1. 判断一个view 是否已经加到window，判断self.window是否为空

- (void)didMoveToWindow

{

if (nil != self.window) {

[self shine];

}

}

2.返回nsstring中某个索引所在的unicode字符

[self.attributedString.string characterAtIndex:i]])

//取出属性字符串的string 调用characterAtIndex：方法得到某个位置的unicode字符

3.判断字符串第0个位置是不是空格

BOOL iswhitespace = [[NSCharacterSet whitespaceAndNewlineCharacterSet] characterIsMember:[@" dsafsdaf" characterAtIndex:0]];

4. 去除文字多余的空格

NSString \*str = @" this is a test . ";

NSCharacterSet \*whitespaces = [NSCharacterSet whitespaceCharacterSet]; NSPredicate \*noEmptyStrings = [NSPredicate predicateWithFormat:@"SELF != ''"];

NSArray \*parts = [str componentsSeparatedByCharactersInSet:whitespaces];

NSArray \*filteredArray = [parts filteredArrayUsingPredicate:noEmptyStrings];

str = [filteredArray componentsJoinedByString:@" "];

NSLog(@"%@",str);

// **this is a test .**

5.从UIColor获取当前颜色透明度

CGFloat currentAlpha = CGColorGetAlpha([(UIColor \*)value CGColor]);

Label传入一个小于1的数值返回一个和原来颜色一样，但是透明度改变的UIColor

[self.textColor colorWithAlphaComponent:percentage];

6.获取当前的时间，返回CFTimeInterval 小数（有可能是毫秒）

CFTimeInterval now = CACurrentMediaTime();

7. 返回一张带圆角的模糊化的图片（uiimage的方法）

参考yy。。kit（优酷ios）

什么是 dSYM 文件

Xcode编译项目后，我们会看到一个同名的 dSYM 文件，dSYM 是保存 16 进制函数地址映射信息的中转文件，我们调试的 symbols 都会包含在这个文件中，并且每次编译项目的时候都会生成一个新的 dSYM 文件，位于 /Users/<用户名>/Library/Developer/Xcode/Archives 目录下，对于每一个发布版本我们都很有必要保存对应的 Archives 文件 ( AUTOMATICALLY SAVE THE DSYM FILES 这篇文章介绍了通过脚本每次编译后都自动保存 dSYM 文件)。

dSYM 文件有什么作用：

当我们软件 release 模式打包或上线后，不会像我们在 Xcode 中那样直观的看到用崩溃的错误，这个时候我们就需要分析 crash report 文件了，iOS 设备中会有日志文件保存我们每个应用出错的函数内存地址，通过 Xcode 的 Organizer 可以将 iOS 设备中的 DeviceLog 导出成 crash 文件，这个时候我们就可以通过出错的函数地址去查询 dSYM 文件中程序对应的函数名和文件名。大前提是我们需要有软件版本对应的 dSYM 文件，这也是为什么我们很有必要保存每个发布版本的 Archives 文件了。

如何将文件一一对应：

每一个 xx.app 和 xx.app.dSYM 文件都有对应的 UUID，crash 文件也有自己的 UUID，只要这三个文件的 UUID 一致，我们就可以通过他们解析出正确的错误函数信息了。

1.查看 xx.app 文件的 UUID，terminal 中输入命令 ：

dwarfdump --uuid xx.app/xx (xx代表你的项目名)

2.查看 xx.app.dSYM 文件的 UUID ，在 terminal 中输入命令：

dwarfdump --uuid xx.app.dSYM

1. crash 文件内第一行 Incident Identifier 就是该 crash 文件的 UUID。

手动裁剪图像，任意粘贴，到想要的位置，实现手动p图

制作常见的电视剧场景，替换人物

类似于秒拍的应用 视频美化处理

[https://github.com/xujingzhou/VideoBeautify](https://github.com/xujingzhou/VideoBeautify" \t "_blank)

关于ios 图形处理的文章，还有其他相关的链接

<http://www.cocoachina.com/ios/20140925/9756.html>

学习opencv的英文网站

http://www.raywenderlich.com/59602/make-augmented-reality-target-shooter-game-opencv-part-1

iOS图像处理第2部分：核心图形,核心图像,GPUImage

<http://www.cocoachina.com/ios/20140925/9756.html>

ffmpeg在iOS的使用 – iFrameExtractor

ios 编译ffmpeg

<http://www.mamicode.com/info-detail-476094.html>

视频直博方案hls rtmp等

<http://www.tuicool.com/articles/MFbeUn>

FFMPEG视音频编解码零基础学习方法

http://blog.csdn.net/leixiaohua1020/article/details/15811977/

ios ui的一些知识点 （一些不常见方法）

<http://blog.csdn.net/zhangao0086/article/details/45622875>

在本地调试远程通知

<http://blog.csdn.net/zhangao0086/article/details/45448785>

可用来伪造HTTP响应，这样不依赖服务器端，iOS的开发人员就可以测试网络服务了。<http://blog.shiqichan.com/using-afnetworking-sdwebimage-and-ohhttpstubs/>

iOS 异步图片加载优化与常用开源库分析

<https://segmentfault.com/a/1190000002776279>

ios 全景图片展示

<https://code.google.com/archive/p/panoramagl/>

开源的360度视频播放器

https://github.com/hanton/HTY360Player

webview 离线缓存

<http://bbs.csdn.net/topics/390831054>

<http://www.keakon.net/2011/08/14/为UIWebView实现离线浏览>

<https://github.com/asialee/CustomURLCache>

Oc中并发编程的相关api和面临的挑战

http://www.cocoachina.com/ios/20130718/6626.html

IOS富文本组件的实现 DTCoreText源码解析 （亮点：HTML+CSS转换成NSAttributeString）

<http://blog.cnbang.net/tech/2630/>

ios中GIF的制作的简单版本记录

http://www.2cto.com/kf/201312/268731.html

iOS开发中实现显示gif图片的方法

http://www.jb51.net/article/72773.htm

加载gif动态图的三种方式

<http://www.bkjia.com/IOSjc/996621.html>

尝试手写一个更好用的performSelector/msgSend(详细修改版)

<http://www.cocoachina.com/ios/20160111/14927.html>

ios 关于ipv6审核的一些问题

<http://www.2cto.com/kf/201605/512206.html>

IPv6迁移过程当中踩的坑

<http://www.cocoachina.com/ios/20160630/16890.html>

ios视频直播

<https://github.com/runner365/LiveVideoCoreSDK>

做一款仿映客的直播App？看我就够了

<http://www.jianshu.com/p/5b1341e97757>

文章重点摘要：

8.0以上IOS还提供现成的 Video ToolBox框架，可以对摄像头和流媒体数据结构进行处理

采集端

1.目前比较知名的有VideoCore

目前国内很多知名的推流框架都是对VideoCore的二次开发。这个框架主要使用C++写的，支持RTMP推流，但对于iOS开发者来说有点晦涩难懂（精通C++的除外）。想开源和免费的可以选择现在的几个知名项目VideoCore + GPUImage+基于GPU的美颜滤镜 ,播放用IJKPlayer自己修改。

2.国内比较火的LiveVideoCoreSDK

框架提供IOS苹果手机的RTMP推流填写RTMP服务地址，直接就可以进行推流，SDK下载后简单的工程配置后能直接运行，实现了美颜直播和滤镜功能，基于OpenGL，前后摄像头随时切换，提供RTMP连接状态的回调。

这个框架是国内比较早的一款推流框架有不少在使用这个SDK，功能非常齐全，作者也比较牛，用来学习推流采集相关内容非常好，但是集成到工程中有些困难（对于我来说）。总的来说这是一款非常厉害的推流SDK，几乎全部使用C++写的，编译效率非常好，如果有实力的话推荐使用这个框架来做自己项目的推流端。

3.可读性比较好的推流LFLiveKit

框架支持RTMP（Real Time Messaging Protocol ）：实时消息传输协议，Adobe公司的。

HlS (HTTP Live Streaming) ：苹果自家的动态码率自适应技术。主要用于PC和Apple终端的音视频服务。包括一个m3u(8)的索引文件，TS媒体分片文件和key加密串文件。

推荐这个框架第一是因为它主要使用OC写的，剩下的用C语言写的，框架文件十分清晰，这对不精通C++的初学者提供了很大的便利，并且拓展性非常强，支持动态切换码率功能，支持美颜功能。

4.美颜功能

美颜的话一般都是使用的GPUImage基于OpenGl开发，纯OC语言，这个框架十分强大，可以做出各种不同滤镜，可拓展性高。如果对美颜没有具体思路可以直接用BeautifyFace，可以加入到项目中，很方便的实现美颜效果。

播放端用的针对RTMP优化过的ijkplayer（），ijkplayer是基于FFmpeg的跨平台播放器，这个开源项目已经被多个 App 使用，其中映客、美拍和斗鱼使用了 ijkplayer（5700+️） 。在本文的末未提供了，已经打包好的ijkplayer，直接拖入项目就可以使用。省去了编译的过程（编译十分麻烦，并且容易出错）。播放端调用ijkplayer参考VoideSun的简仿映客所写，如有关于ijkplayer的问题，可以请教这位大牛。

VoiderSun: http://pan.baidu.com/s/1eRJAcim

https://github.com/GrayJIAXU/520Linkee :

本项目比较完备的实现了作为一个直播App的基本功能，比如本地视频流采集、播放、美颜、礼物、点赞出心等。

Mac搭建nginx+rtmp服务器

<http://www.jianshu.com/p/02222073b3f1>

rtmp协议学习总结

<http://blog.chinaunix.net/uid-17102734-id-3986995.html>

录屏：

<http://www.cocoachina.com/ios/20160317/15716.html>

<http://www.cnblogs.com/huangzizhu/p/5073389.html>

mvvm：

1.简单把网络请求与数据模型封装在一个viewmodel里面

<http://www.cocoachina.com/ios/20160421/15919.html>

2.通过ViewModel规范TableView界面开发

代码地址：<https://github.com/youzan/SigmaTableViewModel>

比较好的利用了mvvm封装各种分组tablelview，但是没有对自定义cell提供便利的接口

<http://www.cocoachina.com/ios/20160428/16002.html>

3.对自定义cell提供便利的接口，也是基于mvvm思想

代码地址：<https://github.com/chentoo/AITableView>

http://www.cocoachina.com/ios/20160216/15232.html

4．比较好的mvvm结合tableview的案例：（作者的github首页还收集了很多关于项目架构的文章，滴滴，新浪微博，蘑菇街）

<https://github.com/lovemo/MVVMFramework>

蘑菇街架构代码：

[https://github.com/mogujie/MGJRouter.git](https://github.com/mogujie/MGJRouter.git" \t "_blank)

iOS组件化思路－大神博客研读和思考

<http://www.jianshu.com/p/afb9b52143d4>

版本号规范：

<http://semver.org/>

持续集成，通过脚本完成：

<https://github.com/fastlane/fastlane>

5. iOS应用架构谈 开篇

http://casatwy.com/iosying-yong-jia-gou-tan-kai-pian.html

6. 围观神仙打架，反革命工程师《iOS应用架构谈 组件化方案》和蘑菇街Limboy的《蘑菇街 App 的组件化之路》的阅读指导

http://reviewcode.cn/article.html?reviewId=20

7.网易电商的app项目架构：LDBus

<https://github.com/Lede-Inc/LDBusBundle_IOS>

# 8.iOS组件化实践方案－LDBusMediator炼就

http://blog.csdn.net/daiyelang/article/details/51648036

9.一些常用的节省开发时间的分类

https://github.com/randomprocess/SUIUtils



10.cocoapods：

参考文章链接：

http://www.jianshu.com/p/32d9cfb91471

自我总结集成的过程：

1. 查看源：

gem sources -l

2. 删除默认的源（速度慢）

gem sources --remove <http://xxxx.com>

3. 添加淘宝的源：

gem sources -a <https://ruby.taobao.org/>

4. 更新源：

sudo gem update --system

5. 安装cocoapods

由于最新的cocoapods版本安装之后 pod setup会很慢，所以我们先安装 0.39版本，等pod setup成功之后，

再升级cocoapods为最新版本

sudo gem install -n /usr/local/bin cocoapods -v 0.39 （os10.11系统用这个）

sudo gem install cocoapods -v 0.39 （os10.11以前用这个）

pod setup的时候在这个~/.cocoapods目录下载仓库文件 会耗费较长时间，可以在pod setup的时候进入这个目录查看

进度，预计150M左右

0.39版本 pod setup 依然很慢的话：

5.1.删除国外的镜像源：

pod repo remove master

5.2.添加下面的镜像源coding 上有每日更新的，建议使用这个

pod repo add master <https://git.coding.net/CocoaPods/Specs.git>

5.3.然后再次执行 pod setup

更新仓库：

pod repo update

6.pod setup成功之后可以搜索一个第三方试试 pod search AFNetworking

7.使用cocoapods 集成第三方框架：

7.1：cd 切换到xocde项目目录

7.2：创建配置文件：vim Podfile

7.3 vim编辑器 写入以下内容：

platform :ios, '8.0'

#use\_frameworks!个别需要用到它，比如reactiveCocoa

target 'MyApp' do

pod 'AFNetworking', '~> 2.6'

pod 'ORStackView', '~> 3.0'

pod 'SwiftyJSON', '~> 2.3'

end

第二种写法：

platform :ios, '8.0'

#use\_frameworks!个别需要用到它，比如reactiveCocoa

def pods

pod 'AFNetworking', '~> 2.6'

pod 'ORStackView', '~> 3.0'

pod 'SwiftyJSON', '~> 2.3'

end

target 'MyApp' do

pods

end

11.混编ObjectiveC++

http://www.jianshu.com/p/1a7d09788ca2

12 . 漫谈iOS AOP编程之路

<http://www.jianshu.com/p/addd4eac54ed>

Aspects AOP 库，封装了 Runtime ， Method Swizzling 这些黑色技巧

13. iOS 启动连续闪退保护方案

<http://wereadteam.github.io/2016/05/23/GYBootingProtection/>

<https://github.com/liuslevis/GYBootingProtection>

14. iOS内存泄漏自动检测工具PLeakSniffer

<http://www.cocoachina.com/ios/20160706/16951.html>

MLeaksFinder：精准 iOS 内存泄露检测工具

http://wereadteam.github.io/2016/02/22/MLeaksFinder/

15：源码解析之–YYAsyncLayer异步绘制

<http://ios.jobbole.com/86878/>

16. iOS初探+load和+initialize

<http://blog.csdn.net/caryaliu/article/details/49284111>

17. 微信读书 iOS 性能优化总结

<http://wereadteam.github.io/2016/05/03/WeRead-Performance/#more>

# 18. iOS 保持界面流畅的技巧

# http://blog.ibireme.com/2015/11/12/smooth\_user\_interfaces\_for\_ios/