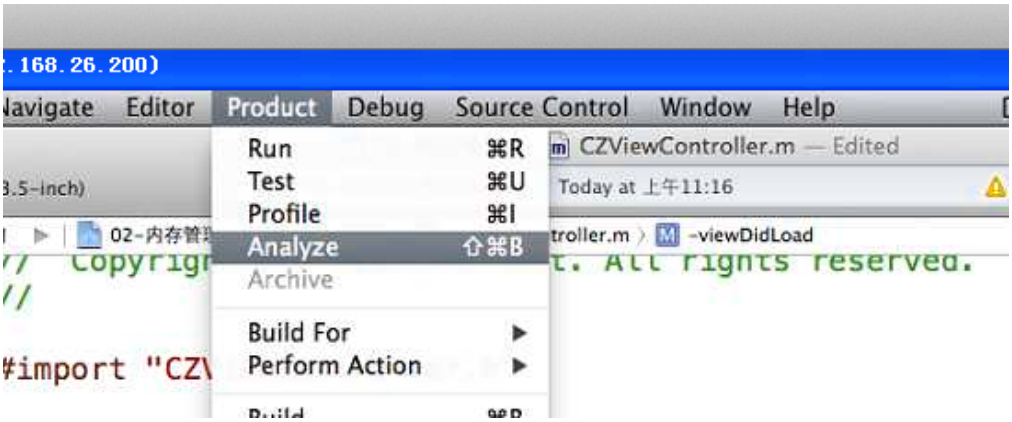
1. 怎么保证MRC多人开发进行内存泄露的检查.

1. 使用Analyze进行代码的静态分析

检测内存泄漏的方法:(leak,内存泄露)



2. (加分)为避免不必要的麻烦, 多人开发时尽量使用ARC(苹果会自动 给我们在需要的时候加release)

2. 非自动内存管理情况下怎么做单例模式.

创建单例设计模式的基本步骤 ·

>声明一个单件对象的静态实例,并初始化为nil。

>创建一个类的类工厂方法,当且仅当这个类的实例为nil时生成一个该类的实例

>实现NScopying协议， 覆盖allocWithZone:方法,确保用户在直接分配和初始化对象时,不会产 生另一个对象。

>覆盖release、autorelease、retain、retainCount方法, 以此确保单例的状态。

>在多线程的环境中,注意使用@synchronized关键字或GCD,确保静态实例被正确的创建和初始化。

3. 对于类方法（静态方法）默认是autorelease的。所有类方法都会这样吗?

```
NSArray *arr = [NSArray array];

int num = arr.count;
NSLog(@"%d", num);
```

1> 系统自带的绝大数类方法(NSArray array]返回的对象,都是经过autorelease的

2>alloc 的就不是...

4．block在ARC中和MRC中的用法有什么区别,需要注意什么

1.对于没有引用外部变量的Block，无论在ARC还是非ARC下，类型都是__NSGlobalBlock__，这种类型的block可以理解成一种全局的block，不需要考虑作用域问题。同时，对他进行Copy或者Retain操作也是无效的

2.应注意避免循环引用

5．什么情况下会发生内存泄漏和内存溢出(内存泄露多了,内存放不下了,就溢出了.)?

当程序在申请内存后，无法释放已申请的内存空间(例如一个对象或者变量使用完成后没有释放,这个对象一直占用着内存)，一次内存泄露危害可以忽略，但内存泄露堆积后果很严重，无论多少内存,迟早会被占光。内存泄露会最终会导致内存溢出！

当程序在申请内存时，没有足够的内存空间供其使用，出现out of memory；比如申请了一个int,但给它存了long才能存下的数，那就是内存溢出。

6．[NSArray arrayWithobject:<id>] 这个方法添加对象后，需要对这个数组做释放操作吗?

不需要 这个对象被放到自动释放池中

7．Json数据的解析，和解析数据的时候有内存泄露吗？有的话 如何解

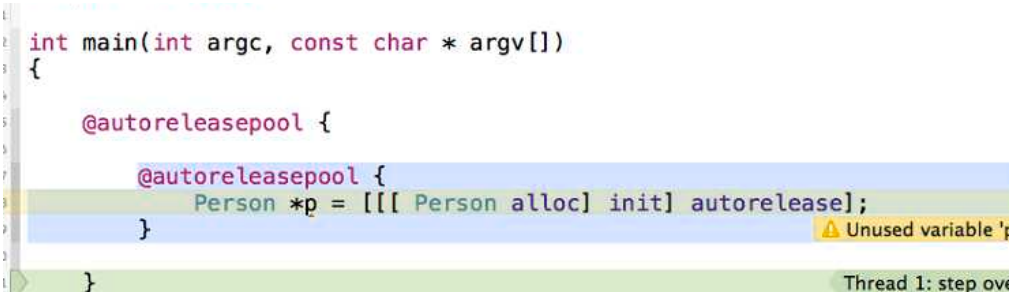
1. JSON解析的方案
- SBJson
- JSONkit
- NSJSONSerialization(ios自带的)
- 可以用静态工具检测下.(一般情况下都用苹果自带的,都没有造成内存泄露)

2. 内存泄漏么?

8. 自动释放池底层怎么实现
(以栈的方式实现的)(系统自动创建,系统自动释放)栈里面的(先进后出)
内存里面有栈,栈里面有自动释放池.

自动释放池以栈的形式实现:当你创建一个新的自动释放池时，它将被添加到栈顶。当一个对象收到发送autorelease消息时,它被添加到当前线程的处于栈顶的自动释放池中,当自动释放池被回收时,它们从栈中被删除，并且会给池子里面所有的对象都会做一次release操作.

只需要创建多个对象的时候...才需要手动管理自动释放池.



19行的时候释放.

手动管理自动释放池