[UIImage系列]－图片的任意旋转、剪切及拉伸

作者 我给你的爱写在西元前 (/u/1aae751ace03) + 关注
2017.03.19 12:33 字数 726 阅读 3 评论 0 喜欢 0
(/u/1aae751ace03)

本文讲述关于UIImage的一系列操作。

首先，来强调一个知识点：

对于UIImage的旋转、剪切等操作，并非是对其进行动画操作，因为动画本身的执行是针对存放UIImage的那个View来进行的，也就是说如果你通过动画来进行裁剪、旋转、变形等操作，他并没有对原有的图片本身有任何实质性的改变。事实上，对于UIImage的这些任意处理，我们是通过获取它的图形上下文Context来进行操作的。

接下来我们具体看看如何进行图片的任意处理：

1. 图片的任意角度旋转
   实现思路：首先将需要处理的图片渲染到她的context上，然后对得到的这个context进行我们需要的旋转处理，最后再将旋转过的context转化为UIImage的类型。

具体的操作步骤如下图所示：

```
@implementation UIImage (ImageRotate)

- (UIImage *)imageRotateInDegree:(float)degree
{
    // 获取图片的宽高
    size_t width = (size_t)(self.size.width * self.scale);
    size_t height = (size_t)(self.size.height * self.scale);

    // 每行图片数据的字节
    size_t bytesPerRow = width * 4;
    // 设置图片的透明度
    CGImageAlphaInfo alphaInfo = kCGImageAlphaPremultipliedFirst;

    // 配置上下文参数
    CGContextRef bmContext = CGBitmapContextCreate(NULL, width, height, 8,
        bytesPerRow, CGColorSpaceCreateDeviceRGB(), kCGBitmapByteOrderDefault |
        alphaInfo);
    if (!bmContext) {
        return nil;
    }
    CGContextDrawImage(bmContext, CGRectMake(0, 0, width, height), self.CGImage);

    // 旋转
    UInt8 *data = (UInt8 *)CGBitmapContextGetData(bmContext);

    vImage_Buffer src = {data, height, width, bytesPerRow};
    vImage_Buffer dest = {data, height, width, bytesPerRow};
    Pixel_8888 bgColor = {0, 0, 0, 0};

    vImageRotate_ARGB8888(&src, &dest, NULL, degree, bgColor,
        kvImageBackgroundColorFill);

    // context ---> UIImage
    CGImageRef rotateImage_ref = CGBitmapContextCreateImage(bmContext);
    UIImage *rotateImage = [UIImage imageWithCGImage:rotateImage_ref scale:self.
        scale orientation:self.imageOrientation];

    return rotateImage;
}
```

创建旋转分类、实现旋转处理

在我们的VC中就只需要引入头文件进行方法调用即可：

```objc
- (void)rotateImageTest
{
    UIImage *image = [UIImage imageNamed:@"pic.jpg"];
    UIImage *newImage = [image imageRotateInDegree:90 * 0.0175];
    UIImageWriteToSavedPhotosAlbum(newImage, nil, nil, nil);
}
```

旋转方法的调用

---

1. 图片的任意位置剪切

   实现思路：首先获取自己需要处理的图片，然后对他的图形上下文进行剪切绘制处理，最后将处理后的内容转化为图片形式。

具体的操作步骤如下图所示：

```objc
@implementation UIImage (ImageCut)

- (UIImage *)imageCutSize:(CGRect)cutRect
{
    CGImageRef subImageRef = CGImageCreateWithImageInRect(self.CGImage, cutRect);
    CGRect smallRect = CGRectMake(0, 0, CGImageGetWidth(subImageRef),
        CGImageGetHeight(subImageRef));

    UIGraphicsBeginImageContext(smallRect.size);

    CGContextRef context = UIGraphicsGetCurrentContext();
    CGContextDrawImage(context, smallRect, subImageRef);

    UIImage *cutImage = [UIImage imageWithCGImage:subImageRef];
    UIGraphicsEndImageContext();

    return cutImage;

}
```

创建剪切分类、实现剪切处理

在我们的VC中就只需要引入头文件进行方法调用即可：

```objc
- (void)cutImageTest
{
    UIImage *image = [UIImage imageNamed:@"pic.jpg"];
    UIImage *newImage = [image imageCutSize:CGRectMake(100, 100, 200, 300)];
    UIImageWriteToSavedPhotosAlbum(newImage, nil, nil, nil);
}
```

剪切方法的调用

---

1. 图片的任意拉伸

   实现思路：首先获取自己需要处理的图片，然后对他的图形上下文进行拉伸处理，最后将处理后的内容转化为图片形式。

具体的操作步骤如下图所示：

```objc
@implementation UIImage (ImageStretch)

- (UIImage *)imageStretchWithSize:(CGSize)size
{
    UIGraphicsBeginImageContext(size);

    // 根据 drawInRect 方法和新的size重新绘制
    [self drawInRect:CGRectMake(0, 0, size.width, size.height)];
    UIImage *image = UIGraphicsGetImageFromCurrentImageContext();

    UIGraphicsEndImageContext();

    return image;

}
```

创建拉伸分类、实现拉伸处理

```objc
- (void)stretchImageTest
{
    UIImage *image = [UIImage imageNamed:@"pic.jpg"];
    UIImage *newImage = [image imageStretchWithSize:CGSizeMake(200, 500)];
    UIImageWriteToSavedPhotosAlbum(newImage, nil, nil, nil);
}
```

拉伸方法的调用

以上就是通过创建分类，处理图形上下文来实现图片的任意旋转、剪切以及拉伸操作的方法。其实通过上述方法，我们可以看出对于图片的这些处理，其根本就在于对图形上下文ImageContext的一系列处理以及他和图片本身的转化。

另外、两个小建议：(1)大家在处理图片的时候尽量去使用同一张图片对其进行不同的操作，这样可以更直观的看出来到底有怎样不同的效果。(2)关于图形上下文这一方面的内容在平时的一般开发中并不常用，知识点上可能会觉得有好多之前未曾接触过的内容，这就需要大家多去搜索和、学习和积累了。

希望我的文章能对大家有所帮助。

我是姣爷、我在简书、和你们一起、加油！

被以下专题收入，发现更多相似内容

+
我的专题

iOS
Dev...