【OC】UIImagePickerController相关设置与问题

作者 寻形觅影 (/u/3af65dbeed74) + 关注
2017.03.28 17:08* 字数 885 阅读 7 评论 0 喜欢 0
(/u/3af65dbeed74)

用户选择头像功能是最常见的调用相机相册场景，下面就一这一场景为例简单介绍一下UIImagePickerController的使用。

一、 相关方法和属性详解：

1、

```
+ (BOOL)isSourceTypeAvailable:(UIImagePickerControllerSourceType)sourceType;
```

用于判断设备是否支持某一数据源。UIImagePickerControllerSourceType是系统枚举值：

```
typedef NS_ENUM(NSInteger, UIImagePickerControllerSourceType) {
    UIImagePickerControllerSourceTypePhotoLibrary,// 图库即相簿
    UIImagePickerControllerSourceTypeCamera,// 相机
    UIImagePickerControllerSourceTypeSavedPhotosAlbum// 相机胶卷
} __TVOS_PROHIBITED;
```

2、

```
+ (nullable NSArray<NSString *> *)availableMediaTypesForSourceType:(UIImagePicker
ControllerSourceType)sourceType;
```

该方法主要用于获得相机模式下支持的媒体类型：sourceType是UIImagePickerControllerSourceTypeCamera时打印数组
为：`("public.image","public.movie")` public.image表示静态图片，public.movie表示视频。当然也可以是图库或者是相机胶卷的，但是结果只有一个`("public.image")`
使用：

```
for (NSString* mediaType in [UIImagePickerController availableMediaTypesForSource
Type:UIImagePickerControllerSourceTypeCamera]) {
        if ([mediaType isEqualToString: (NSString *)kUTTypeImage]) {
            //支持拍照
            break;
        }
        if ([mediaType isEqualToString: (NSString *)kUTTypeMovie]) {
            //支持摄像
            break;
        }
    }
```

从上面的代码可以看到使用了两个常量：kUTTypeImage和kUTTypeMovie 这两个常量。在这里就要提一下UTI：iOS系统中为了更好的进行类型标识，而提供的一套共用的规范，也就是“Uniform Type Identifier”，一般称为“统一类型标识符”,简称为“UTI”。这两个常量是使用UTI定义的常量，表示"public.image","public.movie"。可以查看UTCoreTypes.h 文件, 具体可以自行百度一下UTI-iOS。(主要是码农我也了解的不多😭....)
3、

```
+ (BOOL)isCameraDeviceAvailable:(UIImagePickerControllerCameraDevice)cameraDevice
                   NS_AVAILABLE_IOS(4_0);
```

用于判断设备是否支持前置摄像头 / 后置摄像头。
UIImagePickerControllerCameraDevice是系统枚举值：

```
typedef NS_ENUM(NSInteger, UIImagePickerControllerCameraDevice) {
    UIImagePickerControllerCameraDeviceRear,// 后摄像头
    UIImagePickerControllerCameraDeviceFront // 前摄像头
} __TVOS_PROHIBITED;
```

4、

```
+ (BOOL)isFlashAvailableForCameraDevice:(UIImagePickerControllerCameraDevice)came
raDevice           NS_AVAILABLE_IOS(4_0);
```

用于判断设备前置摄像头 / 后置摄像头是否支持闪光灯。
5、

```
+ (nullable NSArray<NSNumber *> *)availableCaptureModesForCameraDevice:(UIImagePi
ckerControllerCameraDevice)cameraDevice NS_AVAILABLE_IOS(4_0);
```

获得指定摄像头上的可用捕获模式，返回的是NSNumber *类型，捕获模式是枚举类型：

```
typedef NS_ENUM(NSInteger, UIImagePickerControllerCameraCaptureMode) {
    UIImagePickerControllerCameraCaptureModePhoto,//拍照模式
    UIImagePickerControllerCameraCaptureModeVideo//视频录制模式
} __TVOS_PROHIBITED;
```

e.g：

```
for (NSNumber* captureMode in [UIImagePickerController availableCaptureModesForCa
meraDevice:(UIImagePickerControllerCameraDeviceRear)]) {
        if ([captureMode integerValue] == UIImagePickerControllerCameraCaptureMod
ePhoto) {
            JRLog(@"拍照模式");
        }
        if ([captureMode integerValue] == UIImagePickerControllerCameraCaptureMod
eVideo) {
            JRLog(@"视频录制模式 ");
        }
    }
```

6、一些属性：

```
@property(nullable,nonatomic,weak) id <UINavigationControllerDelegate, UIImagePickerControllerDelegate> delegate; // 必须遵循里面的两个代理

@property(nonatomic) UIImagePickerControllerSourceType sourceType; // 指定 ImagePickerController的数据源，默认是UIImagePickerControllerSourceTypePhotoLibrary

@property(nonatomic,copy) NSArray<NSString *> *mediaTypes;// 设置相机模式或者图库、相机胶卷模式下支持的媒体类型，默认是包含kUTTypeImage即("public.image")的数组，所以拍照时可以不用设置；但是当要录像的时候必须设置，可以设置为kUTTypeVideo（视频，但不带声音）或者kUTTypeMovie（视频并带有声音）

@property(nonatomic) BOOL allowsEditing NS_AVAILABLE_IOS(3_1); // 是否允许编辑，默认是NO，关于这里详见（PS.about-1）.

@property(nonatomic) NSTimeInterval videoMaximumDuration NS_AVAILABLE_IOS(3_1); //视频最大录制时长，默认为10分钟。video properties apply only if mediaTypes includes kUTTypeMovie-仅适用于如果媒体类型（mediaTypes）包括kUTTypeMovie视频属性。

@property(nonatomic) UIImagePickerControllerQualityType videoQuality NS_AVAILABLE_IOS(3_1);//设置视频的质量，为枚举类型 ，关于这里详见（PS.about-2）.

@property(nonatomic) BOOL showsCameraControls NS_AVAILABLE_IOS(3_1); //是否显示摄像头控制面板，默认为YES。 // 只有sourceType 先设置为UIImagePickerControllerSourceTypeCamera的情况下可用，否则会崩溃

@property(nullable, nonatomic,strong) __kindof UIView * cameraOverlayView  NS_AVAILABLE_IOS(3_1); //摄像头上覆盖的视图(浮于UIImagePickerController视图的最上方)，可用通过这个视图来自定义拍照或录像界面。值得注意的是当拍照/录像完成后该界面依然存在。

@property(nonatomic) CGAffineTransform cameraViewTransform NS_AVAILABLE_IOS(3_1);   // 设置摄像头拍摄角度的形变，如 cameraPC.cameraViewTransform = CGAffineTransformMakeRotation(M_PI_2);// 平面旋转90度

@property(nonatomic) UIImagePickerControllerCameraCaptureMode cameraCaptureMode NS_AVAILABLE_IOS(4_0); // 设置摄像头捕获模式，默认是UIImagePickerControllerCameraCaptureModePhoto

@property(nonatomic) UIImagePickerControllerCameraDevice      cameraDevice      NS_AVAILABLE_IOS(4_0); // 设置摄像头设备，默认UIImagePickerControllerCameraDeviceRear

@property(nonatomic) UIImagePickerControllerCameraFlashMode   cameraFlashMode   NS_AVAILABLE_IOS(4_0); // 设置闪光灯模式，枚举类型（详见PS.about-3:），默认是UIImagePickerControllerCameraFlashModeAuto.
```

PS.about-1:

一、调用相机时：

allowsEditing = YES:

allowsEditing = NO:

在设置为NO后，拍摄后的图片是不能拖动和缩放的，即不能编辑只能使用原图。
二、使用图库或相机胶卷时：
allowsEditing = YES:

allowsEditing = NO时，只要你选中了图片不会出现上图中界面而知直接会调用代理方法，即无法查看大图及编辑。

PS.about-2：
该属性的枚举类型：

```
typedef NS_ENUM(NSInteger, UIImagePickerControllerQualityType) {
    UIImagePickerControllerQualityTypeHigh = 0,       //高清质量
    UIImagePickerControllerQualityTypeMedium = 1,     // 中等质量，适合WiFi传
输
    UIImagePickerControllerQualityTypeLow = 2,         // 低质量，适合蜂窝网
    UIImagePickerControllerQualityType640x480 NS_ENUM_AVAILABLE_IOS(4_0) =
 3,    // VGA质量，一般不常用
    UIImagePickerControllerQualityTypeIFrame1280x720 NS_ENUM_AVAILABLE_IOS
(5_0) = 4,
    UIImagePickerControllerQualityTypeIFrame960x540 NS_ENUM_AVAILABLE_IOS(
5_0) = 5,
} __TVOS_PROHIBITED;
```

该属性默认是 UIImagePickerControllerQualityTypeMedium，如果相机设备不支持调整视频质量将使用默认值。

PS.about-3:

```
typedef NS_ENUM(NSInteger, UIImagePickerControllerCameraFlashMode) {
    UIImagePickerControllerCameraFlashModeOff  = -1,// 闪光灯关闭
    UIImagePickerControllerCameraFlashModeAuto = 0,// 自动
    UIImagePickerControllerCameraFlashModeOn   = 1// 打开
} __TVOS_PROHIBITED;
```

7、

```
- (void)takePicture NS_AVAILABLE_IOS(3_1); //编程方式拍照，在该方法执行结束会直接调用代
理方法
- (BOOL)startVideoCapture NS_AVAILABLE_IOS(4_0);//编程方式开始录制视频
- (void)stopVideoCapture  NS_AVAILABLE_IOS(4_0);//编程方式停止录制视频
```

二、代理方法及常见内部操作

```
- (void)imagePickerController:(UIImagePickerController *)picker didFinishPickingM
ediaWithInfo:(NSDictionary<NSString *,id> *)info; // 当拍照/录像完成或选择图片完成后都
会走此代理方法

- (void)imagePickerControllerDidCancel:(UIImagePickerController *)picker;// 当点击
原生界面中的取消按钮后会执行此代理方法
```

值得注意的是picker不能关闭自己所以一般要在这两个代理方法内执行dismiss方法！

对于录制好的视频或者照片的一些操作都在 `- (void)imagePickerController:(UIImagePickerController *)picker didFinishPickingMediaWithInfo:(NSDictionary<NSString *,id> *)info` 内进行一般就是照片和视频的存储问题。
info携带信息的key如下所示：

```
UIKIT_EXTERN NSString *const UIImagePickerControllerMediaType __TVOS_PROHIBITED;
// an NSString (UTI, i.e. kUTTypeImage)获取媒体类型信息
UIKIT_EXTERN NSString *const UIImagePickerControllerOriginalImage __TVOS_PROHIBIT
ED; // a UIImage 获取原始照片
UIKIT_EXTERN NSString *const UIImagePickerControllerEditedImage __TVOS_PROHIBITED
; // a UIImage 获取编辑后的照片
UIKIT_EXTERN NSString *const UIImagePickerControllerCropRect __TVOS_PROHIBITED; /
/ an NSValue (CGRect) 获得包含编辑界面的剪裁窗的CGRect值(以此尺寸配合原图得到的新图为正方形
图，同时该图旋转了90度)
UIKIT_EXTERN NSString *const UIImagePickerControllerMediaURL __TVOS_PROHIBITED; /
/ an NSURL //获取拍摄后图片或者视频路径（在SourceType为UIImagePickerControllerSourceTy
peCamera时可以获取到）
UIKIT_EXTERN NSString *const UIImagePickerControllerReferenceURL NS_AVAILABLE_IOS
(4_1) __TVOS_PROHIBITED; //选取到的图片或者视频所在素材库的URL（在SourceType不为UIImageP
ickerControllerSourceTypeCamera时可以获取到）
UIKIT_EXTERN NSString *const UIImagePickerControllerMediaMetadata NS_AVAILABLE_IO
S(4_1) __TVOS_PROHIBITED;  // 获取对象的元数据在SourceType为UIImagePickerControllerS
ourceTypeCamera时可以获取到）
UIKIT_EXTERN NSString *const UIImagePickerControllerLivePhoto NS_AVAILABLE_IOS(9_
1) __TVOS_PROHIBITED;  // a PHLivePhoto，得到LivePhoto对象
```

关于图片和视频的保存到相簿，系统有几个扩展方法：

```
UIKIT_EXTERN void UIImageWriteToSavedPhotosAlbum(UIImage *image, __nullable id co
mpletionTarget, __nullable SEL completionSelector, void * __nullable contextInfo)
 __TVOS_PROHIBITED;
//将照片保存到相簿，其回调方法是：
- (void)image:(UIImage *)image didFinishSavingWithError:(NSError *)error contextI
nfo:(void *)contextInfo;

UIKIT_EXTERN BOOL UIVideoAtPathIsCompatibleWithSavedPhotosAlbum(NSString *videoPa
th) NS_AVAILABLE_IOS(3_1) __TVOS_PROHIBITED;
// 是否允许视频保存到相簿

UIKIT_EXTERN void UISaveVideoAtPathToSavedPhotosAlbum(NSString *videoPath, __null
able id completionTarget, __nullable SEL completionSelector, void * __nullable co
ntextInfo) NS_AVAILABLE_IOS(3_1) __TVOS_PROHIBITED;
// 将视频保存到相簿，其回调方法是：
 - (void)video:(NSString *)videoPath didFinishSavingWithError:(NSError *)error co
ntextInfo:(void *)contextInfo;
```

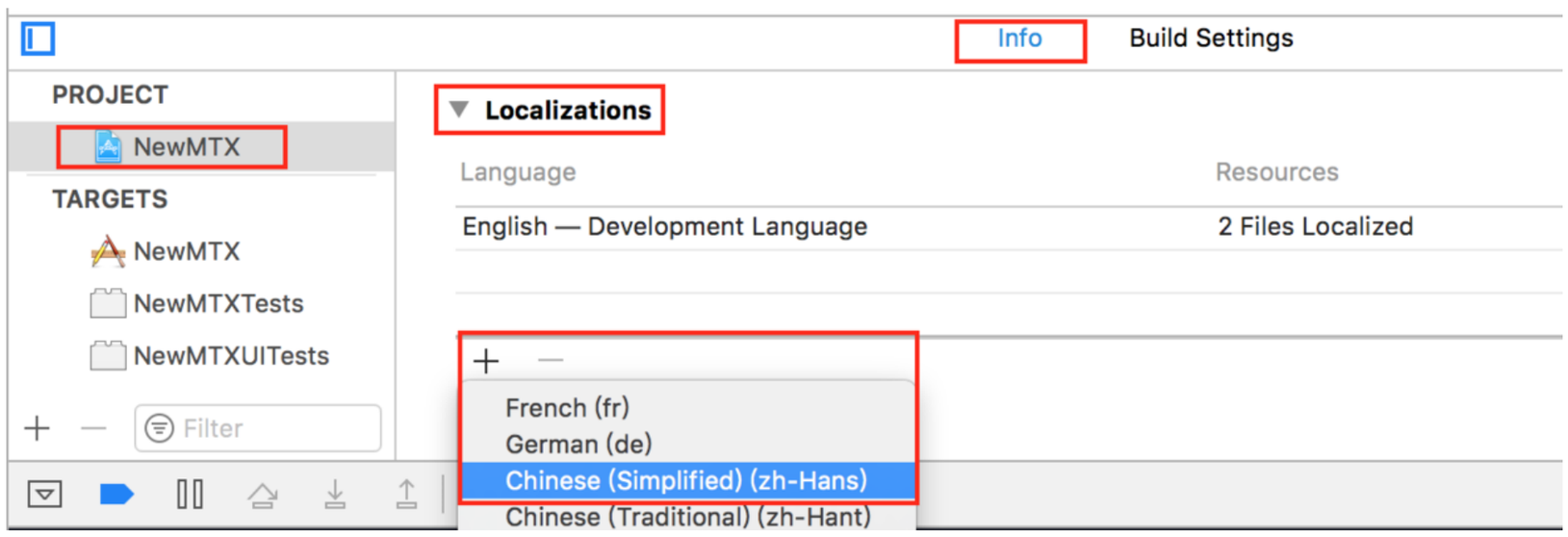
e.g：

```
// UIImagePickControllerDelegate
- (void)imagePickerController:(UIImagePickerController *)picker didFinishPickingM
ediaWithInfo:(NSDictionary<NSString *,id> *)info
{
//以下均是在使用摄像头拍照情况下，在使用相册或图库功能时大致相同
    // 获取媒体类型信息
    NSString *mediaType=[info objectForKey:UIImagePickerControllerMediaType];
    if ([mediaType isEqualToString:(NSString *)kUTTypeImage]) {//如果是拍照
        UIImage *image;
        //如果允许编辑则获得编辑后的照片，否则获取原始照片
        if (picker.allowsEditing) {
            image=[info objectForKey:UIImagePickerControllerEditedImage];//获取编
辑后的照片
        }else{
            image=[info objectForKey:UIImagePickerControllerOriginalImage];//获取
原始照片
        }
        /*
        CGRect cropRect = [[info objectForKey:UIImagePickerControllerCropRect] CG
RectValue];
        UIImage * cropImage=[info objectForKey:UIImagePickerControllerOriginalIma
ge];
        UIImage *rotatedOriginalImage = [cropImage imageRotatedByDegrees:90.0];
        CGImageRef imageRef = CGImageCreateWithImageInRect(rotatedOriginalImage.C
GImage, cropRect);
        UIImage * resultImage = [UIImage imageWithCGImage:imageRef];
        //以此法获得的图片即为编辑后的图片其中-imageRotatedByDegrees: 方法是将图片顺时针转
动90度的方法
        */
        UIImageWriteToSavedPhotosAlbum(image, self, @selector(image: didFinishSav
ingWithError: contextInfo:), nil);// 若保存到相簿后没有其他操作可以：UIImageWriteToSav
edPhotosAlbum(image, nil, nil, nil);

    }else if([mediaType isEqualToString:(NSString *)kUTTypeMovie]){//如果是录制视频
        NSURL *url=[info objectForKey:UIImagePickerControllerMediaURL];//视频路径
        NSString *urlStr=[url path];
        if (UIVideoAtPathIsCompatibleWithSavedPhotosAlbum(urlStr)) {
            //保存视频到相簿，注意也可以使用ALAssetsLibrary来保存
            UISaveVideoAtPathToSavedPhotosAlbum(urlStr, self, @selector(video:did
FinishSavingWithError:contextInfo:), nil);//保存视频到相簿
        }
    }
    [picker dismissViewControllerAnimated:YES completion:^{ }];
}

- (void)imagePickerControllerDidCancel:(UIImagePickerController *)picker
{
    [picker dismissViewControllerAnimated:YES completion:^{ }];
}
// 照片保存到相簿的回调
- (void)image:(UIImage *)image didFinishSavingWithError:(NSError *)error contextI
nfo:(void *)contextInfo
{
    if (error) {
        NSLog(@"保存照片过程中发生错误，错误信息:%@",error.localizedDescription);
    }else{
        NSLog(@"照片保存成功.");
    }
}
// 视频保存到相簿的回调
- (void)video:(NSString *)videoPath didFinishSavingWithError:(NSError *)error con
textInfo:(void *)contextInfo
{
    if (error) {
        NSLog(@"保存视频过程中发生错误，错误信息:%@",error.localizedDescription);
    }else{
        NSLog(@"视频保存成功.");
        //录制完之后可以使用AVPlayer自动播放
        NSURL *url=[NSURL fileURLWithPath:videoPath];
        NSLog(@"视频路径=%@", url);
    }
}
```

```
//该方法是UIImage的分类中自定义的方法，其中DegreesToRadians（）是一个宏定义：#define Degre
esToRadians(x) (M_PI*(x)/180.0)// 将度数转化成弧度
- (UIImage *)imageRotatedByDegrees:(CGFloat)degrees{
    UIView *rotatedViewBox = [[UIView alloc] initWithFrame:CGRectMake(0,0,self.si
ze.height, self.size.width)];
    CGAffineTransform t = CGAffineTransformMakeRotation(DegreesToRadians(degrees)
);
    rotatedViewBox.transform = t;
    CGSize rotatedSize = rotatedViewBox.frame.size;
    UIGraphicsBeginImageContext(rotatedSize);
    CGContextRef bitmap = UIGraphicsGetCurrentContext();
    CGContextTranslateCTM(bitmap, rotatedSize.width/2, rotatedSize.height/2);
    CGContextRotateCTM(bitmap, DegreesToRadians(degrees));
    CGContextScaleCTM(bitmap, 1.0, -1.0);
    CGContextDrawImage(bitmap, CGRectMake(-self.size.height / 2, -self.size.width
 / 2, self.size.height, self.size.width), [self CGImage]);
    UIImage *newImage = UIGraphicsGetImageFromCurrentImageContext();
    UIGraphicsEndImageContext();
    return newImage;
}
```

三、关于界面的一些问题（会不断添加~）

1、原生界面中调用起UIImagePickerController后会发现所有控件上名称都是英文的，若想改为中文有很简单的一种方法：PROJECT --> Info --> Localizations 然后添加简体中文即可。



评论

智慧如你，不想发表一点想法咩~

被以下专题收入，发现更多相似内容

+
我的专题

iOS
Dev...