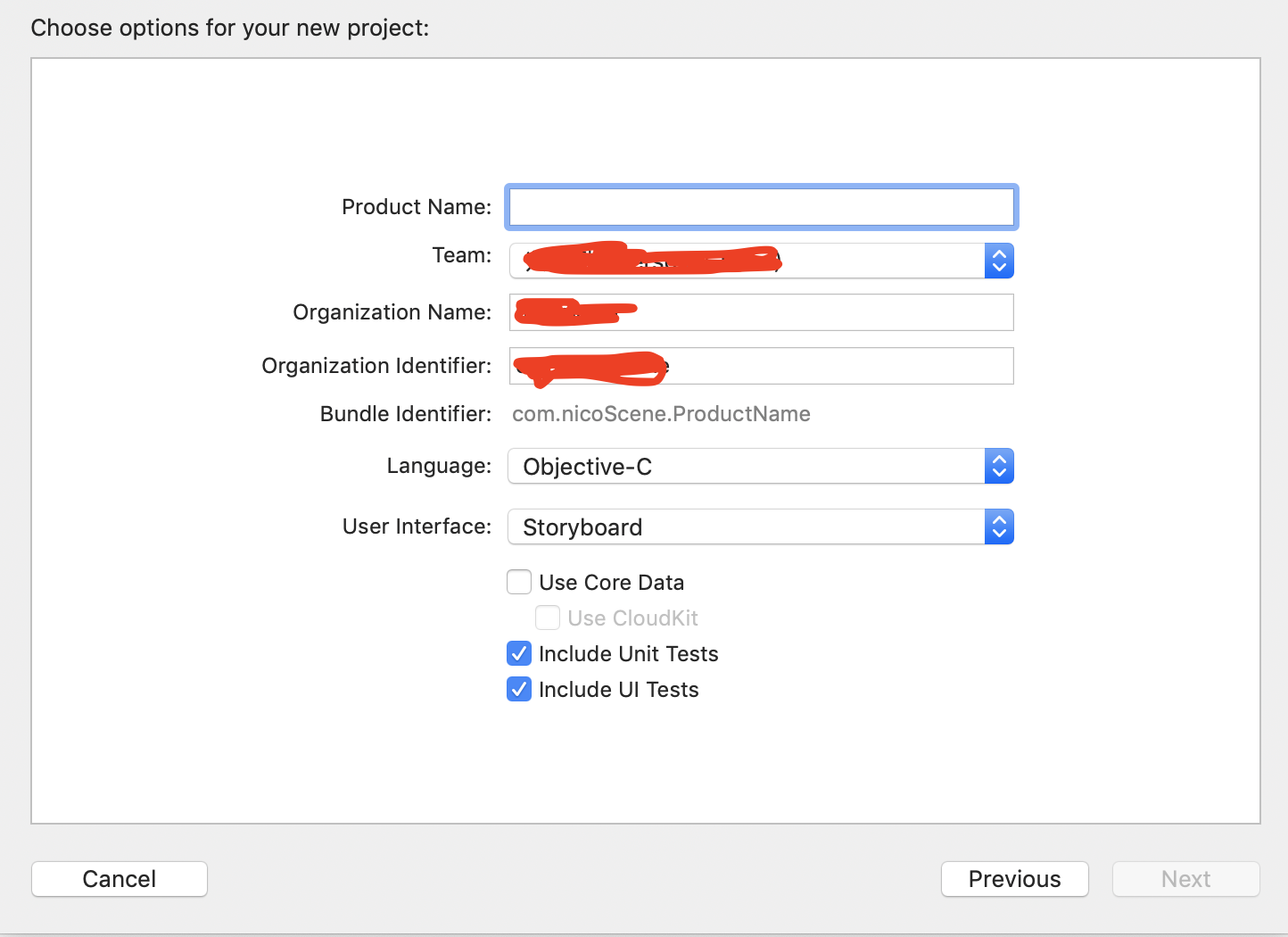
**ios 集成 unity文档记录，主要分以下三部分记录：**

1. **ios空项目创建；**
2. **打开xcode，并选择Create a new Xcode project，下个界面直接next即可。**



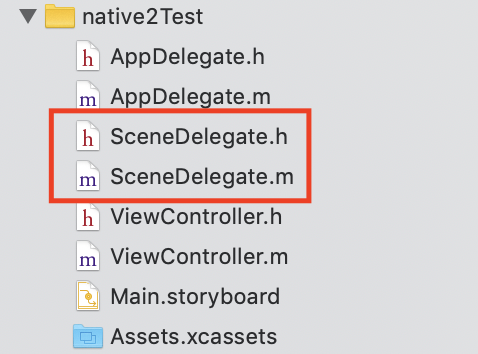
1. **目录如下填写即可，然后点击Next选择目录存储项目，完成项目创建**



1. **直接点击可以运行的。**

注：

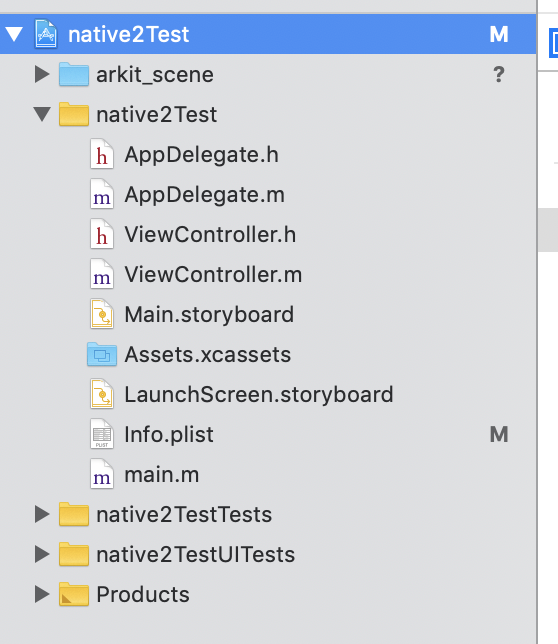
a. Xcode11 创建的项目默认是基于ios13的，会多了以下两个文件



如果不需要操作的话可以参考网上如何去掉，简单点说就是删除Info.plist中的如下选项，然后删除那两个文件：

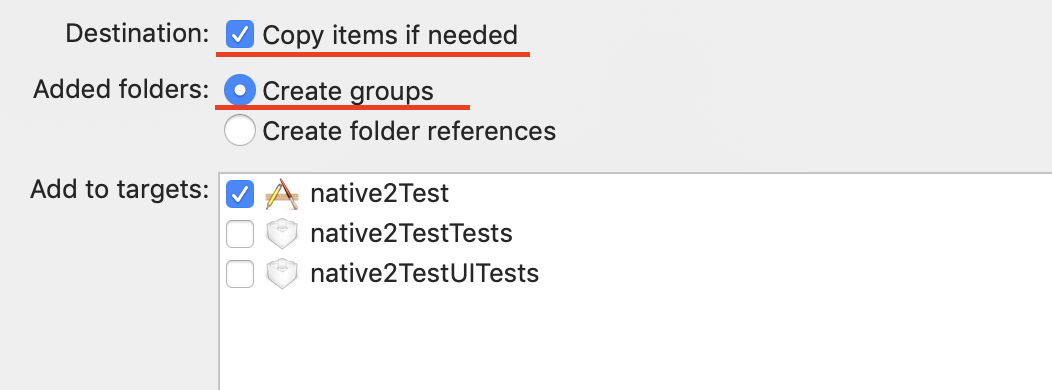


1. **unity空项目创建：**
2. **打开unity hub，添加，然后创建一个空项目即可；**
3. **点击File --> Build Settings， 然后选择iOS, 点击switch platform** (第一次的话会比较慢)；
4. **点击Build**，这里可以直接将Build的输出目录放到上面的空项目中，即native2Test，这里命名为arkit\_scene (这里整的是arkit项目)，成功后目录结构如下；

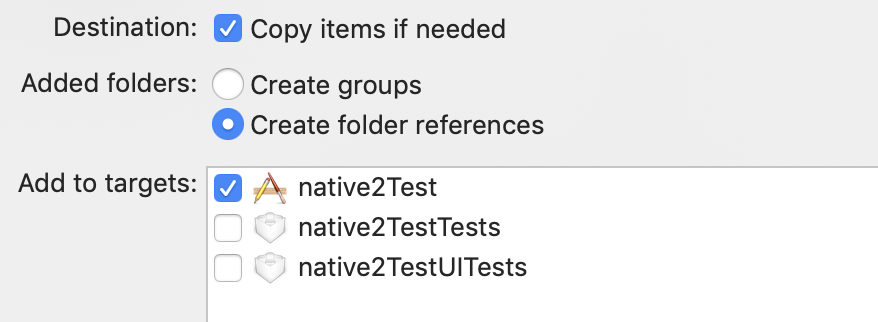


以上完成准备工作。

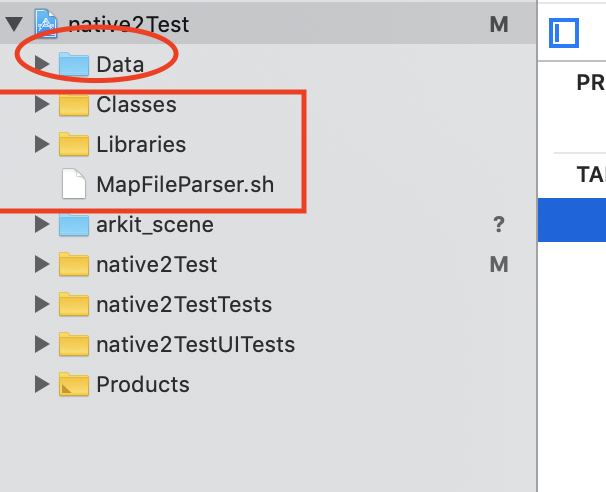
1. **ios集成unity流程**
2. **将unity项目中的Classes、Libraries、MapFileParser.sh拷贝出来**，如下方式选择 (右击项目、选择Add Files to “native2Test”... )



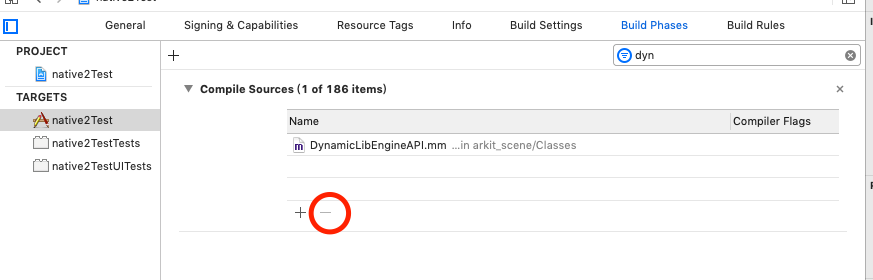
1. **拷贝Data文件夹**



以上两步结束后目录结构如下

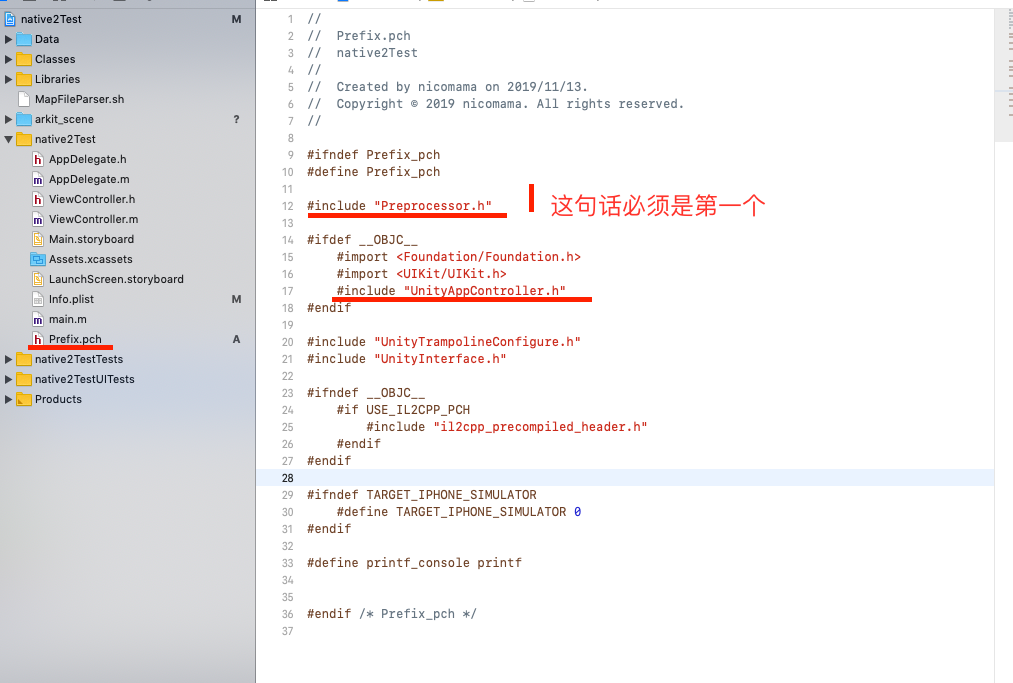


1. **删除Library/libil2cpp**，这里删除的时候是选择Remove References；
2. **选择 Targets --> Build Phases**，过滤出 DynamicLibEngineApp，然后删除掉



注：这里不删除的话链接过程会报arm错误。

1. **操作pch文件**，如果原工程中没有pch文件的话，则新创建一个，然后将Classes/Prefix.pch中的内容拷贝到新的里面。

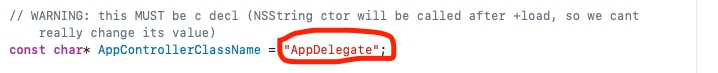


1. **处理main.mm**

将Classes/main.mm 拷贝到 native2Test.main.m中，注意保留原main.m中到前两个头文件包含，别删除了。

然后将main.m改名为main.mm，删除掉Classes/main.mm，依旧是remove reference。

然后修改如下语句，将引号中的内容替换为AppDelegate



1. **修改AppDelegate.m**

//

// AppDelegate.m

// nativeTest

//

// Created by nicomama on 2019/11/13.

// Copyright © 2019 nicomama. All rights reserved.

//

#import "AppDelegate.h"

#import "ViewController.h"

#import "UnityAppController.h"

@interface AppDelegate ()

//@property (strong, nonatomic) UIWindow \* window;

@end

@implementation AppDelegate

-(UIWindow \*)unityWindow{

return UnityGetMainWindow();

}

-(void)showUnityWindow{

UIButton \*button = [UIButton buttonWithType:UIButtonTypeSystem];

button.frame = CGRectMake(0, 0, 44, 44);

button.backgroundColor = [UIColor redColor];

[button setTitle:@"返回" forState:UIControlStateNormal];

[button addTarget:self action:@selector(hideUnityWindow) forControlEvents:UIControlEventTouchUpInside];

[self.unityWindow addSubview:button];

// [[UIApplication sharedApplication] setStatusBarHidden:YES];

[self.unityWindow makeKeyAndVisible];

}

-(void)hideUnityWindow{

// [[UIApplication sharedApplication] setStatusBarHidden:NO];

// [[UIApplication sharedApplication] setStatusBarStyle:UIStatusBarStyleLightContent animated:YES];

[self.window makeKeyAndVisible];

}

- (BOOL)application:(UIApplication \*)application didFinishLaunchingWithOptions:(NSDictionary \*)launchOptions {

// Override point for customization after application launch.

self.window = [[UIWindow alloc]initWithFrame:[UIScreen mainScreen].bounds];

self.window.backgroundColor = [UIColor whiteColor];

ViewController \* vc = [[ViewController alloc]init];

self.window.rootViewController = vc;

self.unityController = [[UnityAppController alloc]init];

[self.unityController application:application didFinishLaunchingWithOptions:launchOptions];

[self.window makeKeyAndVisible];

return YES;

}

- (void)applicationWillResignActive:(UIApplication \*)application {

[self.unityController applicationWillResignActive:application];

}

- (void)applicationDidEnterBackground:(UIApplication \*)application {

[self.unityController applicationDidEnterBackground:application];

}

- (void)applicationWillEnterForeground:(UIApplication \*)application {

[self.unityController applicationWillEnterForeground:application];

}

- (void)applicationDidBecomeActive:(UIApplication \*)application {

[self.unityController applicationDidBecomeActive:application];

}

- (void)applicationWillTerminate:(UIApplication \*)application {

[self.unityController applicationWillTerminate:application];

}

@end

1. **修改AppDelegate.h**

//

// AppDelegate.h

// nativeTest

//

// Created by nicomama on 2019/11/13.

// Copyright © 2019 nicomama. All rights reserved.

//

#import <UIKit/UIKit.h>

@class UnityAppController;

@interface AppDelegate : UIResponder <UIApplicationDelegate>

@property (nonatomic, strong) UnityAppController \* unityController;

@property (strong, nonatomic) UIWindow \*window;

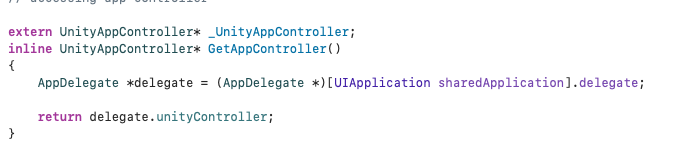
@property (strong, nonatomic) UIWindow \*unityWindow;

-(void)showUnityWindow;

-(void)hideUnityWindow;

@end

1. **修改Classes/UnityAppController.h**



同时包含头文件 #import "AppDelegate.h"

1. **添加Framework，选择Targets-->General --> Frameworks...**，对照unity项目，将所有用到的东西都添加进来：

其他包都根据查找直接添加进来即可。

注意，libiconv.2.dylib添加方式如下：

点击 + ，点击 add files ...，按住command + shift + G， 然后在弹出都框中输入/usr/lib，找到libiconv.2.dylib，添加进去即可。

1. **进行其他设置：**

· Enabled BitCode 设置为NO；

· Header search path 和 library search path参照 unity项目，全部拖好；

· other linker flags 和 unity 项目保持一致；

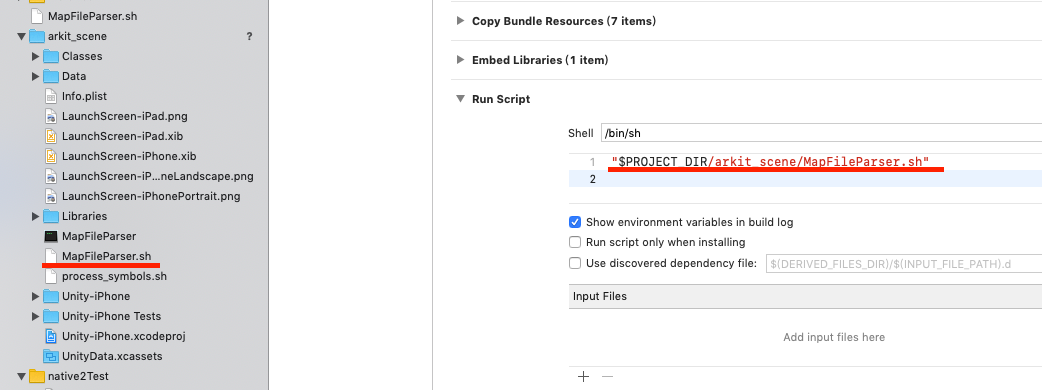
· other c flag / other c++ flag 和 unity 项目保持一致；

· 添加User-Defined (target --> Building settings， 点击加号)

· prefix header， 将之前创建的prefix.pch拖进来。

1. **设置Run scripts** (Targets --> Build Phases)，点击 + 添加run scripts

设置，注意路径



1. **在Info.plist中添加一个key value**，通过vscode打开添加即可

<key>NSCameraUsageDescription</key>

<string>cameraDesciption</string>

1. **在ViewController.m中创建一个按钮**，用于ios打开unity游戏 (这里用到了一张图片，自己注意随便加一张)。

//

// ViewController.m

// nativeTest

//

// Created by nicomama on 2019/11/13.

// Copyright © 2019 nicomama. All rights reserved.

//

#import "ViewController.h"

#import "AppDelegate.h"

@interface ViewController ()

@end

@implementation ViewController

- (void)viewDidLoad {

[super viewDidLoad];

// Do any additional setup after loading the view.

//1.创建按钮（UIButton）

UIButton \*button = [[UIButton alloc]init];

//2.设置按钮上显示的文字

[button setTitle:@"点我吧" forState:UIControlStateNormal];

[button setTitle:@"摸我干啥" forState:UIControlStateHighlighted];

//设置文字颜色

[button setTitleColor:[UIColor redColor] forState:UIControlStateHighlighted];

[button setTitleColor:[UIColor blueColor] forState:UIControlStateHighlighted];

//3.加载图片

UIImage \*imgNormal = [UIImage imageNamed:@"btn1"];

UIImage \*imgHighlighted = [UIImage imageNamed:@"btn1"];

//4.设置背景图片

[button setBackgroundImage:imgNormal forState:UIControlStateNormal];

[button setBackgroundImage:imgHighlighted forState:UIControlStateHighlighted];

//5.设置frame属性（位置和大小）

button.frame = CGRectMake(50, 50, 100, 100);

//6.通过代码为控件注册一个单机事件

[button addTarget:self action:@selector(buttonPrint) forControlEvents:UIControlEventTouchUpInside];

//7.把动态创建的控件添加到控制器的view中

[self.view addSubview:button];

}

- (void)buttonPrint{

printf("测试打印");

[(AppDelegate \*)[UIApplication sharedApplication].delegate showUnityWindow];

}

@end

以上操作完成就可以构建发布到手机上看效果了。我这边已经完成了。

1. **注意事项，会将上述的部分流程独立为注意事项：**

这里要注意3.4和3.13

1) 3.4：这里不删除的话会导致链接的时候报错；

2) 3.13：这里要添加以下key value，不然会导致ios跳转unity崩溃。

1. **unity项目更新后如何操作。**

当unity项目更新后，构建完成会替换掉原来的项目，此时只需要做以下修改即可：

修改UnityAppController.h中的内容，修改成和上面一样的内容即可，这样的话就可以完成更新替换了。