**[iOS 扩展机制category与associative](http://blog.csdn.net/qi_ruihua/article/details/17299749)**

category和associative作为objective-c 扩展机制的两个特性，category可以通过它来扩展方法；associative可以通过它来扩展属性。

在iOS开发过程中，前者category比较常见，也比较简单，这里就不说了，这里主要说一下associative；

后者associative相对用的就比较少，要用associative就必须使用#import<objc/runtime.h>，然后调用objc\_setAssociatedObject 和 objc\_getAssociatedObject  方法分别为属性添加setter 和  getter方法，就可以实现属性扩展了。

下面介绍一下这两个方法：

①：void objc\_setAssociatedObject(id object, void \*key, id value, objc\_AssociationPolicy policy)

其中的参数分别是：

**Parameters**

object：  The source object for the association.

key： The key for the association.

value：  The value to associate with the key key for object. Pass nil to clear an existing association.

policy：  The policy for the association

其中的policy有

enum {

   OBJC\_ASSOCIATION\_ASSIGN = 0,

   OBJC\_ASSOCIATION\_RETAIN\_NONATOMIC = 1,

   OBJC\_ASSOCIATION\_COPY\_NONATOMIC = 3,

   OBJC\_ASSOCIATION\_RETAIN = 01401,

   OBJC\_ASSOCIATION\_COPY = 01403

};

②：id objc\_getAssociatedObject(id object, void \*key)

**Parameters**

object：  The source object for the association.

key：  The key for the association.

**Return Value**

The value associated with the key key for object.

都比较简单，下面就通过一个demo来说明吧！

我这里是扩展UIImageview为其添加一个方法和一个属性。

category的头文件：

**[cpp]** [view plain](http://blog.csdn.net/crayondeng/article/details/9123329?reload" \o "view plain" \t "_blank)[copy](http://blog.csdn.net/crayondeng/article/details/9123329?reload" \o "copy" \t "_blank)

1. #import <UIKit/UIKit.h>
3. @interface UIImageView (associate)
5. @property(nonatomic,strong)NSString\* myString;
7. -(void)Output;
9. @end

category的实现文件：

**[cpp]** [view plain](http://blog.csdn.net/crayondeng/article/details/9123329?reload" \o "view plain" \t "_blank)[copy](http://blog.csdn.net/crayondeng/article/details/9123329?reload" \o "copy" \t "_blank)

1. #import <objc/runtime.h>
2. #import "UIImageView+associate.h"
4. static void \* MyKey = (void \*)@"MyKey";
6. @implementation UIImageView (associate)
8. -(NSString\*)myString {
9. return objc\_getAssociatedObject(self, MyKey);
10. }
12. -(void)setMyString:(NSString \*)string {
13. objc\_setAssociatedObject(self, MyKey, string, OBJC\_ASSOCIATION\_COPY\_NONATOMIC);
14. }
16. -(void)Output {
17. NSLog(@"output mystring:%@",self.myString);
18. }
19. @end

说明：头文件中添加了一个属性和一个方法，在实现文件中使用associative特性为属性重写了setter和getter方法，都比较简单。

测试一下：

**[cpp]** [view plain](http://blog.csdn.net/crayondeng/article/details/9123329?reload" \o "view plain" \t "_blank)[copy](http://blog.csdn.net/crayondeng/article/details/9123329?reload" \o "copy" \t "_blank)

1. UIImageView \*imageView = [[UIImageView alloc] initWithImage:[UIImage imageNamed:@"Icon@2x.png"]];
2. imageView.bounds = CGRectMake(50, 50, 100, 100);
3. imageView.myString = @"hello world";
4. [self.view addSubview:imageView];
5. [imageView Output];

运行后，模拟器上就显示一个图片，终端输出：**output mystring:hello world**