

iOS应用开发Tips

UC浏览器
客户端研发部
iPhone 组
梁劲
2014.06.21



内容概要

- 提高程序稳定性
- 让程序变得流畅
- 实用工具介绍



稳定性

- 追溯崩溃现场
- 常见的问题代码
- 提早发现问题



追溯崩溃现场

开源项目: PLCrashReport

官网: www.plcrashreporter.org

作用:

1. 使用公开的API来生成崩溃报告
2. 可跟踪所有线程堆栈
3. 不依赖LLDB/GDB
4. 生成的日志格式与Apple日志格式一致
5. 易于集成到应用
6. 可自定义扩展



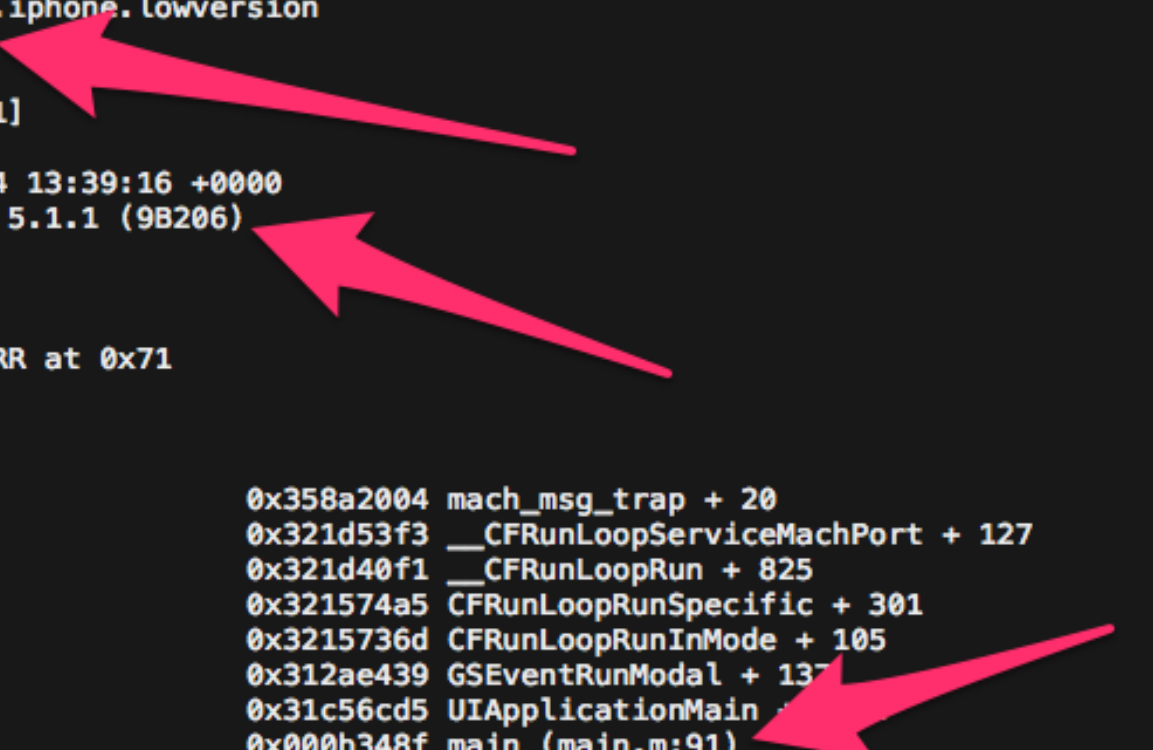
PLCrashReport日志例子

```
Hardware Model:      iPod4,1
Process:             UCWEB [8757]
Path:                /var/mobile/Applications/9E2756A4-A094-4303-8499-A64DE21F2A99/UCWEB.app/UCWEB
Identifier:          com.ucweb.iphone.lowversion
Version:             9.4.0.342
Code Type:           ARM
Parent Process:      launchd [1]

Date/Time:           2013-12-14 13:39:16 +0000
OS Version:          iPhone OS 5.1.1 (9B206)
Report Version:      104

Exception Type:      SIGSEGV
Exception Codes:     SEGV_ACCERR at 0x71
Crashed Thread:      13

Thread 0:
0  libsystem_kernel.dylib      0x358a2004 mach_msg_trap + 20
1  CoreFoundation              0x321d53f3 __CFRunLoopServiceMachPort + 127
2  CoreFoundation              0x321d40f1 __CFRunLoopRun + 825
3  CoreFoundation              0x321574a5 CFRunLoopRunSpecific + 301
4  CoreFoundation              0x3215736d CFRunLoopRunInMode + 105
5  GraphicsServices            0x312ae439 GSEventRunModal + 137
6  UIKit                       0x31c56cd5 UIApplicationMain + 1181
7  UCWEB                        0x000b348f main (main.m:91)
```



稳定性

- 追溯崩溃现场
- 常见的问题代码
- 提早发现问题



常见的问题代码(1)

assign类型delegate需要在不使用时置空
常见于UITableView、UIWebView、
UIGesture等

```
- (void)dealloc
{
    self.webView.delegate = nil;
    [_webView release];
    _webView = nil;

    [super dealloc];
}
```



常见的问题代码(2)

一对多的观察者，在回调时候注意观察者列表被修改

```
- (void)notifyObserversWithRecorderCreated:(PagePerformanceRecorder *)recorder
{
    for (id <Observer> observer in self.observers)
    {
        [observer recorderFactory:self didCreatedRecorder:recorder];
    }
}
```



常见的问题代码(3)

使用NSKeyedUnarchiver反序列化时
添加@catch防止文件被破坏从而触发
exception的情况

```
@try {  
    arr = [NSKeyedUnarchiver unarchiveObjectWithFile:@"data.plist"];  
}  
@catch (NSException *exception) {  
    handler  
}  
@finally {  
    statements  
}
```



常见的问题代码(4)

注意多线程操作令对象在子线程析构

```
@implementation JViewController

- (void)viewDidLoad
{
    [super viewDidLoad];
    [NSThread detachNewThreadSelector:@selector(doJob) toTarget:self withObject:nil];
}

- (void)dealloc
{
    [self.view removeFromSuperview];
    [super dealloc];
}
```



常见的问题代码(5)

注意空指针。
系统函数部分不允许空指针传入参数

```
- (NSString*)makeNSString:(const char*)utf8  
{  
    return [NSString stringWithUTF8String:utf8];  
}
```



稳定性

- 追溯崩溃现场
- 常见的问题代码
- 提早发现问题



提早发现问题手段(1)

在一个悬崖边上立着一个警示牌，上面醒目地写着：“WARNING：前面是悬崖”。

然后，所有经过此地的程序员都掉了下去……



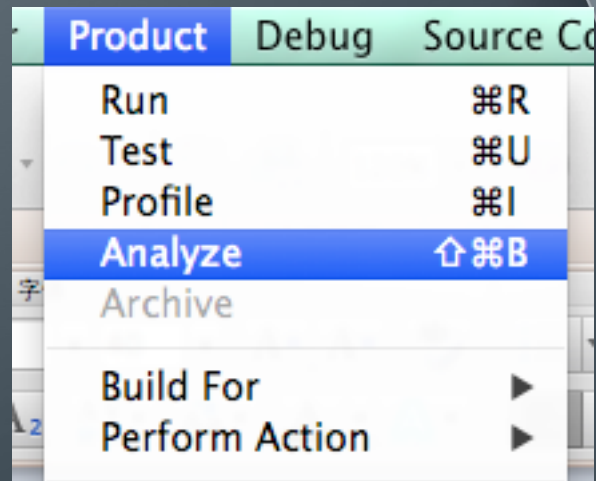
提早发现问题手段(1)

- 
- Build target TestConnection
- Project TestConnection | Configuration Debug | Destination iPhone Retina (4-inch) | SDK Simulator - iOS 7.0
 - Precompile TestConnection/TestConnection-Prefix.pch
 - Compile main.m ...in /Users/liangjin/Desktop/UCProject/TestConnection/TestConnection
 - Compile JURLCache.m ...in /Users/liangjin/Desktop/UCProject/TestConnection/TestConnection
 - Compile AppDelegate.m ...in /Users/liangjin/Desktop/UCProject/TestConnection/TestConnection
 - Compile JTestProtocol.m ...in /Users/liangjin/Desktop/UCProject/TestConnection/TestConnection
 - Compile JViewController.m ...in /Users/liangjin/Desktop/UCProject/TestConnection/TestConnection
 - Link /Users/liangjin/Desktop/UCProject/TestConnection/DerivedData/TestConnection/Build/Products/Debug-iphonesimulator/TestConnection.app/TestConnection
 - CompileStoryboard TestConnection/Base.lproj/Main_iPad.storyboard
 - CompileStoryboard TestConnection/Base.lproj/Main_iPhone.storyboard
 - Copy TestConnection/en.lproj/InfoPlist.strings
 - CompileAssetCatalog DerivedData/TestConnection/Build/Products/Debug-iphonesimulator/TestConnection.app TestConnection/Images.xcassets
 - Process TestConnection-Info.plist ...in /Users/liangjin/Desktop/UCProject/TestConnection/TestConnection
 - Generate TestConnection.app.dSYM ...in /Users/liangjin/Desktop/UCProject/TestConnection/DerivedData/TestConnection/Build/Products/Debug-iphonesimulator
 - Touch /Users/liangjin/Desktop/UCProject/TestConnection/DerivedData/TestConnection/Build/Products/Debug-iphonesimulator/TestConnection.app
- Build succeeded 14-6-20 下午8:49
No issues



提早发现问题手段(2)

使用静态分析发现问题



```
- (id)initWithRequest:(NSURLRequest *)request cachedResponse:(NSCachedURLResponse *)cachedResponse {
    NSMutableURLRequest* mRequest = [request mutableCopy];
    self = [super initWithRequest:mRequest cachedResponse:cachedResponse client:client];
    if (self) {
        assert(cachedResponse == nil);
    }
    return self;
}
```

1. Method returns an Objective-C object with a +1 retain count

2. Object leaked: object allocated and stored into 'mRequest' is not referenced later in this execution path and has a retain...



提早发现问题手段(3)

Debug版下多使用assert

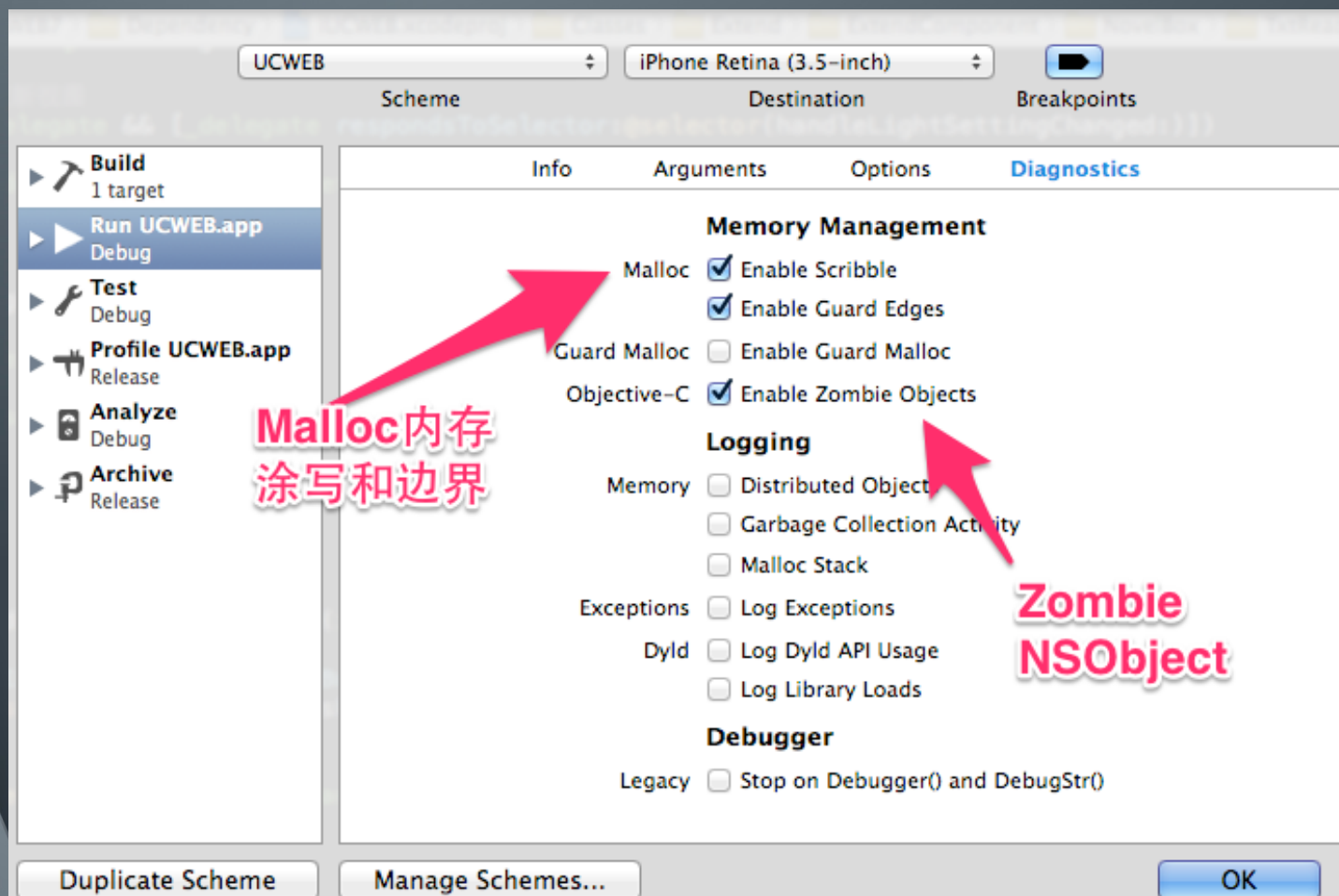
```
@implementation JViewController  
  
- (void)dealloc  
{  
    assert([NSThread isMainThread]);  
    [super dealloc];  
}
```

```
- (void)didSkinItemClicked:(id)sender  
{  
    NSInteger skinID = ((UIButton*)sender).tag;  
    assert(NBSkin_white <= skinID && skinID <= NBSkin_black);  
}
```



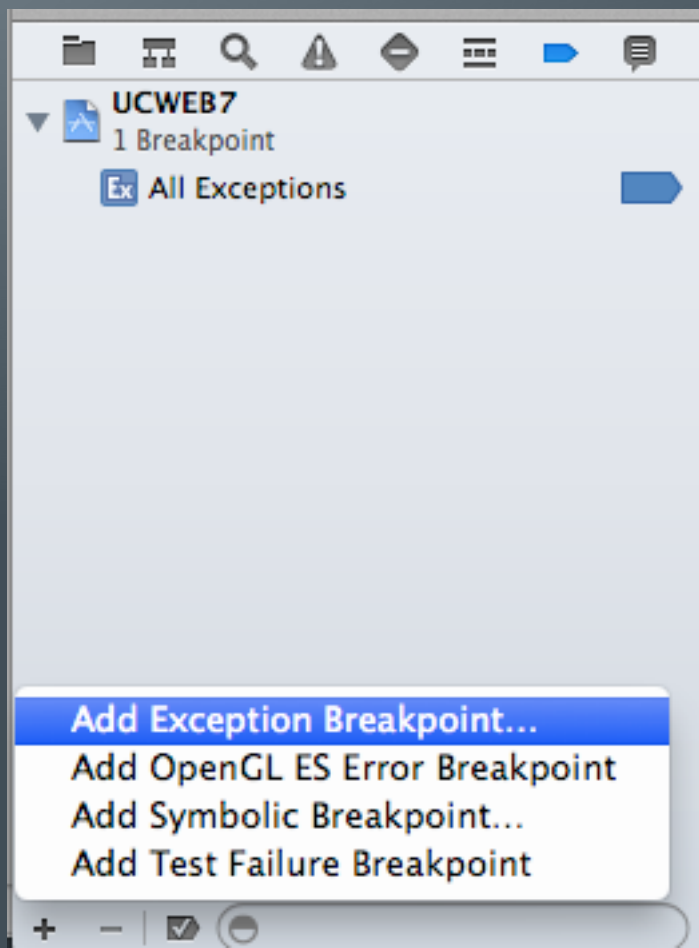
提早发现问题手段(4)

调试时打开内存诊断参数



提早发现问题手段(5)

添加Exception断点



稳定性小结

编码的时候不要掉以轻心

逻辑代码应该考虑好各种路径



让程序变得流畅

- 快速响应操作
- 任务执行迅速



快速响应操作(1)

- 长耗时操作转移到子线程进行

- 1) IO操作
- 2) 大批量运算
- 3) 联网数据处理



快速响应操作(2)

- 长耗时任务分片

例子:

需要删除包含1000个文件的文件夹

```
bool shouldContinue = true;
NSEnumerator* e = [filePathArray objectEnumerator];
NSString* path = nil;
while (shouldContinue
      && (path = [e nextObject]))
{
    [[NSFileManager defaultManager] removeItemAtPath:path error:nil];
}
```



快速响应操作(3)

- 减少频率太高的动画

人眼觉得流畅的帧率是24 FPS(帧/秒)
60FPS 已经极度流畅并且接近硬件极限

```
- (void)startAnimation
{
    self.animTimer = [NSTimer scheduledTimerWithTimeInterval:0.008
                                                            target:self
                                                            selector:@selector(timeoutProcess)
                                                            userInfo:nil
                                                            repeats:NO];
}
```

浪费!



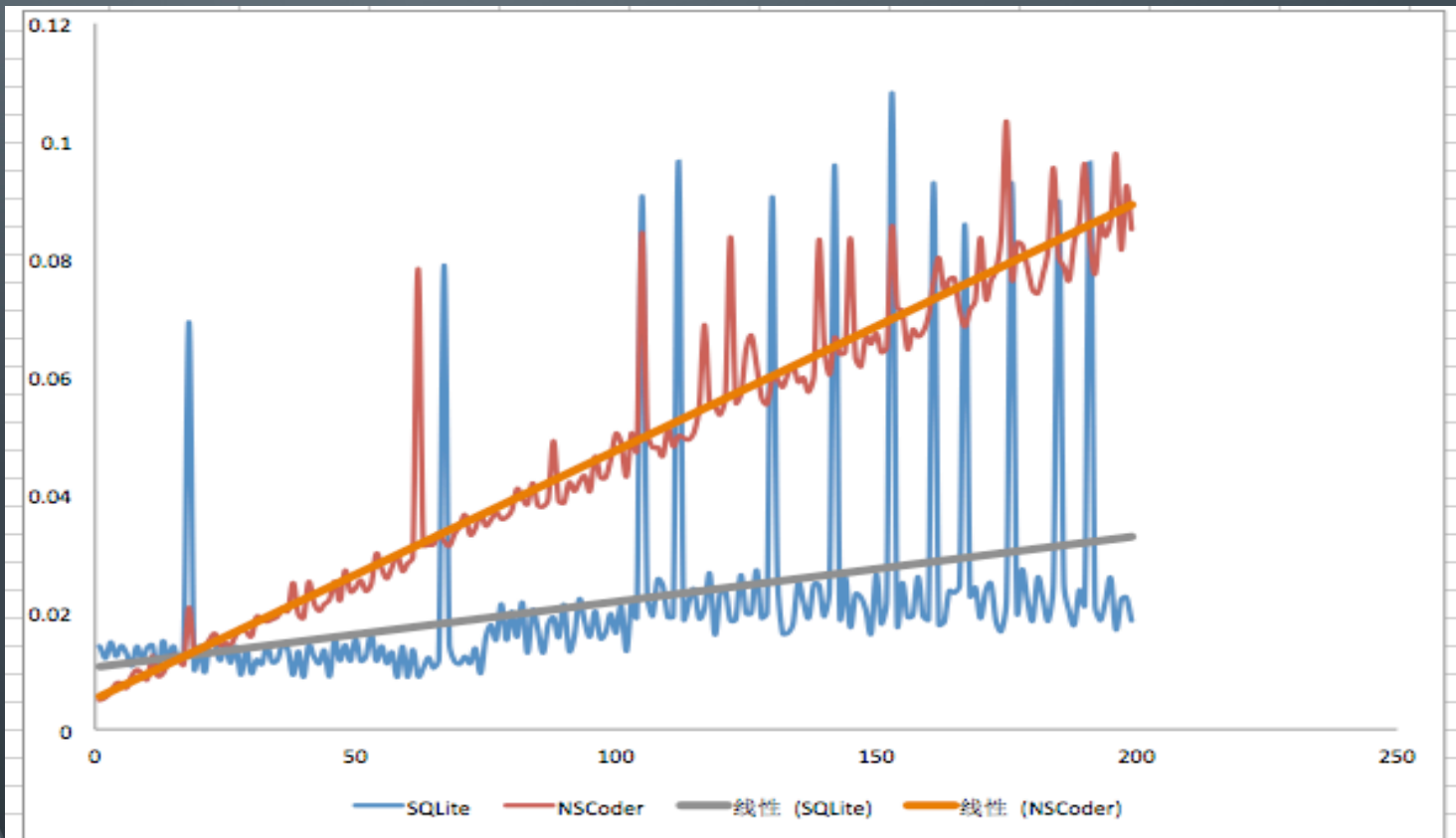
让程序变得流畅

- 快速响应操作
- 任务执行迅速



任务执行迅速(1)

对于频繁修改的持久层
使用sqlite 代替 plist



任务执行迅速(2)

对于大内存传递
考虑使用NoCopy的方法托管内存
避免冗余复制和内存峰值

```
- (NSData*)bigDataToNSData:(char*)bigChar withLength:(int)length  
{  
    return [NSData dataWithBytesNoCopy:bigChar length:length freeWhenDone:YES];  
}
```



任务执行迅速(3)

使用instrument找出瓶颈

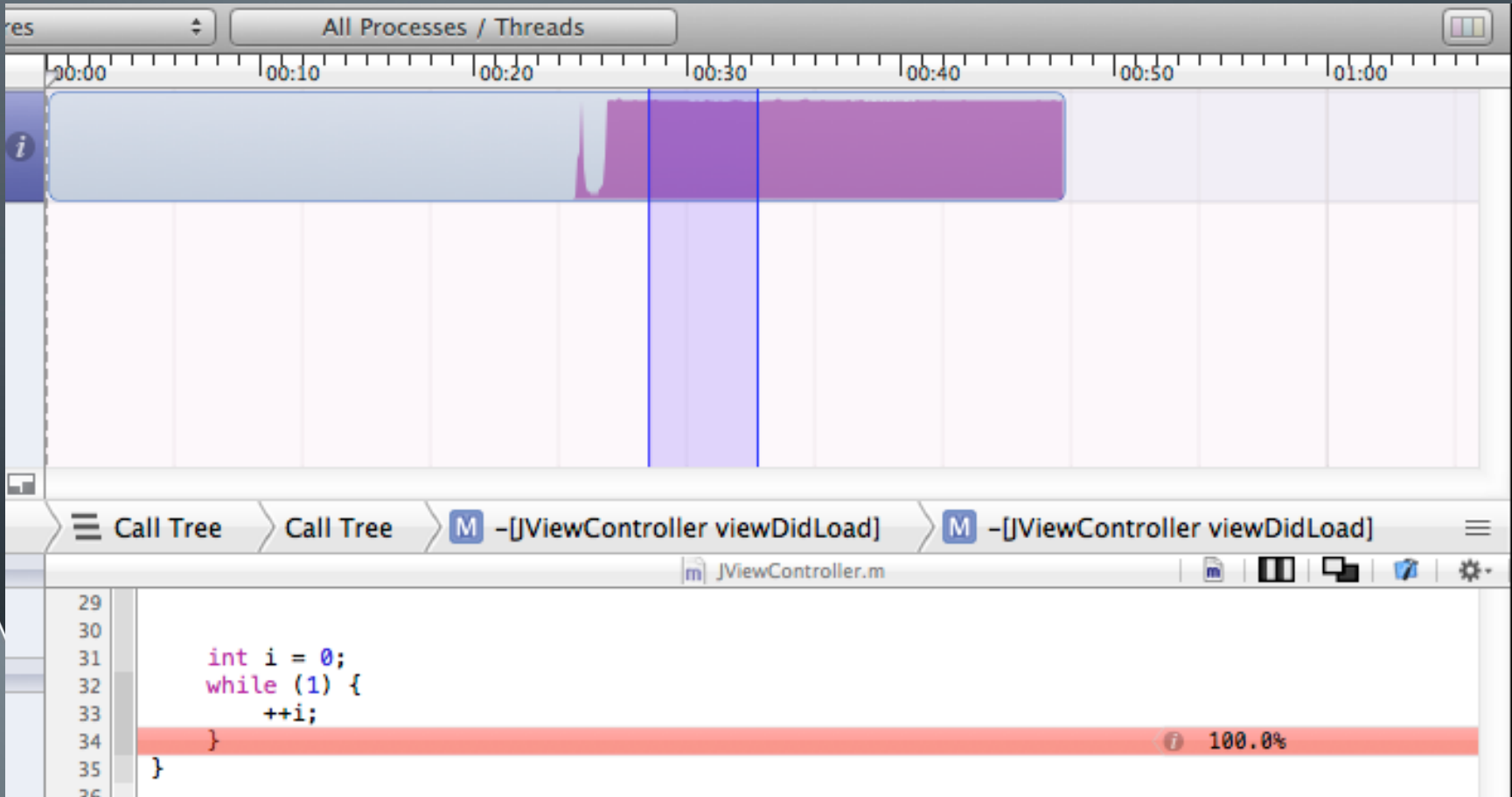
Choose a Template for the Trace Document:

The screenshot displays the 'Choose a Template for the Trace Document' dialog in Xcode. On the left, there are two category lists: 'iOS' and 'iOS Simulator'. The 'iOS' list includes 'All', 'Memory', 'CPU', 'I/O Activity', and 'Graphics'. The 'iOS Simulator' list includes 'All', 'Memory', 'CPU', and 'File System'. The main area shows eight instrument templates arranged in a 2x4 grid. The 'Time Profiler' template is highlighted with a grey border. The templates are: Blank (a dashed box), Allocations (a 2x2 cube with water), Leaks (a pipe with water dripping), Activity Monitor (a laptop with a heart rate line), Zombies (a 2x2 cube with yellow and grey faces), Time Profiler (a speedometer showing 45), System Trace (a laptop with code), and Automation (a robotic device).

Category	Template Name	Icon Description
iOS	All	Dashed box
	Memory	2x2 cube with water
	CPU	Pipe with water dripping
	I/O Activity	Laptop with heart rate line
	Graphics	Laptop with heart rate line
iOS Simulator	All	2x2 cube with yellow and grey faces
	Memory	Speedometer showing 45
	CPU	Laptop with code
	File System	Robotic device

任务执行迅速(3)

使用instrument找出瓶颈



小结

定期关注效率问题

起码需要处理好80%的情况



好用工具介绍

事半功倍
节省时间
避免重复造轮子



好用工具介绍(1)

- pngquant
- 对图片资源进行优化。半透明图片使用png，不透明图片使用jpg



Original PNG: **75,628** bytes

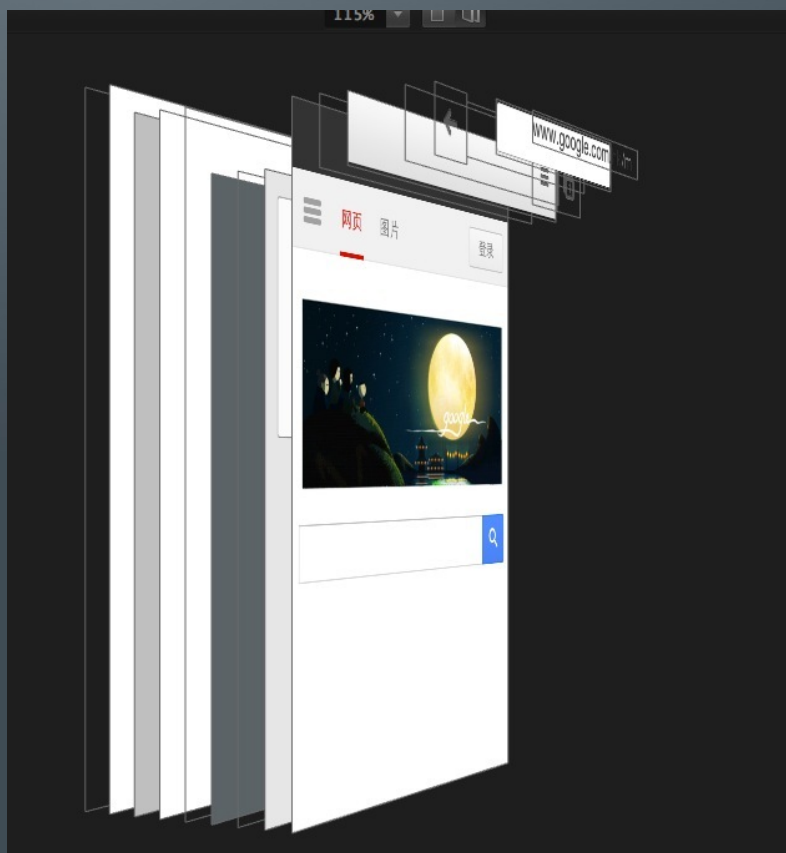


pngquant: **19,996** bytes (73% smaller)



好用工具介绍(2)

- Reveal
- 实时了解应用的View层次结构



好用工具介绍(3)

MSLeakHunter

What it looks like

- When you run the app with a leak hunter enabled, and it finds a possible object that is leaking, this is what you'll see:

```
2012-10-20 18:11:04.145 MSVCLeakHunterSampleProject[72927:c07] -[UINavigationController viewDidAppear:]
2012-10-20 18:11:04.146 MSVCLeakHunterSampleProject[72927:c07] -[MSMenuVC viewDidAppear:]
2012-10-20 18:12:20.204 MSVCLeakHunterSampleProject[72927:c07] -[MSMenuVC viewDidDisappear:]
2012-10-20 18:12:20.205 MSVCLeakHunterSampleProject[72927:c07] -[MSOKVC viewDidAppear:]
2012-10-20 18:12:21.062 MSVCLeakHunterSampleProject[72927:c07] -[MSOKVC viewDidDisappear:]
2012-10-20 18:12:21.063 MSVCLeakHunterSampleProject[72927:c07] -[MSMenuVC viewDidAppear:]
2012-10-20 18:12:21.063 MSVCLeakHunterSampleProject[72927:c07] -[MSOKVC dealloc]
2012-10-20 18:12:22.770 MSVCLeakHunterSampleProject[72927:c07] -[MSMenuVC viewDidDisappear:]
2012-10-20 18:12:22.770 MSVCLeakHunterSampleProject[72927:c07] -[MSLeakingVC viewDidAppear:]
2012-10-20 18:12:23.641 MSVCLeakHunterSampleProject[72927:c07] -[MSLeakingVC viewDidDisappear:]
2012-10-20 18:12:23.642 MSVCLeakHunterSampleProject[72927:c07] -[MSMenuVC viewDidAppear:]
2012-10-20 18:12:33.642 MSVCLeakHunterSampleProject[72927:c07] [MSVCLeakHunter] POSSIBLE LEAK OF VIEW CONTROLLER MSLeakingVC <0x719cf00>
```

screenshot from the sample project

THANK YOU

Q & A

