Runtime 10种用法（没有比这更全的了）

作者 wg689 (/u/e03c395306c5) + 关注
2016.04.29 10:51 字数 2888 阅读 5221 评论 17 喜欢 203 阅读 5221 评论 17 喜欢 203
(/u/e03c395306c5)

阅读了多篇运行时的文章,感觉都很不错,从几篇文章里面提取一些个人认为比较重要的,偏实战的知识点的摘录,另外还结合了个人的创造形成此文。再牛逼的技术和实战结合才有意义，本篇介绍技术尽量和实战联系起来，有些没讲明白的，我给的链接里面都有，所以我就不赘述了，如果觉得可以就点个赞吧，哈哈。不需要打赏。

1)替换系统方法,

2)字典转model,

3)归档,

4)万能控制器跳转

四个偏实战的方法从不同的文章中聚合到这里.没讲明白的麻烦看客去我给的链接里面看看，他们都讲了

对文章的内容我用一张图介绍吧(图只管明了,我喜欢,大家也喜欢)

#📣appstore审核问答群:369250107,建了个仓库:,github[专门解决苹果拒绝各种问题汇总的仓库](https://github.com/wg689/Solve-App-Store-Review-Problem)

runtime 文章收集:

---

runtime 文章专题 (http://www.jianshu.com/collection/dc947eab6af3)
(13+关于runtime的)

如下4篇内容有相似的 小白都可以看懂runtime,值得细看

文① OC最实用的runtime总结，面试、工作你看我就足够了！
(http://www.jianshu.com/p/ab966e8a82e2)

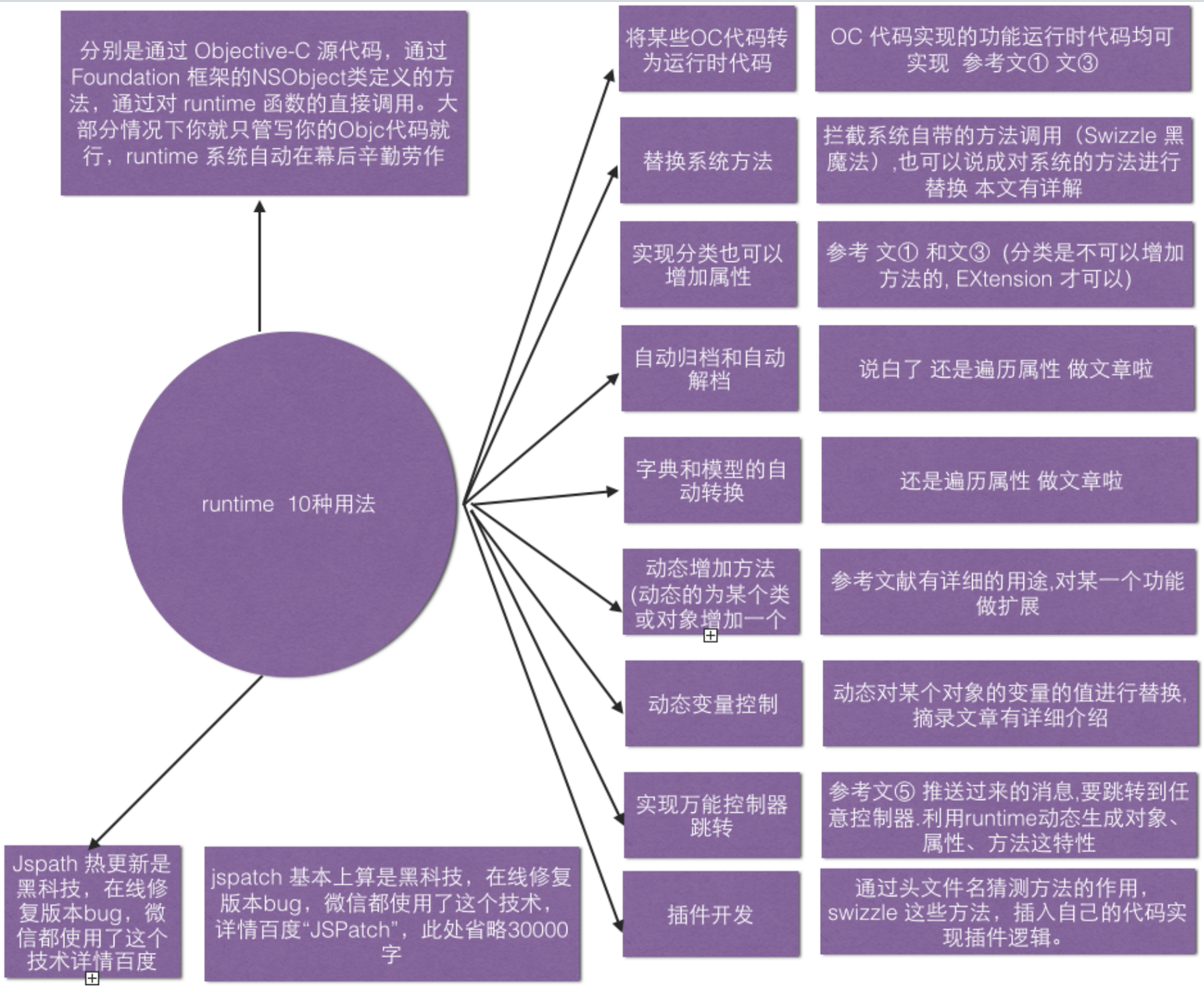
文② 让你快速上手Runtime (http://www.jianshu.com/p/e071206103a4)

文③ runtime详解 (http://www.jianshu.com/p/46dd81402f63)

文④ 详解runtime运行时机制 (http://www.jianshu.com/p/1e06bfee99d0)

文⑤ 万能控制器跳转 (http://www.cocoachina.com/ios/20150824/13104.html)

先用一张图对文章做一个介绍吧(图只管,易懂方便回忆,我喜欢,大家也喜欢)



what(runtime 是什么)

Runtime基本是用C和汇编写的，可见苹果为了动态系统的高效而作出的努力。你可以在这里 (http://opensource.apple.com//source/objc4/)下到苹果维护的开源代码。苹果和GNU各自维护一个开源的runtime版本，这两个版本之间都在努力的保持一致。Objective-C 从三种不同的层级上与 Runtime 系统进行交互，***分别是通过 Objective-C 源代码***，***通过 Foundation 框架的NSObject类定义的方法***，通过对 runtime 函数的直接调用。大部分情况下你就只管写你的Objc代码就行，runtime 系统自动在幕后辛勤劳作着。

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，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。

where(哪些地方使用runtime)

1.将某些OC代码转为运行时代码，探究底层，比如block的实现原理（上边已讲到）；

2.拦截系统自带的方法调用（Swizzle 黑魔法）,也可以说成对系统的方法进行替换，比如拦截imageNamed:、viewDidLoad、alloc；文 / 滕先洪（简书作者） 原文链接：http://www.jianshu.com/p/ab966e8a82e2 (http://%E6%96%87%EF%BC%8F%E6%BB%95%E5%85%88%E6%B4%AA%EF%BC%88%E7%AE%80%E4%B9%A6%E4%BD%9C%E8%80%85%EF%BC%89%20%C2%A0%E5%8E%9F%E6%96%87%E9%93%BE%E6%8E%A5%EF%BC%9Ahttp://www.jianshu.com/p/ab966e8a82e2)

需求：比如iOS6 升级 iOS7 后需要版本适配，根据不同系统使用不同样式图片（拟物化和扁平化），如何通过不去手动一个个修改每个UIImage的imageNamed：方法就可以实现为该方法中加入版本判断语句？

步骤：

a、为UIImage建一个分类（UIImage+Category）

b、在分类中实现一个自定义方法，方法中写要在系统方法中加入的语句，比如版本判断[参考]

```
+ (UIImage*)xh_imageNamed:(NSString*)name {

doubleversion = [[UIDevicecurrentDevice].systemVersiondoubleValue];

if(version >=7.0) {

// 如果系统版本是7.0以上，使用另外一套文件名结尾是‘_os7’的扁平化图片

name = [name stringByAppendingString:@"_os7"];    }

return[UIImagexh_imageNamed:name];

}
```

c、分类中重写UIImage的load方法，实现方法的交换（只要能让其执行一次方法交换语句，load再合适不过了）

```
+ (void)load {

// 获取两个类的类方法

Method  m1 = class_getClassMethod([UIImageclass],@selector(imageNamed:));

Method m2 = class_getClassMethod([UIImageclass],@selector(xh_imageNamed:));

// 开始交换方法实现

method_exchangeImplementations(m1, m2);

}
```

注意：自定义方法中最后一定要再调用一下系统的方法，让其有加载图片的功能，但是由于方法交换，系统的方法名已经变成了我们自定义的方法名（有点绕，就是用我们的名字能调用系统的方法，用系统的名字能调用我们的方法），这就实现了系统方法的拦截！

利用以上思路，我们还可以给 NSObject 添加分类，统计创建了多少个对象，给控制器添加分类，统计有创建了多少个控制器，特别是公司需求总变的时候，在一些原有控件或模块上添加一个功能，建议使用该方法！

3.实现分类也可以增加属性；

4.实现NSCoding的自动归档和自动解档；(不用对每个属性edcode和decode了,如果几十个属性一个个的encode和decode真的很麻烦啊,使用运行时可以遍历出每个对象的属性,数组的方式遍历eccode,decode)

用运行时的归档方法

```objc
- (void)encodeWithCoder:(NSCoder *)encoder

{
    unsigned int count = 0;

    Ivar *ivars = class_copyIvarList([Movie class], &count);

    for (int i = 0; i<count; i++) {
        // 取出i位置对应的成员变量
        Ivar ivar = ivars[i];
        // 查看成员变量

        const char *name = ivar_getName(ivar);
        // 归档
        NSString *key = [NSString stringWithUTF8String:name];
        id value = [self valueForKey:key];
        [encoder encodeObject:value forKey:key];
    }
    free(ivars);
}
```

```objc
- (id)initWithCoder:(NSCoder *)decoder

{
    if (self = [super init]) {
        unsigned int count = 0;
        Ivar *ivars = class_copyIvarList([Movie class], &count);
        for (int i = 0; i<count; i++) {
            // 取出i位置对应的成员变量
            Ivar ivar = ivars[i];
            // 查看成员变量
            const char *name = ivar_getName(ivar);
            // 归档
            NSString *key = [NSString stringWithUTF8String:name];
            id value = [decoder decodeObjectForKey:key];
            // 设置到成员变量身上
            [self setValue:value forKey:key];
        }
        free(ivars);
    }
    return self;
}
```

不用运行时的归档方法:(还好只有5个属性,如果20个,30个或者后台突然增加了属性,这么直接写死估计代码就不灵了)

```objc
- (void)encodeWithCoder:(NSCoder *)encoder
{
    [encoder encodeObject:_nickname forKey:@"nickname"];
    [encoder encodeObject:_username forKey:@"username"];
    [encoder encodeObject:_imageStr forKey:@"imageStr"];
    [encoder encodeObject:_payOrderId forKey:@"payOrderId"];
    [encoder encodeObject:_expertId forKey:@"expertId"];

}


- (id)initWithCoder:(NSCoder *)decoder
{
    if (self = [super init]) {
        _nickname = [decoder decodeObjectForKey:@"nickname"];
        _username = [decoder decodeObjectForKey:@"username"];
        _imageStr = [decoder decodeObjectForKey:@"imageStr"];
        _payOrderId = [decoder decodeObjectForKey:@"payOrderId"];
        _expertId = [decoder decodeObjectForKey:@"expertId"];
    }
    return self;
}
```

5.实现字典和模型的自动转换(核心就是可以遍历出字典中的每个属性,json解析中大牛框架都用了这个特性,包括MJEXtension,YYModel，jsonModel都是将json转换为字典,再遍历字典中的每个属性来进行modle的转换)。

MJextension 使用运行时转换的json转model的部分代码摘录

```
unsigned int outCount = 0;
/**
    class_copyIvarList 成员变量，提示有很多第三方框架会使用 Ivar，
        能够获得更多的信息
    但是: 在 swift 中，由于语法结构的变化，使用 Ivar 非常不稳定，经常
        会崩溃!
    class_copyPropertyList 属性
    class_copyMethodList 方法
    class_copyProtocolList 协议
    */
objc_property_t *properties = class_copyPropertyList(c, &
    outCount);

// 2.遍历每一个成员变量
for (unsigned int i = 0; i<outCount; i++) {
    MJProperty *property = [MJProperty
        cachedPropertyWithProperty:properties[i]];
    property.srcClass = c;
    [property setKey:[self propertyKey:property.name]
        forClass:self];
    [property setObjectClassInArray:[self
        propertyObjectClassInArray:property.name] forClass:
        self];
```

YYModel json转model 核心代码 摘录

```
        initWithProperty:properties[i]];
        if (info.name) propertyInfos[info.name] = info;
    }
    free(properties);
}

unsigned int ivarCount = 0;
Ivar *ivars = class_copyIvarList(cls, &ivarCount);
if (ivars) {
    NSMutableDictionary *ivarInfos = [NSMutableDictionary new];
    _ivarInfos = ivarInfos;
    for (unsigned int i = 0; i < ivarCount; i++) {
        YYClassIvarInfo *info = [[YYClassIvarInfo alloc] initWithIvar:
            ivars[i]];
        if (info.name) ivarInfos[info.name] = info;
    }
    free(ivars);
}
```

JsonModel json字典转model 摘录

```
while (class != [JSONModel class]) {
    //JMLog(@"inspecting: %@",
        NSStringFromClass(class));

    unsigned int propertyCount;
    objc_property_t *properties =
        class_copyPropertyList(class, &propertyCount)
        ;

    //loop over the class properties
    for (unsigned int i = 0; i < propertyCount; i++)
    {

        JSONModelClassProperty* p =
            [[JSONModelClassProperty alloc] init];

        //get property name
        objc_property_t property = properties[i];
        const char *propertyName = property_getName
            (property);
        p.name = @(propertyName);

        //JMLog(@"property: %@", p.name);
```

获取属性的列表的方法是字典转模型的比较核心的方法,

> OBJC_EXPORT Ivar *class_copyIvarList(Class cls, unsigned int *outCount)
>
> __OSX_AVAILABLE_STARTING(__MAC_10_5, __IPHONE_2_0);

致此可以下个结论吗?

基本上主流的json 转model 都少不了,使用运行时动态获取属性的属性名的方法,来进行字典转模型替换,字典转模型效率最高的(耗时最短的)的是KVC,其他的字典转模型是在KVC 的key 和Value 做处理,动态的获取json 中的key 和value ,当然转换的过程中,第三方框架需要做一些判空啊,镶嵌的逻辑处理, 再进行KVC 转模型.这句代码 [xx setValue:value forKey:key];无论JsonModle,YYKIt,MJextension 都少不了[xx setValue:value forKey:key];这句代码的,不信可以去搜,这是字典转模型的核心方法,

6)动态增加方法（动态的为某个类或对象增加一个方法,摘录文章中有详细介绍）

7)动态变量控制（动态对某个对象的变量的值进行替换,摘录文章有详细介绍）

8)实现万能控制器跳转

产品来一变态需求,推送过来的消息,要跳转到任意控制器.利用runtime动态生成对象、属性、方法这特性，我们可以先跟服务端商量好，定义跳转规则，比如要跳转到A控制器，需要传属性id、type，那么服务端返回字典给我，里面有控制器名，两个属性名跟属性值，客户端就可以根据控制器名生成对象，再用kvc给对象赋值，这样就搞定了

9)插件开发

> 插件入门
>
> XCode 有个很坑爹的地方，就是它并不官方支持插件开发，官方没有文档，XCode 也没有开源，但由于 XCode 是 Objective-C 写的，OC 动态性太强大，导致在这么封闭的情况下民间还是可以做出各种插件，其核心开发方式就是：
>
> dump 出 Xcode 所有头文件，知道 Xcode 里有哪些类和接口。
>
> 通过头文件方法名猜测方法的作用，swizzle 这些方法，插入自己的代码实现插件逻辑。
>
> 通过 NSNotificationCenter 监听各种事件的发生。
>
> 更详细的开发教程网上有不少文章，有兴趣的自行搜索吧。

10)Jspath 热更新 也是使用运行时，jspatch 基本上算是黑科技，在线修复版本bug，微信都使用了这个技术，详情百度“JSPatch”，此处省略30000字

- 作者开发经验总结的文章推荐,持续更新学习心得笔记

[Runtime 10种用法（没有比这更全的了）](http://www.jianshu.com/p/3182646001d1)

[成为iOS顶尖高手，你必须来这里(这里有最好的开源项目和文章)]
(http://www.jianshu.com/p/8dda0caf47ea)

[iOS逆向Reveal查看任意app 的界面](http://www.jianshu.com/p/060745d5ecc2)

[JSPatch （实时修复App Store bug）学习(一)](http://www.jianshu.com/p/344db07a2374)

[iOS 高级工程师是怎么进阶的(补充版20+点)](http://www.jianshu.com/p/1f2907512046)

[扩大按钮(UIButton)点击范围(随意方向扩展哦)](http://www.jianshu.com/p/ce2d3191224f)

[最简单的免证书真机调试(原创)](http://www.jianshu.com/p/c724e6282819)

[通过分析微信app,学学如何使用@2x,@3x图片](http://www.jianshu.com/p/99f1f924ae45)

[TableView之MVVM与MVC之对比](http://www.jianshu.com/p/d690b5d97201)

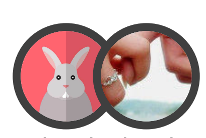
[使用MVVM减少控制器代码实战(减少56%)](http://www.jianshu.com/p/f85363c82ea1)

[ReactiveCocoa添加cocoapods 配置图文教程及坑总结]
(http://www.jianshu.com/p/66f0c7e1ced8)

我发布的文章 (/nb/2432502)

如果觉得我的文章对您有用，请随意打赏。您的支持将鼓励我继续创作!

打赏支持

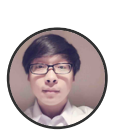


(/u/(c5f9d45f072b4)2af8)

喜欢 | 203

更多分享

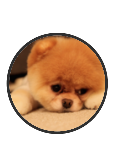
(http://cwb.assets.jianshu.io/notes/images/3723392/weib



写下你的评论...

17条评论 只看作者

按时间正序 按时间倒序 按喜欢排序



lifution (/u/bb834b9e6c9d)

2楼 · 2016.04.30 23:34

(/u/bb834b9e6c9d)

👍

赞 回复

> wg689 (/u/e03c395306c5)：@i_Steven (/users/bb834b9e6c9d) 谢谢
>
> 2016.05.03 13:53 回复
>
> ---
>
> 添加新评论

wzxjiang (/u/389c20d5a244)
3楼 · 2016.05.01 13:39
(/u/389c20d5a244)
不错

赞 回复

> wg689 (/u/e03c395306c5)：@WzxJiang (/users/389c20d5a244) 谢谢
>
> 2016.05.03 13:53 回复
>
> ---
>
> 添加新评论

worm (/u/b656eaa51771)
4楼 · 2016.05.12 23:23
(/u/b656eaa51771)
不错

赞 回复

lyking (/u/2289411316bc)
5楼 · 2016.07.10 17:02
(/u/2289411316bc)
不错，谢谢分享

赞 回复

下弦月丿 (/u/3ab8aeac97e4)
6楼 · 2016.07.12 11:24
(/u/3ab8aeac97e4)
不错！

赞 回复

不知蜕变的挣扎 (/u/ff69c3d53433)
7楼 · 2016.08.22 09:14
(/u/ff69c3d53433)
mark

赞 回复

指尖猿 (/u/231071c62af8)
8楼 · 2016.09.22 13:04
(/u/231071c62af8)
学习了.准备模仿

赞 回复

新地球说着一口陌生腔调 (/u/e26ea111cb5a)
9楼 · 2016.10.26 15:14
(/u/e26ea111cb5a)
我想知道微信使用jspatch解决了什么bug

赞 回复

> wg689 (/u/e03c395306c5)：@新地球说着一口陌生腔调 (/users/e26ea111cb5a) 的确使用了，解决了啥bug我不知道呢
>
> 2016.10.26 15:18 回复

新地球说着一口陌生腔调 (/u/e26ea111cb5a)：@wg689 (/users/e03c395306c5) 你能具体解释是如何进行热修复的吗?

2016.10.26 21:17 回复

wg689 (/u/e03c395306c5)：@新地球说着一口陌生腔调 (/users/e26ea111cb5a) 这个需要10000字，你百度哈😄

2016.10.26 22:04 回复

添加新评论 | 还有1条评论， 展开查看

---

hehtao (/u/471312f45b9f)

10楼 · 2016.12.01 16:04

(/u/471312f45b9f)
大神,求教个问题,怎么拦截系统的代理方法?不用继承的

赞 回复

wg689 (/u/e03c395306c5)：@hehtao (/users/471312f45b9f) 直接拦截那个方法

2016.12.01 18:27 回复

hehtao (/u/471312f45b9f)：@wg689 (/users/e03c395306c5) 可是在run time里我没找到获取代理方法实现的方法，直接用getinstanceMethod拿不到

2016.12.01 18:31 回复

添加新评论
