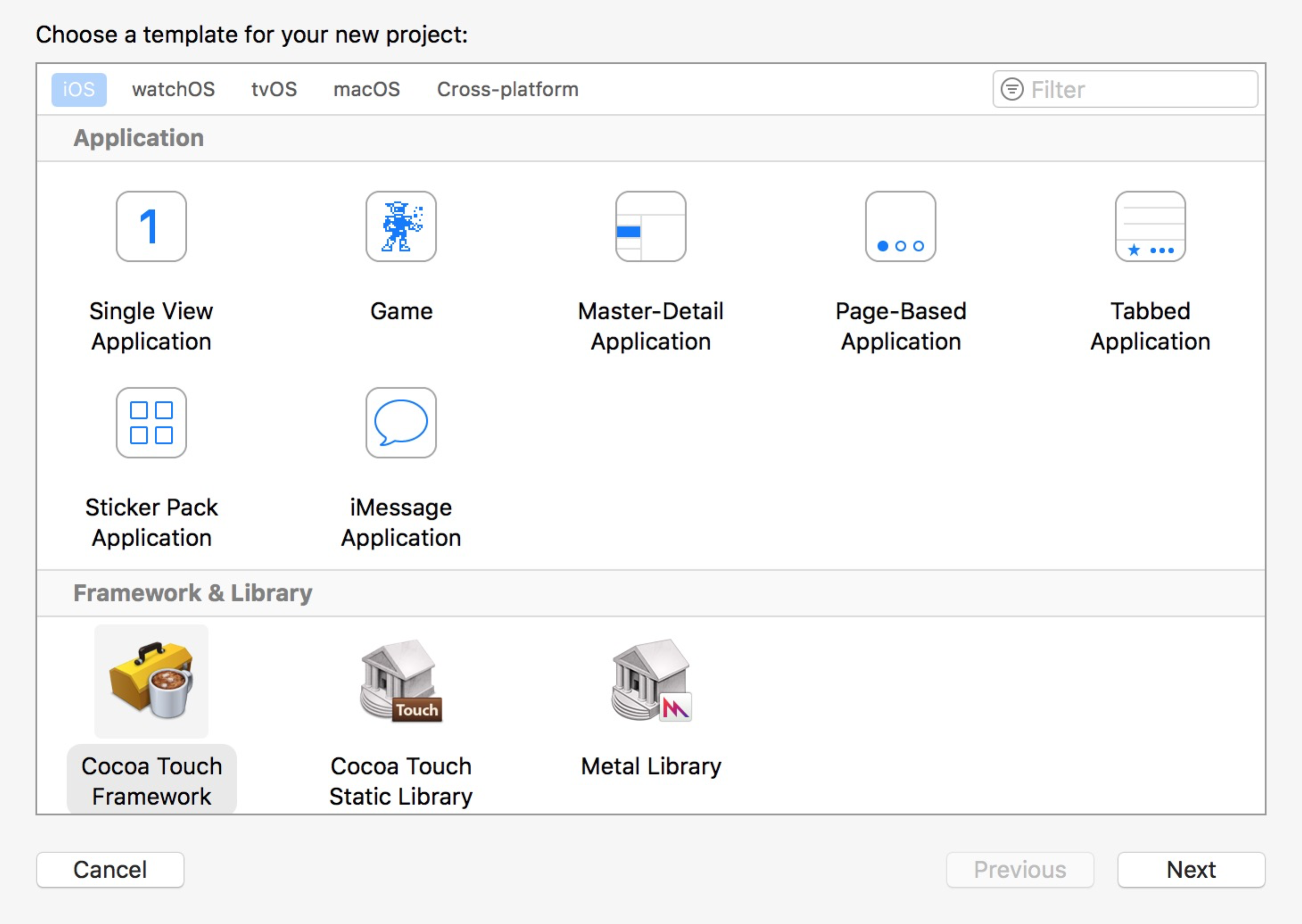
iOS FrameWork制作流程

## 1 简介

Mac OS X扩展了framework的功能，让我们能够利用它来共享代码和资源。通过framework我们可以共享所有形式的资源，如动态共享库，nib文件，图像字符资源以及文档等。  
 系统会在需要的时候将framework载入内存中，多个应用程序可以同时使用同一个framework。这种方法可以使得你的代码易分享，在多个工程中复用，并且可以隐藏实现细节，控制公开的头文件。

## 2 FrameWork制作步骤

2.1打开Xcode，创建新工程。

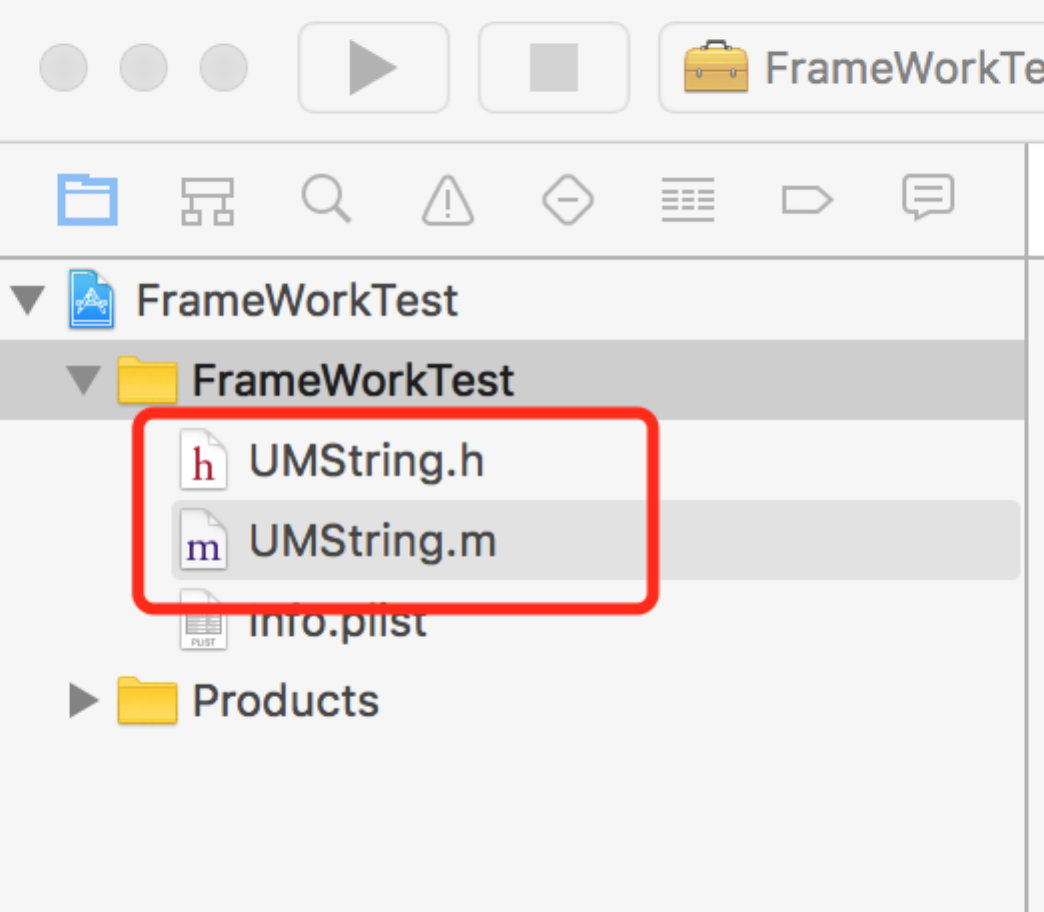


点击Next创建Cocoa Touch FrameWork

2.2创建功能类。

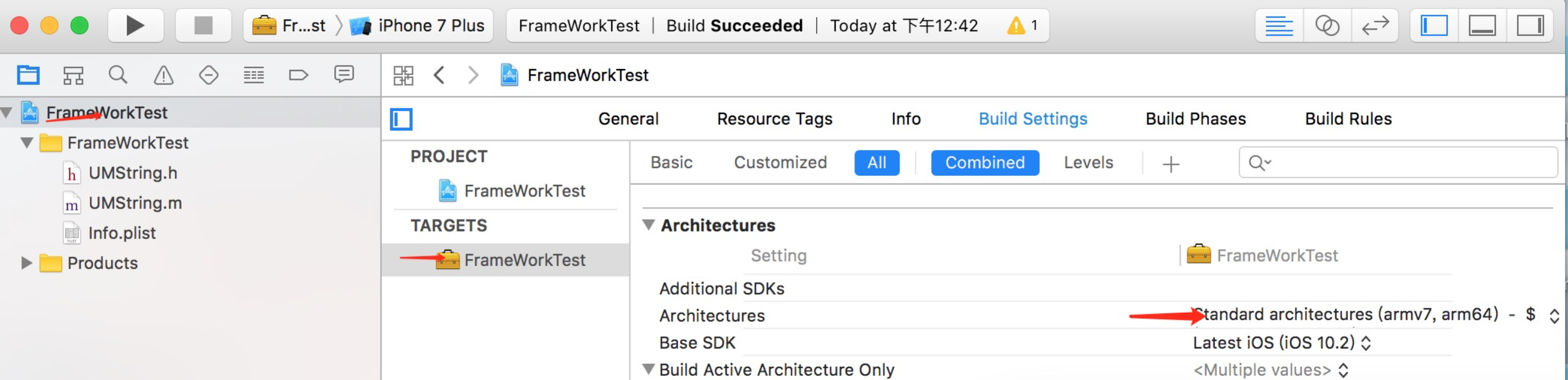
这里我把FrameWorkTest.h去掉，导入我创建的UMString.h和UMString.m类，将它们加入项目中。

如图所示：

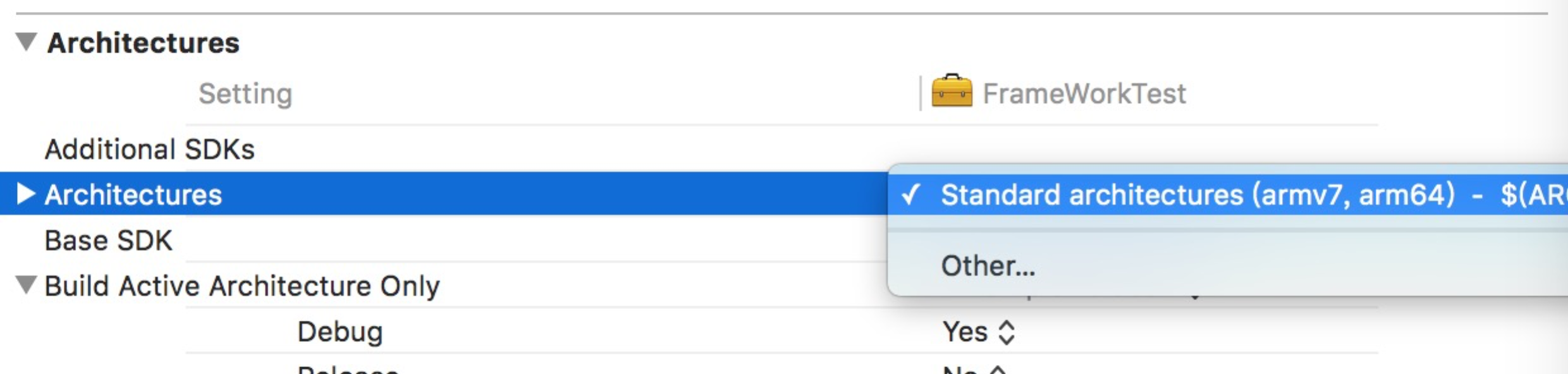


## 3 进行整个工程的配置

3.1 TARGETS - >Build Settings -> Architectures:添加 ***armv7s***；

****

上述3步骤完成后会出现other选项，点击后如下图操作：



注：此处为什么添加armv7s？

首先了解一下什么是ARM，ARM是微处理器，而armv6, armv7, armv7s是ARM CPU的不同指令集，

armv6设备：iPhone， iPhone2， iPhone 3G，第一代、第二代iPod Touch

armv7设备：iPhone 3GS，iPhone 4，iPhone 4S，iPad ，iPad 2，iPod Touch 3G，iPod Touch 4

armv7s设备：iPhone 5，iPad4

arm64设备：iPhone 5S，iPad Air，iphone6，iphone6plus，iPhone6s，iPhone6s Plus等

    Xcode6更新后，默认不支持armv7s，也就是默认不会编译armv7s架构，虽然其处理器架构会向下兼容，但是无法进行相关优化操作。如果你提供的Framework默认不支持armv7s架构，但是使用者要支持armv7s架构，这时就会有冲突，Xcode无法构建代码原因是丢失某架构（armv7s架构）链接库的错误，所以为了避免冲突，Framework默认支持armv7s架构，使用过程中由使用者自行决定是否支持。

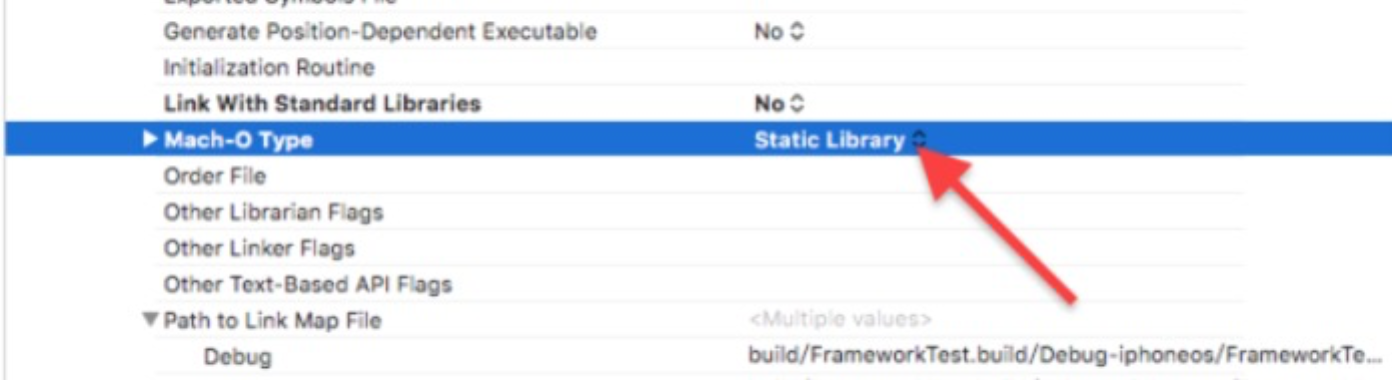
3.2 "Build Active Architecture Only" 设置为 "NO"



注：Build active Architecture Only为什么设置为NO？

      在目标设备上，执行设备对应的指令集。Build active Architecture Only 设置为YES，只会选择编译、链接对应的指令集，设置为NO时，会涵盖所有指令集，在必要的时候选择执行对应的指令集。所以一般在Debug时会选择设置为YES（效率会高点，虽然也没什么卵用），Release时会选择设置为NO，以支持所有可能的架构。

3.3 "Mach-O Type" 设置为 "Static Library"



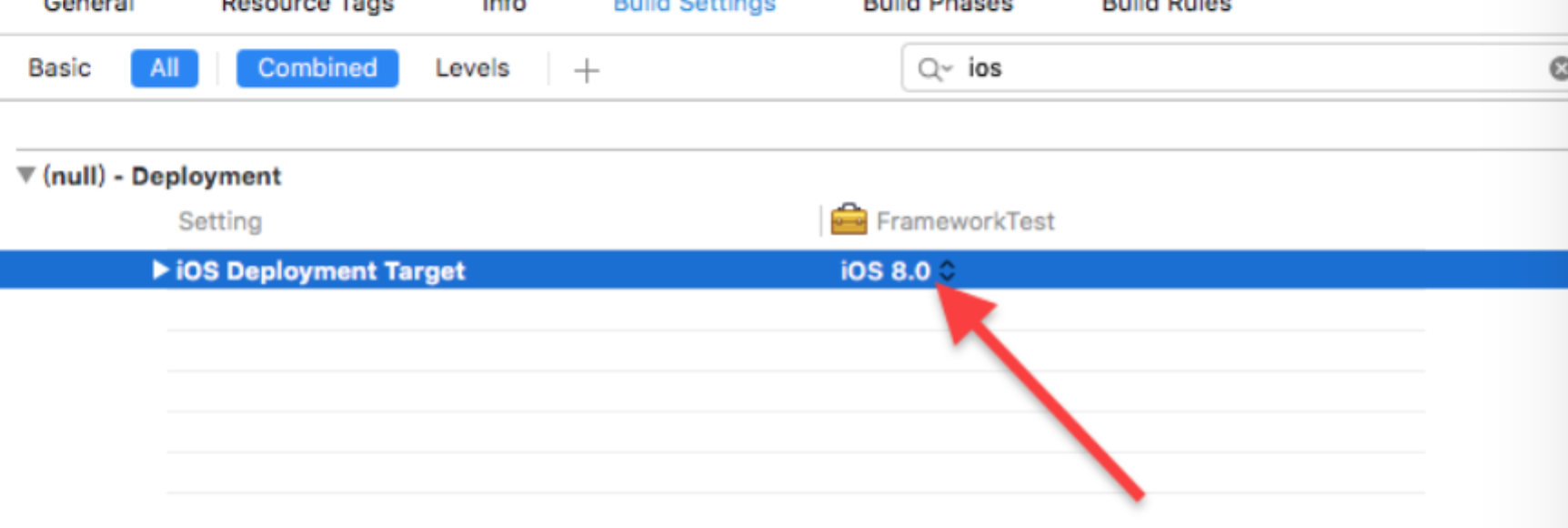
注：为什么Mach-O Type设置为 Static Library？

      请参阅下面的链接文件，写的很详细。

      参考文章：[iOS专题2:静态库和动态库详解 - 简书](http://www.jianshu.com/p/c8366e4f9378" \t "_blank)

3.4 "iOS Deployment Target" 设置为 "iOS 8.0"

这是支持最低运行iOS系统版本（可自行选择，但是要注意的是，iOS SDK中，已经废弃的方法，这个需要配合 Base SDK 设置）。

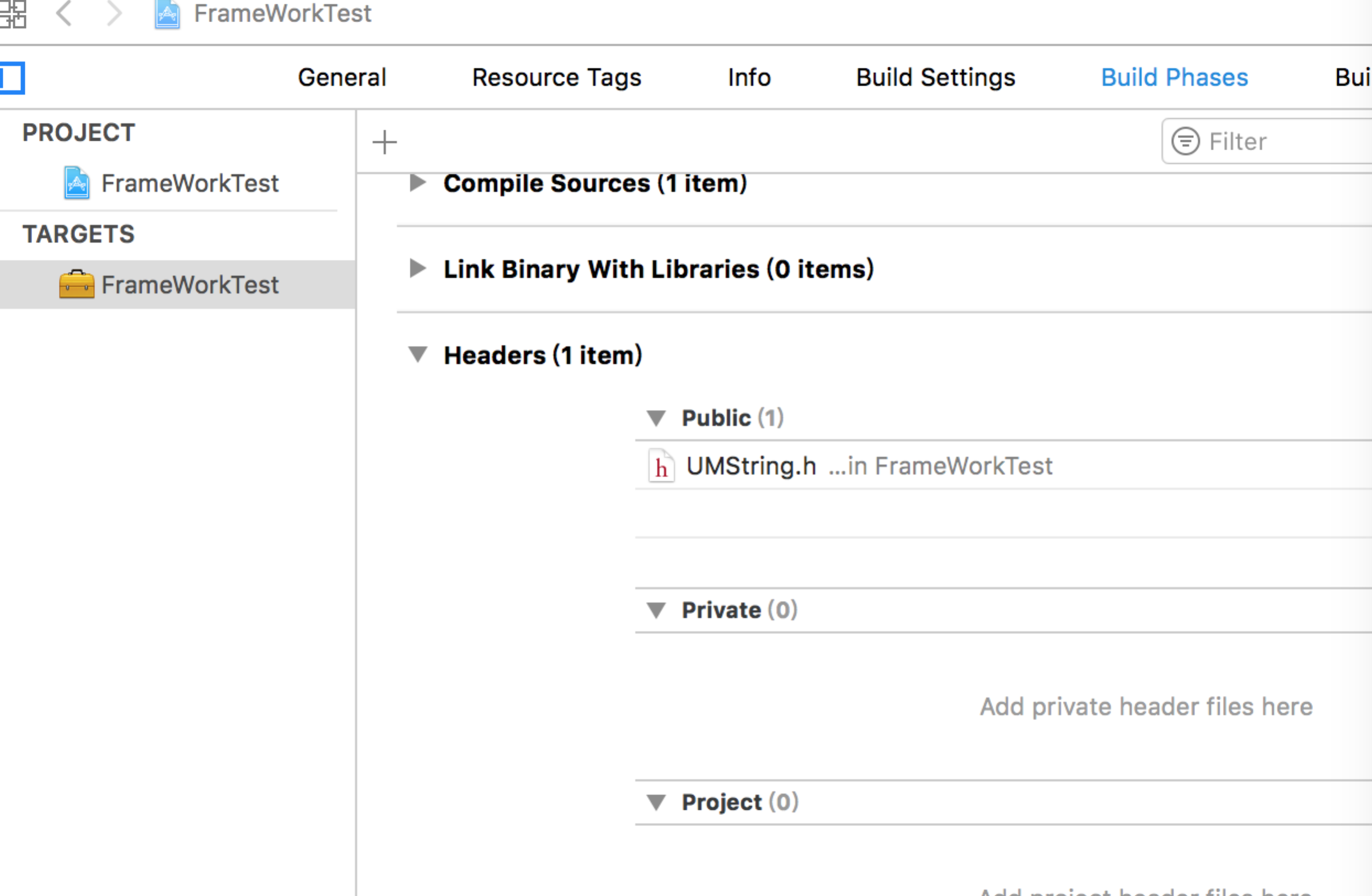


在Xcode7及以上版本中 Dead Code Stripping、Link with Standard Libraries不再需要设置成NO！

## 4 头文件处理

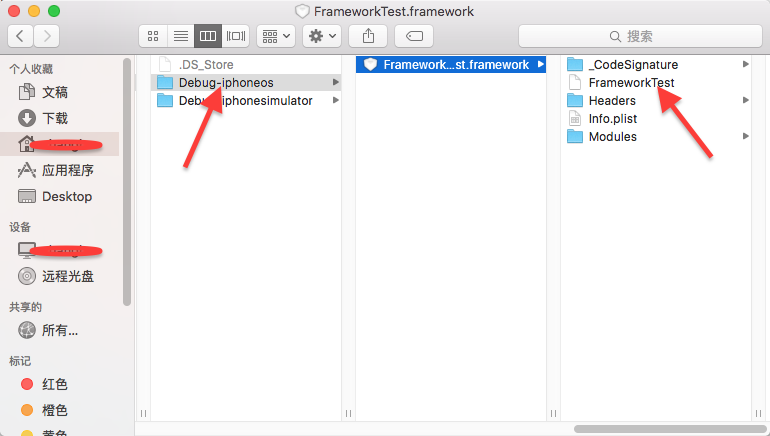
下面我们将头文件暴露在外，实现文件隐藏起来，如下：

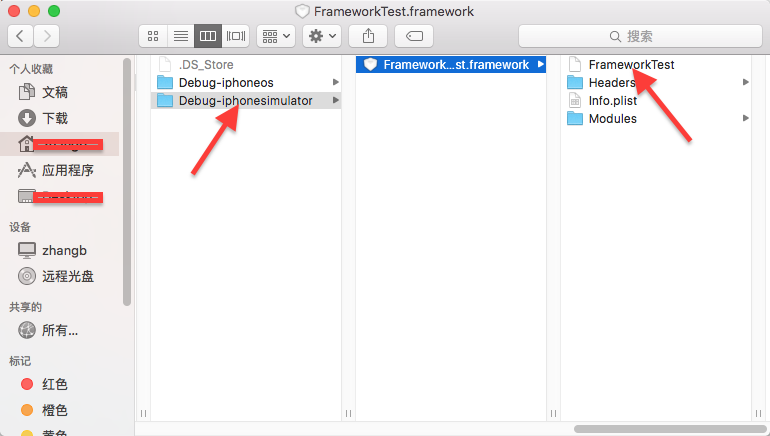
TARGETS - >Build Settings -> Build Phases中，将Headers中的Project内的头文件拖到Public中。如下：



## 5 制作FrameWork文件

5.1 来到工程目录树，Products下的文件都是红色的，现在我们选中.framework文件，分别真机和模拟器运行一遍（成功运行），然后Show in Finder 找到对应的 .framework文件(上级目录可以看出是真机还是模拟器文件夹)，真机和模拟器运行成功的文件是在这俩个文件夹内的。如下图：

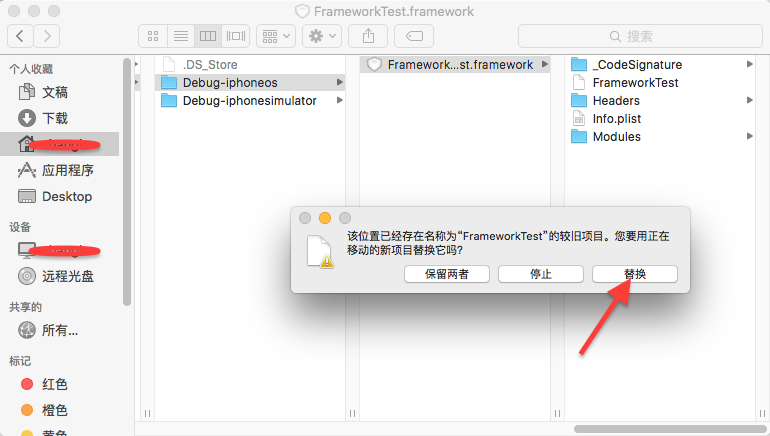




5.2 我们需要将上图内的两个文件（FrameworkTest）合并成一个新的文件，并放在上述.framework文件中（真机or模拟器），使之可以在模拟器和真机上均能运行。这需要用到命令行工具：



打开控制台输入 lipo -create iphoneos下frameworkTest的路径 simulator下frameworkTest的路径 -output 新的路径，这样就完成了模拟器和真机版本的合并，新路径下的frameworkTest就是你合并后的文件，将这个文件名字改成和你未合并之前的Test一样的名字，放到framework文件夹下，替换掉原来的FrameworkTest文件。



注：为什么要替换FrameworkTest文件？

  因为原来的FrameworkTest文件要么只能在真机上运行，要么只能在模拟器上运行。合成文件为的就是让.framework文件既能在真机上运行也能在模拟器上运行，毕竟不能让使用者只能在某个特定的情况下才可以使用，那样就背离我们的初衷啦！