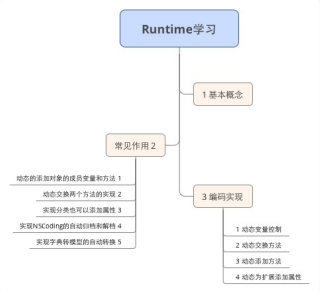
谈Runtime机制和使用的整体化梳理

5 回复　58 查看

(https://www.shiyanlou.com/user/8490) 实验楼管理员 (https://www.shiyanlou.com/vip)　　2016-01-21 18:00

技术分享 (https://www.shiyanlou.com/questions/?tag=技术分享)

相比“凌波微步”的swift，Object-C被誉为“如来神掌”。传说Runtime就是支持这“如来神掌”说法的最好体现。听起来总是这么的神秘高级，于是总能在各个论坛看到碎片资料，时间一长总记不住哪里是哪里，每次都要打开好几个网页。这种记不住象现显然是知识体系还不完整重要体现。还是自己从Runtime的思想到动手代码呈现上做出总结尚为上策。



分享到微博

全部回答

实验楼管理员 (https://www.shiyanlou.com/user/8490) (https://www.shiyanlou.com/vip)

(https://www.shiyanlou.com/user/8490)

一.基本概念

- RunTime简称运行时,就是系统在运行的时候的一些机制，其中最主要的是消息机制。
- 对于C语言，函数的调用在编译的时候会决定调用哪个函数（ C语言的函数调用请看这里 ）。编译完成之后直接顺序执行，无任何二义性。
- OC的函数调用成为消息发送。属于动态调用过程。在编译的时候并不能决定真正调用哪个函数（事实证明，在编译阶段，OC可以调用任何函数，即使这个函数并未实现，只要申明过就不会报错。而C语言在编译阶段就会报错）。
- 只有在真正运行的时候才会根据函数的名称找 到对应的函数来调用。

官网文档还提供关于传统和现代版本Runtime的说明

> In the legacy runtime, if you change the layout of instance variables in a class, you must recompile classes that inherit from it.
>
> In the modern runtime, if you change the layout of instance variables in a class, you do not have to recompile classes that inherit from it.In addition, the modern runtime supports instance variable synthesis for declared properties (see Declared Properties in The Objective-C Programming Language).

二.知晓OC的方法调用在Runtime中具体的实现

1.OC代码调用一个方法

```
[self.loginBt login];
```

2.在编译时RunTime会将上述代码转化成[发送消息]

```
objc_msgSend(self.loginB,@selector(login));
```

三.常见的作用

> 既然是“如来神掌”，简直可以无法无天啦，当街拦下一个人问道“这是马还是鹿啊？”，那人看是Runtime大人惧怕道“Runtime大人，您说是马就是马，是鹿就是鹿~”。Runtime大快“wow哈哈哈~，见你乖巧，我也不为难于你。你缺头驴是吧？，本大人现在造一头送于你，迁回家便是！喔~哈哈哈”。

呵呵，扯远了，回到Runtime作用上。无所不能的事情就不一一介绍了，梳理下较为可能用的几个地方：

1. 动态的添加对象的成员变量和方法

2. 动态交换两个方法的实现

3. 实现分类也可以添加属性

4. 实现NSCoding的自动归档和解档

5. 实现字典转模型的自动转换

2016-01-21 18:01

四.编写代码实现

1. 动态变量控制

1）Sense：

```
Teacher: What's your name?
XiaoMing: My name is XiaoMing.
Teacher: Pardon?
XiaoMing: My name is __
```

在程序当中，假设XiaoMing的name原来的值为XiaoMing，后来被Runtime偷换了一个名字叫Minggo。那么，Runtime是如何做到的呢？

2）Step：

①动态获取XiaoMing类中的所有属性[当然包括私有]

```
Ivar *ivar = class_copyIvarList([self.xiaoMing class], &count);
```

②遍历属性找到对应name字段

```
const char *varName = ivar_getName(var);
```

③修改对应的字段值成Minggo

```
object_setIvar(self.xiaoMing, var, @"Minggo");
```

3）Show Code：

```
-(void)answer{
    unsigned int count = 0;
    Ivar *ivar = class_copyIvarList([self.xiaoMing class], &count);
    for (int i = 0; i<count; i++) {
        Ivar var = ivar[i];
        const char *varName = ivar_getName(var);
        NSString *name = [NSString stringWithUTF8String:varName];

        if ([name isEqualToString:@"_englishName"]) {
            object_setIvar(self.xiaoMing, var, @"Minggo");
            break;
        }
    }
    NSLog(@"XiaoMing first answer is %@",self.xiaoMing.englishName);
    self.nameTf.text = self.xiaoMing.englishName;
}
```

2016-01-21 18:02

实验楼管理员 (https://www.shiyanlou.com/user/8490) (https://www.shiyanlou.com/vip)

(https://www.shiyanlou.com/user/8490)

2. 动态交换方法

1）Sense：

```
Teacher: What's your name?
XiaoMing: My name is XiaoMing.
Teacher: Pardon?
XiaoMing: My name is __
```

在程序当中，假设XiaoMing的第一次回答为firstSay，后来被Runtime交换了一个名字叫secondSay的方法，最终再调用firstSay的时候，其实是调用了secondSay的实现。那么，Runtime是如何做到的呢？

2）Step：

①动态找到firstSay和secondSay方法

```
Method m1 = class_getInstanceMethod([self.xiaoMing class], @selector(firstSay));
Method m2 = class_getInstanceMethod([self.xiaoMing class], @selector(secondSay));
```

②交换两个方法

```
method_exchangeImplementations(m1, m2);
```

3）Show Code：

```
-(void)answer{

    Method m1 = class_getInstanceMethod([self.xiaoMing class], @selector(firstSay));
    Method m2 = class_getInstanceMethod([self.xiaoMing class], @selector(secondSay));

    method_exchangeImplementations(m1, m2);
    NSString *secondName = [self.xiaoMing firstSay];

    self.nameTf.text = secondName;
    NSLog(@"XiaoMing:My name is %@",secondName);
}
```

2016-01-21 18:02

实验楼管理员 (https://www.shiyanlou.com/user/8490) (https://www.shiyanlou.com/vip)

(https://www.shiyanlou.com/user/8490)

3. 动态添加方法

1）Sense：

```
Teacher: Where is LiLei from?
XiaoMing: I don't know.
Teacher: Guess?.
LiHua: He is from __
```

在程序当中，假设XiaoMing的中没有guess这个方法，后来被Runtime添加一个名字叫guess的方法，最终再调用guess方法做出相应。那么，Runtime是如何做到的呢？

2）Step：

①动态给XiaoMing类中添加guess方法：

```
class_addMethod([self.xiaoMing class], @selector(guess), (IMP)guessAnswer, "v@:");
```

这里参数地方说明一下：

- (IMP)guessAnswer 意思是guessAnswer的地址指针;
- "v@:" 意思是，v代表无返回值void，如果是i则代表int；@代表 id sel; : 代表 SEL _cmd;
- “v@:@@” 意思是，两个参数的没有返回值。

②调用guess方法响应事件：

```
[self.xiaoMing performSelector:@selector(guess)];
```

③编写guessAnswer的实现：

```
void guessAnswer(id self,SEL _cmd){
    NSLog(@"He is from GuangTong");
}
```

这个有两个地方留意一下：

1.void的前面没有+、-号，因为只是C的代码。

2.必须有两个指定参数(id self,SEL _cmd)

3）Show Code：

```
-(void)answer{
    class_addMethod([self.xiaoMing class], @selector(guess), (IMP)guessAnswer, "v@:");
    if ([self.xiaoMing respondsToSelector:@selector(guess)]) {

        [self.xiaoMing performSelector:@selector(guess)];

    } else{
        NSLog(@"Sorry,I don't know");
    }
    self.cityTf.text = @"GuangTong";
}

void guessAnswer(id self,SEL _cmd){

    NSLog(@"He is from GuangTong");

}
```

2016-01-21 18:02

4. 动态为Category扩展加属性

这一点上有两点要表达一下：第一，XCode运行你在Category的.h文件申明@Property，编译通过，但运行时如果没有Runtime处理，进行赋值取值，就马上报错。第二，这一点是iOS面试当中经常面到的问题：如何给扩展添加属性？。

1）Sense：

```
Teacher: What's your Chinese name?
XiaoMing: I have no one.
LiHua: You should have one.
LiHua: Your Chinese name is __
```

在程序当中，假设XiaoMing的中没有chineseName这个属性，后来被Runtime添加一个名字叫chineseName的属性。那么，Runtime是如何做到的呢？

2）Step：

①申明chineseName属性

```
#import "XiaoMing.h"

@interface XiaoMing (MutipleName)

@property(nonatomic,copy) NSString *chineseName;

@end
```

②动态添加属性和实现方法

```
#import "XiaoMing+MutipleName.h"
#import <objc/runtime.h>

@implementation XiaoMing (MutipleName)

char cName;

-(void)setChineseName:(NSString *) chineseName{
       objc_setAssociatedObject(self, &cName, chineseName, OBJC_ASSOCIATION_COPY_NONATOMIC);
}

-(NSString *)chineseName{
    return objc_getAssociatedObject(self, &cName);
}

@end
```

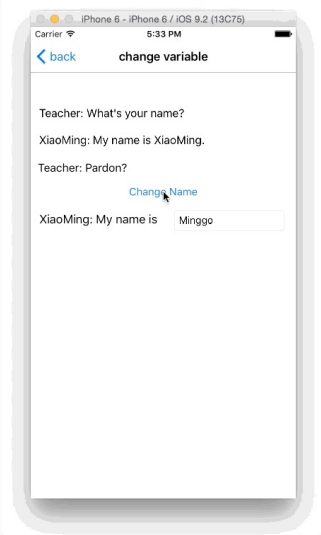
③使用chineseName属性

```
-(void)answer{
    NSLog(@"My Chinese name is %@",self.xiaoMing.chineseName);
    self.chineseNameTf.text = self.xiaoMing.chineseName;
}
```

3）Show Code：

上边就是最要的Code了。以下更精彩。

五.效果图更直观



六.源码下载地址更详细

https://github.com/minggo620/iOSRuntimeLearn.git (https://github.com/minggo620/iOSRuntimeLearn.git)

文章地址：http://www.jianshu.com/p/8916ad5662a2

作者：minggo

2016-01-21 18:03

登录后才能回答问题哟~

我要提问

标签

Linux (https://www.shiyanlou.com/questions/?tag=Linux) Python (https://www.shiyanlou.com/questions/?tag=Python)

C/C++ (https://www.shiyanlou.com/questions/?tag=C/C++) 实验环境 (https://www.shiyanlou.com/questions/?tag=实验环境)

技术分享 (https://www.shiyanlou.com/questions/?tag=技术分享) 功能建议 (https://www.shiyanlou.com/questions/?tag=功能建议)

课程需求 (https://www.shiyanlou.com/questions/?tag=课程需求) Java (https://www.shiyanlou.com/questions/?tag=Java)

其他 (https://www.shiyanlou.com/questions/?tag=其他) NodeJS (https://www.shiyanlou.com/questions/?tag=NodeJS)

SQL (https://www.shiyanlou.com/questions/?tag=SQL) Hadoop (https://www.shiyanlou.com/questions/?tag=Hadoop)

Web (https://www.shiyanlou.com/questions/?tag=Web) 常见问题 (https://www.shiyanlou.com/questions/?tag=常见问题)

PHP (https://www.shiyanlou.com/questions/?tag=PHP) Shell (https://www.shiyanlou.com/questions/?tag=Shell)

HTML (https://www.shiyanlou.com/questions/?tag=HTML) Git (https://www.shiyanlou.com/questions/?tag=Git)

HTML5 (https://www.shiyanlou.com/questions/?tag=HTML5) 信息安全 (https://www.shiyanlou.com/questions/?tag=信息安全)

网络 (https://www.shiyanlou.com/questions/?tag=网络) GO (https://www.shiyanlou.com/questions/?tag=GO)

NoSQL (https://www.shiyanlou.com/questions/?tag=NoSQL) Android (https://www.shiyanlou.com/questions/?tag=Android)

训练营 (https://www.shiyanlou.com/questions/?tag=训练营) Ruby (https://www.shiyanlou.com/questions/?tag=Ruby)

Perl (https://www.shiyanlou.com/questions/?tag=Perl)

相关问题

brackets：前端开发工程师必备编辑器 (https://www.shiyanlou.com/questions/3197)

优秀的计算机编程类博客和文章 (https://www.shiyanlou.com/questions/3200)

40个Java集合面试问题和答案 (https://www.shiyanlou.com/questions/3187)

程序员和设计师必备的20个CSS工具 (https://www.shiyanlou.com/questions/3186)

Web开发的十佳HTML5响应式框架 (https://www.shiyanlou.com/questions/3177)