类说明：

AppDelegate

程序入口

ViewController

第一个controller的view装载SHShadowedLayer和SMTransformLayer 并接收交互，进行形变和逛变

主要函数

```
//这两个函数都是根据手所在的位置和中心的距离和整个view的宽和高的比例作为系数，去调整3D旋转的角度。
- (void)pan:(UIPanGestureRecognizer*)pan
{
CGPoint location = [pan locationInView:self.view];
CGSize viewSize = self.view.bounds.size;
location = CGPointMake((location.x - viewSize.width / 2) / viewSize.width,
                       (location.y - viewSize.height / 2) /
viewSize.height);
[CATransaction begin];
[CATransaction setDisableActions:YES];
transformLayer.transform =
CATransform3DRotate(CATransform3DMakeRotation(M_PI * location.x, 0, 1, 0), -
M_PI * location.y, 1, 0, 0);
// if I weren't using SMShadowedTransformLayer, here I would simply do:
// [layer setTransform:layer.transform animatePerFrame:YES];
[CATransaction commit];
}

- (void)tap:(UITapGestureRecognizer*)tap
{
CGPoint location = [tap locationInView:self.view];
CGSize viewSize = self.view.bounds.size;
location = CGPointMake((location.x - viewSize.width / 2) / viewSize.width,
                       (location.y - viewSize.height / 2) /
viewSize.height);
[CATransaction begin];
[CATransaction setAnimationDuration:1.f];
transformLayer.transform =
CATransform3DRotate(CATransform3DMakeRotation(M_PI * location.x, 0, 1, 0), -
M_PI * location.y, 1, 0, 0);
// if I weren't using SMShadowedTransformLayer, here I would simply do:
// [layer setTransform:layer.transform animatePerFrame:YES];
[CATransaction commit];
}
```

SHShadowedLayer

它是负责光效果的。负责维护两个**CALayer *shadowLayer; CALayer *specularLayer;** 同时维护

- @property (nonatomic, assign) BOOL showShadow;
- @property (nonatomic, strong) UIColor *shadowingColor;
- @property (nonatomic, assign) float currentShadowOpacity;
- @property (nonatomic, assign) BOOL showHighlight;
- @property (nonatomic, strong) UIColor *specularColor;
- @property (nonatomic, assign) float specularFalloff;

- @property (nonatomic, assign) float currentSpecularOpacity;

而这些属性是**shadowLayer**和**specularLayer**共有的。这两个layer只负责颜色而已。实际上两个layer颜色各自负责迎光和斜光。迎光显示白色，斜光导致开始变黑。

主要函数

```
//这个函数是为了同步两个layer的mask属性
- (CALayer*)constructShadowMask
{
if (!self.contents)
    return nil;
CALayer *mask = [CALayer layer];
mask.frame = shadowLayer.bounds;
mask.contents = self.contents;
mask.contentsRect = self.contentsRect;
mask.contentsCenter = self.contentsCenter;
mask.contentsGravity = self.contentsGravity;
mask.contentsScale = self.contentsScale;

return mask;
}
```

SMTransformLayer

继承自CATransformLayer，它是特殊的不持有很多平面特性的layer，但是它具有perspective的效果。

维护的属性有

- @property (nonatomic, readonly) NSMutableSet *shadowedLayers;

这个属性是自己的sublayer里的shadowLayer的交集。

因为这个东西可一进行三维旋转