链滴

iOS 开发 -------- 自定义系统相机

作者：HuixiaZhang
原文链接：https://ld246.com/article/1539224923486
来源网站：链滴

一.初衷

看到各种APP的自定义相机界面,之后再看我们自己的APP全是用的系统相机,感觉有点遗憾,就简单写一下.

二.简单代码(用的是AVFoundation)

```
#import "CustomCameraViewController.h"
#import <AVFoundation/AVFoundation.h>
#import "PhotoHandleViewController.h"
@interface CustomCameraViewController ()<AVCaptureMetadataOutputObjectsDelegate,AV
apturePhotoCaptureDelegate>
// 捕获设备,前置,后置摄像头,麦克风
@property (nonatomic,strong) AVCaptureDevice *device;
@property (nonatomic,strong) AVCapturePhotoSettings *settings;
// 输入设备
@property (nonatomic,strong) AVCaptureDeviceInput *input;
@property (nonatomic,strong) AVCaptureMetadataOutput *output;
// 输出图片
@property (nonatomic,strong) AVCapturePhotoOutput *photoOutput;
// 摄像头
@property (nonatomic,strong) AVCaptureSession *session;
// 实时显示捕获的图像
@property (nonatomic,strong) AVCaptureVideoPreviewLayer *layer;
// 聚焦点
@property (nonatomic,strong) UIView *focusView;
@end
```

初始化自定义相机

```
- (void)customCamera {
    // AVMediaTypeVideo 代表视频 (默认使用后置);
    self.device = [AVCaptureDevice defaultDeviceWithMediaType:AVMediaTypeVideo];

    // 设备输入
    self.input = [[AVCaptureDeviceInput alloc] initWithDevice:self.device error:nil];

    // 输出对象
    self.output = [[AVCaptureMetadataOutput alloc] init];
    self.photoOutput = [[AVCapturePhotoOutput alloc] init];

    // 会话 结合输入输出
    self.session = [[AVCaptureSession alloc] init];

    if ([self.session canSetSessionPreset:AVCaptureSessionPreset1280x720]) {

        self.session.sessionPreset = AVCaptureSessionPreset1280x720;
    }

    if ([self.session canAddInput:self.input]) {

        [self.session addInput:self.input];
    }
```

```
    if ([self.session canAddOutput:self.photoOutput]) {

        [self.session addOutput:self.photoOutput];
    }

    // 预览

    self.layer = [[AVCaptureVideoPreviewLayer alloc] initWithSession:self.session];
    self.layer.frame = CGRectMake(20, 70, self.view.frame.size.width - 40, self.view.frame.size.he
ght / 2);
    self.layer.videoGravity = AVLayerVideoGravityResizeAspectFill;

    [self.view.layer addSublayer:self.layer];

    // 开始启动
    [self.session startRunning];

    if ([self.device lockForConfiguration:nil]) {

        if ([self.device isFlashModeSupported:AVCaptureFlashModeAuto]) {

            [self.device setFlashMode:AVCaptureFlashModeAuto];
        }

        // 自动白平衡
        if ([self.device isWhiteBalanceModeSupported:AVCaptureWhiteBalanceModeAutoWhite
alance]) {

            [self.device setWhiteBalanceMode:AVCaptureWhiteBalanceModeAutoWhiteBalance];
        }
        [self.device unlockForConfiguration];
    }
}
```

聚焦点

```
// 聚焦点

- (void)focusGesture:(UITapGestureRecognizer *)recognizer {

    CGPoint point = [recognizer locationInView:recognizer.view];

    [self focusAtPoint:point];
}

- (void)focusAtPoint:(CGPoint)point {

    CGSize size = self.view.bounds.size;
    CGPoint focusPoint = CGPointMake(point.y / size.height, 1 - point.x / size.width);
    NSError *error;

    if ([self.device lockForConfiguration:&error]) {

        if ([self.device isFocusModeSupported:AVCaptureFocusModeAutoFocus]) {
```

```
            [self.device setFocusPointOfInterest:focusPoint];
            [self.device setFocusMode:AVCaptureFocusModeAutoFocus];
        }

        if ([self.device isExposureModeSupported:AVCaptureExposureModeAutoExpose]) {
            [self.device setExposurePointOfInterest:focusPoint];
            [self.device setExposureMode:AVCaptureExposureModeAutoExpose];

        }

        [self.device unlockForConfiguration];

    }
    _focusView.center = point;
    _focusView.hidden = NO;

    [UIView animateWithDuration:0.3 animations:^{

        _focusView.transform = CGAffineTransformMakeScale(1.25, 1.25);
    } completion:^(BOOL finished) {

        [UIView animateWithDuration:0.5 animations:^{
            _focusView.transform = CGAffineTransformIdentity;
        } completion:^(BOOL finished) {

            _focusView.hidden = YES;
        }];
    }];
}
```

截取图片

```
- (void)shutterCamera {

    AVCaptureConnection *videoConnection = [self.photoOutput connectionWithMediaType:
VMediaTypeVideo];
    if (!videoConnection) {

        NSLog(@"拍照失败");
        return;
    }

    [self.photoOutput capturePhotoWithSettings:self.settings delegate:self];

}

- (void)captureOutput:(AVCapturePhotoOutput *)captureOutput didFinishProcessingPhotoSa
pleBuffer:(nullable CMSampleBufferRef)photoSampleBuffer previewPhotoSampleBuffer:(nulla
le CMSampleBufferRef)previewPhotoSampleBuffer resolvedSettings:(AVCaptureResolvedPhot
Settings *)resolvedSettings bracketSettings:(nullable AVCaptureBracketedStillImageSettings *
bracketSettings error:(nullable NSError *)error {

    if (photoSampleBuffer == NULL) {
```

```
        return;
    }

    NSData *imgData = [AVCapturePhotoOutput JPEGPhotoDataRepresentationForJPEGSampl
Buffer:photoSampleBuffer previewPhotoSampleBuffer:previewPhotoSampleBuffer];
    // 照片处理界面
    PhotoHandleViewController *photoVC = [[PhotoHandleViewController alloc] init];
    photoVC.img = [UIImage imageWithData:imgData];
    [self.navigationController pushViewController:photoVC animated:YES];
}
```

照片处理PhotoHandleViewController

```
#import "PhotoHandleViewController.h"
#import "CustomCollectionViewCell.h"

@interface PhotoHandleViewController () <UICollectionViewDelegate,UICollectionViewDataS
urce,UICollectionViewDelegateFlowLayout>
@property (nonatomic,strong) UICollectionView *bottomC;
@property (nonatomic,strong) NSMutableArray *filterArray;
@property (nonatomic,strong) UIImageView *imgV;
@end

self.imgV = [[UIImageView alloc] initWithFrame:CGRectMake(20, 70, self.view.frame.size.width
- 40, self.view.frame.size.height / 2)];
    self.imgV.image = self.img;
    self.imgV.contentMode = UIViewContentModeScaleAspectFill;
    self.imgV.clipsToBounds = YES;
    [self.view addSubview:self.imgV];

// 滤镜效果
    self.filterArray = [[NSMutableArray alloc] initWithObjects:
                        @"OriginImage",
                        @"CIPhotoEffectChrome",
                        @"CIPhotoEffectFade",
                        @"CIPhotoEffectInstant",
                        @"CIPhotoEffectProcess",
                        @"CIPhotoEffectTransfer",
                        @"CISRGBToneCurveToLinear",
                        @"CIColorInvert",
                        @"CIColorPosterize",
                        @"CIFalseColor",
                        @"CIXRay",
                        @"CIThermal",
                        @"CISepiaTone",
                        @"CIColorMonochrome",
                        nil];

    UICollectionViewFlowLayout *flowLay = [[UICollectionViewFlowLayout alloc] init];
    flowLay.itemSize = CGSizeMake(100, 150);
    flowLay.scrollDirection = UICollectionViewScrollDirectionHorizontal;
    self.bottomC = [[UICollectionView alloc] initWithFrame:CGRectMake(0, CGRectGetMaxY(self
imgV.frame) + 20, self.view.frame.size.width, 200) collectionViewLayout:flowLay];
```

```
    self.bottomC.backgroundColor = [UIColor whiteColor];
    self.bottomC.delegate = self;
    self.bottomC.dataSource = self;
    [self.view addSubview:self.bottomC];

    [self.bottomC registerClass:[CustomCollectionViewCell class] forCellWithReuseIdentifier:@"
ell"];
```

处理滤镜

```
- (void)collectionView:(UICollectionView *)collectionView didSelectItemAtIndexPath:(NSIndex
ath *)indexPath {

    [self fliterEvent:self.filterArray[indexPath.item]];
}

#pragma mark 滤镜处理事件

- (void)fliterEvent:(NSString *)filterName
{
    if ([filterName isEqualToString:@"OriginImage"]) {
        self.imgV.image = self.img;

    }else{
        //将UIImage转换成CIImage
        CIImage *ciImage = [[CIImage alloc] initWithImage:[self fixOrientation:self.img]];

        //创建滤镜
        CIFilter *filter = [CIFilter filterWithName:filterName keysAndValues:kCIInputImageKey, ci
mage, nil];

        //已有的值不改变，其他的设为默认值
        [filter setDefaults];

        //获取绘制上下文
        CIContext *context = [CIContext contextWithOptions:nil];

        //渲染并输出CIImage
        CIImage *outputImage = [filter outputImage];

        //创建CGImage句柄
        CGImageRef cgImage = [context createCGImage:outputImage fromRect:[outputImage ex
ent]];

        //获取图片
        UIImage *image = [UIImage imageWithCGImage:cgImage];

        //释放CGImage句柄
        CGImageRelease(cgImage);

        self.imgV.image = image;
    }
}
```

此非常之重要,由于通过cgimageref 得到的图片 会逆时针转90度的,因此用以下方法得到正确图片

```
#pragma mark -------

- (UIImage *)fixOrientation:(UIImage *)aImage {

    // No-op if the orientation is already correct
    if (aImage.imageOrientation == UIImageOrientationUp)
        return aImage;

    // We need to calculate the proper transformation to make the image upright.
    // We do it in 2 steps: Rotate if Left/Right/Down, and then flip if Mirrored.
    CGAffineTransform transform = CGAffineTransformIdentity;

    switch (aImage.imageOrientation) {
        case UIImageOrientationDown:
        case UIImageOrientationDownMirrored:
            transform = CGAffineTransformTranslate(transform, aImage.size.width, aImage.size.he
ght);
            transform = CGAffineTransformRotate(transform, M_PI);
            break;

        case UIImageOrientationLeft:
        case UIImageOrientationLeftMirrored:
            transform = CGAffineTransformTranslate(transform, aImage.size.width, 0);
            transform = CGAffineTransformRotate(transform, M_PI_2);
            break;

        case UIImageOrientationRight:
        case UIImageOrientationRightMirrored:
            transform = CGAffineTransformTranslate(transform, 0, aImage.size.height);
            transform = CGAffineTransformRotate(transform, -M_PI_2);
            break;
        default:
            break;
    }

    switch (aImage.imageOrientation) {
        case UIImageOrientationUpMirrored:
        case UIImageOrientationDownMirrored:
            transform = CGAffineTransformTranslate(transform, aImage.size.width, 0);
            transform = CGAffineTransformScale(transform, -1, 1);
            break;

        case UIImageOrientationLeftMirrored:
        case UIImageOrientationRightMirrored:
            transform = CGAffineTransformTranslate(transform, aImage.size.height, 0);
            transform = CGAffineTransformScale(transform, -1, 1);
            break;
        default:
            break;
    }

    // Now we draw the underlying CGImage into a new context, applying the transform
```

```
    // calculated above.
    CGContextRef ctx = CGBitmapContextCreate(NULL, aImage.size.width, aImage.size.height,
                                             CGImageGetBitsPerComponent(aImage.CGImage), 0,
                                             CGImageGetColorSpace(aImage.CGImage),
                                             CGImageGetBitmapInfo(aImage.CGImage));
    CGContextConcatCTM(ctx, transform);
    switch (aImage.imageOrientation) {
        case UIImageOrientationLeft:
        case UIImageOrientationLeftMirrored:
        case UIImageOrientationRight:
        case UIImageOrientationRightMirrored:
            // Grr...
            CGContextDrawImage(ctx, CGRectMake(0,0,aImage.size.height,aImage.size.width), aIm
ge.CGImage);
            break;

        default:
            CGContextDrawImage(ctx, CGRectMake(0,0,aImage.size.width,aImage.size.height), aIm
ge.CGImage);
            break;
    }

    // And now we just create a new UIImage from the drawing context
    CGImageRef cgimg = CGBitmapContextCreateImage(ctx);
    UIImage *img = [UIImage imageWithCGImage:cgimg];
    CGContextRelease(ctx);
    CGImageRelease(cgimg);
    return img;
}
```

最后附上demo地址:https://github.com/HuixiaZhang/CustomCamera