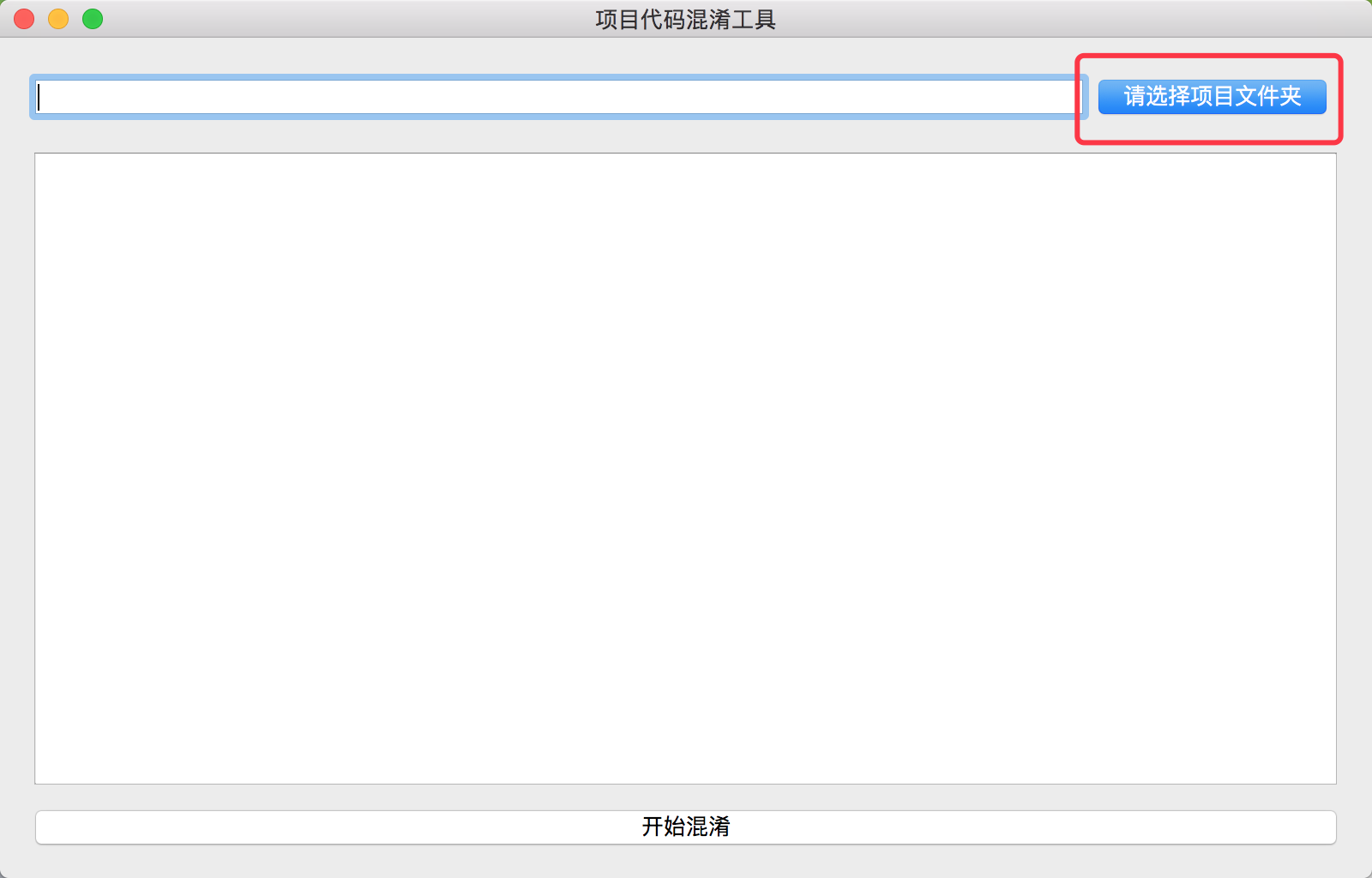
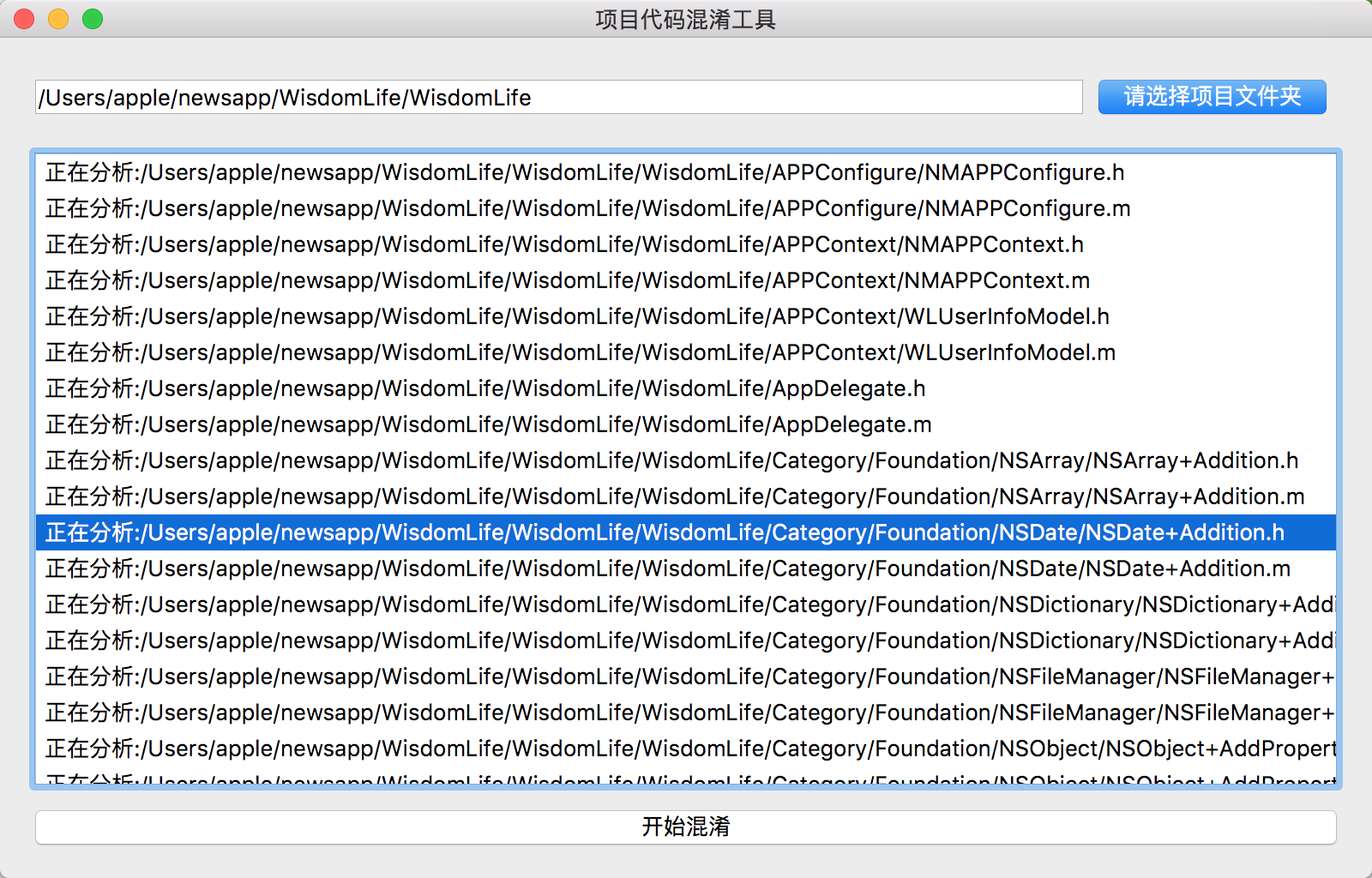
**使用该混淆工具之前，一定要备份原有项目，最好拷贝一份完整的项目文件单独进行混淆，防止混淆失败的时候破坏项目的结构！(该版本的混淆会根据类名混淆的结果修改对应类的xib的文件名，也会修改storyboard和xib文件中的内容)**

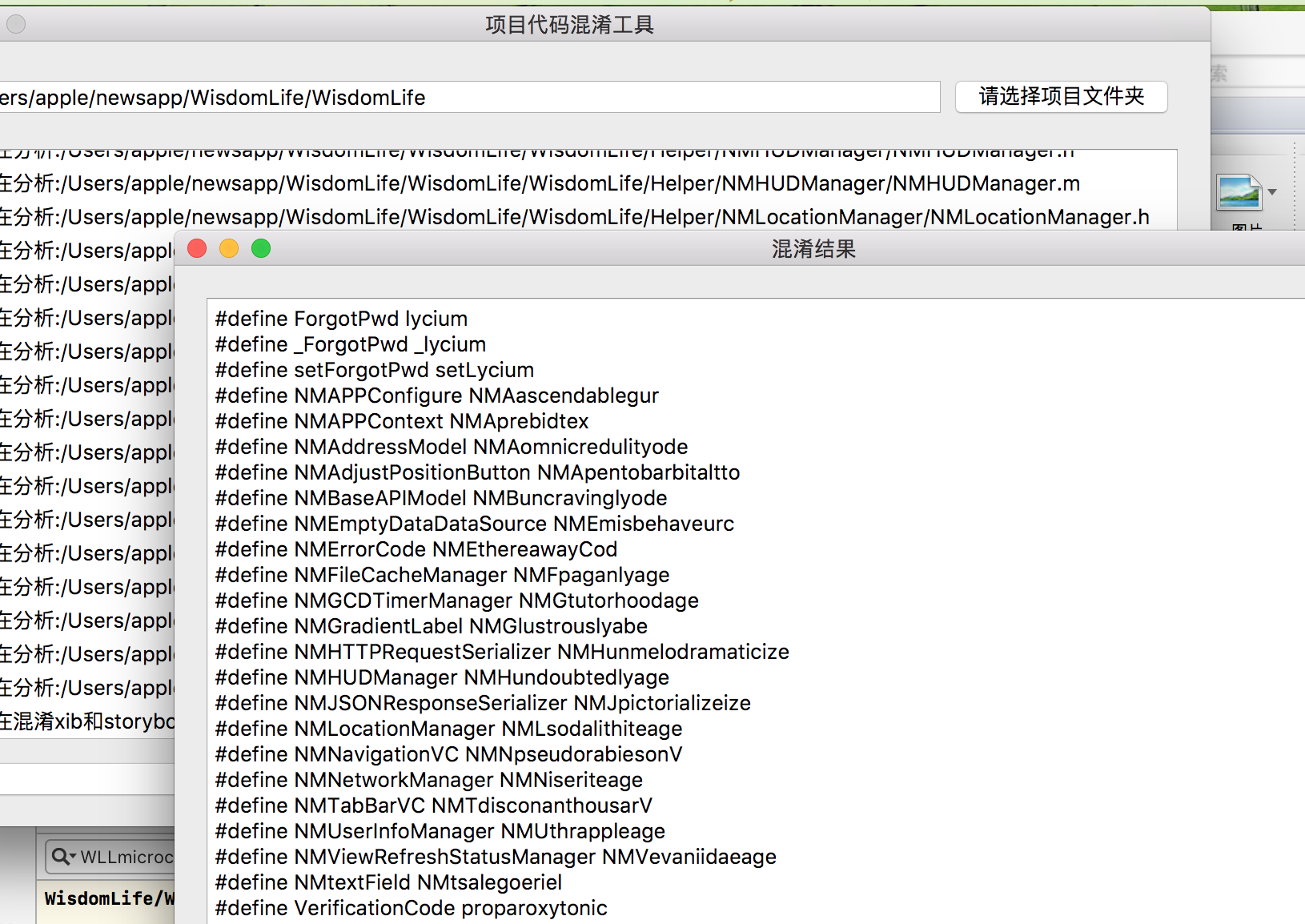
1. 点击“请选择项目文件夹”，选择要混淆的项目源代码工程



2. 点击开始混淆，进行源文件分析



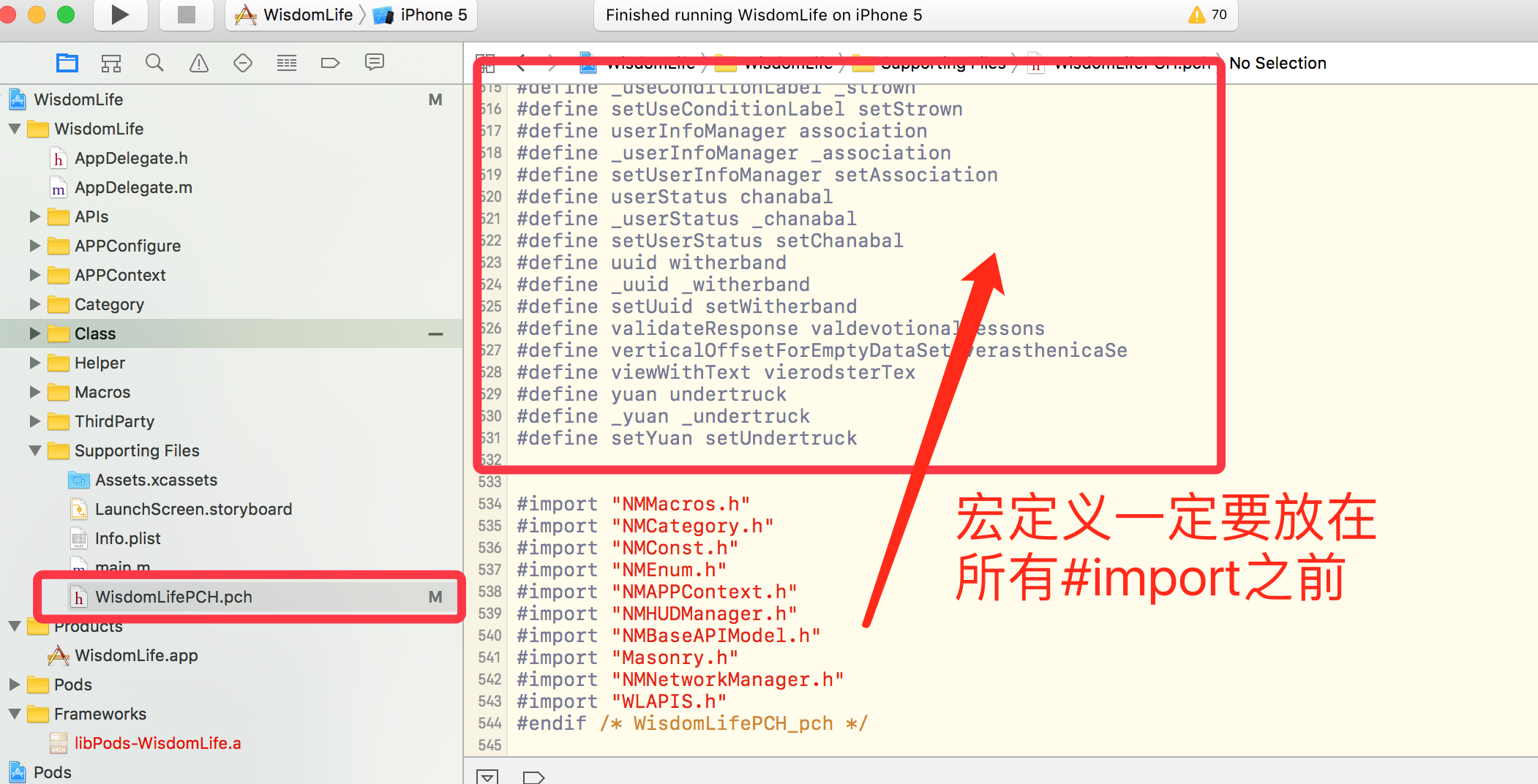
3. 等待分析完成，会弹出混淆结果，文本框中的代码可以复制或者剪切出来



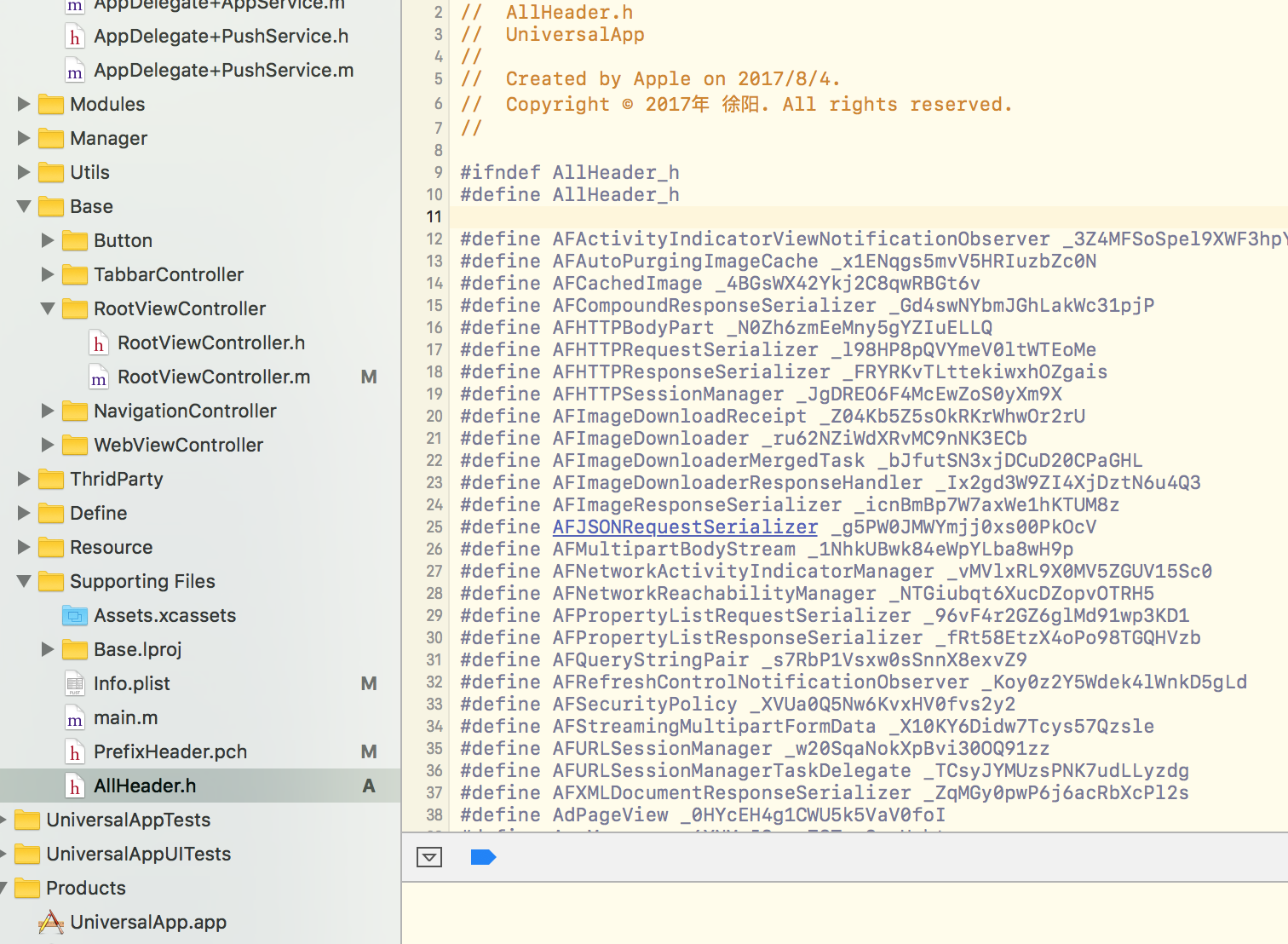
4. 将生成的宏定义代码粘贴到项目的XXX.pch 预编译头文件中：

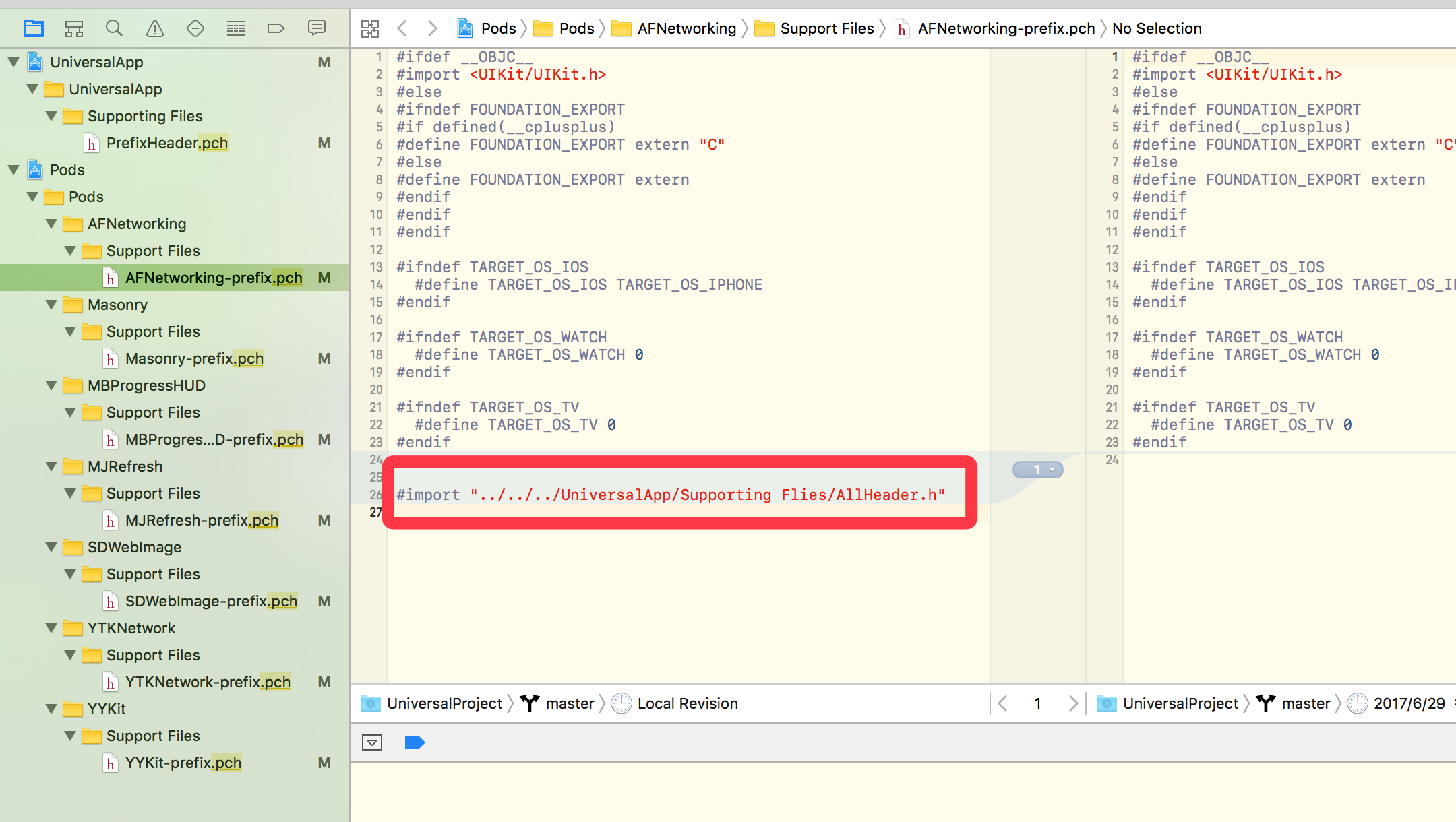
以cocos2d-x的项目为例，需要同时拷贝两分同样的宏定义代码到引擎的cocos2d-prefix.pch和ios目录下的Prefix.pch中，**一定要将宏定义拷贝到所有#import文件指令的上面，否则可能出现编译失败的情况，因为宏定义要在包含头文件之前执行**。

（注：cocos2d-prefix.pch可能需要手动右击add到项目中）



对于使用了cocoapods的项目，对pods库也进行了混淆，最好新建一个单独的头文件存放用于混淆的宏定义代码，因为不仅项目的Prefix.pch需要添加，pods库的预编译头文件也需要加，便于在每个pods库（已经编译成framework或者.a静态库的pods库不需要加）的pch文件里面加入：

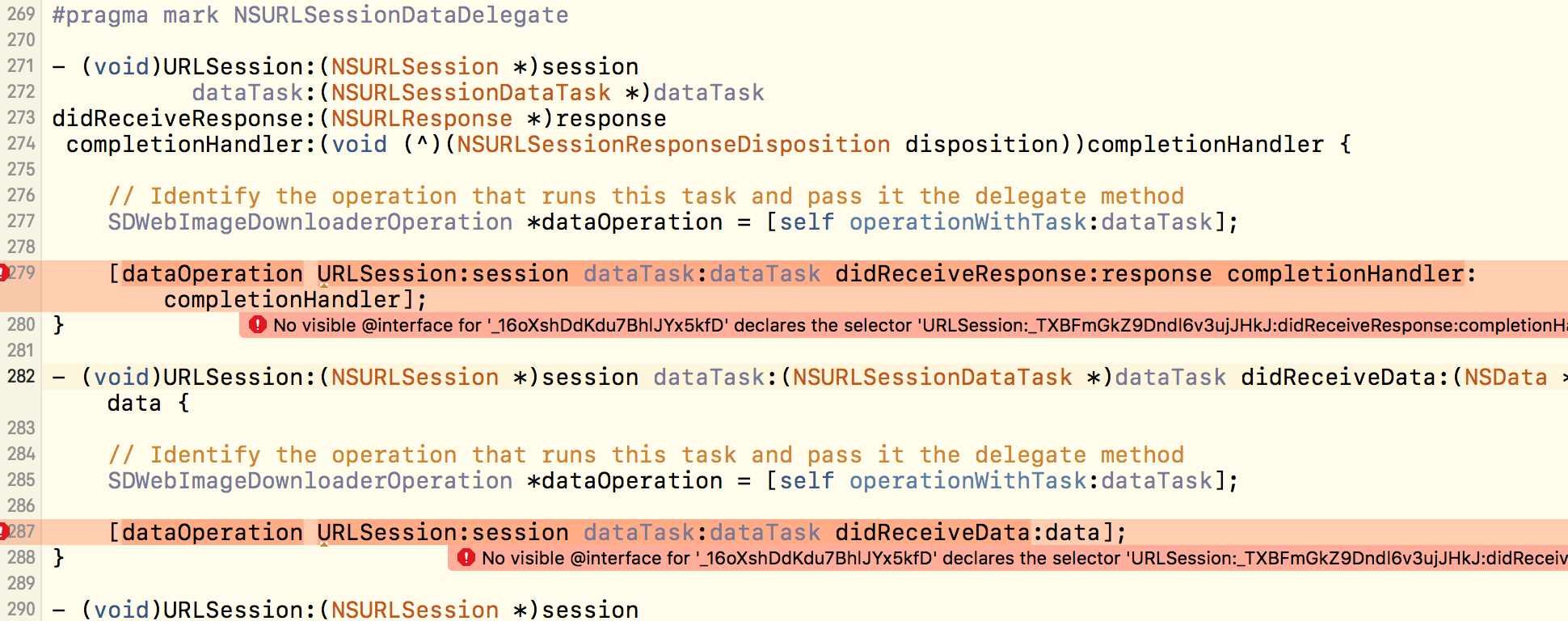




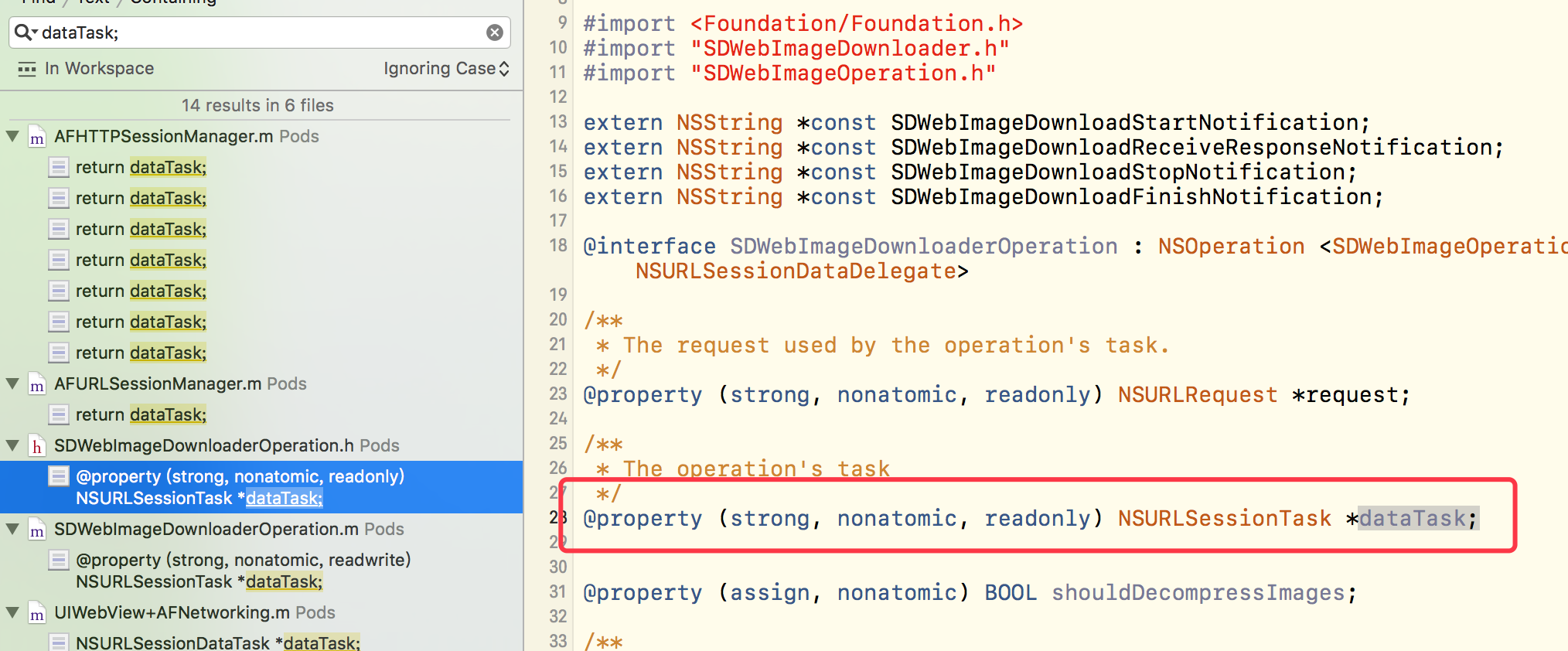
5. 编译过程中可能会遇到报错的情况(工具没有做到100%完美，识别出的标识符可能与C/C++库的函数或者系统类型冲突)，根据报错信息，删除对应的#define宏定义即可（如果是property属性报错，有可能需要删掉三个宏定义（例如属性@property (nonatomic,assign) int age, 混淆后会自动生成对 age, \_age 和 setAge的宏定义，所以需要删除这三个对应的宏定义））

可能遇到的编译错误和解决方法示例：

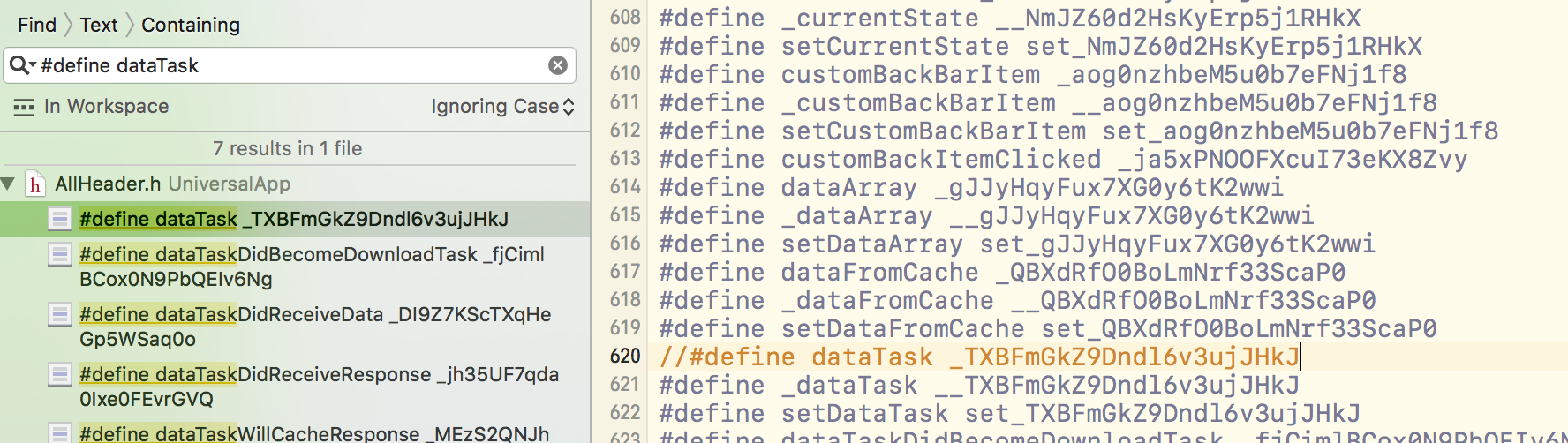
(1).No visible @interface for '\_16oXshDdKdu7BhlJYx5kfD' declares the selector 'URLSession:\_TXBFmGkZ9Dndl6v3ujJHkJ:didReceiveResponse:completionHandler:'



可以看出，是dataTask字段被重新宏定义了，所以识别不出来

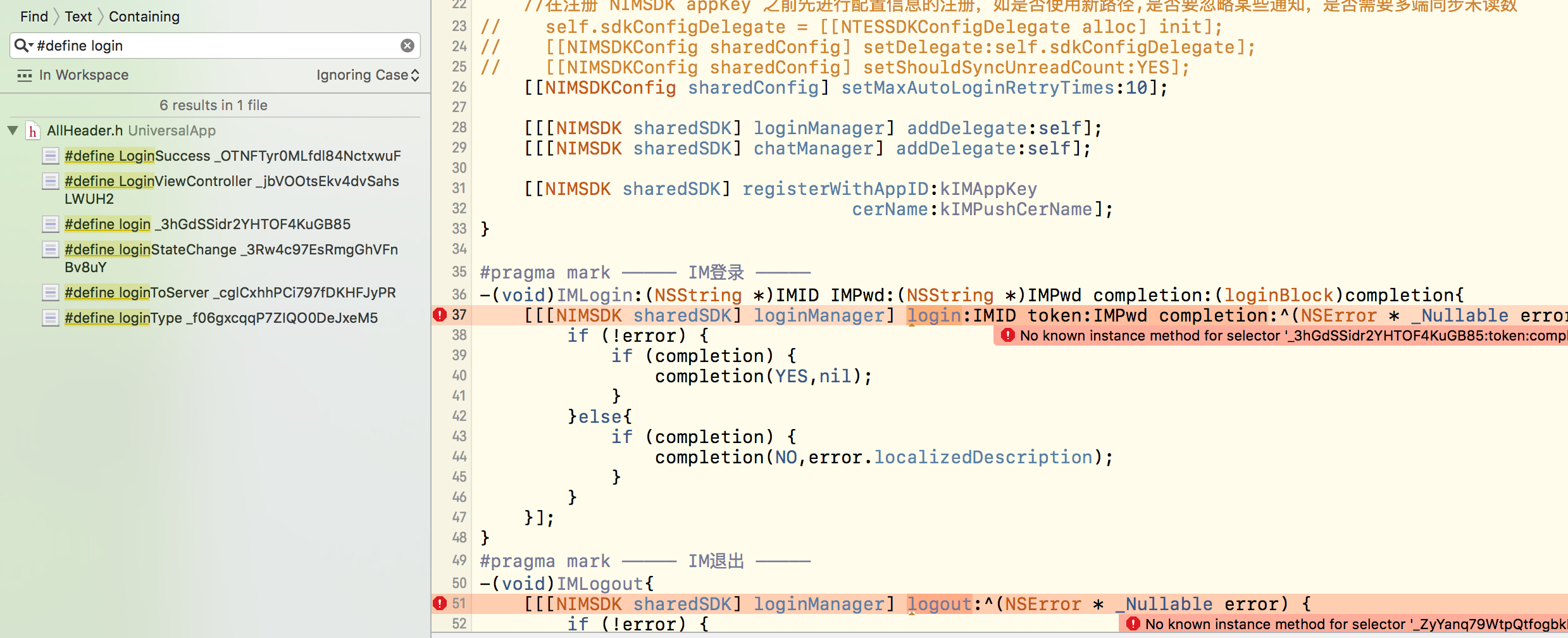


混淆的时候会混淆项目中的属性和方法，如果属性名称和方法中的某个名称重复，就会导致这种错误，所以解决方法就是删除或者注释dataTask对应的宏定义，

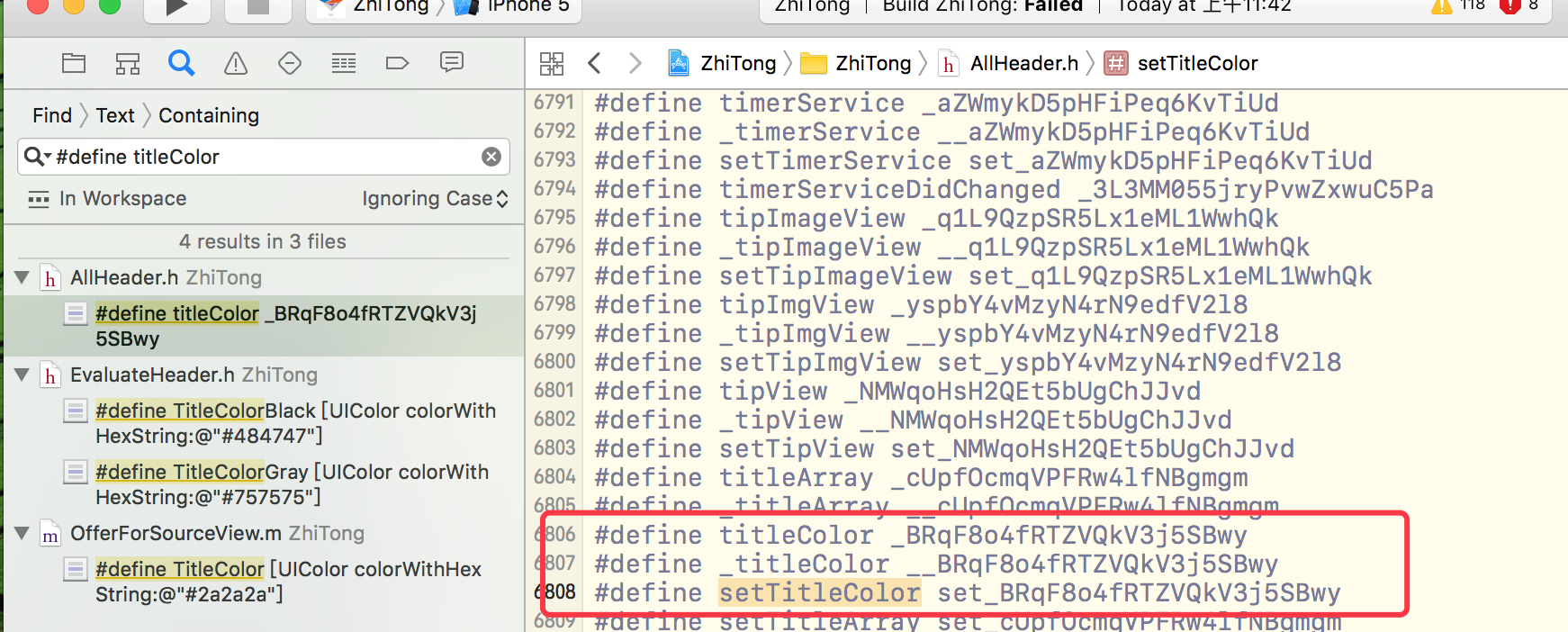
****

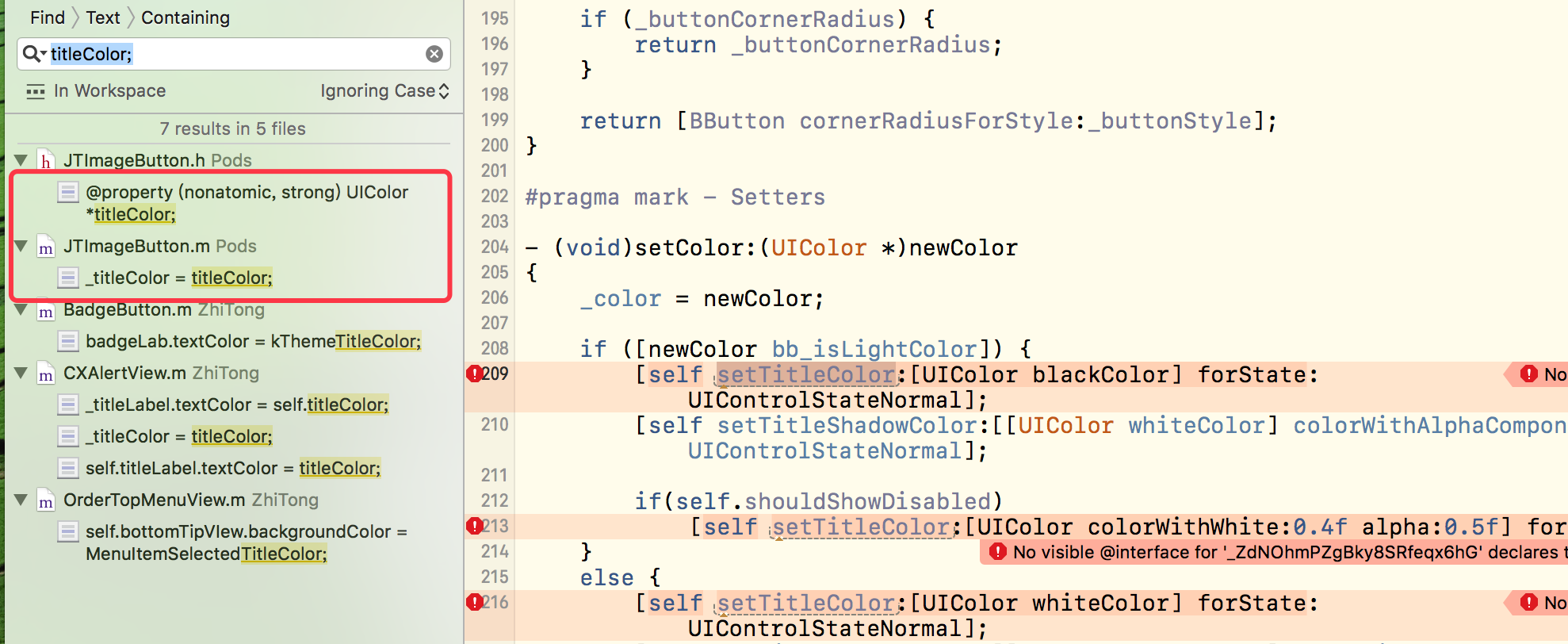
(2).No known instance method for selector '\_3hGdSSidr2YHTOF4KuGB85:token:completion:'

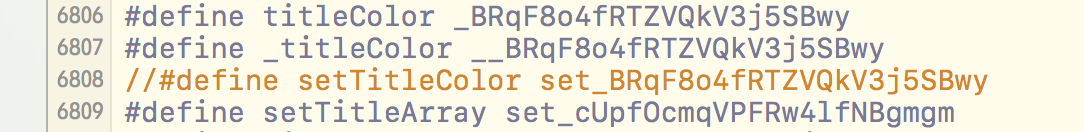
这种也是找不到方法名称的错误，可以直接搜索 \_3hGdSSidr2YHTOF4KuGB85 这个字符串，找到对应的宏定义删除或者注释



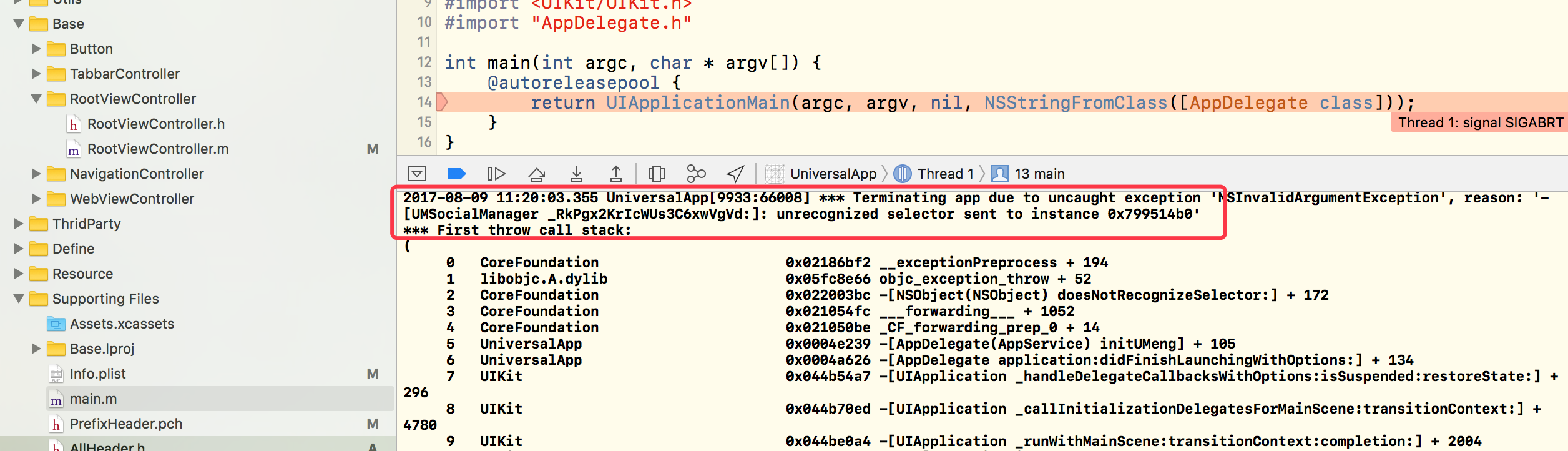
(3). 项目或者第三方库中的属性或者方法名称，与UIKit等系统的标识符名称重复，也会导致编译报错，如下所示，titleColor这个自己定义的属性，导致生成的setTitleColor宏定义与系统UIButton的setTitleColor重复了，所以需要注释掉



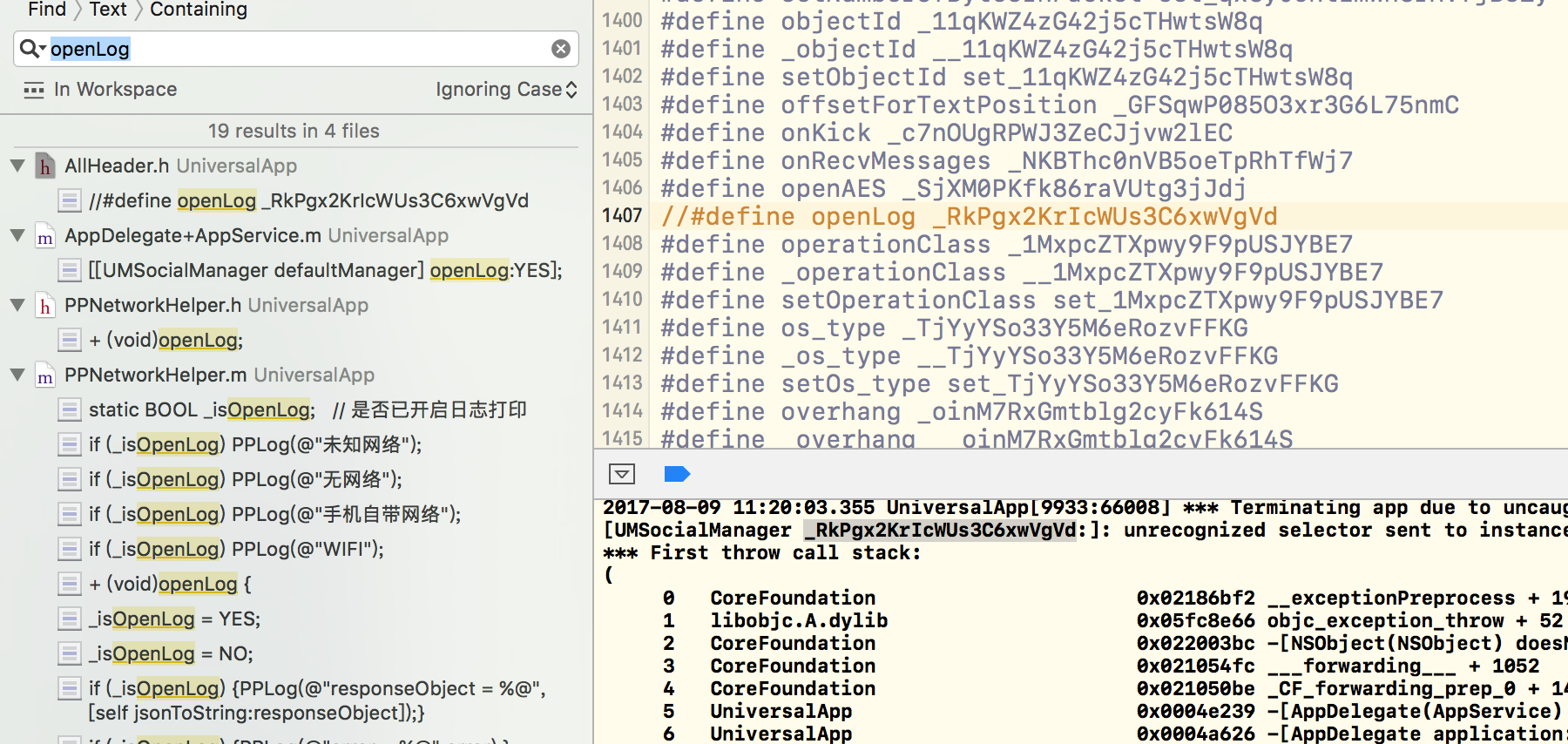


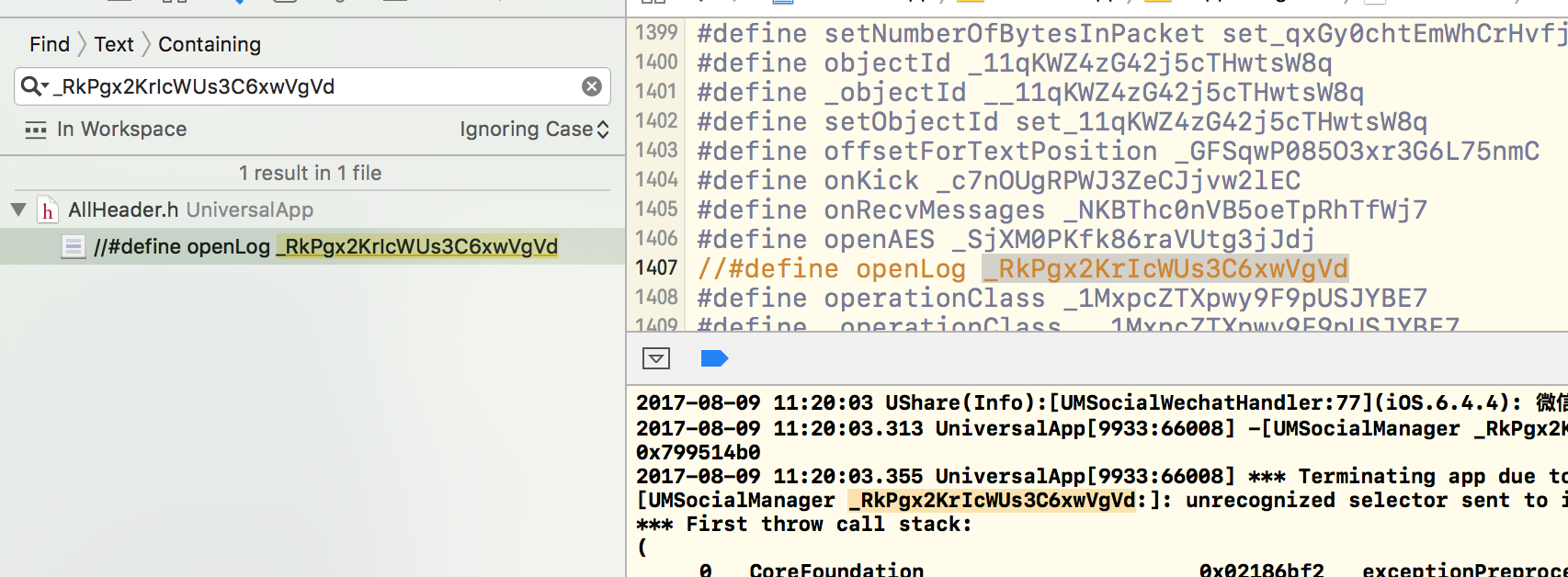


(4). 运行时崩溃



这种很可能是项目中使用了和第三方SDK同样的属性名称，或者方法名称导致的，由于宏定义无法改变已经编译成.a库或者framework里的标识符，所以只能搜索对应的宏定义删除或注释了，如下图，PPNetworkHelper中定义了与友盟SDK中一样的函数名称：openLog，找到对应宏定义注释掉，重新编译运行后一切正常。





6. 个别项目混淆后运行也可能出问题（很可能是项目代码中的一些属性名称和方法名称与第三方SDK的重名了，导致#define的东西把第三方SDK调用的地方也改了，故有可能会导致崩溃），所以混淆后最好所有功能都走一遍。

7. 使用该工具混淆的项目使用Hopper Disassembler反编译后的效果如下图：

