跑马灯动画

1. 简述:跑马灯在app中一般用来展示警示、提示性语句，循环地跑动指定的文字来给予用户提醒。跑马灯的实现在安卓中系统有相应的封装，只要简单调用即可。Ios则需要开发者自己去封装了。
2. 原理

   跑马灯的原理是什么呢?动画效果上来讲，是一行文本的滚动，在没结束但快结束前，文本又重头接了进来。那么，从实现上来说呢?没错，我们需要两个一样的文本，一种方式：在一个文本快滚动结束时让第二个文本跑起来，第一个文本跑动结束后复位并等待第二个文本滚动结束，这样无限循环下去；另一种方式：还是两个文本，只不过这两个文本都是最后面跟着一串空字符串的，然后第二个文本紧跟第一个文本，然后第一个文本位置x从0到-自身宽度的时候迅速让两个文本复位，这样视觉上就达到了和方式一一样的效果，实现上来讲，第二种实现也是比较简单的。
3. 实现

   就第二种方式，我们来做个跑马灯

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

@interface RunHorseLampView : UIControl

/**
 移动一个单位宽度文字所需的时间，默认为5秒
 */
@property (assign, nonatomic) CGFloat duration_perwidth;

/**
 跑马灯点击回调
 */
@property (copy, nonatomic) void(^RunHourseLampViewClickBlock)();

/**
 * 开始动画
 */
- (void)startRuning:(NSString *)text;

/**
 * 结束动画
 *
 */
- (void)stopRuning;

@end


@interface RunHorseLampView ()

@property (strong, nonatomic) UILabel *firstLabel;
@property (strong, nonatomic) UILabel *secondLabel;

@property (strong, nonatomic) NSLayoutConstraint *firstLabelLeft;//默认为self.width，移动结束时为-self.width
@property (strong, nonatomic) NSLayoutConstraint *firstLabelWidth;

@property (assign, nonatomic) CGFloat duration;//移动整个文本所需的时间

@property (assign, nonatomic) BOOL appIsActive;

@end
```

```
- (void)dealloc{
    [[NSNotificationCenter defaultCenter] removeObserver:self];
}

- (instancetype)initWithFrame:(CGRect)frame{
    if(self = [super initWithFrame:frame]){
        self.layer.masksToBounds = YES;
        self.duration_perwidth = 8.0f;
        self.appIsActive = YES;
        [self addSubview:self.firstLabel];
        [self addSubview:self.secondLabel];

        __weak typeof(self) weakSelf = self;
        [self.firstLabel cwn_makeConstraints:^(UIView *maker) {
            weakSelf.firstLabelLeft = [maker.leftToSuper(0) lastConstraint];
            weakSelf.firstLabelWidth =  [maker.topToSuper(0).bottomToSuper(0).width(self.frame.size.width) lastConstraint];
        }];

        [self.secondLabel cwn_makeConstraints:^(UIView *maker) {
            maker.leftTo(weakSelf.firstLabel, 1, 0).centerYto(weakSelf.firstLabel, 0).widthTo(weakSelf.firstLabel, 1, 0).heightTo(weakSelf.firstLabel, 1, 0);
        }];

        [[NSNotificationCenter defaultCenter] addObserver:self selector:@selector(applicationWillResignActive) name:UIApplicationWillResignActiveNotification object:nil];
        [[NSNotificationCenter defaultCenter] addObserver:self selector:@selector(applicationDidBecomeActive) name:UIApplicationDidBecomeActiveNotification object:nil];
    }
    return self;
}
- (void)addAnimation{
    [self layoutIfNeeded];

    __weak typeof(self) weakSelf = self;
    weakSelf.firstLabelLeft.constant =  -weakSelf.firstLabelWidth.constant;
    [UIView animateWithDuration:self.duration delay:0 options:UIViewAnimationOptionCurveLinear animations:^{
        [weakSelf layoutIfNeeded];
    } completion:^(BOOL finished) {
        if(finished == NO){
            if(weakSelf.appIsActive == YES){//不正常被中断，需要重启
                if([weakSelf.firstLabel.text length] > 0)
                    [weakSelf startRuning:weakSelf.firstLabel.text];
            }
        }else{//正常结束，递归动画
            CALayer *layer = weakSelf.firstLabel.layer.presentationLayer;
            CGRect frame = [layer frame];

            if(CGRectGetMinX(frame) == -_firstLabelWidth.constant){
                weakSelf.duration = weakSelf.firstLabelWidth.constant * 1.0 / weakSelf.frame.size.width * weakSelf.duration_perwidth;
                weakSelf.firstLabelLeft.constant = 0;
                [weakSelf addAnimation];
            }else{
//                DLog(@"");
            }
        }
    }];
}

- (void)startRuning:(NSString *)text{
    if([text length] == 0){
        [self stopRuning];
        return;
    }

    [self layoutIfNeeded];
    self.firstLabelLeft.constant = 0;

    self.firstLabel.text = text;
    self.secondLabel.text = text;

    CGSize size = [self.firstLabel sizeThatFits:CGSizeMake(self.frame.size.width, self.frame.size.height)];
    self.firstLabelWidth.constant = size.width >= self.frame.size.width ? size.width : self.frame.size.width;

    self.duration = self.firstLabelWidth.constant * 2.0 / self.frame.size.width * self.duration_perwidth;

    [self performSelector:@selector(delayRuning:) withObject:text afterDelay:0.44];
}
```

```
- (void)startRuning:(NSString *)text{
    if([text length] == 0){
        [self stopRuning];
        return;
    }

    [self layoutIfNeeded];
    self.firstLabelLeft.constant = 0;

    self.firstLabel.text = text;
    self.secondLabel.text = text;

    CGSize size = [self.firstLabel sizeThatFits:CGSizeMake(self.frame.size.width, self.frame.size.height)];
    self.firstLabelWidth.constant = size.width >= self.frame.size.width ? size.width : self.frame.size.width;

    self.duration = self.firstLabelWidth.constant * 2.0 / self.frame.size.width * self.duration_perwidth;

    [self performSelector:@selector(delayRuning:) withObject:text afterDelay:0.44];
}

- (void)delayRuning:(NSString *)text{//延时，等上一个结束
    [self addAnimation];
}

- (void)stopRuning{
    //清空文本
    self.firstLabel.text = @"";
    self.secondLabel.text = @"";

    //移除所有动画
    [self.layer removeAllAnimations];
}


#pragma mark - 监听事件处理

- (void)applicationWillResignActive{
    _appIsActive = NO;
}

- (void)applicationDidBecomeActive{
    _appIsActive = YES;
    if([[self.firstLabel text] length] > 0){
        [self startRuning:self.firstLabel.text];
    }
}

- (void)touchesBegan:(NSSet<UITouch *> *)touches withEvent:(UIEvent *)event{
    if(self.RunHourseLampViewClickBlock){
        self.RunHourseLampViewClickBlock();
    }
}

#pragma mark - 控件get方法

- (UILabel *)firstLabel{
    if(!_firstLabel){
        _firstLabel = [[UILabel alloc] init];
        _firstLabel.font = [UIFont systemFontOfSize:14];
        _firstLabel.textColor = HexColor(0x333333);
    }
    return _firstLabel;
}

- (UILabel *)secondLabel{
    if(!_secondLabel){
        _secondLabel = [[UILabel alloc] init];
        _secondLabel.font = [UIFont systemFontOfSize:14];
        _secondLabel.textColor = HexColor(0x333333);
    }
    return _secondLabel;
}
```