# UIImagePickerController

## 1.+(BOOL)isSourceTypeAvailable:(UIImagePickerControllerSourceType)sourceType; 检查指定源是否在设备上可用。

//检查照片源是否可用

## [UIImagePickerController isSourceTypeAvailable:UIImagePickerControllerSourceTypePhotoLibrary]2.allowsEditing 默认NO

是否允许编辑

允许编辑.

[imagePicker setAllowsEditing:YES];

## 3. videoMaximumDuration

设置UIImagePicker的最大视频持续时间.默认10分钟

## 4. + availableMediaTypesForSourceType: // 指定源可用的媒体种类

 //获得相机模式下支持的媒体类型

NSArray\* availableMediaTypes = [UIImagePickerController availableMediaTypesForSourceType:UIImagePickerControllerSourceTypeCamera];

## 5. sourceType

设置UIImagePicker照片源类型，默认有3种。

照片源类型

UIImagePickerControllerSourceTypeCamera 照相机

UIImagePickerControllerSourceTypePhotoLibrary 照片库（通过同步存放的，用户不能删除）

UIImagePickerControllerSourceTypeSavedPhotosAlbum 保存的照片（通过拍照或者截屏保存的，用户可以删除）

## 6.UIImagePicker使用步骤:

    检查指定源是否可用. isSourceTypeAvailable:方法.  
    检查可用媒体(视频还是只能是图片) availableMediaTypesForSourceType:方法.  
    设置界面媒体属性mediaTypes property.  
    显示界面使用presentViewController:animated:completion:方法.iPad中是popover形式.需要确保sourceType有效.   
    相关操作,移除视图.   
  
  
如果想创建一个完全自定义界面的image picker来浏览图片,使用 Assets Library Framework Reference中的类. (AV Foundation Programming Guide 中的 “Media Capture and Access to Camera” )  
  
 

## 7.设置源

    + availableMediaTypesForSourceType: // 指定源可用的媒体种类  
    + isSourceTypeAvailable: // 指定源是否在设备上可用  
      sourceType   
  
// 运行相关接口前需要指明源类型.必须有效,否则抛出异常. picker已经显示的时候改变这个值,picker会相应改变来适应.默认 UIImagePickerControllerSourceTypePhotoLibrary.

## 8.设置picker属性

      allowsEditing //是否可编辑  
      delegate   
      mediaTypes   
  
// 指示picker中显示的媒体类型.设置每种类型之前应用availableMediaTypesForSourceType:检查一下.如果为空或者array中类型都不可用,会发生异常.默认 kUTTypeImage, 只能显示图片.  
  
    

## 9.video选取参数

      videoQuality //视频拍摄选取时的编码质量.只有mediaTypes包含kUTTypeMovie时有效.  
      videoMaximumDuration //秒,video最大记录时间,默认10分钟.只用当mediaTypes包含kUTTypeMovie时有效.

## 10.自定义界面

      showsCameraControls   
  
// 指示 picker 是否显示默认的camera controls.默认是YES,设置成NO隐藏默认的controls来使用自定义的overlay view.(从而可以实现多选而不是选一张picker就dismiss了).只有 UIImagePickerControllerSourceTypeCamera源有效,否则NSInvalidArgumentException异常.  
  
      cameraOverlayView   
  
//自定义的用于显示在picker之上的view.只有当源是UIImagePickerControllerSourceTypeCamera时有效.其他时候使用抛出NSInvalidArgumentException异常.  
  
      cameraViewTransform   
  
//预先动画.只影响预先图像,对自定义的overlay view和默认的picker无效.只用当picker的源是UIImagePickerControllerSourceTypeCamera时有效,否则NSInvalidArgumentException异常.

## 11.选取媒体

    – takePicture   
  
//使用摄像头选取一个图片。自定义overlay可以多选。已经有图片正在选取是调用无效，必须要等delegate收到 imagePickerController:didFinishPickingMediaWithInfo:消息后才能再次选取。非UIImagePickerControllerSourceTypeCamera源会导致异常。  
  
    – startVideoCapture   
  
    – stopVideoCapture   
  
//结束视频选取，之后系统调用delegate的 imagePickerController:didFinishPickingMediaWithInfo:方法。

## 12.设置摄像头

      cameraDevice //使用的镜头（默认后置的）  
    + isCameraDeviceAvailable: // 摄像设备是否可用.  
    + availableCaptureModesForCameraDevice: // 设备可用的选取模式   
      cameraCaptureMode //相机捕获模式  
      cameraFlashMode //闪光灯模式(默认自动)  
    + isFlashAvailableForCameraDevice: // 是否有闪光能力

## 13.UIImagePickerControllerDelegate

使用UIImageWriteToSavedPhotosAlbum保存图像, UISaveVideoAtPathToSavedPhotosAlbum保存视频. 4.0后使用writeImageToSavedPhotosAlbum:metadata:completionBlock:保存元数据.  
  
    - (void)imagePickerController:(UIImagePickerController \*)picker didFinishPickingMediaWithInfo:(NSDictionary \*)info  
  
//包含选择的图片或者一个视频的URL,详见“Editing Information Keys.”  
  
//如果是设置可编辑属性,那么picker会预显示选中的媒体,编辑后的与初始的都会保存在info中.  
  
    – imagePickerControllerDidCancel:  
    – imagePickerController:didFinishPickingImage:editingInfo://Deprecated in iOS 3.0  
  
NSString \*const UIImagePickerControllerMediaType;// 媒体类型  
NSString \*const UIImagePickerControllerOriginalImage;// 原始未编辑的图像  
NSString \*const UIImagePickerControllerEditedImage;// 编辑后的图像  
NSString \*const UIImagePickerControllerCropRect;// 源图像可编辑(有效?)区域  
NSString \*const UIImagePickerControllerMediaURL;// 视频的路径  
NSString \*const UIImagePickerControllerReferenceURL;// 原始选择项的URL  
NSString \*const UIImagePickerControllerMediaMetadata;// 只有在使用摄像头并且是图像类型的时候有效.包含选择图像信息的字典类型

## 14. UIImagePickerController小例子

UIImagePickerController的代理需要遵守这两个协议.<UIImagePickerControllerDelegate, UINavigationControllerDelegate>

#pragma mark 选择照片

- (void)selectPhoto

{

// 1. 首先判断照片源是否可用

if ([UIImagePickerController isSourceTypeAvailable:UIImagePickerControllerSourceTypePhotoLibrary]) {

// 0）实例化控制器

UIImagePickerController \*picker = [[UIImagePickerController alloc]init];

// 1）设置照片源

[picker setSourceType:UIImagePickerControllerSourceTypePhotoLibrary];

// 2) 设置允许修改

[picker setAllowsEditing:YES];

// 3) 设置代理

[picker setDelegate:self];

// 4) 显示控制器

[self presentViewController:picker animated:YES completion:nil];

} else {

NSLog(@"照片源不可用");

}

}

#pragma mark - imagePicker代理方法

- (void)imagePickerController:(UIImagePickerController \*)picker didFinishPickingMediaWithInfo:(NSDictionary \*)info

{

UIImage \*image = info[@"UIImagePickerControllerEditedImage"];

[\_imageButton setImage:image forState:UIControlStateNormal];

// 关闭照片选择器

[self dismissViewControllerAnimated:YES completion:nil];

// 需要将照片保存至应用程序沙箱，由于涉及到数据存储，同时与界面无关

// 可以使用多线程来保存图像

dispatch\_async(dispatch\_get\_global\_queue(DISPATCH\_QUEUE\_PRIORITY\_DEFAULT, 0), ^{

// 保存图像

// 1. 取图像路径

NSArray \*docs = NSSearchPathForDirectoriesInDomains(NSDocumentDirectory, NSUserDomainMask, YES);

NSString \*imagePath = [docs[0]stringByAppendingPathComponent:@"abc.png"];

// 2. 转换成NSData保存

NSData \*imageData = UIImagePNGRepresentation(image);

[imageData writeToFile:imagePath atomically:YES];

});

}