区。