绘制圆形进度条

1. 简述：

项目需求，在 banner 上放置一个圆形进度条，随 scrollView 拖拽比例显示进度，在一定拖拽距离后进度条开始作循环转动并进行页面刷新。

2. 实现：

(1) MYColorfulCircleProgressView

```
@interface MYColorfulCircleProgressView : UIView

/**
 * 动画状态
 */
@property (assign, nonatomic, readonly) BOOL isLoading;

/**
 * 自动刷新
 * @ note 在进度达到1时自动执行动画，默认值为YES
 */
@property (assign, nonatomic) BOOL autoAnimating;

/**
 * 设置进度
 *
 * @ param 进度条进度 范围: 0-1
 */
- (void)setProgress:(float)progress;

/**
 * 开始动画
 *
 */
- (void)beginAnimating;
```

```
/**
 * 结束动画
 *
 */
- (void)stopAnimating;

@end
```

```
@interface MYColorfulCircleProgressView ()

@property (strong, nonatomic) CAShapeLayer *backgroundLayer;
@property (strong, nonatomic) CAShapeLayer *cycleLayer;

@property (strong, nonatomic) CABasicAnimation *animation;

@end

@implementation MYColorfulCircleProgressView

- (instancetype)initWithCoder:(NSCoder *)aDecoder{
    if(self = [super initWithCoder:aDecoder]){
        [self layoutIfNeeded];
        [self initialState:self.frame];
    }
    return self;
}

- (instancetype)initWithFrame:(CGRect)frame{
    if(self = [super initWithFrame:frame]){
        [self  initialState:frame];
    }
    return self;
}
```

```
- (void)initialState:(CGRect)frame{
    if(frame.size.width != frame.size.height){
        self.frame = CGRectMake(frame.origin.x, frame.origin.y, frame.size.width, frame.size.width);
    }
    _backgroundLayer = [CAShapeLayer layer];
    _backgroundLayer.bounds = CGRectMake(2, 2, frame.size.width - 4, frame.size.height - 4);
    _backgroundLayer.position = CGPointMake(frame.size.width/2, frame.size.height/2);
    _backgroundLayer.strokeColor = [UIColor grayColor].CGColor;
    _backgroundLayer.fillColor = [UIColor clearColor].CGColor;
    _backgroundLayer.lineCap = kCALineCapRound;
    _backgroundLayer.allowsEdgeAntialiasing = YES;
    [_backgroundLayer setLineWidth:2];
    [self.layer addSublayer:_backgroundLayer];
    UIBezierPath *path = [UIBezierPath bezierPathWithArcCenter:_backgroundLayer.position radius:self.frame.size.width/2 - 2 startAngle:-M_PI_2 endAngle:2 *
        M_PI - M_PI_2 clockwise:YES];
    [_backgroundLayer setPath:path.CGPath];
    _backgroundLayer.opacity = 0;
```

```
_cycleLayer = [CAShapeLayer layer];
_cycleLayer.bounds = CGRectMake(2, 2, frame.size.width - 4, frame.size.height - 4);
_cycleLayer.position = CGPointMake(frame.size.width/2, frame.size.height/2);
_cycleLayer.strokeColor = [UIColor redColor].CGColor;
_cycleLayer.fillColor = [UIColor clearColor].CGColor;
_cycleLayer.opacity = 1;
_cycleLayer.lineCap = kCALineCapRound;
_cycleLayer.allowsEdgeAntialiasing = YES;
[_cycleLayer setLineWidth:2];
[self.layer addSublayer:_cycleLayer];
UIBezierPath *path1 = [UIBezierPath bezierPathWithArcCenter:_backgroundLayer.position radius:self.frame.size.width/2 - 2 startAngle:-M_PI_2 endAngle:-
    M_PI_2 + 2 * M_PI clockwise:YES];
_cycleLayer.strokeEnd = 0;
[_cycleLayer setPath:path1.CGPath];
```

```
CAGradientLayer *gradientLayer = [CAGradientLayer layer];
CAGradientLayer *gradientLayer1 = [CAGradientLayer layer];
// 设置 gradientLayer 的 Frame
gradientLayer1.frame = CGRectMake(0, 0, frame.size.width/2, frame.size.height);
// 创建渐变色数组，需要转换为CGColor颜色
gradientLayer1.colors = @[(id)[UIColor redColor].CGColor,
                          (id)[UIColor yellowColor].CGColor];
// 设置两种颜色变化点，取值范围 0.0~1.0
gradientLayer1.locations = @[@(0.5f), @(0.9f)];
// 设置渐变颜色方向，左上点为(0,0), 右下点为(1,1)
gradientLayer1.startPoint = CGPointMake(0.5, 1);
gradientLayer1.endPoint = CGPointMake(0.5, 0);
// 添加渐变色到创建的 UIView 上去
[gradientLayer addSublayer:gradientLayer1];

CAGradientLayer *gradientLayer2 = [CAGradientLayer layer];
gradientLayer2.frame = CGRectMake(frame.size.width/2, 0, frame.size.width/2, frame.size.height);
gradientLayer2.colors = @[(id)[UIColor yellowColor].CGColor,
                          (id)[UIColor blueColor].CGColor];
gradientLayer2.locations = @[@(0.1f), @(0.5f)];
gradientLayer2.startPoint = CGPointMake(0.5, 0);
gradientLayer2.endPoint = CGPointMake(0.5, 1);
[gradientLayer addSublayer:gradientLayer2];

[self.layer addSublayer:gradientLayer];
[gradientLayer setMask:_cycleLayer];

_isLoading = NO;
_autoAnimating = YES;
}
```

```
#pragma mark public methods

- (void)setProgress:(float)progress{
    if(progress < 0){
        if(_cycleLayer.strokeEnd != 0)
        _cycleLayer.strokeEnd = 0;
        _backgroundLayer.opacity = 0;
        return;
    }
    if(progress >= 1){
        progress = 1;
        if(_autoAnimating)
        [self beginAnimating];
        else{
            _cycleLayer.strokeEnd =  1;
            _backgroundLayer.opacity = 0.3;
        }
    }else{
        _cycleLayer.strokeEnd = progress;
        _backgroundLayer.opacity = 0.3 * progress;
    }
}
```

```
- (void)beginAnimating{
    _isLoading = YES;
    UIBezierPath *path = [UIBezierPath bezierPathWithArcCenter:_backgroundLayer.position radius:self.frame.size.width/2 - 2 startAngle:-M_PI_2 endAngle:2 *
        M_PI - M_PI_2 clockwise:YES];
    [_backgroundLayer setPath:path.CGPath];
    _backgroundLayer.opacity = 0.3;

    UIBezierPath *path1 = [UIBezierPath bezierPathWithArcCenter:_backgroundLayer.position radius:self.frame.size.width/2 - 2 startAngle:-M_PI endAngle:-M_PI_2
        clockwise:YES];
    [_cycleLayer setPath:path1.CGPath];
    _cycleLayer.strokeEnd =  1;

    _animation = [CABasicAnimation animationWithKeyPath:@"transform.rotation.z"];
    _animation.duration = 0.66;
    _animation.timingFunction = [CAMediaTimingFunction functionWithName:kCAMediaTimingFunctionLinear];
    _animation.repeatCount = CGFLOAT_MAX;
    _animation.toValue = @(2 * M_PI);
    _animation.beginTime = CACurrentMediaTime();
    [_cycleLayer addAnimation:_animation forKey:@"rotation"];
}
```

```
- (void)stopAnimating{
    _isLoading = NO;
    [_cycleLayer removeAnimationForKey:@"rotation"];

    [CATransaction begin];
    [CATransaction setDisableActions:YES];
    _cycleLayer.strokeEnd = 0;
    _backgroundLayer.opacity = 0;
    [CATransaction commit];

    UIBezierPath *path1 = [UIBezierPath bezierPathWithArcCenter:_backgroundLayer.position radius:
        self.frame.size.width/2 - 2 startAngle:-M_PI_2 endAngle:-M_PI_2 + 2 * M_PI clockwise:YES];
    _cycleLayer.path = path1.CGPath;

    CABasicAnimation *animation = [CABasicAnimation animationWithKeyPath:@"opacity"];
    animation.duration = 0.33;
    animation.toValue = @0;
    [self.layer addAnimation:animation forKey:@"removeSubViews"];

    CABasicAnimation *animation1 = [CABasicAnimation animationWithKeyPath:@"opacity"];
    animation1.duration = 0.33;
    animation1.fromValue = @0.3;
    animation1.toValue = @0;
    animation1.removedOnCompletion = NO;
    [self.backgroundLayer addAnimation:animation1 forKey:@"removeSubView"];
}
```

(2) MYNormalCircleProgressView

```objc
#import <UIKit/UIKit.h>

@interface MYNormalCircleProgressView : UIView

@property (strong, nonatomic) UIColor *lineColor;
@property (assign, nonatomic) CGFloat lineWidth;

@property (assign, nonatomic) BOOL isLoading;

- (void)startAnimating;
- (void)stopAnimating;

@end
```

```objc
#import "MYNormalCircleProgressView.h"

@interface MYNormalCircleProgressView ()

@property (assign, nonatomic) CGSize size;
@property (strong, nonatomic) CAShapeLayer *circleLayer;

@end

@implementation MYNormalCircleProgressView

- (instancetype)initWithCoder:(NSCoder *)aDecoder{
    if(self = [super initWithCoder:aDecoder]){
        self.lineColor = [UIColor grayColor];
        self.lineWidth = 2.0f;
        self.circleLayer = [[CAShapeLayer alloc] init];
        self.circleLayer.fillColor = [UIColor clearColor].CGColor;
        self.circleLayer.allowsEdgeAntialiasing = YES;
    }
    return self;
}

- (void)drawRect:(CGRect)rect{
    [super drawRect:rect];
    CGFloat radius = MIN(rect.size.width, rect.size.height);
    self.size = CGSizeMake(radius, radius);
}
```

```
#pragma mark Private Methods

- (void)drawCircleProgressLayer{
    [self.layer setOpacity:1];

    UIBezierPath *path = [UIBezierPath bezierPathWithArcCenter:CGPointMake(self.size.width/2, self
        .size.height/2) radius:self.size.width/2 - self.lineWidth startAngle:0.1 endAngle:M_PI_2  * 3.7
        clockwise:YES];
    self.circleLayer.path = path.CGPath;
    self.circleLayer.frame = CGRectMake(0, 0, self.size.width, self.size.height);
    self.circleLayer.lineWidth = self.lineWidth;
    self.circleLayer.lineCap = kCALineCapRound;
    self.circleLayer.strokeColor = self.lineColor.CGColor;
    [self.layer addSublayer:self.circleLayer];
```

```
    // 创建 CAGradientLayer 对象
    CAGradientLayer *gradientLayer = [CAGradientLayer layer];
    CAGradientLayer *gradientLayer1 = [CAGradientLayer layer];
    // 设置 gradientLayer 的 Frame
    gradientLayer1.frame = CGRectMake(0, self.size.height/2, self.size.width, self.size.height/2);
    // 创建渐变色数组，需要转换为CGColor颜色
    gradientLayer1.colors = @[(id)[UIColor groupTableViewBackgroundColor].CGColor, (id)[UIColor
        grayColor].CGColor];
    // 设置两种颜色变化点，取值范围 0.0~1.0
    gradientLayer1.locations = @[@(0.1f), @(0.9f)];
    // 设置渐变颜色方向，左上点为(0,0), 右下点为(1,1)
    gradientLayer1.startPoint = CGPointMake(1, 0.5);
    gradientLayer1.endPoint = CGPointMake(0, 0.5);
    // 添加渐变色到创建的 UIView 上去
    [gradientLayer addSublayer:gradientLayer1];

    CAGradientLayer *gradientLayer2 = [CAGradientLayer layer];
    gradientLayer2.frame = CGRectMake(0, 0, self.size.width, self.size.height/2);
    gradientLayer2.colors = @[(id)[UIColor grayColor].CGColor, (id)[UIColor grayColor].CGColor];
    gradientLayer2.locations = @[@(0.1f), @(0.9f)];
    gradientLayer1.startPoint = CGPointMake(1, 0.5);
    gradientLayer1.endPoint = CGPointMake(0, 0.5);
    [gradientLayer addSublayer:gradientLayer2];

    [self.layer addSublayer:gradientLayer];
    [gradientLayer setMask:_circleLayer];
}
```

```
#pragma mark Public Methods

- (void)startAnimating{
    if(self.isLoading){
        return;
    }

    [self drawCircleProgressLayer];

    CABasicAnimation *animation = [CABasicAnimation
        animationWithKeyPath:@"transform.rotation.z"];
    animation.duration = 0.88;
    animation.toValue = @(M_PI *2);
    animation.removedOnCompletion = NO;
    animation.repeatCount = INT_MAX;
    animation.timingFunction = [CAMediaTimingFunction functionWithName:
        kCAMediaTimingFunctionLinear];
    animation.fillMode = kCAFillModeForwards;
    [self.layer addAnimation:animation forKey:@"rotation"];
    _isLoading = YES;
}
```

```
- (void)stopAnimating{
    if(!_isLoading)
        return;

    [CATransaction begin];
    [CATransaction setDisableActions:YES];
    [self.layer setOpacity:0];
    [CATransaction commit];

    CABasicAnimation *animation = [CABasicAnimation animationWithKeyPath:@"opacity"];
    animation.duration = 0.33;
    animation.fromValue = @1;
    animation.toValue = @0;
    animation.removedOnCompletion = YES;
    animation.fillMode = kCAFillModeForwards;
    animation.delegate = self;
    [self.layer addAnimation:animation forKey:@"opacity"];
}

#pragma mark CAAnimationDelegate

- (void)animationDidStop:(CAAnimation *)anim finished:(BOOL)flag{
    if(flag){
        _isLoading = NO;
        [self.layer.sublayers makeObjectsPerformSelector:@selector(removeFromSuperlayer)];
    }
}
```

(3) MYCustomCircleProgressView

```
@interface MYCustomCircleProgressView : UIView

@property (strong, nonatomic) UIColor *lineColor;
@property (assign, nonatomic) CGFloat lineWidth;

@property (assign, nonatomic) BOOL isLoading;

- (void)startAnimating;
- (void)stopAnimating;

@end
```

```
#import "MYCustomCircleProgressView.h"

@interface MYCustomCircleProgressView ()

@property (assign, nonatomic) CGSize size;
@property (nonatomic, strong) CAShapeLayer *circleLayer;

@end

@implementation MYCustomCircleProgressView

- (instancetype)initWithCoder:(NSCoder *)aDecoder{
    if(self = [super initWithCoder:aDecoder]){
        self.lineColor = [UIColor grayColor];
        self.lineWidth = 2.0f;
        self.circleLayer = [[CAShapeLayer alloc] init];
        self.circleLayer.fillColor = [UIColor clearColor].CGColor;
        self.circleLayer.allowsEdgeAntialiasing = YES;
    }
    return self;
}

- (void)drawRect:(CGRect)rect{
    [super drawRect:rect];
    CGFloat radius = MIN(rect.size.width, rect.size.height);
    self.size = CGSizeMake(radius, radius);
}
```

```
- (void)drawCircleProgressLayer{
    [self.layer setOpacity:1];

    UIBezierPath *path = [UIBezierPath bezierPathWithArcCenter:CGPointMake(self.size.width/2, self.size.height/2) radius:self.size.
        width/2 - self.lineWidth startAngle:0.1 endAngle:M_PI * 2 clockwise:YES];
    self.circleLayer.path = path.CGPath;
    self.circleLayer.frame = CGRectMake(0, 0, self.size.width, self.size.height);
    self.circleLayer.lineWidth = self.lineWidth;
    self.circleLayer.lineCap = kCALineCapRound;
    self.circleLayer.allowsEdgeAntialiasing = YES;
    self.circleLayer.strokeColor = self.lineColor.CGColor;
    [self.layer addSublayer:self.circleLayer];
```

```
    // 创建 CAGradientLayer 对象
    CAGradientLayer *gradientLayer = [CAGradientLayer layer];
    CAGradientLayer *gradientLayer1 = [CAGradientLayer layer];
    // 设置 gradientLayer 的 Frame
    gradientLayer1.frame = CGRectMake(0, self.size.height/2, self.size.width, self.size.height/2);
    // 创建渐变色数组，需要转换为CGColor颜色
    gradientLayer1.colors = @[(id)[UIColor groupTableViewBackgroundColor].CGColor, (id)[UIColor grayColor].CGColor];
    // 设置两种颜色变化点，取值范围 0.0~1.0
    gradientLayer1.locations = @[@(0.1f), @(0.9f)];
    // 设置渐变颜色方向，左上点为(0,0), 右下点为(1,1)
    gradientLayer1.startPoint = CGPointMake(1, 0.5);
    gradientLayer1.endPoint = CGPointMake(0, 0.5);
    // 添加渐变色到创建的 UIView 上去
    [gradientLayer addSublayer:gradientLayer1];

    CAGradientLayer *gradientLayer2 = [CAGradientLayer layer];
    gradientLayer2.frame = CGRectMake(0, 0, self.size.width, self.size.height/2);
    gradientLayer2.colors = @[(id)[UIColor grayColor].CGColor, (id)[UIColor grayColor].CGColor];
    gradientLayer2.locations = @[@(0.1f), @(0.9f)];
    gradientLayer1.startPoint = CGPointMake(1, 0.5);
    gradientLayer1.endPoint = CGPointMake(0, 0.5);
    [gradientLayer addSublayer:gradientLayer2];

    [self.layer addSublayer:gradientLayer];
    [gradientLayer setMask:_circleLayer];
}
```

```
#pragma mark Public Methods

- (void)startAnimating{
    if(self.isLoading){
        return;
    }

    [self drawCircleProgressLayer];

    CABasicAnimation *animation = [CABasicAnimation animationWithKeyPath:@"transform.rotation.z"];
    animation.duration = 3.0f;
    animation.toValue = @(M_PI *2);
    animation.removedOnCompletion = NO;
    animation.repeatCount = INT_MAX;
    animation.timingFunction = [CAMediaTimingFunction functionWithName:kCAMediaTimingFunctionLinear];
    animation.fillMode = kCAFillModeForwards;
    [self.layer addAnimation:animation forKey:@"rotation"];
    _isLoading = YES;

    CABasicAnimation *interAnimation1 = [CABasicAnimation animationWithKeyPath:@"strokeEnd"];
    interAnimation1.duration = 1.0f;
    interAnimation1.fromValue = @0;
    interAnimation1.toValue = @1;
    interAnimation1.beginTime = 0;
    interAnimation1.timingFunction = [CAMediaTimingFunction functionWithName:kCAMediaTimingFunctionEaseIn];
```

```
    CABasicAnimation *interAnimation2 = [CABasicAnimation animationWithKeyPath:@"strokeStart"];
    interAnimation2.duration = 1.0f;
    interAnimation2.fromValue = @0;
    interAnimation2.toValue = @1;
    interAnimation2.beginTime = 1;
    interAnimation2.timingFunction = [CAMediaTimingFunction functionWithName:kCAMediaTimingFunctionEaseOut];

    CAAnimationGroup *interAnimations = [CAAnimationGroup animation];
    interAnimations.duration = 2.0f;
    interAnimations.fillMode = kCAFillModeForwards;
    interAnimations.animations = @[interAnimation1, interAnimation2];
    interAnimations.repeatCount = INT_MAX;
    interAnimations.removedOnCompletion = NO;
    [self.circleLayer addAnimation:interAnimations forKey:@"interAnimations"];
}
```

```
- (void)stopAnimating{
    if(!_isLoading)
        return;

    [CATransaction begin];
    [CATransaction setDisableActions:YES];
    [self.layer setOpacity:0];
    [CATransaction commit];

    CABasicAnimation *animation = [CABasicAnimation animationWithKeyPath:@"opacity"];
    animation.duration = 0.33;
    animation.fromValue = @1;
    animation.toValue = @0;
    animation.removedOnCompletion = YES;
    animation.fillMode = kCAFillModeForwards;
    animation.delegate = self;
    [self.layer addAnimation:animation forKey:@"opacity"];
}

#pragma mark CAAnimationDelegate

- (void)animationDidStop:(CAAnimation *)anim finished:(BOOL)flag{
    if(flag){
        _isLoading = NO;
        [self.layer.sublayers makeObjectsPerformSelector:@selector(removeFromSuperlayer)];
    }
}
```