定制相机相册

相关链接：

* 定制相机相册相关代码
* iOS开发中自定义相册功能性能改善
* LGPhotoBrowser
* iOS开发中dismiss到最底层控制器的方法
* iOS实时获取当前的屏幕方向之重力感应
* IOS 图片上传处理 图片压缩 图片处理
* iOS开发-自定义专属相册 (详细)| 干货

数据分了好3个逻辑数据：

1. 选择证照页面中的modelArray：

   其中装载的是从服务器得到的图片数据，以及从相册和相机中获取的图片上传服务器后返回的图片数据。

2. 用于存储单个相册中所有图片数据的数组albumModelArray。

3. 定制相册页面中的modelAdditionArray：

   其中装载了从定制相册中选中的图片的数据，并将从相册中获取的数据传递到定制相机中。

4. 相机中用于存储从相册中获取的图片以及相机排到的图片modelTakePhotosArray：

   进入页面的时候，先将相册中选中的图片赋值给modelTakePhotosArray，刷新collectionView显示

   相册中图片，拍照后将拍照数据存入其中，如果点击完成，可将modelTakePhotosArray中数据直接

   上传，如果点击去相册，则赋值给modelAdditionArray刷新定制相册页面，如果点击取消，则直接

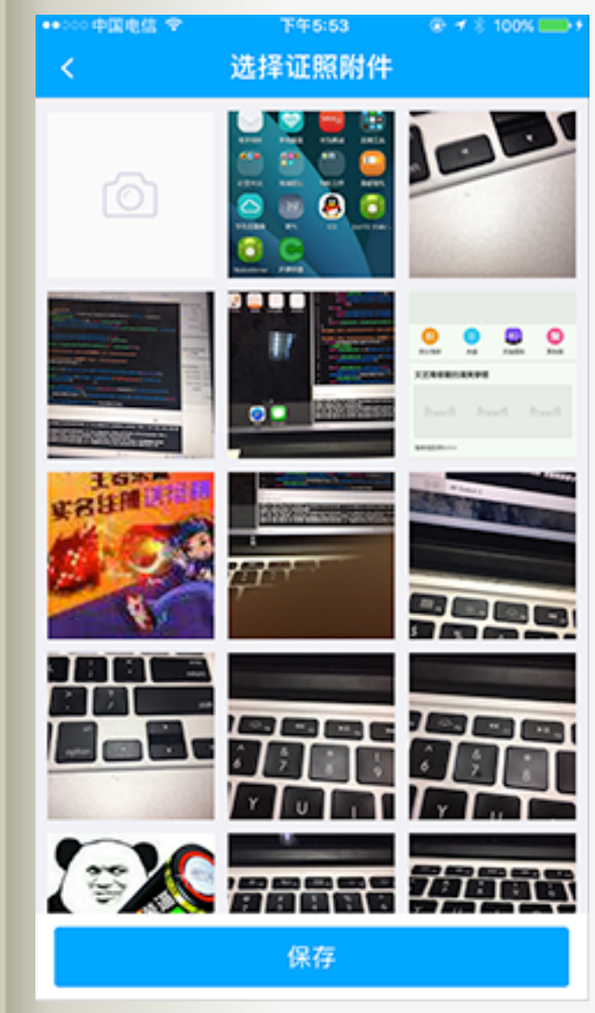
   丢弃modelTakePhotosArray中数据，保持modelAdditionArray中数据不变返回定制相册页面。

主要分为三个部分：

1. 选择证照附件页面（起中转作用）

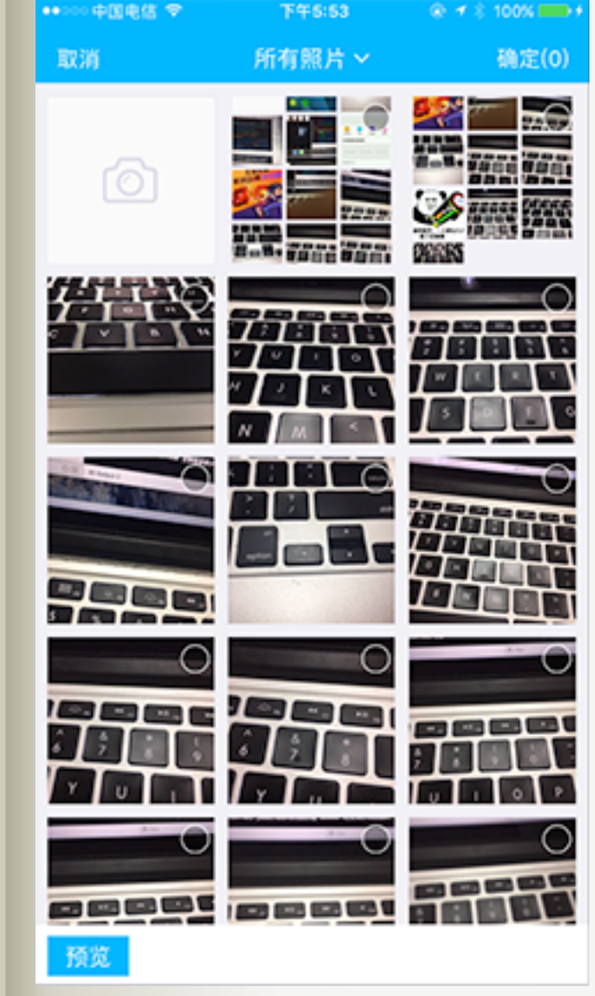
此页面一方面接收来自服务器的相关数据，对其内容进行处理，展示到页面上，另一方面，

接收来自相册或者照相机传过来的图片，对图片进行压缩等，上传到服务器中



2. 相机胶卷模块

此页面是自定义相册部分，从手机相册中获取相关相册和图片进行展示，并对图片的处理
主要难点：大量显示图片卡顿.



3. 照相模块

此模块是自定义相机部分，里边添加了可以将拍的照片显示到一个collectionView上进行显示

重点功能使用方法：

1. 使用<Photos/Photos.h>定制相册过程

1> 导入框架头：#import <Photos/Photos.h>

2> 获取所有胶卷名字

```
#pragma mark 获取所有的胶卷的名字
+ (NSMutableArray *)getAlbumObjects;

#pragma mark 获取胶卷的名字相关数据
+ (NSMutableArray *)getAlbumObjects {

    NSMutableArray *array = [NSMutableArray array];

    // 获得相机胶卷
    PHAssetCollection *cameraRoll = [PHAssetCollection
fetchAssetCollectionsWithType:PHAssetCollectionTypeSmartAlbum
subtype:PHAssetCollectionSubtypeSmartAlbumUserLibrary options:nil].lastObject;
    [array addObject:cameraRoll];

    // 获得所有的自定义相簿
    PHFetchResult<PHAssetCollection *> *assetCollections = [PHAssetCollection
fetchAssetCollectionsWithType:PHAssetCollectionTypeAlbum subtype:PHAssetCollectionSubtypeAlbumRegular
options:nil];

    for (PHAssetCollection *assetCollection in assetCollections) {
```

```
        PHFetchResult<PHAsset *> *assets = [PHAsset fetchAssetsInAssetCollection:assetCollection options:nil];

        if (assets.count) { // 如果相册里有图片，则添加到数组中
            [array addObject:assetCollection];
        }
    }

    return array;

}
```

3> 根据胶卷名字，获取所有的相册的封面图

```
#pragma mark 获取封面图片方法
+ (void)getCoverImageWith:(PHAssetCollection *)assetCollection withBlcok:(void(^)(UIImage *image))image;

#pragma mark 获取封面图片方法
+ (void)getCoverImageWith:(PHAssetCollection *)assetCollection withBlcok:(void(^)(UIImage *image))image {
    PHFetchResult *assetResult = [PHAsset fetchAssetsInAssetCollection:assetCollection options:nil];
    [[PHImageManager defaultManager] requestImageForAsset:assetResult.firstObject targetSize:CGSizeZero
contentMode:PHImageContentModeDefault options:[PHImageRequestOptions new] resultHandler:^(UIImage * _Nullable
result, NSDictionary * _Nullable info) {
        image(result);
    }];

}
```

4> 获取每个相册的相片数量，用于显示在相册列表

```
#pragma mark 获取指定胶卷儿的照片的数量
+ (NSUInteger)getAlbumCountWith:(PHAssetCollection *)assetCollection;

#pragma mark 获取某胶卷儿的照片数量
+ (NSUInteger)getAlbumCountWith:(PHAssetCollection *)assetCollection {
    // 获得某个相簿中的所有PHAsset对象
    PHFetchResult<PHAsset *> *assets = [PHAsset fetchAssetsInAssetCollection:assetCollection options:nil];
    return assets.count;
}
```

5> 获取单个相册的所有的缩略图，一次获取pageNumber张，并且可以通过页面滚动动态加载

这样的话就可以解决一次加载成百上千张造成的整个页面的卡顿甚至直接崩溃的问题

```
#pragma mark 滚动代理相关
- (void)scrollViewDidScroll:(UIScrollView *)scrollView {

    /**
     基本逻辑:
     如果上一阶段的pageNumber中的图片添加完毕，每一组图片的高度大约为3000，
     数值方向的偏移量大于3000，并且不是最后一张，就加载下一组的图片，大约一组
     图片用零点几秒时间，无影响
     */

    if (scrollView == self.collectionView) {

        if (self.isAddPage && self.collectionView.contentOffset.y > self.currentPage*3000-700 &&
self.currentPage < self.totalPage ) {

            self.isAddPage = NO;

            // 获取单个相册中的200张图片
            [AlbumTool getAlbumThumbnailWithAssetCollection:[self.albumsArray objectAtIndex:self.albumIndex]
withPage:self.currentPage+1 andComplete:^(NSMutableArray *modelArray, NSInteger totalPhotos, NSInteger
totalPage, NSInteger currentPage) {

                [self.albumModelArray addObjectsFromArray:modelArray];
                self.currentPage = currentPage; // 指定当前页数
```

```
                self.totalPage = totalPage; // 获取相册所有页数
                self.isAddPage = YES; // 如果有了返回数据，则可以进行添加了

                // 新添加一步，对两个数据源进行比较，用于初始化显示
                [self compareTwoModelArray]; // 可以一次匹配100张，这个也许可以
            }];
        }
    }
}

#pragma mark 比较两个数组中元素
- (void)compareTwoModelArray {

    /**
     匹配规则：
     新添加判断，一次只匹配比较pageNumber张
     */

    NSLog(@"self.modelAdditionArray.count; ==== %lu",(unsigned long)self.modelAdditionArray.count);

    NSInteger compareNumber = 0;

    if (self.currentPage*PageNumber + PageNumber < self.albumModelArray.count) {
        compareNumber = self.currentPage*PageNumber + PageNumber;
    } else {
        compareNumber = self.albumModelArray.count;
    }

    for (CertificateCellModel *model in self.modelAdditionArray) {

        for (NSInteger i = self.currentPage*PageNumber; i < compareNumber; i ++) {

            CertificateCellModel *modelOne = [self.albumModelArray objectAtIndex:i];

            if ([model.localIdentifier isEqualToString:modelOne.localIdentifier]) {

                NSLog(@"匹配到一张");
                modelOne.cellImageType = CertificateCellImageSelect;
                break;
            }
        }
    }

    // 在第一次添加的时候给一个第一次添加状态，其他的滑动启动这个方法，不会到顶部
    if (self.isFirstAdd) {
        // 数据加载完毕，将页面回到顶部
        self.collectionView.contentOffset=CGPointMake(0, 0);
        self.isFirstAdd = NO;
    }

    // 刷新数据
    [self.collectionView reloadData];
}
```

获取缩略图方法：可以传入页面数量和单个相册的相关数据，返回获取的所有图片数据，相册的总页数，相册总图片数，

当前处在第几页。

```
#pragma mark 根据胶卷儿获取缩略图
+ (void)getAlbumThumbnailWithAssetCollection:(PHAssetCollection *)assetCollection withPage:(NSInteger)page
andComplete:(void(^)(NSMutableArray *modelArray,NSInteger totalPhotos, NSInteger totalPage, NSInteger
currentPage))modelArrayBlock;

#pragma mark 获取所有的缩略图
+ (void)getAlbumThumbnailWithAssetCollection:(PHAssetCollection *)assetCollection withPage:(NSInteger)page
andComplete:(void(^)(NSMutableArray *modelArray,NSInteger totalPhotos, NSInteger totalPage, NSInteger
currentPage))modelArrayBlock {
    [self enumerateAssetsInAssetCollection:assetCollection original:NO withPage:page
andComplete:^(NSMutableArray *modelArray,NSInteger totalPhotos, NSInteger totalPage, NSInteger currentPage) {
        modelArrayBlock(modelArray,totalPhotos,totalPage,page);
    }];
}

#pragma mark 缩略图的获取
+ (void)enumerateAssetsInAssetCollection:(PHAssetCollection *)assetCollection original:(BOOL)original
withPage:(NSInteger)page andComplete:(void(^)(NSMutableArray *modelArray,NSInteger totalPhotos, NSInteger
totalPage, NSInteger currentPage))modelArrayBlock {
    NSLog(@"相簿名:%@", assetCollection.localizedTitle);

    PHImageRequestOptions *options = [[PHImageRequestOptions alloc] init];
```

```
    // 同步获得图片，只会返回1张图片
    options.synchronous = YES;

    // 获得某个相簿中的所有PHAsset对象
    PHFetchResult<PHAsset *> *assets = [PHAsset fetchAssetsInAssetCollection:assetCollection options:nil];

//    NSLog(@"assets.count ========== %lu",(unsigned long)assets.count);

    /**
     如果我在这里添加一个GCD呢，会怎样呢？先试试
     */

    dispatch_async(dispatch_get_global_queue(DISPATCH_QUEUE_PRIORITY_DEFAULT, 0), ^{

        NSMutableArray *modelArray = [NSMutableArray array];
        NSUInteger totalPage = assets.count/PageNumber;

        if (assets.count % PageNumber) {
            totalPage ++;
        }

        for (NSInteger i = page*PageNumber; i < page*PageNumber + PageNumber; i ++) {

            if (i >= assets.count) {
                break;
            } else {

                PHAsset *asset = [assets objectAtIndex:assets.count-i-1];
                // 是否要原图
                CGSize size = original ? CGSizeMake(asset.pixelWidth, asset.pixelHeight) : CGSizeMake(125,
125);

                // 从asset中获得图片
                [[PHImageManager defaultManager] requestImageForAsset:asset targetSize:size
contentMode:PHImageContentModeDefault options:options resultHandler:^(UIImage * _Nullable result, NSDictionary
* _Nullable info) {

                    /**
                     为了优化，直接将model在这里创建
                     */

                    CertificateCellModel *model = [[CertificateCellModel alloc] init];
                    model.itemImage = result;
                    model.page = page;
                    model.assetCollection = assetCollection;
                    model.localIdentifier = asset.localIdentifier;
                    model.cellImageType = CertificateCellImageDeselect;
                    [modelArray addObject:model];
                }];
            }
        }

        dispatch_async(dispatch_get_main_queue(), ^{

            modelArrayBlock(modelArray, assets.count, totalPage, page);
            //回到主线程
        });

    });
}
```

6> 在定制相册中点击选中时，去获取原图，由于原图有时候会特别特别大，所以取原图时对原图进行处理，使其又相对清晰

```
#pragma mark 获取指定某张图片的原图
+ (void)photoWithAssetCollection:(PHAssetCollection *)assetCollection withLocalIdentifier:(NSString
*)localIdentifier andPage:(NSInteger)page withBlcok:(void(^)(UIImage *bigImage))bigImage;

#pragma mark 获取单张原图
/**
 基本逻辑：
 首先知道相册，然后再知道相册中的第几个，那么直接匹配localIdentifier即可
 */
+ (void)photoWithAssetCollection:(PHAssetCollection *)assetCollection withLocalIdentifier:(NSString
*)localIdentifier andPage:(NSInteger)page withBlcok:(void(^)(UIImage *bigImage))bigImage {

    PHImageRequestOptions *options = [[PHImageRequestOptions alloc] init];
    // 同步获得图片，只会返回1张图片
    options.synchronous = YES;

    // 获得某个相簿中的所有PHAsset对象
```

```
PHFetchResult<PHAsset *> *assets = [PHAsset fetchAssetsInAssetCollection:assetCollection options:nil];


for (NSInteger i = page*PageNumber; i < page*PageNumber + PageNumber; i ++) {

    if (i >= assets.count) {
        break;
    } else {

        PHAsset *asset = [assets objectAtIndex:assets.count-i-1];
        // 是否要原图
        if ([asset.localIdentifier isEqualToString:localIdentifier]) {

            // 是否要原图
            CGSize size = CGSizeMake(480.0, 640.0);

            if (asset.pixelWidth > 480 || asset.pixelHeight > 640) {
                size = CGSizeMake(480.0, 640.0);
            } else {
                size = CGSizeMake(asset.pixelWidth, asset.pixelHeight);
            }

            // 从asset中获得图片
            [[PHImageManager defaultManager] requestImageForAsset:asset targetSize:size
contentMode:PHImageContentModeDefault options:options resultHandler:^(UIImage * _Nullable result, NSDictionary
* _Nullable info) {

                bigImage(result);
            }];
            break;
        }
    }
}
```

2. 定制相机部分

直接拷贝地址：AVFoundation拍照和录制视频

要注意的地方，由于直接获取xib控件中的宽高会都返回错误值1000，所以不能基于xib控件中的尺寸来适配相机屏幕

```
@property (strong,nonatomic) AVCaptureVideoPreviewLayer *captureVideoPreviewLayer;//相机拍摄预览图层

//创建视频预览层，用于实时展示摄像头状态
    _captureVideoPreviewLayer=[[AVCaptureVideoPreviewLayer alloc]initWithSession:self.captureSession];
    // 从新写一遍，不然获取的xib中的layer宽高都是1000
    self.viewContainer.layer.frame = CGRectMake(0, 0, WIDTH, HEIGHT-168);
    CALayer *layer=self.viewContainer.layer;
    layer.masksToBounds=YES;

    _captureVideoPreviewLayer.frame = layer.frame;
    _captureVideoPreviewLayer.videoGravity=AVLayerVideoGravityResizeAspectFill;//填充模式
    //将视频预览层添加到界面中
    //[layer addSublayer:_captureVideoPreviewLayer];
    [layer insertSublayer:_captureVideoPreviewLayer below:self.focusCursor.layer];
```

照相机拍出照片的转向

1> 点击拍照按钮的一瞬间获取当前的重力方向，获取一次方向直接关掉定时器，不然手机会发热

* 重力感应部分

```
#import <CoreMotion/CoreMotion.h> // 导入头

@property (nonatomic,strong) CMMotionManager *mManager; // 重力感应管理

// 重力感应管理懒加载
- (CMMotionManager *)mManager {
    if (!_mManager) {
        _mManager = [[CMMotionManager alloc] init];
    }
    return _mManager;
```

```
}

/**
* 开启重力感应
*/
+ (void)startUpdateAccelerometerWithManager:(CMMotionManager *)manager Result:(void (^)(UIImageOrientation
direction))result;

+ (void)startUpdateAccelerometerWithManager:(CMMotionManager *)manager Result:(void (^)(UIImageOrientation
direction))result {
    if ([manager isAccelerometerAvailable] == YES) {
        //回调会一直调用,建议获取到就调用下面的停止方法，需要再重新开始，当然如果需求是实时不间断的话可以等离开页面之后再stop

        NSTimeInterval updateInterval = 1/15.0;

        [manager setAccelerometerUpdateInterval:updateInterval];
        [manager startAccelerometerUpdatesToQueue:[NSOperationQueue currentQueue]
withHandler:^(CMAccelerometerData *accelerometerData, NSError *error)
         {
             double x = accelerometerData.acceleration.x;
             double y = accelerometerData.acceleration.y;
             if (fabs(y) >= fabs(x))
             {
                 if (y >= 0){
                     //Down
                     NSLog(@"倒立");
                     result(UIImageOrientationDown);
                 }
                 else{
                     //Portrait
                     NSLog(@"正常");
                     result(UIImageOrientationUp);
                 }
             }
             else
             {
                 if (x >= 0){
                     //Right
                     NSLog(@"右边");
                     result(UIImageOrientationRight);
                 }
                 else{
                     //Left

                     NSLog(@"左边");
                     result(UIImageOrientationLeft);
                 }
             }

             [self stopUpdateWithManager:manager]; // 调用一次直接关闭
         }];
    }
}

#pragma mark 关闭更新
+ (void)stopUpdateWithManager:(CMMotionManager *)manager {
    if ([manager isAccelerometerActive] == YES) {
        NSLog(@"停止");
        [manager stopAccelerometerUpdates];
        manager = nil;
    }
}
```

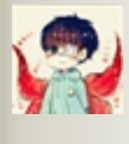
* 直接根据重力感应获取的方向来调整排到的图片的旋转角度

```
/**
 * 处理图片方向
 */
+ (UIImage *)image:(UIImage *)image rotation:(UIImageOrientation)orientation;

#pragma mark 图片处理部分
+ (UIImage *)image:(UIImage *)image rotation:(UIImageOrientation)orientation {

    switch (orientation) {
        case UIImageOrientationLeft: {
            UIImage *newImage = [UIImage imageWithCGImage:image.CGImage scale:1.0
orientation:UIImageOrientationUp];
            return newImage;
            break;
        }
```

```
        case UIImageOrientationRight:{
            // 调整好
            UIImage *newImage = [UIImage imageWithCGImage:image.CGImage scale:1.0
orientation:UIImageOrientationDown];

            return newImage;
            break;
        }
        case UIImageOrientationDown:{
            // 这个是对的
            UIImage *newImage = [UIImage imageWithCGImage:image.CGImage scale:1.0
orientation:UIImageOrientationLeft];
            return newImage;
            break;
        }
        default:
            return image;
            break;
    }

    return image;
}
```

2> 将拍到的图片保存到相册中

```
#pragma mark 保存图片
+ (void)saveImage:(UIImage *)image withLocalIdentifier:(void(^)(NSString *localIdentifier))localIdentifier;

#pragma mark 保存图片
+ (void)saveImage:(UIImage *)image withLocalIdentifier:(void(^)(NSString *localIdentifier))localIdentifier {

//    UIImageWriteToSavedPhotosAlbum(image, nil, nil, nil);

    __block NSString *assetId = nil;

    [[PHPhotoLibrary sharedPhotoLibrary] performChanges:^{ // 这个block里保存一些"修改"性质的代码
        // 新建一个PHAssetCreationRequest对象，保存图片到"相机胶卷"
        // 返回PHAsset(图片)的字符串标识
        assetId = [PHAssetCreationRequest
creationRequestForAssetFromImage:image].placeholderForCreatedAsset.localIdentifier;

    } completionHandler:^(BOOL success, NSError * _Nullable error) {
        if (error) {
            NSLog(@"保存图片到相机胶卷中失败");
            return;
        }

        localIdentifier(assetId); // 将图片保存到相册中，并获取其唯一标识assetId及localIdentifier，用于后期的筛选
        NSLog(@"成功保存图片到相机胶卷中");
    }];

}
```

AnchoriteFiliGod
关注 - 0
粉丝 - 0

« 上一篇：万能星用于测试多个应用程序

posted on 2016-11-09 10:12 编辑

最新IT新闻:

· Google阻止一起利用恶意AdSense 广告的攻击
· 下一代iPhone：远程无线充电将为十年来最大创新
· 史上最严！谷歌公布安卓7.0 AOSP规范：快充都不能自己搞
· 乐视体育和兴业银行联合发了一张信用卡 除此外还搞了好多事
· B站VIP大会员制度大变：无法直接购买
» 更多新闻...

最新知识库文章:

· 循序渐进地代码重构
· 技术的正宗与野路子
· 陈皓：什么是工程师文化?
· 没那么难，谈CSS的设计模式
· 程序猿媳妇儿注意事项
» 更多知识库文章...