



# iOS Runtime

## Runtime 介绍

Objective-C 是一个动态语言，这意味着它不仅需要一个编译器，也需要一个运行时系统来动态得创建类和对象、进行消息传递和转发。理解 Objective-C 的 Runtime 机制可以帮我们更好的了解这个语言，适当的时候还能对语言进行扩展，从系统层面解决项目中的一些设计或技术问题。了解 Runtime，要先了解它的核心 - 消息传递（Messaging）。

## Runtime objc\_msgSend执行流程

- 在OC中的方法调用，其实都是转换为objc\_msgSend函数的调用，给receiver（方法调用者）发送一条消息(selector 方法名)
- objc\_msgSend的执行流程可以分为3个阶段
  - 消息发送
  - 动态方法解析
  - 消息转发

ps：如果这三个阶段都找不到方法的话就会报经典错误 unrecognized selector sent to instance

## 消息发送

1. 通过 receiver（消息接收者）的isa 指针 找到 receiver 的 Class
2. 在 Class 的 cache（方法缓存）的散列表中寻找对应的 IMP（方法实现）
3. 如果在 cache（方法缓存）中没有找到对应的 IMP，就继续在 Class 的 methodlist（方法列表）中找对应的 selector，如果找到，填充到 cache（方法缓存）中，并返回 selector；
4. 如果在 class 中，没有找到这个 selector，就继续在它的 superClass 中的 cache（方法缓存）中查找，如果查到缓存到 Class 中的 cache（方法缓存）中
5. 如果没有查到，就在 superClass 中的 methodlist 中查找，并且缓存到 Class 中的 cache（方法缓存）中





## 动态方法解析

源码是否曾经有过动态解析 如果没有 则看开发者有没有实现 `+resolveInstanceMethod:` , `resolveClassMethod:` 方法来动态解析方法 如果实现了该方法, 标记为已经动态解析。会重新回到消息发送流程中(在Class的cache中查找方法)这一步开始执行。 如果已经动态解析过 则会进入下一步 消息转发阶段。

例子:

```
ZQPerson * p = [[ZQPerson alloc] init];
    [p test];
打印结果: -[ZQPerson other]
```

- person类中没有实现test方法 而是在 `resolveInstanceMethod` 中动态指向了other方法的实现

person.h

```
@interface ZQPerson : NSObject
- (void)test;
@end
```

person.m

```
- (void)other{
    NSLog(@"%s", __func__);
}

+ (BOOL)resolveInstanceMethod:(SEL)sel{
    Method otherMethod = class_getInstanceMethod(self, @selector(other));
    if (sel == @selector(test)) {
        class_addMethod(self, sel, method_getImplementation(otherMethod), method_getTypeEnc);
        return YES;
    }
    return [super resolveInstanceMethod:sel];
}
```

## 消息转发



[首页](#) ▼[探索掘金](#)[登录](#)

返回值位nil，就会调用obj\_msgSend(返回值, SEL)，如果 forwardingTargetForSelector 返回为 nil，则要实现 methodSignatureForSelector , forwardInvocation ,这两个方法。

方法签名：返回值类型，参数类型

```
- (NSMethodSignature *)methodSignatureForSelector:(SEL)aSelector{
    if (aSelector == @selector(test)) {
        return [NSMethodSignature signatureWithObjCTypes:"v@:"];
    }
    return [super methodSignatureForSelector:aSelector];
}

//NSInvocation封装了一个方法调用，包括：方法调用者、方法名、方法参数
//    anInvocation.target // 方法调用者
//    anInvocation.selector 方法名
//    [anInvocation getArgument:NULL atIndex:0]; 参数
- (void)forwardInvocation:(NSInvocation *)anInvocation{
    anInvocation.target = [[ZQCat alloc] init]; //指定消息转发接收的对象
    [anInvocation invoke]; //执行方法

    //或者这样写
    [anInvocation invokeWithTarget:[ZQCat alloc] init];
}
```

关注下面的标签，发现更多相似文章

[iOS](#)

**GGCode** Lv1 iOS页面仔  
获得点赞 1 · 获得阅读 316

[关注](#)

## 安装掘金浏览器插件

打开新标签页发现好内容，掘金、GitHub、Dribbble、ProductHunt 等站点内容轻松获取。快来安装掘金浏览器插件获取高质量内容吧！

相关推荐



[首页](#) ▼[探索掘金](#)[登录](#)

老司机技术周报 · 1天前 · iOS / SwiftUI

## 【WWDC20】10037 - SwiftUI 中的 App 要领

👍 8

💬

FengyunSky · 4天前 · iOS

## 一文读懂iOS线程调用栈原理

👍 12

💬

ZenonHuang · 7天前 · iOS

## iOS 的自动构建流程

👍 61

💬 7

JeremyHuang37 · 1天前 · iOS

## 【译】自定义 Collection View Layout -- 一个简单的模板

👍

💬 2

掘金酱 · 27天前 · iOS / Android / 前端 / 后端 / 程序员

## 🏆 掘金征文 | 2020与我的年中总结

👍 94

💬 109

Chouee · 3天前 · iOS

## 造轮子 - UITableView字母索引条

👍 7

💬 1

路过看风景 · 2天前 · iOS

## CocoaPods原理 及 组件化

👍 3

💬

阿里巴巴淘系技术 · 5天前 · iOS

## Apple Widget：下一个顶级流量入口？

👍 9

💬

路过看风景 · 1天前 · iOS

## iOS事件处理 UIResponder

👍

💬



