- 面试题出自技术群分享
- 欢迎转载，转载请注明出处：pmst-swiftgg
- 调试好可运行的源码 objc-runtime，官网找 objc4
- 完成度：60%
- 最后修订：2020/05/24

一面

1. __block 的作用和原理? 怎么做到 block 内部修改的?如何做变量提升的?怎么修改的内存位置?

block 面试题解答

2. block 在内存层面是如何分配的? 多个场景解释. (借助 clang 查看)

block 面试题解答

3. load 和 initial 方法的相同点与不同点? 调用方式的区别? (方法调用方式请与 messageSend 进行辨析?)如果父类、本类、子类分别都实现了 Load、initial 方法, 调用顺序分别是什么?

load 和 initialize 区别。

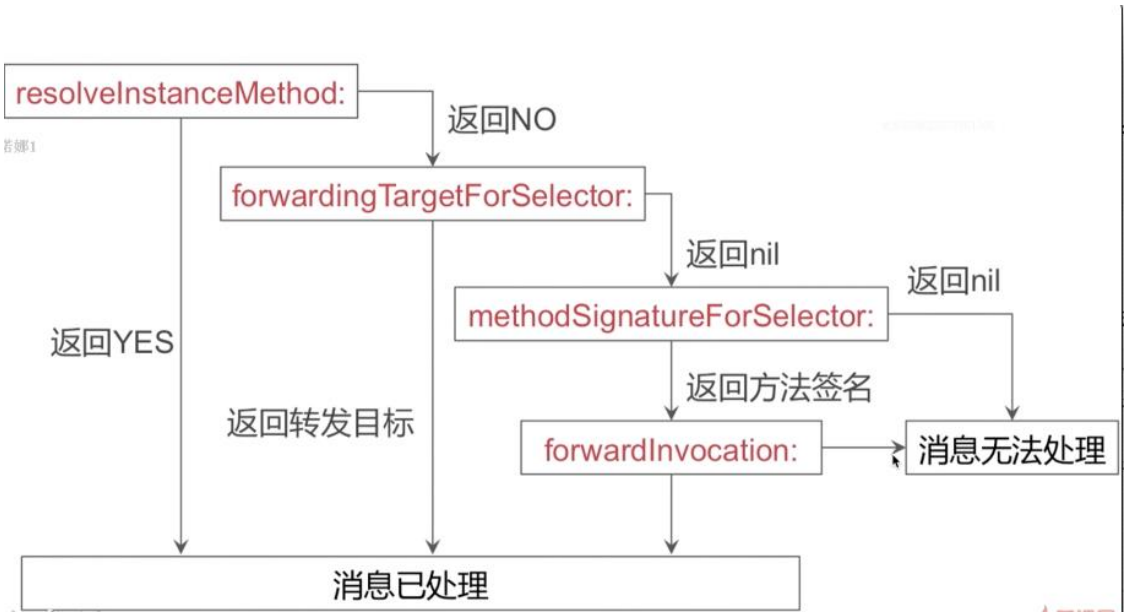
4. 发送消息的流程?

OC 中的方法调用，编译后的代码最终都会转成 `objc_msgSend(id , SEL, ...)` 方法进行调用，这个方法第一个参数是一个消息接收者对象，`runtime` 通过这个对象的 isa 指针找到这个对象的类对象，从类对象中的 cache 中查找(**哈希查找，bucket 桶实现**)是否存在 `SEL` 对应的 IMP，若不存在，则会在 `method_list` 中查找（二分查找或者顺序查找），如果还是没找到，则会到 `super_class` 中查找，仍然没找到的话，就会调用 `_objc_msgForward(id, SEL, ...)` 进行消息转发。

5. 自动释放池工作原理?什么时候会释放?

autoreleasePool 就是一个基于栈节点的双向链表，@autoreleasePool{} 分别对应 [autoreleasePool push] 和 [autoreleasePool pop]，前者就是将一个哨兵对象（nil） push 到栈中，花括号内所有发送一次 autorelease 消息的对象都会 push 到当前栈中，等到作用域结束 [autoreleasePool pop] 调用之时，会批量的 pop 对象直至碰到哨兵对象，对于 pop 出来的对象发送一次 release 消息。

6. 消息转发具体经过几步?具体到方法名, 参数,返回值



各个阶段的应用：

`resolveInstanceMethod:`

1. CoreData 数据模型属性动态创建，允许动态添加方法；

`forwardingTargetForSelector:`

1. 解决 NSTimer，CADisplayLink 循环引用问题；

`forwardInvocation:`

1. Aspects 库的实现；
2. NSProxy 多代理、多继承，解决 NSTimer，CADisplayLink 循环引用问题；
3. UIAppearance 实现；

7. 分类、扩展的区别?应用场景有哪些?

category:

- 运行时添加分类属性/协议/方法
- 分类添加的方法会“覆盖”原类方法，因为方法查找的话是从头至尾，一旦查找到了就停止了
- 同名分类方法谁生效取决于编译顺序，image 读取的信息是倒叙的，所以编译越靠后的越先读入
- 名字相同的分类会引起编译报错；

extension:

- 编译时决议
- 只以声明的形式存在，多数情况下就存在于 .m 文件中；
- 不能为系统类添加扩展

8. 算法:字符串中最大不重复子串 abcdcde -> abcd、abcdcdefg -> cdefg、abcdecfgh -> decfgh

```
#define MAX(a,b) ((a) < (b) ? (b) : (a))

char * maxWindow(char *s) {
    int dict[128] = {0};
    for(int i = 0; i < 128;i++)dict[i] = -1;

    int ansL = 0,ansR = 0;
    int head = 0;
    int res = 0;
    int len = strlen(s);

    for(int i = 0;i < len;i++){
        char ss = s[i];
        int k = dict[ss];
        if(k != -1){
            head = MAX(head, k);
        }
        dict[s[i]] = i + 1; // 存的是当前字符的下一个序号作为 head

        if (res < i - head +1) {
            ansL = head;
            ansR = i;
            res = i - head +1;
        }
    }

    char *ans = (char *)malloc(sizeof(char) * (ansR-ansL+2));
    int k = 0;
    for(int i = ansL; i < ansR+1;i++){
        ans[k++] = s[i];
    }
    ans[ansR-ansL+1] = '\0';
    return ans;
}
```

9. assign 与 weak 的辨析? weak 实现原理? (map 的 key value 具体是谁? 考虑如果有多个 weak 指针指向同一个对象的场景, 怎么进行 map 设计, 这么存?)

weak 面试题解答

key 就是对象指针进行哈希算法后的值，本来全局的 SideTables 表个数为 8 或 64，因此必定存在多个对象共用同一个 SideTable 的情况，不过 SideTable 中的 weak_table_t 和 RefcountMap 又是个哈希表，此时的 key 是对指针进行取反操作，另外还做了哈希碰撞处理。

```
// objc-private.h
#if TARGET_OS_IPHONE && !TARGET_OS_SIMULATOR
    enum { StripeCount = 8 };
#else
    enum { StripeCount = 64 };
#endif

static unsigned int indexForPointer(const void *p) {
  uintptr_t addr = reinterpret_cast<uintptr_t>(p);
  return ((addr >> 4) ^ (addr >> 9)) % StripeCount;
}
```

10. 对于结构体,如何访问成员变量.比如:类中成员变量(ivar)

通过实例对象首地址+偏移量的方式取值

```
@interface Person : NSObject
{
    NSString *_name;
    NSUInteger _age;
    NSDate *_birthDay;
    NSUInteger _value;
}
@end

@implementation Person

+ (instancetype)personWithName:(NSString *)name
                           age:(NSUInteger)age
                      birthday:(NSDate *)day
                         value:(NSUInteger)value {
    Person *p = [Person new];
    p->_name = name;
    p->_age = age;
    p->_birthDay = day;
    p->_value = value;
    return p;
}
@end

  @implementation ViewController

- (void)viewDidLoad {
    [super viewDidLoad];
    Person *p = [Person personWithName:@"pmst" age:18 birthday:[NSDate
now] value:0x12345678];
```

```
    Ivar name_ivar = class_getInstanceVariable(Person.class, "_name");
    ptrdiff_t name_offset = ivar_getOffset(name_ivar);
    void **pp = (__bridge void *)p + name_offset;
    NSLog(@"name : %@",(__bridge id)(*pp));

    Ivar age_ivar = class_getInstanceVariable(Person.class, "_age");
    ptrdiff_t age_offset = ivar_getOffset(age_ivar);
    void **ppp = (__bridge void *)p + age_offset;
    NSLog(@"name : %d",(__bridge id)(*ppp));

    Ivar day_ivar = class_getInstanceVariable(Person.class, "_birthDay
");
    ptrdiff_t day_offset = ivar_getOffset(day_ivar);
    void **pppp = (__bridge void *)p + day_offset;
    NSLog(@"name : %@",(__bridge id)(*pppp));

    Ivar value_ivar = class_getInstanceVariable(Person.class, "_value");
    ptrdiff_t value_offset = ivar_getOffset(value_ivar);
    void **ppppp = (__bridge void *)p + value_offset;
    NSLog(@"name : 0x%x",(__bridge id)(*ppppp));
}


@end
```

runtime 提供了 object_getIvar 方法返回是对象，注意 id *location = (id *)((char *)obj + offset) 其实也就是对象起始位置+偏移量，不过 setIvar 同样是针对 oc 对象，两者都没有获取基本数据类型的接口：

```
id object_getIvar(id obj, Ivar ivar)
{
    if (!obj  ||  !ivar  ||  obj->isTaggedPointer()) return nil;

    ptrdiff_t offset;
    objc_ivar_memory_management_t memoryManagement;
    _class_lookUpIvar(obj->ISA(), ivar, offset, memoryManagement);

    id *location = (id *)((char *)obj + offset);

    if (memoryManagement == objc_ivar_memoryWeak) {
        return objc_loadWeak(location);
    } else {
        return *location;
    }
}

static ALWAYS_INLINE
void _object_setIvar(id obj, Ivar ivar, id value, bool assumeStrong)
{
```

```
    if (!obj  ||  !ivar  ||  obj->isTaggedPointer()) return;

    ptrdiff_t offset;
    objc_ivar_memory_management_t memoryManagement;
    _class_lookUpIvar(obj->ISA(), ivar, offset, memoryManagement);

    if (memoryManagement == objc_ivar_memoryUnknown) {
        if (assumeStrong) memoryManagement = objc_ivar_memoryStrong;
        else memoryManagement = objc_ivar_memoryUnretained;
    }

    id *location = (id *)((char *)obj + offset);

    switch (memoryManagement) {
    case objc_ivar_memoryWeak:       objc_storeWeak(location, value); b
reak;
    case objc_ivar_memoryStrong:     objc_storeStrong(location, value);
 break;
    case objc_ivar_memoryUnretained: *location = value; break;
    case objc_ivar_memoryUnknown:    _objc_fatal("impossible");
    }
}
```

11. 评价下, 如何代码:

```
@interface homeViewControler : UIViewController
{
    someManager *_manager;
}
@property (nonatomic, assign) NSNumber *flag;
@property (nonatomic, strong) NSString *name;
@property (nonatomic, strong) UIButton *button;
@end

@implementation homeViewControler

- (void)viewDidLoad
{
    self.button.onClick = ^{
        if (self.flag) {
            self.name = @"the name";
            [_manager reloadData:self.name];
        }
        else
        {
            self.name = nil;
            [_manager clearData];
        }
    };
}
```

@end

二面

1. 方法交换和分类同时去 hook 同一个方法, 结果会怎么样? 具体交换的是什么? 交换时是如何处理传参数? 如果使用 NSInvocation 的话, 是否能处理方法有返回值的场景?具体怎么处理的? ### 2. 介绍 Weex/RN 的渲染原理? 与 Flutter 渲染的区别? 目录树的步骤是在哪里的? iOS/android 渲染出来的树是否一致? ### 3. Weex JS 运行环境是多个还是一个? ### 4. Weex/RN 与 H5、native 相比的优缺点? ### 5. Weex/RN 与 H5 白屏的原因? ### 6. 你认为 c++、与大前端相关的语言,比如 objc、swift、js 相比它的优缺点?

关于 Weex / RN / Flutter 可以提 pr 到仓库，请在 rn_flutter_weex 目录下创建对应的解答，然后修改当前 md 文档附上链接即可

三面

websocket 协议 与 MQTT 协议的区别? MQTT 是否支持搭配 websocket 实现聊天功能?

弱网情况下 IM 即时通讯的优化? 比如: 网络抖动.

弱网环境下整体优化方案

什么是 SNI? SNI 是什么的缩写? iOS 上想设置 SNI , 怎么设?

排序算法, 字母和数字排序, 字母优先级高于数字: abc123.