# [如何在原生工程中引入Cordova工程](http://blog.csdn.net/e20914053/article/details/50170487)

<http://blog.csdn.net/e20914053/article/details/50170487>

标签： [ios](http://www.csdn.net/tag/ios" \t "http://blog.csdn.net/e20914053/article/details/_blank)[cordova](http://www.csdn.net/tag/cordova" \t "http://blog.csdn.net/e20914053/article/details/_blank)[混合开发](http://www.csdn.net/tag/%e6%b7%b7%e5%90%88%e5%bc%80%e5%8f%91" \t "http://blog.csdn.net/e20914053/article/details/_blank)

2015-12-04 09:28 1985人阅读 [评论](http://blog.csdn.net/e20914053/article/details/50170487" \l "comments)(2) [收藏](http://blog.csdn.net/e20914053/article/details/javascript:void(0);" \o "收藏" \t "http://blog.csdn.net/e20914053/article/details/_blank) [举报](http://blog.csdn.net/e20914053/article/details/50170487" \l "report" \o "举报)

IMG_256 分类：

混合开发 iOS开发（4） IMG_257 Cordova

版权声明：本文为博主原创文章，未经博主允许不得转载。

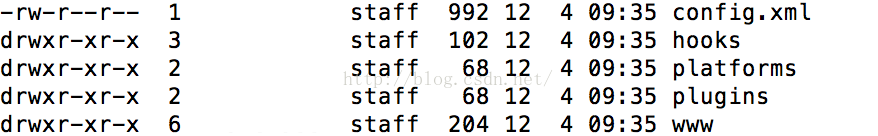
如今混合开发方兴未艾，有的项目可能一开始是原生开发的，后期需要加入混合开发，如将Cordova工程引进到原生工程中。那么该如何操作呢？下面就来描述一下步骤。

1、首先我们来创建一个Cordova工程，取名MyCordova。在创建Cordova工程之前，需要先安装Cordova，具体安装方法网上很多，在此不累述。

**[plain]** [view plain](http://blog.csdn.net/e20914053/article/details/50170487" \o "view plain) [copy](http://blog.csdn.net/e20914053/article/details/50170487" \o "copy)

1. cordova create MyCordova

进入MyCordova工程目录，其结构如下：



2、接下来添加**[iOS](http://lib.csdn.net/base/ios" \o "iOS知识库" \t "http://blog.csdn.net/e20914053/article/details/_blank)**平台，添加命令如下：

**[html]** [view plain](http://blog.csdn.net/e20914053/article/details/50170487" \o "view plain) [copy](http://blog.csdn.net/e20914053/article/details/50170487" \o "copy)

1. cordova platform add ios

该命令需要在MyCordova工程根目录下执行。执行成功后，我们进入MyCordova工程下的platforms目录下，我们发现它增加了一个名为ios的文件目录。

IMG_259

3、回到MyCordova工程根目录，运行刚才添加的iOS工程。

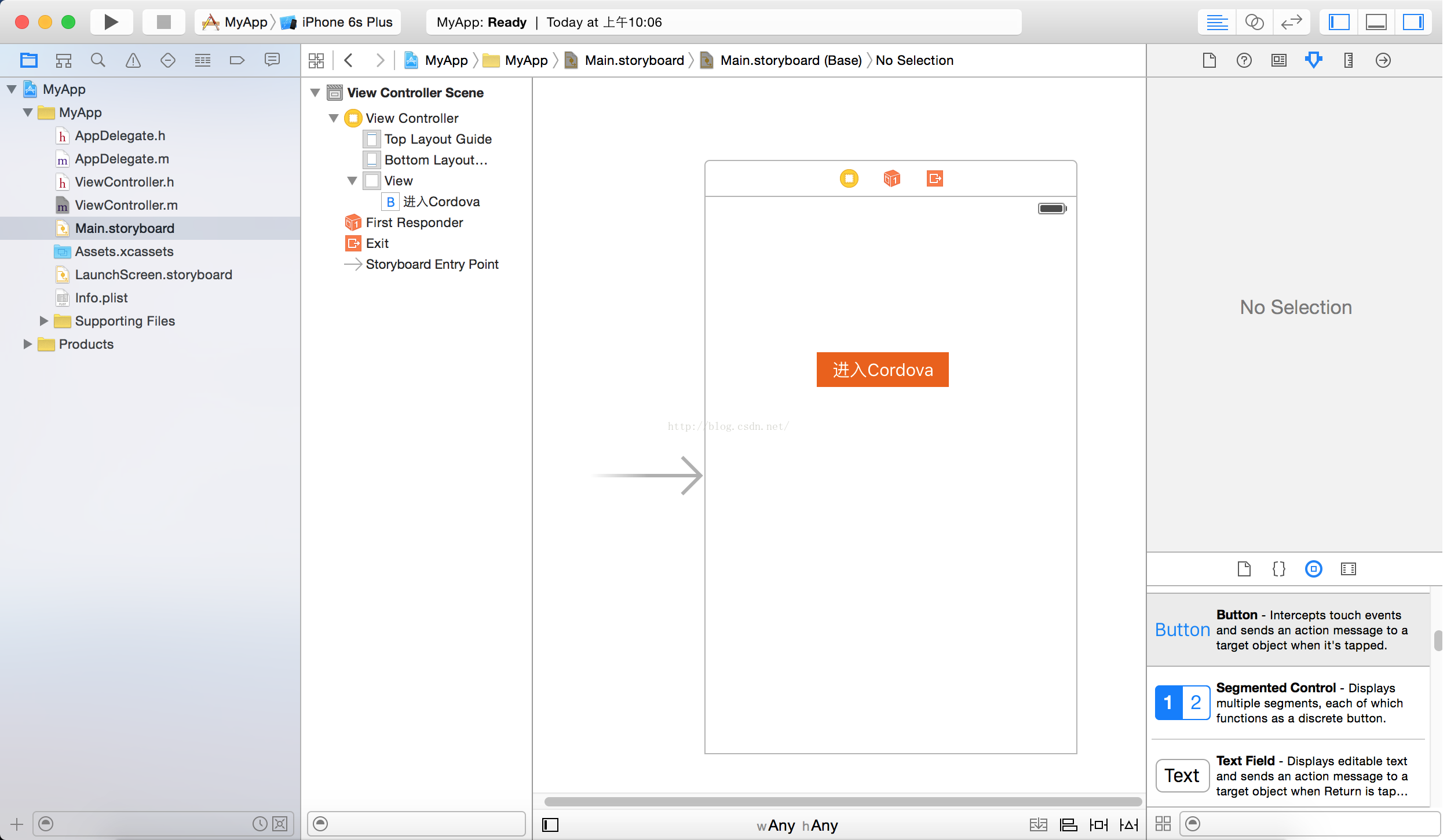
**[plain]** [view plain](http://blog.csdn.net/e20914053/article/details/50170487" \o "view plain) [copy](http://blog.csdn.net/e20914053/article/details/50170487" \o "copy)

1. cordova run ios

运行效果如下：

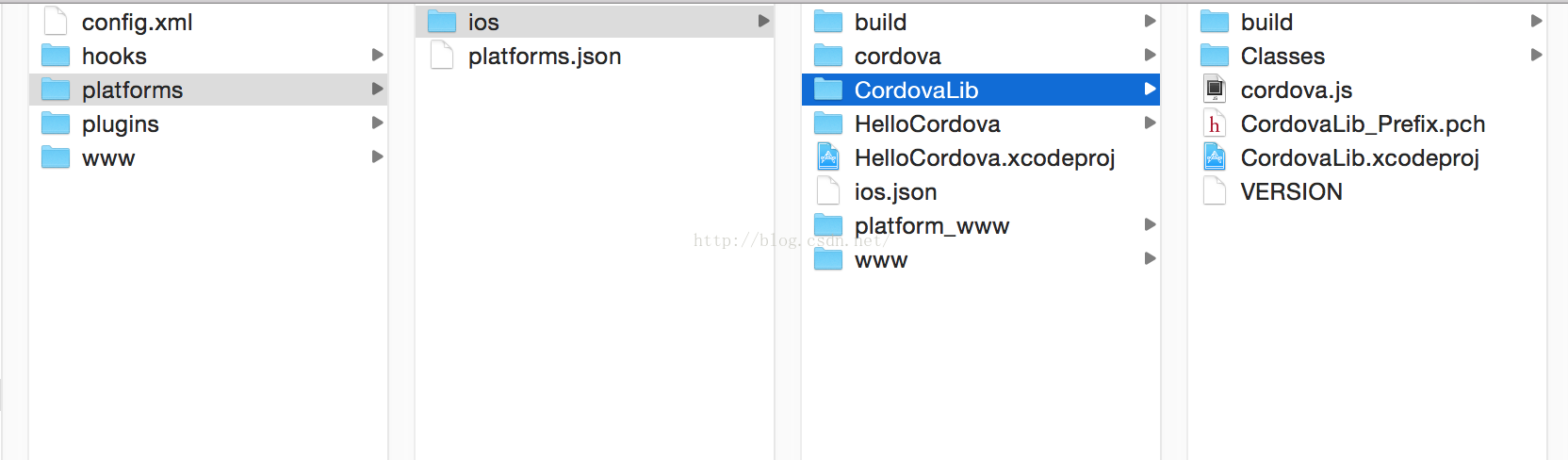


4、通过xcode创建一个原生工程MyApp。如果原生工程已经存在，可以忽略此步骤。

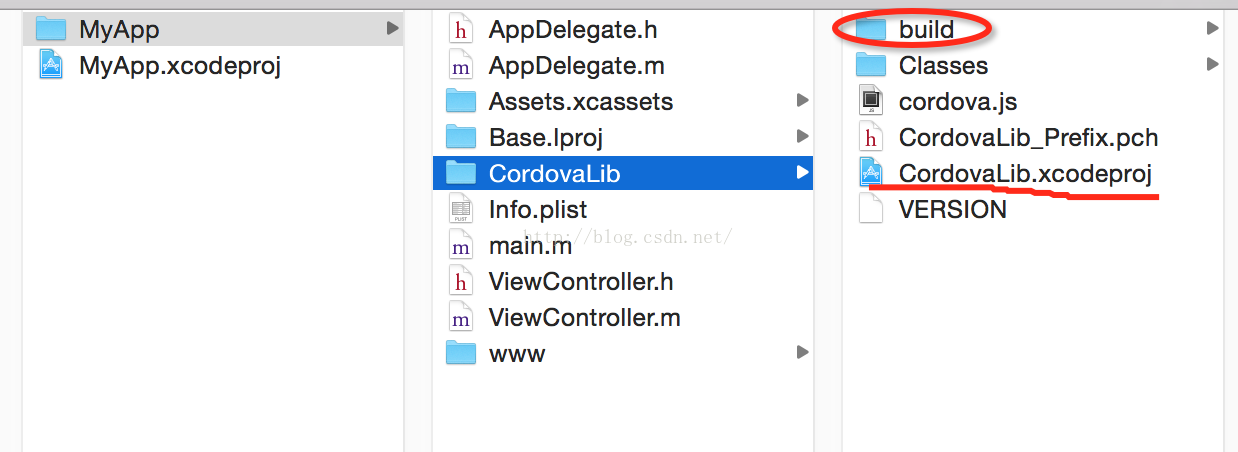


5、将MyCordova工程中iOS下的CordovaLib文件夹和www文件夹拷贝到MyApp工程目录下。

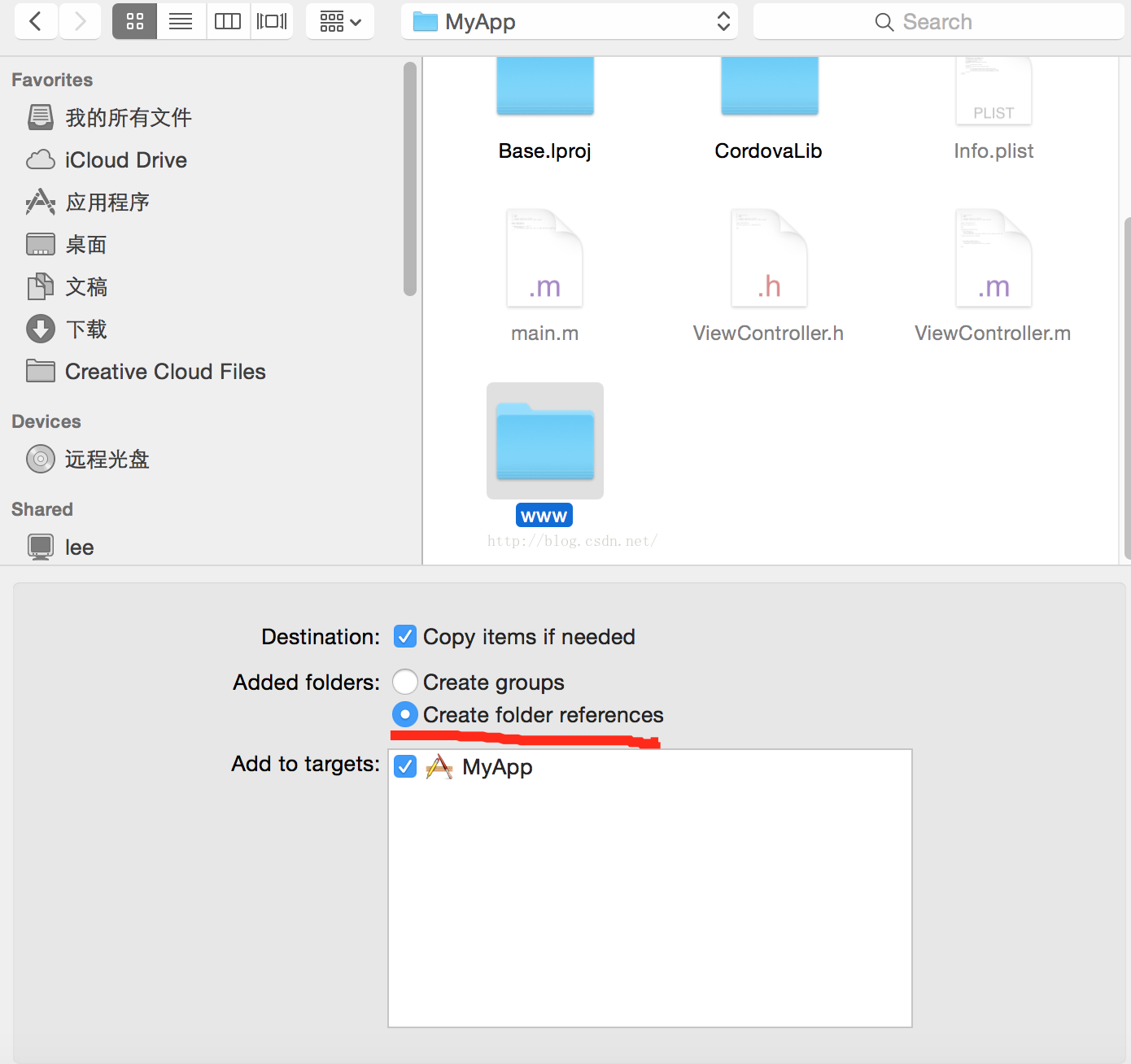
MyCordova目录：



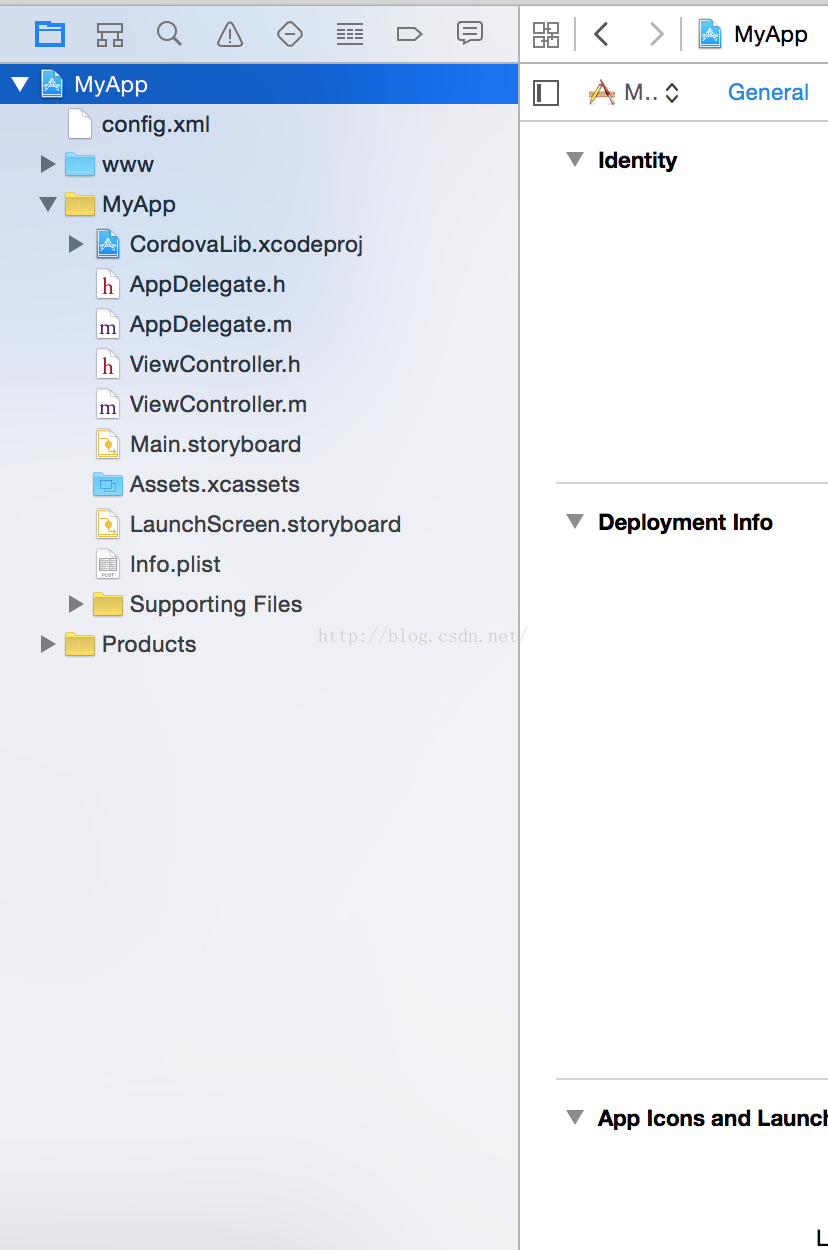
MyApp目录：



6、参看上图。删除CordovaLib下面的build文件夹，此文件夹是在执行cordova run ios命令过程中产生的，如果你没有执行过该命令就不会产生这个文件夹。然后通过xcode的Add files to “MyApp” ...将CordovaLib.xcodeproj文件和www文件夹添加到MyApp工程中。注意，在添加www文件夹时要勾选Create folder references。如下：

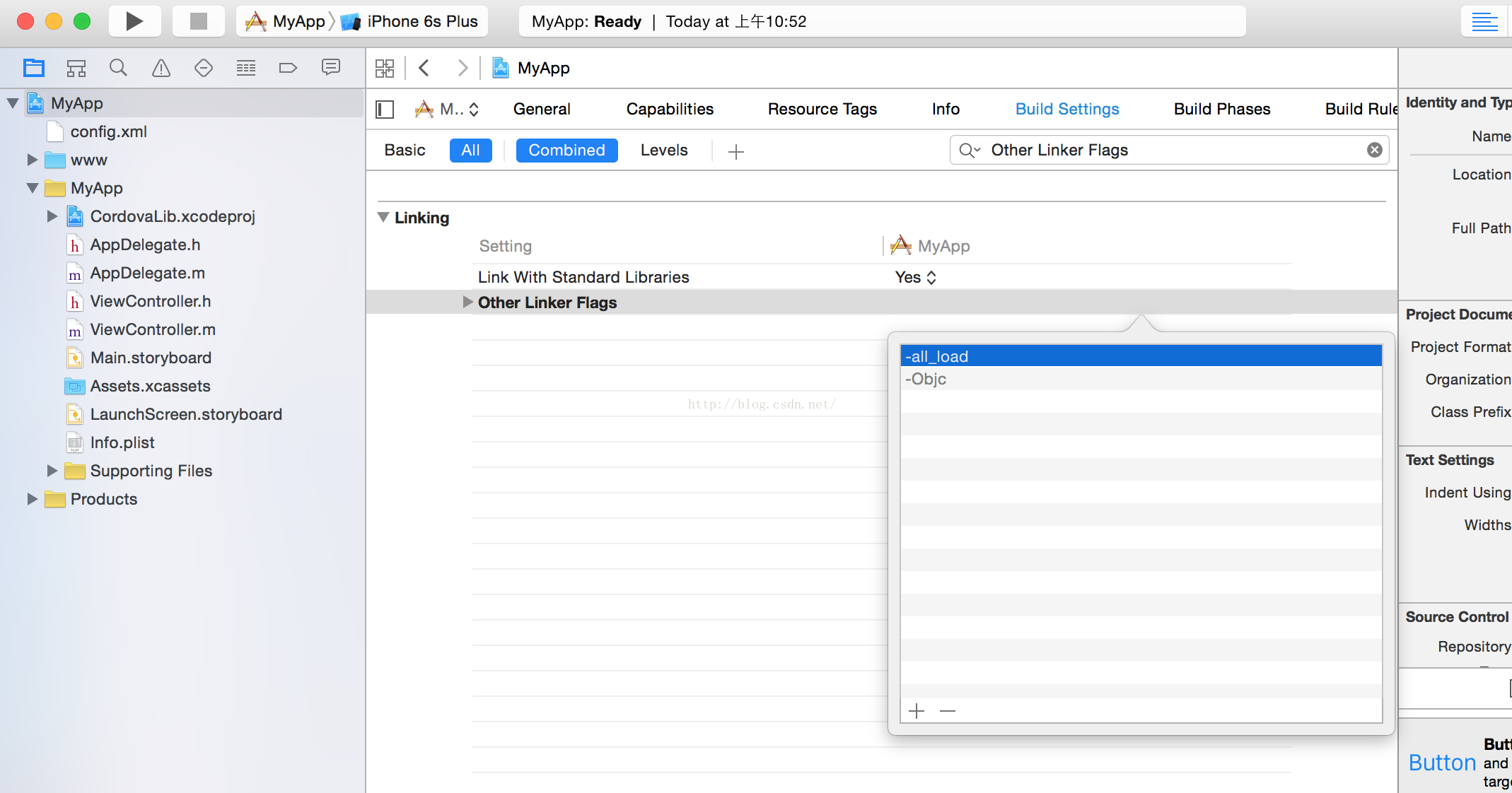


7、将MyCordova工程根目录下的config.xml也添加到MyApp工程中。至此，所需的文件拷贝添加工作已经完成，其文件结构如下：

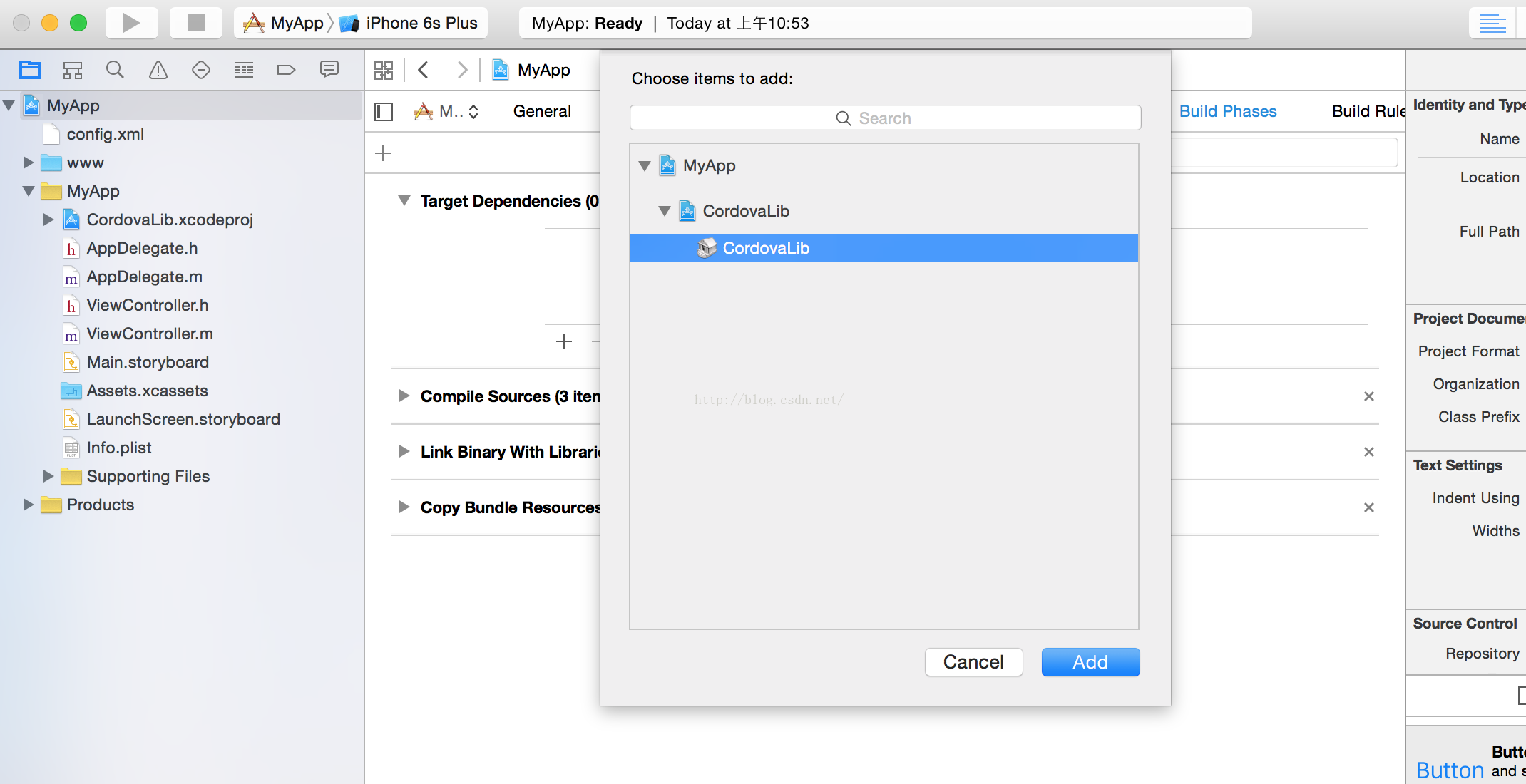


下面开始对对MyApp工程进行配置工作。

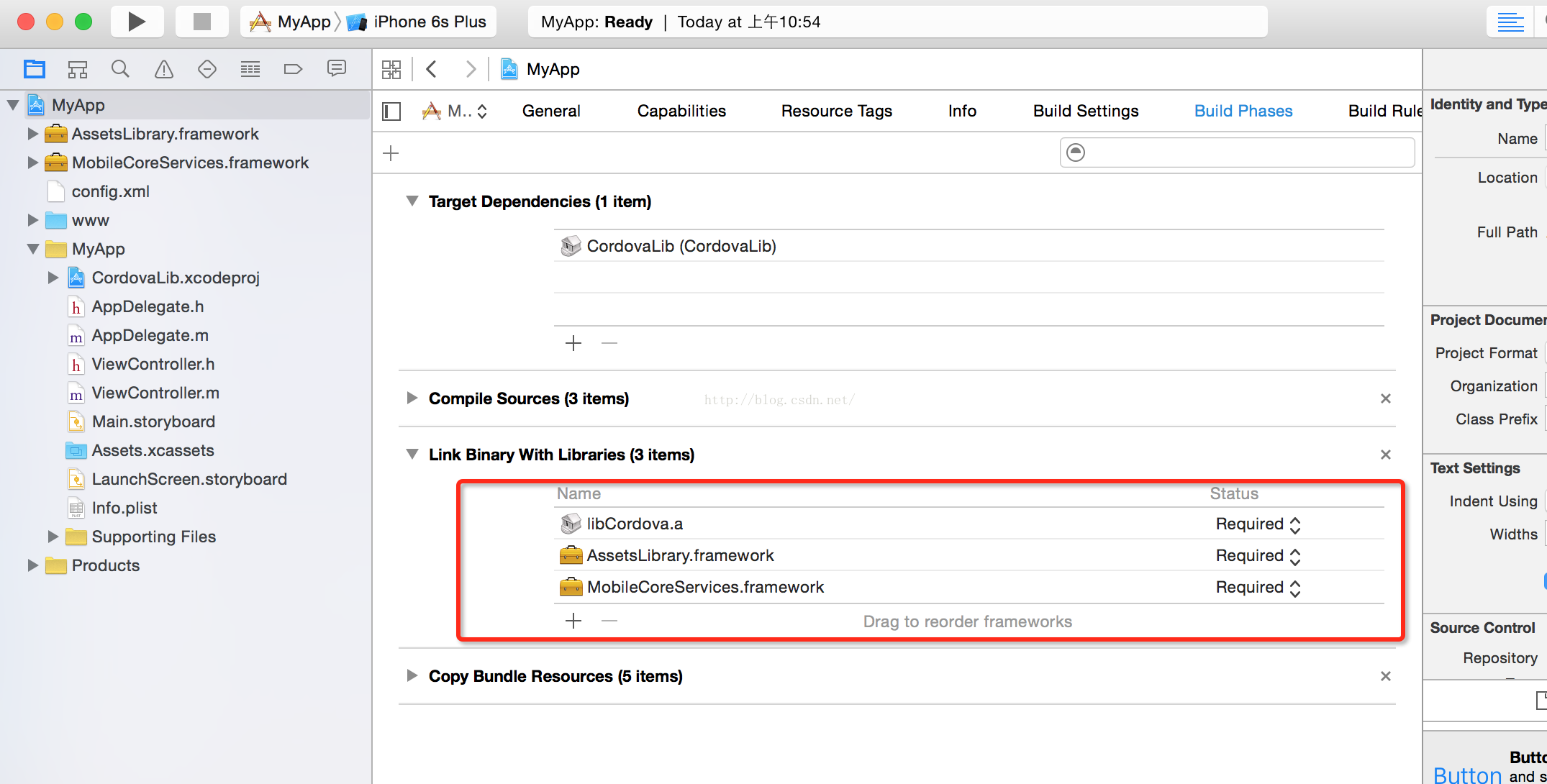
8、选择MyApp工程的Build Settings->Other Linker Flags, 设置-Objc -all\_load



9、选择MyApp工程的Build Phases->Target Dependencies添加CordovaLib

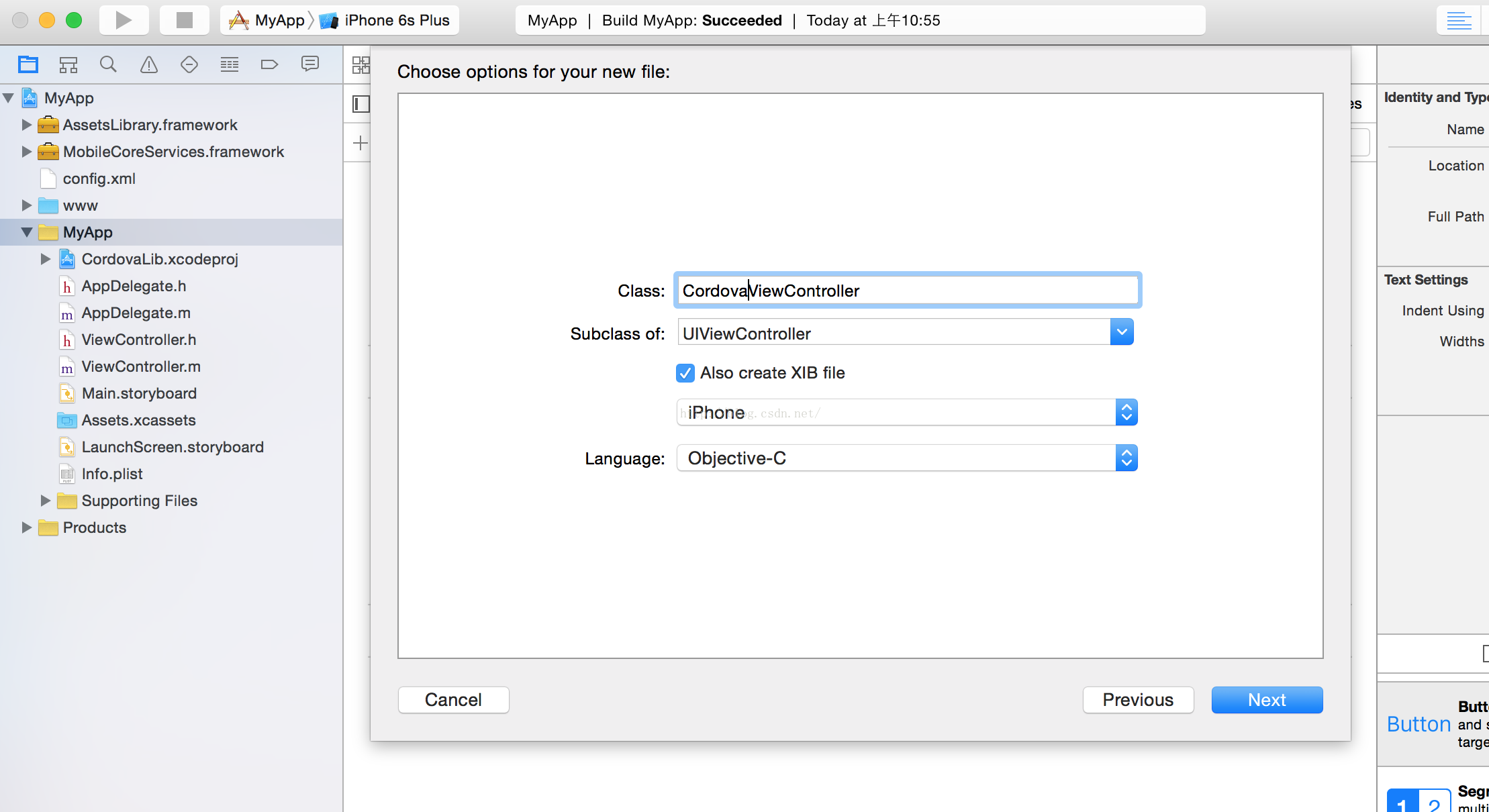


10、选择MyApp工程的Build Phases->Link Binary With Librarys添加libCordova.a、 MobileCoreServices.framework、AssetsLibrary.framework相关框架。



到此MyApp工程已经顺利导入MyCordova工程了，点击Product->Build编译通过。下面再来创建并弹出Cordova页面。

11、创建一个视图控制器CordovaViewController。



其中CordovaViewController.h文件内容如下：

**[objc]** [view plain](http://blog.csdn.net/e20914053/article/details/50170487" \o "view plain) [copy](http://blog.csdn.net/e20914053/article/details/50170487" \o "copy)

1. #import <Cordova/CDVViewController.h>
2. #import <Cordova/CDVCommandDelegateImpl.h>
3. #import <Cordova/CDVCommandQueue.h>
5. **@interface** CordovaViewController : CDVViewController
7. **@end**
9. **@interface** CordovaCommandDelegate : CDVCommandDelegateImpl
10. **@end**
12. **@interface** CordovaCommandQueue : CDVCommandQueue
13. **@end**

CordovaViewController.m文件内容如下：

**[objc]** [view plain](http://blog.csdn.net/e20914053/article/details/50170487" \o "view plain) [copy](http://blog.csdn.net/e20914053/article/details/50170487" \o "copy)

1. #import "CordovaViewController.h"
3. **@implementation** CordovaViewController
5. - (**id**)initWithNibName:(NSString\*)nibNameOrNil bundle:(NSBundle\*)nibBundleOrNil
6. {
7. **self** = [**super** initWithNibName:nibNameOrNil bundle:nibBundleOrNil];
8. **if** (**self**) {
9. // Uncomment to override the CDVCommandDelegateImpl used
10. // \_commandDelegate = [[MainCommandDelegate alloc] initWithViewController:self];
11. // Uncomment to override the CDVCommandQueue used
12. // \_commandQueue = [[MainCommandQueue alloc] initWithViewController:self];
13. }
14. **return** **self**;
15. }
17. - (**id**)init
18. {
19. **self** = [**super** init];
20. **if** (**self**) {
21. // Uncomment to override the CDVCommandDelegateImpl used
22. // \_commandDelegate = [[MainCommandDelegate alloc] initWithViewController:self];
23. // Uncomment to override the CDVCommandQueue used
24. // \_commandQueue = [[MainCommandQueue alloc] initWithViewController:self];
25. }
26. **return** **self**;
27. }
29. - (**void**)didReceiveMemoryWarning
30. {
31. [**super** didReceiveMemoryWarning];
32. }
34. #pragma mark View lifecycle
36. - (**void**)viewWillAppear:(**BOOL**)animated
37. {
39. [**super** viewWillAppear:animated];
40. }
42. - (**void**)viewDidLoad
43. {
44. [**super** viewDidLoad];
45. }
47. - (**void**)viewDidUnload
48. {
49. [**super** viewDidUnload];
50. }
52. #pragma mark UIWebDelegate implementation
54. - (**void**)webViewDidFinishLoad:(UIWebView\*)theWebView
55. {
56. theWebView.backgroundColor = [UIColor blackColor];
58. **return** [**super** webViewDidFinishLoad:theWebView];
59. }

62. **@end**
64. **@implementation** CordovaCommandDelegate

67. #pragma mark CDVCommandDelegate implementation
69. - (**id**)getCommandInstance:(NSString\*)className
70. {
71. **return** [**super** getCommandInstance:className];
72. }
74. - (NSString\*)pathForResource:(NSString\*)resourcepath
75. {
76. **return** [**super** pathForResource:resourcepath];
77. }
79. **@end**
81. **@implementation** CordovaCommandQueue
83. - (**BOOL**)execute:(CDVInvokedUrlCommand\*)command
84. {
85. **return** [**super** execute:command];
86. }
88. **@end**

12、为MyApp工程中的“进入Cordova”UIButton绑定事件方法，来弹出CordovaViewController视图控制器。运行效果如下：（左边为原生视图控制器，右边为弹出的CordovaViewController视图控制器）

是不是跟在MyCordova工程中通过cordova run ios命令运行出来的效果一样呢！

ok，至此原生工程导入Cordova工程的方法步骤全部结束。