iOS1605模拟面试：

1、怎么解决UITabView卡顿问题？

1.使用cell复用机制,同时可以节约内存(cell有个复用队列)

, 2.使用异步加载图片,3.图片不要太大(图片多大,就会占用多大的内存)

2、xcode的instruments是什么？

一个检测内存泄露的工具

7、请实现图片缓存的封装？

1)、根据图片URL查找内存是否有这张图片，有则返回图片，没有则进入第二步

2)、查找物理存储(沙盒)是否有这张图片，有则返回图片，没有则进入第三步

3)、从网络上下载该图片，下载完后保存到内存和物理存储上，并返回该图片

9、做过断点续传吗？怎么实现的？

断点续传业务逻辑

1.判断该文件是否存在

if (![manager fileExistsAtPath:self.filePath])

2.1不存在

1)通过NSFileManager创建这个文件

[manager createFileAtPath:self.filePath contents:nil attributes:nil];

2)设置NSFileHandle为读写(fileHandleForUpdatingAtPath)

self.fileHandle = [NSFileHandle fileHandleForUpdatingAtPath:self.filePath];

3)发起网络连接

self.downConnection = [NSURLConnection connectionWithRequest:request delegate:self];

———————————————————————————————————

2.2存在

1)设置NSFileHandle为读写(fileHandleForUpdatingAtPath)

self.fileHandle = [NSFileHandle fileHandleForUpdatingAtPath:self.filePath];

2)将NSFileHandle移动到文件最末尾(seekToEndOfFile)

[self.fileHandle seekToEndOfFile];

3)记录文件已下载长度(offsetInFile)

long long fileLength = self.fileHandle.offsetInFile;

(关键点)

4)设置RANGE, [request addValue:[NSString stringWithFormat:@"bytes=%lld-",fileLength] forHTTPHeaderField:@"Range"];

[request addValue:[NSString stringWithFormat:@"bytes=%lld-",fileLength] forHTTPHeaderField:@"Range"];

5)发起网络连接,下载文件后面的部分

self.downConnection = [NSURLConnection connectionWithRequest:request delegate:self];

10、瀑布流实现，瀑布流的实现原理？

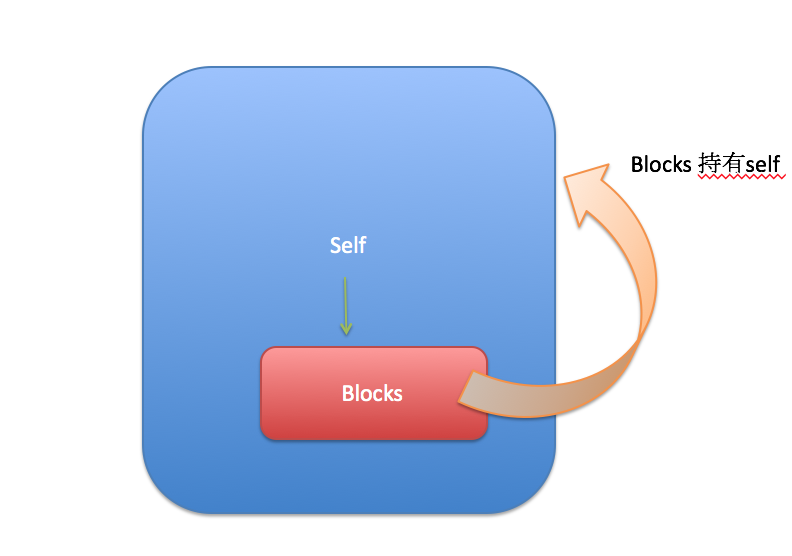
通过自定义collecitonViewlayout来实现瀑布流.瀑布流原理,cell高度不一致,但是间隙一致.使用自定义的layout时,需要返回每个cell的高度.并且cell的布局方式是依次接最短的cell

12、什么情况下会造成循环引用，如何处理？循环引用的原理，举几个例子？为什么系统的block，AFN网络请求的block内使用self不会造成循环引用?

代理互相持有,block使用self会引起循环引用.

代理改assign, block使用\_\_weak

AFN网络请求中,self并不持有这个block,所以不会循环引用.



13、如何避免内存泄漏，谈一下runtime？

遵守黄金法则

RunTime简称运行时, 在编译的时候并不能决定真正调用哪个函数, 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。使用runtime可以动态生成类,动态给类添加属性,成员变量,动态添加方法.JsonModel就是用的runtime

(oc就是对runtime的一个封装)

//动态创建类

objc\_allocateClassPair([NSObject class], "myClass", 0)

//添加成成员量

class\_addIvar(MyClass, "name", sizeof(NSString \*), 0, "@")

//添加方法

class\_addMethod([MyClass class], @selector(test:), (IMP)test, "v@:@");

//runtime掉用方法

objc\_msgSend(instance, @selector(test:),@"hello");

16、atomic(缺省) / nonatomic 什么时候用到？

(银行取钱和网上转账的时候,就用到,取钱的时候,就会锁定资源,别的地方用不到,保护资金安全)

Atomic加线程锁,多线程时会使用到

17、实现ARC和MRC混编？

fno-objc-arc

18、通知中心的原理，用代理可以实现吗？（）

不能.

通知中心基于观察者模式. 根据通知名称创建数组, 数组中存放对象的指针，当对象进行设置

[[NSNotificationCenterdefaultCenter]postNotificationName:ANOTIFNAME object:theValue];的时候添加到对应的数组中, 调用的时候，找到对应的数组，对里面的指针（对象）执行其对应的方法，顺序或者多线程执行.如果对象已经释放了，如P1已经是个野指针，call 的时候，进行方法调用，就会出现崩溃,通知必须释放

19、继承的坏处是什么？

①：父类的内部细节对子类是可见的,不利于保护。

　　②：子类从父类继承的方法在编译时就确定下来了，所以无法在运行期间改变从父类继承的方法的行为。

　　③：如果对父类的方法做了修改的话（比如增加了一个参数），则子类的方法必须做出相应的修改。所以说子类与父类是一种高耦合，违背了面向对象思想(高内聚,低耦合)。

优点:可以获取父类的所有属性和方法,节省代码量

20、自定义TabBar有几种方法？

1).对系统tabBar的图片标题进行修改

2).将系统tabBar隐藏,自己创建一个视图来做tabbar

21、基类是增加耦合还是减少耦合？

增加(子类和父类是增加耦合的)

22、\_\_BLock 在ARC与MRC是否一样？为什么？

是不一样的。  
在MRC中**block variable在block中使用是不會retain(引用计数加1)的**  
**但是ARC中**block則是會Retain的。  
取而代之的是用**weak或是**unsafe\_unretained來更精確的描述weak reference的目的

23、你的参与的项目中使用过什么核心技术？

蓝牙,推送,支付,分享,二维码扫描,购物车等,具体根据项目(旅游类:定位)

24、使用第三方框架的缺点？

1.出现bug时不方便调试,2.需要特定的功能时,第三方框架可能缺失

25、什么情况下会发生内存泄漏和内存溢出？

内存溢出 out of memory，是指程序在申请内存时，没有足够的内存空间供其使用(即内存给全部用完了,只能去占用别人的空间)

内存泄露 memory leak，是指程序在申请内存后，无法释放已申请的内存空间

申请的对象不释放就会泄露.内存被消耗完了就会溢出.

26、AFNetworking是怎么封装的？如何实现网络请求的封装？

对NSURLSession进行封装.传入接口参数,通过block返回数据(data或error)

29、socket的原理？

对TCP,UDP的一层封装().是长连接

30、如何进行代码性能的优化？

使用复用

使用cache(缓存)

使用正确的图片

不要阻塞主线程

 尽可能使Views不透明(clear color),这样大量使用内存

避免庞大的XIB,造成加载缓慢,(分开封装)

31、NSOperationQueue和GCD在实现多线程上，有什么区别？

1) GCD是底层的C语言构成的API，而NSOperationQueue及相关对象是Objc的对象。在GCD中，在队列中执行的是由block构成的任务，这是一个轻量级的数据结构；而Operation作为一个对象，为我们提供了更多的选择；

2) 在NSOperationQueue中，我们可以随时取消已经设定要准备执行的任务(当然，已经开始的任务就无法阻止了)，而GCD没法停止已经加入queue的block(其实是有的，但需要许多复杂的代码)；

3) NSOperation能够方便地设置依赖关系，我们可以让一个Operation依赖于另一个Operation，这样的话尽管两个Operation处于同一个并行队列中，但前者会直到后者执行完毕后再执行；

4) 我们能将KVO应用在NSOperation中，可以监听一个Operation是否完成或取消，这样子能比GCD更加有效地掌控我们执行的后台任务；

5) 在NSOperation中，我们能够设置NSOperation的priority优先级，能够使同一个并行队列中的任务区分先后地执行，而在GCD中，我们只能区分不同任务队列的优先级，如果要区分block任务的优先级，也需要大量的复杂代码；

6) 我们能够对NSOperation进行继承，在这之上添加成员变量与成员方法，提高整个代码的复用度，这比简单地将block任务排入执行队列更有自由度，能够在其之上添加更多自定制的功能。

32、JSonModel的作用是什么，能自己写出来吗？

将json转化成模型.通过runtime遍历类的成员列表,再使用KVC进行赋值.

34、动画实现?以及动画的类型有哪些？

切换帧来实现动画.

一种是UIView层面的，

  一种是使用CATransition进行更低层次的控制

36、MJRefresh上下拉的实现原理？

在tableView上加上一个View，注意不是headerView，而是一个Y值为负数的普通View，下拉时候监听偏移量，改变View的内容显示。  
在tableView下面加上一个footerView，监听偏移量，当footerView完全显示的时候加载更多数据。

37、jsonmodel 的实现原理？

通过runtime遍历类的成员列表,再使用KVC进行赋值.

38、SDWebimage 的实现原理？

开启线程,异步下载图片

40、代码量多大？

1万到2万

41、objective-c框架？

UIKit,Foundation(基本数据类型),MapKit,AVFoundation(视频)

42、你在使用SVN时，会出现什么问题 ？

问题1：’.’ is not a working copy. Can’t open file ‘.svn/entries’: 系统找不到指定的路径。  
  
解答：原因是输入的访问路径不正确，如svn://192.168.6.200/如果最后少写了“/”，就会出现这种错误提示。  
  
问题2：将文件checkout之后，没有出现SVN的图标，是怎么回事？  
  
解答：有些时候在客户端Checkout文件后，SVN的系统图标也会不显示，可以执行一下“Clean up”，就会出现SVN的系统图标。  
  
问题3：为什么添加的文件，别人看不到，版本库里也没有？  
  
解答：最可能的原因是，你只是执行了“Add”而没有“Commit”，这样只是在本地注明某个文件是预定要增加的，而没有实际添加到版本库中，要添加到版本库必须执行“Commit”。删除文件也是一样。  
  
问题4：“Commit failed。……You have  
to update your working copy first” 提交失败，需要首先执行更新操作。  
  
解答：多人同时修改同一文件，在提交前其他人已经抢先提交到SVN服务器中，导致该错误；解决方法：对工作复本中的文件进行更新即可。  
  
问题5：更新时提示文件发生冲突：“One or more files are not a conflicted state。”  
  
解答：多人同时修改同一文件的同一部分，SVN无法自动进行合并，会导致该错误；解决方法：对工作复本中的文件和服务器的文件进行比较，手工合并即可。  
  
问题6：“Commit failed；File already exists”提交失败，文件\*\*已存在。  
  
  
解答：版本管理系统在改变你的计算机上的工作副本时，是非常的小心的。在做任何事情之前，它都尽可能把您的意图写到你的计算机上的日志文件中去。但如果偶然地操作中断了(例如：突然停电了，您的计算机死机了)，那么日志文件记录就可能同您最后的工作状态不一致。一种建议解决途径：先把要提交的东西拷出来放到其它目录，再更新本地文件，然后把拷出来的文件重新放回去提交。  
  
问题7：Working copy’\*\*’locked. Please execute the ’Clean up’command.  
  
解答：Subversion客户端在提交内容之前会在本地的工作拷贝写日志，防止其他客户端再次作操作，如果这个提交过程中发生错误，就会存在未清理的日志，解决这个问题之需要执行“清理”操作，整理你的计算机上的工作副本，清理错误的日志记录，使您可以继续操作。  
  
问题8：执行clean up时，出现错误“Subversion reported an error while doing a cleanup!” '\*\*'  
is not a working copy directory ”  
  
解答：遇到这种情况，先删除隐藏文件夹.svn中的tmp下面的临时文件，再执行clean up。

46、GCD在是时候能用？GCD的优缺点

多线程.

1）优点：最高效，避开并发陷阱。

2）缺点：基于C实现。

47、你怎么把一个点曲线移动到另外一点？

使用贝塞尔曲线绘图

48、登录你是怎么实现的？

把用户名\密码post到服务器,服务器返回登录结果,

49、你在项目中哪里用到多线程？

异步下载

50、单例在ARC和MRC中有哪些不同

没有

写法:+(Person \*)shareInstance

{

static Person \*p = nil;//statioc 代表这片内存空间不会释放

static dispatch\_once\_t onceToken;//确保多线程安全

dispatch\_once(&onceToken,^{

p = [[Person alloc]init];

});

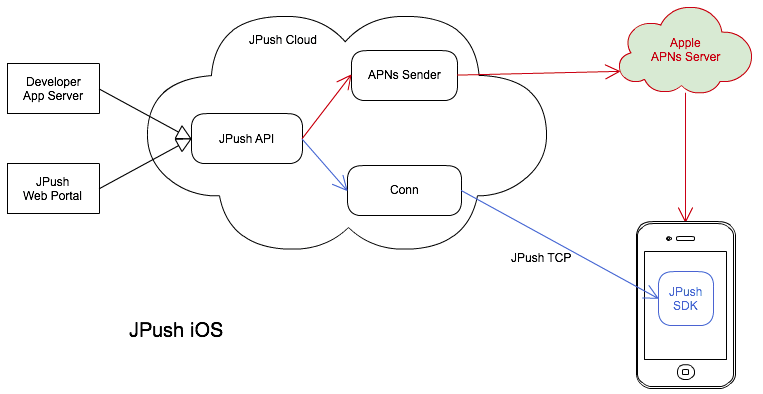
return p;

}

52、极光推送的原理？

1.当app在前台时,可以保持和极光服务器的长连接,这个时候可以使用直推

2.当APP在后台时,只能通过APNS推



53、实现登录的时候，密码怎样加密的；

MD5加密 AES加密

56、如何解决视频播放时卡的问题？

使用硬件解码渲染

根据网络带宽选择合适的片源

71、快捷支付是怎么弄的

集成支付宝

73、如何使用autolayout布局

加约束 种类:同级视图之间的间隙 父视图之间的间隙 等高\等宽\等比例\横向居中\纵向居中

原理:在布局视图时,不再指定frame,一切通过约束计算而来的 ,使用autolayout时,设置它的宽高无限,设置origin也无效

3、block实现原理？block为什么能完成回调？－函数指针

block就是匿名函数,就是一个函数指针,只要调用这个函数指针,就可以运行函数指针内部的代码(回调)

4、block类型

5、动画实现原理？核心动画的实现以及动画的类型？

6、传输一张图片，有哪几种协议方式，列举一下？

通过HTTP协议post上传

8、怎么封装一个网络请求？

传入一个借口地址和参数,返回数据或错误

11、怎么解藕

减少类与类之间的参数传递,方法调用,使用mvc设计模式的一种作用

14、什么时候用到runloop？runloop的作用？

1.子线程中做下载(AFNetworing).

2.拖动scrollView时,依然可以通过定时器自动滚动

Run Loop是一让线程能随时处理事件但不退出的机制, 线程执行了这个函数后，就会一直处于这个函数内部 “接受消息->等待->处理” 的循环中，直到这个循环结束（比如传入 quit 的消息），函数返回。让线程在没有处理消息时休眠以避免资源占用、在有消息到来时立刻被唤醒。

15、怎么减小ipa的大小？

App Thinning 会自动检测用户的设备类型（即型号名称）并且只下载当前设备所适用的内容换句话说，如果你使用的是 iPad Mini 1（1x分辨率且非 retina 显示屏）那么只会下载 1x分辨率（下文会有更多介绍）所使用的文件。更强大和更高分辨率的 ipad（如iPad Mini 3或 4）所使用的资源将不会被下载。因为用户仅需下载他/她自己当前使用的特定设备所需的内容，这不仅加快了下载速度，还节约了设备的存储空间。

27、抽屉效果怎么封装更好用？

暴露出必须使用的接口

28、isa指针？

OC语法最后Runtime转化成可被Runtime执行的C代码。在OC中定义一个创建NSObject类，转化成C语言后，其表现是创建了一个结构体。其中isa就是这个结构中的一个指针。

33、KVO和KVC以及他们的区别？

KVC 键值编码

KVO 键值观察

KVO 是基于KVC上的,要观察某个对象的属性发生变化,就给它添加一个观察者,设置好观察者的属性,一旦属相发生变化,立即出发ObserveForkeyPath回调事件,判断Kyepath是不是你观察的键,然后做响应的操作

35、APNs推送机制？以及简单实现？

45、单例在外面alloc 你怎么保证他是单例 ，就是不调用加方法？

重写allocWithZone方法

+(id)allocWithZone:(NSZone \*)zone

{

    if (single == nil) {

        single = [[super allocWithZone:zone] init];

    }

    return single;

}

single 就是静态本类对象

51、项目支付有哪些？支付宝中遇到的问题？？？支付宝的支付流程以及简单的实现。

微信支付,支付宝支付,银联支付

1、订单签名错误，请检查PARTNER对应的密钥是否正确

这个问题网上很多人问，但千篇一律都是没有给出解决办法的。很不幸的，我也碰到了。一开始以为是自己密钥生成有问题，对着支付宝的开发文档，重复生成密钥，每步都严格按照要求，结果还是无法解决。几近崩溃！后来没办法了，问了支付宝集成开发的技术支持，原来这个不是密钥的问题，而是没有申请到对应的接口。app应用内支付，要开通的产品是：安全支付，版本不知道它是怎样区分的，我v1.1和v2.0我都见过，如果开通了这个接口，那这个问题就马上解决了（前提是你的密钥生成不能有问题）。

2、服务器端验签不成功

这个问题大多是因为支付宝公钥生成不正确。很奇怪的是，支付宝集成开发的文档居然没有说怎样生成支付宝公钥，这个问题也是搞了好久才明白过来。支付宝公钥文件的格式如下：

-----BEGIN PUBLIC KEY-----

xxxxx  
xxxxx

xxxxx

-----END PUBLIC KEY-----

中间内容部分是签约后台上获取的支付宝公钥，将空格去掉，换成换行就可以了。文件保存为alipay\_public\_key.pem。（当然名字可以改的，前提是服务器端的验签代码也要改成相应的文件名）

54、内存管理机制；

黄金法则

ARC 与MRC 区别

ARC 是编译器帮你在适当的位置添加release.retain语句

54、怎么进行系统优化？

57、你项目中最得意的技术，以及项目中的核心技术，你项目遇到的最大问题或bug.这些不知道怎么回答

在项目当中使用到不会的技术.例如视频流的接收,显示.

通过请教同事,网上找资料来学习解决

58、特别是项目中遇到的难点，和最大的bug

59、蓝牙的实现原理以及简单的实现？

60、OC与JS如何实现交互？在面试中遇到OC与JS交互，蓝牙，APNs，支付等问题该如何回答？

- (void)webViewDidFinishLoad:(UIWebView \*)webView

{

NSString \*title = [webView stringByEvaluatingJavaScriptFromString:@"document.title"];

NSLog(@"%@", title);

}

64、最近IPV6上架的问题，及什么是IPV6

苹果宣布6月1日后所有应用必须支持IPv6-only网络. IPv4的下一个版本,主要解决ip地址空间不足的问题.

61、微信如何搜索附近的人？模糊搜索和精准搜索

定位,向服务器发送位置信息.服务器返回附近的人

62、即时通讯的UI布局，要是你做你会用多少个cell

三种,语音,图片,文字

63、说说直播，你了解多少

<https://www.zybuluo.com/qvbicfhdx/note/126161>

65、消息机制是什么，用它可以做什么

对象之间互相传递消息.

KVO,通知,代理,Block,Target-Action

66、instruments有哪些常用的工具，如何测试核心动画性能

Time Profile**r**: 时间分析工具用来检测应用CPU的使用情况

Allocations: 分配工具。它能给出你所有创建和存储它们的内存的详细信息，它也显示你保留了每个对象的计数

Leaks :内存泄露

Activity monitor: CPU,内存使用监测

Zombie: 查找僵尸对象

核心动画性能测试:

http://wiki.jikexueyuan.com/project/ios-core-animation/performance-tuning.html

67、GCD信号机制？

在使用NSOperationQueue进行多线程编程时，可通过[queue setMaxConcurrentOperationCount:5]来设置线程池中最多并行的线程数,在GCD中信号量机制也和它相似，可以控制并发的线程数量。

1.首先熟悉下几个函数

dispatch\_semaphore\_create 创建一个信号量,设置一个初始值  
dispatch\_semaphore\_signal 发送一个信号,信号通知,信号量＋1  
dispatch\_semaphore\_wait 等待信号,信号量－1

当一个信号量被通知 ，信号量就会加1，当一个信号等待，信号总量就减1，当减到信号量小于0时，线程会被阻塞，信号量不会在减了。直到信号量大于0时，线程会再次启动执行

|  |
| --- |
| dispatch\_group\_t group=dispatch\_group\_create();      dispatch\_semaphore\_t semaphore=dispatch\_semaphore\_create(1);      dispatch\_queue\_t queue=dispatch\_get\_global\_queue(DISPATCH\_QUEUE\_PRIORITY\_DEFAULT, 0);      for (int i=0; i<100; i++) {          //信号量减1,如果同时开启10个以上的线程，则信号量小于等于0，此时就会阻塞该线程。          dispatch\_semaphore\_wait(semaphore, DISPATCH\_TIME\_FOREVER);          dispatch\_group\_async(group, queue, ^{              NSLog(@"test %d",i);          //每个线程执行减1后通过信号量通知加1，这样始终保持线程在10个之内          dispatch\_semaphore\_signal(semaphore);          });      }      dispatch\_group\_wait(group, DISPATCH\_TIME\_FOREVER); |

68、什么是RAC和MVVM。应用场景

ReactiveCocoa（简称为RAC）,是由Github开源的一个应用于iOS和OS开发的新框架,Cocoa是苹果整套框架的简称，因此很多苹果框架喜欢以Cocoa结尾。

 Model-View-ViewModel 的简写. MVVM 在使用当中，通常还会利用双向绑定技术，使得 Model 变化时，ViewModel 会自动更新，而 ViewModel 变化时，View 也会自动变化。

69、

70、响应链

响应者对象（Responder Object），顾名思义，指的是有响应和处理事件能力的对象。响应者链就是由一系列的响应者对象构成的一个层次结构。

UIResponder是所有响应对象的基类，在UIResponder类中定义了处理上述各种事件的接口。我们熟悉的UIApplication、 UIViewController、UIWindow和所有继承自UIView的UIKit类都直接或间接的继承自UIResponder，所以它们的实例都是可以构成响应者链的响应者对象。

对象以消息的形式将事件发送给第一响应者，使其有机会首先处理事件。如果第一响应者没有进行处理，系统就将事件（通过消息）传递给响应者链中的下一个响应者，看看它是否可以进行处理。依次进行传递.



72、如何使用xib和纯代码方式动态获取cell的高度

- (CGRect)rectForRowAtIndexPath:(NSIndexPath \*)indexPath;

一般我们在模型中,自己计算字符串所占用的高度,然后把高度设置给cell

使用自动布局

CGFloat height = [cell.contentView systemLayoutSizeFittingSize:UILayoutFittingCompressedSize].height;

71、推送过来的信息，点击提示信息后，如何跳转到当前的页面，像我们微信或qq一样，点击信息，跳转到当前对应的界面？

当用户通过点击通知消息进入应用时

- (BOOL)application:(UIApplication\*)application didFinishLaunchingWithOptions:(NSDictionary\*)launchOptions

中会有推送消息的userInfo信息，此时我们可以通过获得推送消息内容。如果remoteNotification不为空，则说明用户通过推送消息进入，那么可以声明一个属性

NSDictionary\* remoteNotification = [launchOptions objectForKey:UIApplicationLaunchOptionsRemoteNotificationKey];

@property (nonatomic) BOOL isLaunchedByNotification;

用于标识用户是否通过点击通知消息进入本应用。此时，

- (void)application:(UIApplication\*)application didReceiveRemoteNotification:(NSDictionary\*)userInfo

一定会被调用，iOS7可以使用

- (void)application:(UIApplication\*)application didReceiveRemoteNotification:(NSDictionary\*)userInfo fetchCompletionHandler:(void (^)(UIBackgroundFetchResult))completionHandler

因为此方法的调用时，MainViewController已经被初始化，所以我们已经可以在MainViewController注册推送消息的监听，用于展示对应的视图，如下：

//订阅展示视图消息，将直接打开某个分支视图

[[NSNotificationCenter defaultCenter] addObserver:self selector:@selector(presentView:) name:@"PresentView" object:nil];//弹出消息框提示用户有订阅通知消息。主要用于用户在使用应用时，弹出提示框

[[NSNotificationCenter defaultCenter] addObserver:self selector:@selector(showNotification:) name:@"Notification" object:nil];

**1.nil Nil NULL NSNull 之间的区别**

nil 是对objective c id 对象赋空值

对于objective c 集合类对象 比如数组对象，字典对象，当我们不需要再使用他们的时候，对他们release 的同时最好也把他们赋值为nil，这样确保安全性，下面一个简单的一个例子：如果不赋值nil，可能导致程序崩溃

**Nil**: A null pointer to an Objective-C class.表示对类进行赋空值

**NULL**: A null pointer to anything else,  is for C-style memory pointers. 用于对非对象指针赋空值，比如C指针

NSNull 对于像NSArray这样的类型，nil或NULL不能做为加到其中的Object，如果定义了一个NSArray，为其分配了内存，又想设置其中的内容为空，则可以用[NSNULL null] 返回的对象来初始化NSArray中的内容

2.setValue和setObject的区别

1, setObject：forkey：中value是不能够为nil的，不然会报错。

setValue：forKey：中value能够为nil，但是当value为nil的时候，会自动调用removeObject：forKey方法

2, setValue：forKey：中key的参数只能够是NSString类型，而setObject：forKey：的可以是任何类型

[NSNull null]表示的是一个空对象，并不是nil，注意这点

3. **iOS推送和本地通知的区别**

* 注意：这里说的推送通知跟NSNotification有所区别
* NSNotification是抽象的，不可见的
* 推送通知是可见的（能用肉眼看到）
* **iOS中提供了2种推送通知**
  + 本地推送通知（Local Notification）
  + 远程推送通知（Remote Notification）
* **推送通知的作用**
  + 让app不在前台时,告示用户App的最新情况
* **推送通知的呈现方式**
  + 在屏幕顶部一块横幅
  + 在屏幕中间弹出UIAlertView
  + 同时可以播放音效
  + 锁屏时展示
  + App图标右上角额数字提示

总结：两者呈现形式一样，作用一样

区别：本地通知要求那个app要在后台运行，不能在应用程序切换器里划掉。推送通知是依靠[苹果公司](https://www.baidu.com/s?wd=%E8%8B%B9%E6%9E%9C%E5%85%AC%E5%8F%B8&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1Y3P16LPH64m163mvc4mvfz0ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EPWD3PjmvrH6YP1mvrHm1Pjbz)的服务器来实现消息的提醒，不管你的那个app是否在后台运行，只要允许了它推送通知并且你的设备连接到了互联网它就能向你推送。

4.对于语句NSString \* testObject = [[NSData alloc]init];testObject在编译时和运行时分别是什么数据类型？

编译时是nsstring, 运行时是nsdata的一个实例，没有类型之分。

编译时创建了一个NSString 对象，指派了空间。运行时就指向该内存引用地址了。

5.详解 CALayer 和 UIView 的区别和联系

1、 UIView继承UIResponder,CALayer继承NSObject,UIView比CALayer多了一个事件处理的功能，最明显的区别是 View可以接受并处理事件，而 Layer 不可以

2、**UIView主要是对显示内容的管理而 CALayer 主要侧重显示内容的绘制。**

**3、**UIView来自CALayer，是CALayer的高层实现和封装，UIView的所有特性来源于CALayer支持。

6、@class关键字的作用以及#include和#import的区别

#include 是包含的意思， 相当于拷贝头文件中的声明内容，会报重复定义的错误

#Import是导入的意思，可以解决重复导入的问题，他会做一次判断，如果已经导入一次就不导入了

@class 是告诉编译器，这只是一个类，在别处有定义但是不知道这个类中的信息

7、Runloop

1、runloop是事件接收和分发机制的一个实现。一个runloop就是一个事件处理循环，用来不停的调配工作以及处理输入事件。使用run loop的目的是使你的线程在有工作的时候工作，没有的时候休眠。

2、Run loops是线程的基础架构部分，每个线程，包括程序的主线程（main thread）都有与之相应的run loop对象。

3、主线程的run loop默认是启动的。对其它线程来说，run loop默认是没有启动的，如果你需要更多的线程交互则可以手动配置和启动，如果线程只是去执行一个长时间的已确定的任务则不需要。

8、Runtime

* 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，
* OC的函数调用成为消息发送。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。

用法：

• 在程序运行过程中, 动态创建一个类(比如KVO的底层实现)

* 在程序运行过程中, 动态地为某个类添加属性\方法, 修改属性值\方法
* 遍历一个类的所有成员变量(属性)\所有方法

9、KVO原理(基于Runtime机制)

1、当一个object有观察者时，动态创建这个object的类的子类

2、对于每个被观察的property，重写其set方法

3、在重写的set方法中调用- willChangeValueForKey:和- didChangeValueForKey:通知观察者

4、当一个property没有观察者时，删除重写的方法

5、当没有observer观察任何一个property时，删除动态创建的子类

10、KVC

通过KVC可以直接用字符串的名字(key)来访问类属性的机制。而不是通过调用Setter、Getter方法访问。

11、atomic和nonatomic区别

atomic和nonatomic区别用来决定编译器生成的getter和setter是否为原子操作。atomic提供多线程安全,是描述该变量是否支持多线程的同步访问，如果选择了atomic 那么就是说，系统会自动的创建lock锁，锁定变量。nonatomic禁止多线程，变量保护，提高性能。

retain 使用了引用计数，是指针拷贝，copy 是内容拷贝。assign就是直接赋值，当数据为基础数据类型 （NSInteger，CGFloat）和C数据类型（int, float, double, char）时使用。

12、ios中观察者模式与通知机制，及KVO

在ios开发中，通知机制和KVO都是依靠观察者模式得以实现。通知机制与KVO的不同之处在于，前者是一个中心对象为所有观察者提供变更通知，后者是被观察对象直接向观察者发送通知。

**以下关注观察者模式与通知机制实现：**

1、观察者模式中Subject对象，又可称目标对象，是通知的发布者，又是被观察者。提供注册和取消注册的方法；Observer对象，又可称为观察者，是通知的订阅者。Observer类中，相识Subject,以达到能够接收通知。

2、ios中的通知机制，是Cocoa Touch框架为开发者开发的类，让开发者不必自己写观察者模式，用到它便能实现。对于我而言，通知机制的所有使用，都集中到一个类中。而导致我用通知机制去理解观察者模式时，迷糊于Subject和Observer,谁是通知的发送者，因为类中集结了观察者和发送者。

13、Category和Extension的区别

**Category**

* 用于给class及其subclass添加新的方法
* 有自己单独的 .h 和 .m 文件
* (1) Category的方法不一定非要在@implementation中实现，也可以在其他位置实现，但是当调用Category的方法时，依据继承树没有找到该方法的实现，程序则会崩溃。
* (2) Category理论上不能添加变量，但是可以使用@dynamic 来弥补这种不足。

**Extension**

* Extension常被称为是匿名的Category
* 用于给类添加新方法，但只作用于原始类，不作用于subclass
* 只能对有implementation源代码的类写Extension，对于没有implementation源代码的类，比如framework class，是不可以的
* Extension可以给原始类添加新方法，以及新属性

14、isa

isa：是一个Class 类型的指针. 每个实例对象有个isa的指针,他指向对象的类，而Class里也有个isa的指针, 指向meteClass(元类)。元类保存了类方法的列表。当类方法被调用时，先会从本身查找类方法的实现，如果没有，元类会向他父类查找该方法。同时注意的是：**元类（meteClass）也是类，它也是对象。**元类也有isa指针,它的isa指针最终指向的是一个根元类(root meteClass).根元类的isa指针指向本身，这样形成了一个封闭的内循环。

**每一个对象本质上都是一个类的实例。其中类定义了成员变量和成员方法的列表。对象通过对象的isa指针指向类。**

**每一个类本质上都是一个对象，类其实是元类（meteClass）的实例。元类定义了类方法的列表。类通过类的isa指针指向元类。**

**所有的元类最终继承一个根元类，根元类isa指针指向本身，形成一个封闭的内循环。**

15、GCD、NSOperation区别

那这两者直接有什么区别呢？

1> GCD是纯C语言的API，NSOperationQueue是基于GCD的OC版本封装

2> GCD只支持FIFO的队列，NSOperationQueue可以很方便地调整执行顺序、设置最大并发数量

3> NSOperationQueue可以在轻松在Operation间设置依赖关系，而GCD需要写很多的代码才能实现

4> NSOperationQueue支持KVO，可以监测operation是否正在执行(isExecuted)、是否结束(isFinished)，是否取消(isCanceld)

5> GCD的执行速度比NSOperationQueue快

任务之间不太互相依赖：GCD

任务之间有依赖\或者要监听任务的执行情况：NSOperationQueue

16、UDP、TCP、socket区别

IP协议对应于网络层，TCP协议对应于传输层，而HTTP协议对应于应用层，socket则是对TCP/IP协议的封装和应用(程序员层面上)。主要解决数据如何在网络中传输，HTTP是应用层协议，主要解决如何包装数据

首先说一下他们直接的联系，UDP和TCP就像声明的一个协议，是需要传送的东西也就是内容，而scoket就像是一个通道，用于传送这些内容，也就是用socket来实现。

UDP:UDP是一种面向无连接的用户数据报服务(user data protocol)，不需要和服务器也能交互，只需要知道ip和监听端口，不需要链接没有目的的socket，只是将数据报投递出去，不管接收方是否成功接收到，因此是一种不可靠的传输，可能会造成数据丢包，但由于这些特征，传输效率要优于TCP。

TCP:TCP是一种面向连接的传输控制协议(transform contorl protocol),必须要和服务器交互，具有高安全性，可靠性，需要和服务器进行三次握手，能根据具体网络拥堵情况进行延时。

Socket:Socket有两种连接操作方式，面向连接的和面向无连接的。使用UDP就是面向无连接的，使用TCP就是面向连接的。使用UDP无需要指定一个socket目的地，而是用TCP必须要指定一个socket目的地，需要进行预链接，否则连接不到。

socket就像是API，二UDP/TCP就是协议，使用scoket来实现内容的传送。

17、类别和继承的区别

1. 类别 category

这是 Objective-C 语言的一个特性,可以在不改变类名和原来类的实现的前 下,实现对类的方法扩展。

以下两种方式最后使用类别。

1)针对系统 供的一些类,例如:NSString,NSArray,NSNumber 等类,系统 本身不 倡使用继承去扩展方法,因为这些类内部实现对继承有所限制,所以最 后使用类别来进行方法扩展。

2)类别支持开发人员针对自己构建的类,把相关的方法分组到多个单独的 文件中,对于大型而复杂的类,这有助于 高可维护性,并简化单个源文件的管 理。

2. 继承

这个是面向对象语言都有的一个特性,子类会继承父类的方法和属性。 对于以下情况,无法使用类别,必须使用继承。

1)新扩展的方法与原方法同名,但是还需要使用父类的实现。因为使用类

别,会覆盖原类的实现,无法访问到原来的方法。

2)扩展类的属性,这个类别无法做到。

18、 ViewControler 的生命周期

创建时: 1.alloc 创建对象,分配空间

2.init(initWithNibName)初始化对象,初始化数据 3.loadView 从 nib 载入视图,通常这一步不需要去干涉,除非你没有使用 xib 文件创建 4.viewDidLoad 载入完成,可以自定义数据和动态加入自定义控件

5.viewWillAppear 视图将出现在屏幕上,马上被展现 6.viewDidAppear 已经出现在屏幕上,渲染完成

销毁时:

7.viewWillDisappear

8.viewDidDisappear

9.viewUnload

10.dealloc

19、使用支付宝进行一个完整的支付功能，大致有以下步骤：

a 与支付宝进行签约，获得商户ID（partner）和账号ID（seller）

b 下载相应的公钥私钥文件（加密签名用）

c 下载支付宝SDK

d 生成订单信息

e  调用支付宝客户端，有支付宝客户端跟支付宝安全服务器打交道

f  支付完毕后返回支付结果给客户端和服务器

20、MRC 和ARC block区别

\*\_NSConcreteGlobalBlock:全局静态 block 不会访问任何外部变量

\*\_NSConcreteStackBlock:保存在栈区的block,当函数返回时会被销毁

\*\_NSConcreteMallocBlock:保存在堆区 block 引用计数为0时,会被销毁;

* NSGlobalBlock：在block内部没有引用任何外部变量

对NSGlobalBlock的retain、copy、release操作都无效。

* NSStackBlock：在block内部引用外部变量
* 先讨论下MRC模式

栈block在当函数退出的时候，该空间就会被回收，因此如果再调用该block会导致crash：retain、release这种类型的block不起作用。可以通过将block拷贝到堆上来解决这个问题：

在ARC情况下

生成的block也是NSStackBlock，只是当赋值给strong对象时，系统会主动对其进行copy:

• NSMallocBlock

如果NSStackBlock需要在其作用域外部使用的时候，在MRC的模式下需要手动将其copy到堆上，NSMallocBlock支持retain、release，会对其引用计数＋1或－1，copy不会生成新的对象，只是增加了一次引用，类似retain；而在ARC模式下会自动对其进行copy，不需要自己手动去管理。

MRC下 \_\_block修饰的变量，并不改变引用计数,但是block内部对引入的外部对象，会更改引用计数。所以要及时对block进行release.

ARC下 \_\_block修饰的引用计数会增加，同时block内部持有的对象引用计数会增加，所以需要使用weak.

21、介绍一下XMPP?

基于XML的点对点通讯协议,实现通讯功能.

优点:可以跨平台开发.

缺点:丢包,只能发文字(图片发送发的是链接).

22、应用程序如何省电?

获取请求不能过频.优化算法.

8.OC内存管理的机制是什么? assign、retain、copy、strong、weak的区别?

OC 内存管理机制是 ARC,自动引用计数器.

assign:用于非指针类型变量,基本数据类型,C 数据类型。若 述 id 类型, retainCount 不+1;

retain:强指针,retainCount +1 copy: 浅拷贝,并 retainCount +1。 strong:有 arc 后导入的,强指针,retainCount +1;(等价于 retain) weak:有 arc 后导入的,弱指针,retainCount 不+1, 只能 述对象,并且

最后会将该对象设置为 nil 来防止野指针。