AdWhirl Open Source SDK for iPhone 2.2.1
Setup Instructions

11/09

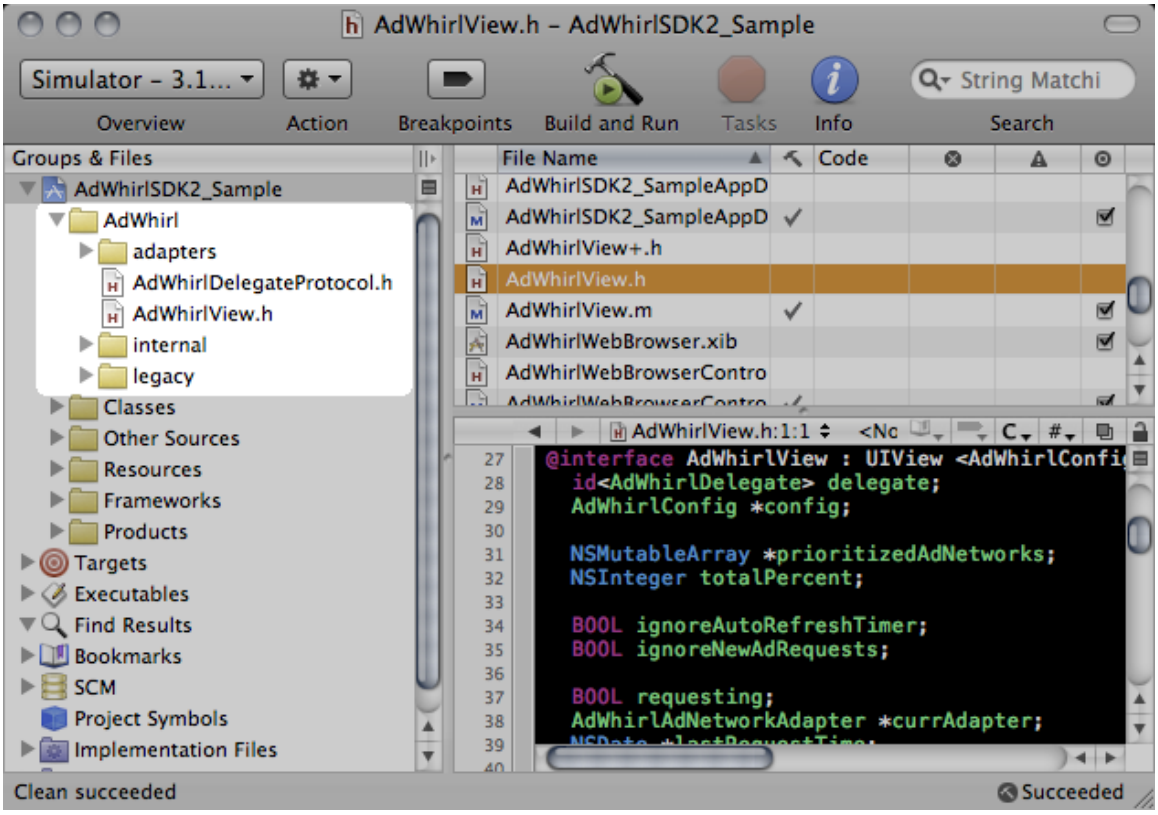
AdWhirl Open Source SDK Setup Instructions for iPhone SDK 2.2.1

The AdWhirl Open Source SDK contains the code for your iPhone application to display ads from different ad networks.
These instructions are for developers who need to deploy to devices running iPhone OS 2.2.1 . To support iPhone OS 2.2.1, build a binary with iPhone Device 3.0 as the Base SDK, and iPhone OS 2.2.1 as the deployment target. This way, you may be able to support both iPhone OS 2.2.1 and 3.0 devices with just one binary, depending on the ad networks you use. However, if you use an ad network that provides separate binaries for 2.2.1 and 3.0 , you may still have to build two different binaries in order to take full advantage of the features offered by the 3.0 SDKs.
The instructions below assume that you are familiar with Xcode and understand how to change build settings and add frameworks to your Xcode projects.

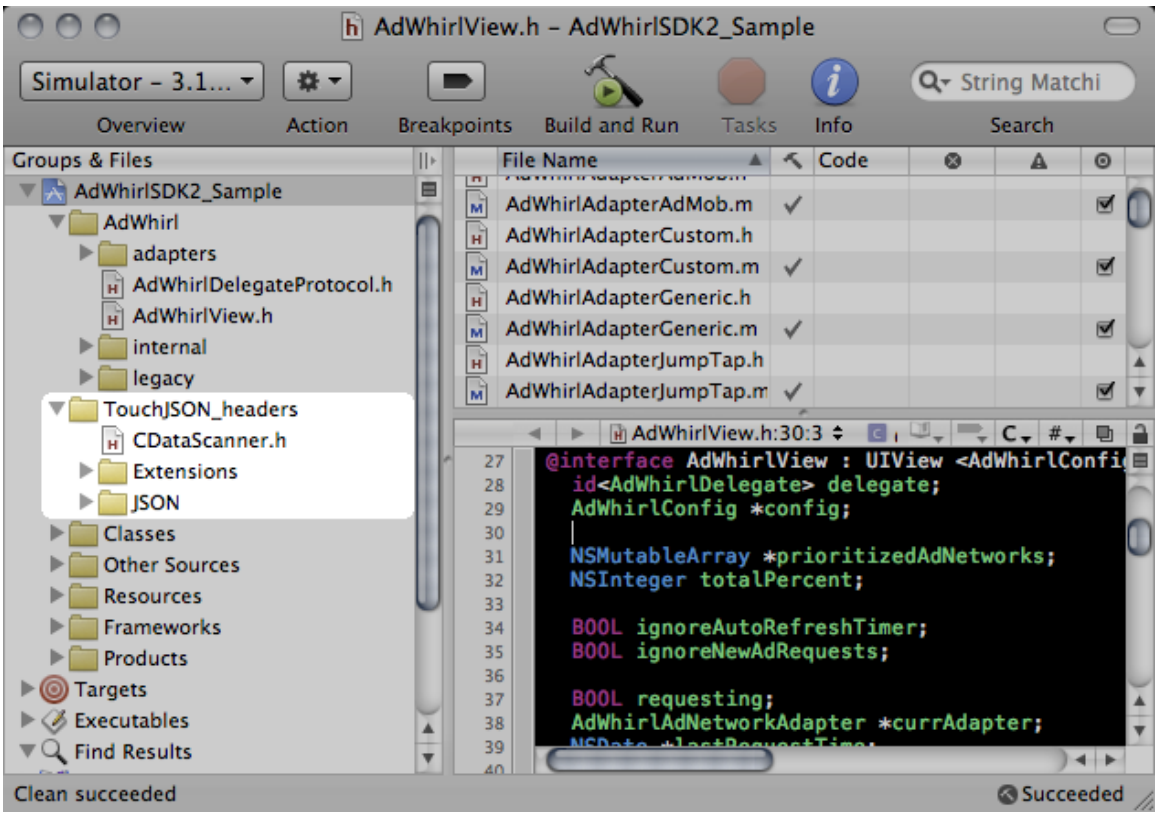
Note: When adding files to your Xcode project, make sure "Copy items into destination group's folder (if needed)" is checked, and select "Recursively create groups for any added folders".

Setting up the AdWhirl SDK

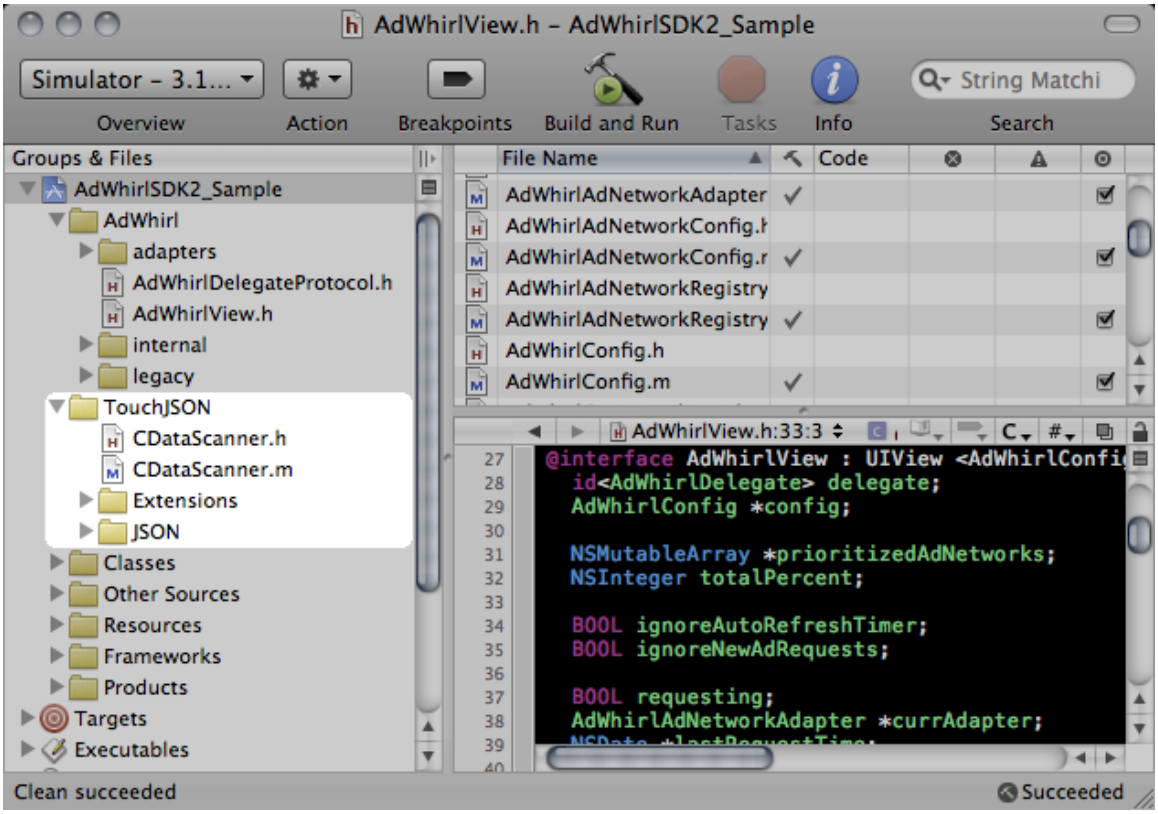
1. Add the `AdWhirl` folder from the SDK into your project. In the `AdWhirl` folder, you'll find two files (`AdWhirlDelegateProtocol.h, AdWhirlView.h`) and three folders (`adapters, internal, legacy`).



2. Add TouchJSON into your project. If you plan to include the AdMob SDK, drag the `TouchJSON_headers` directory into your project.



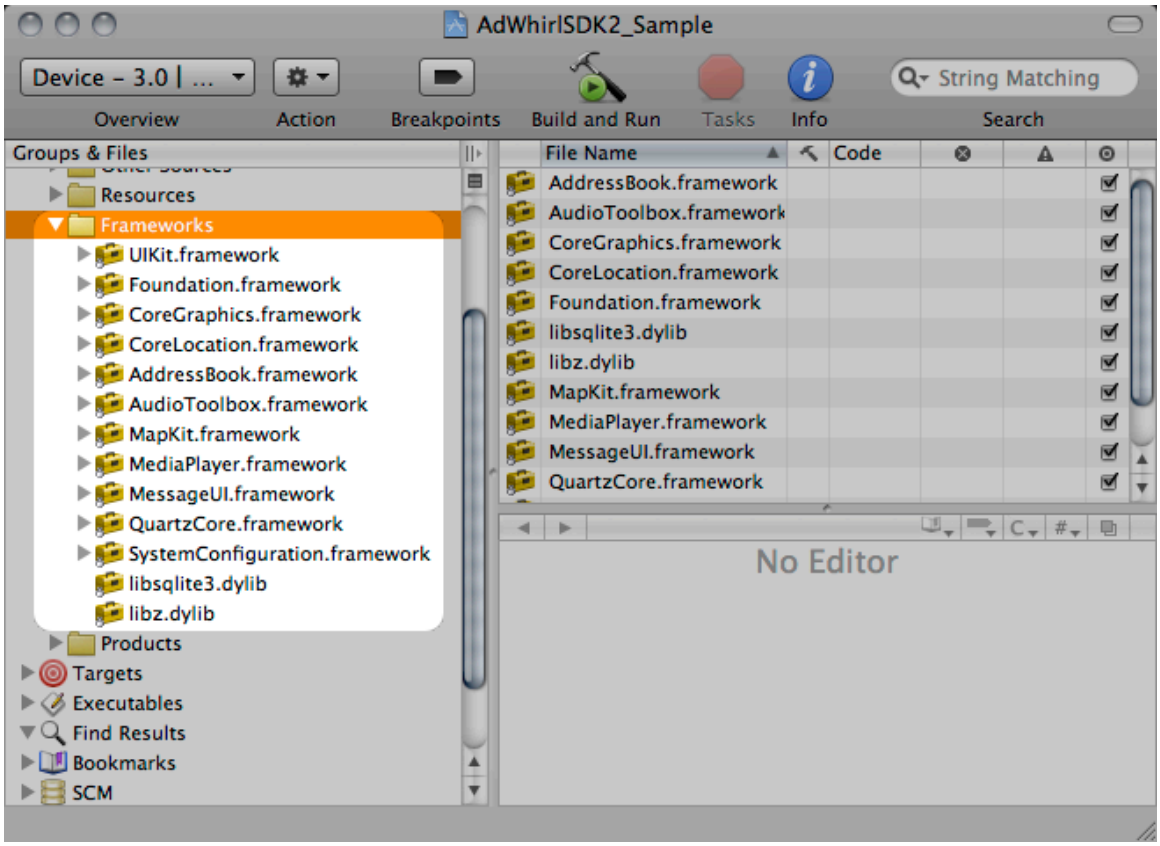
Otherwise, drag the `TouchJSON` directory into your project.



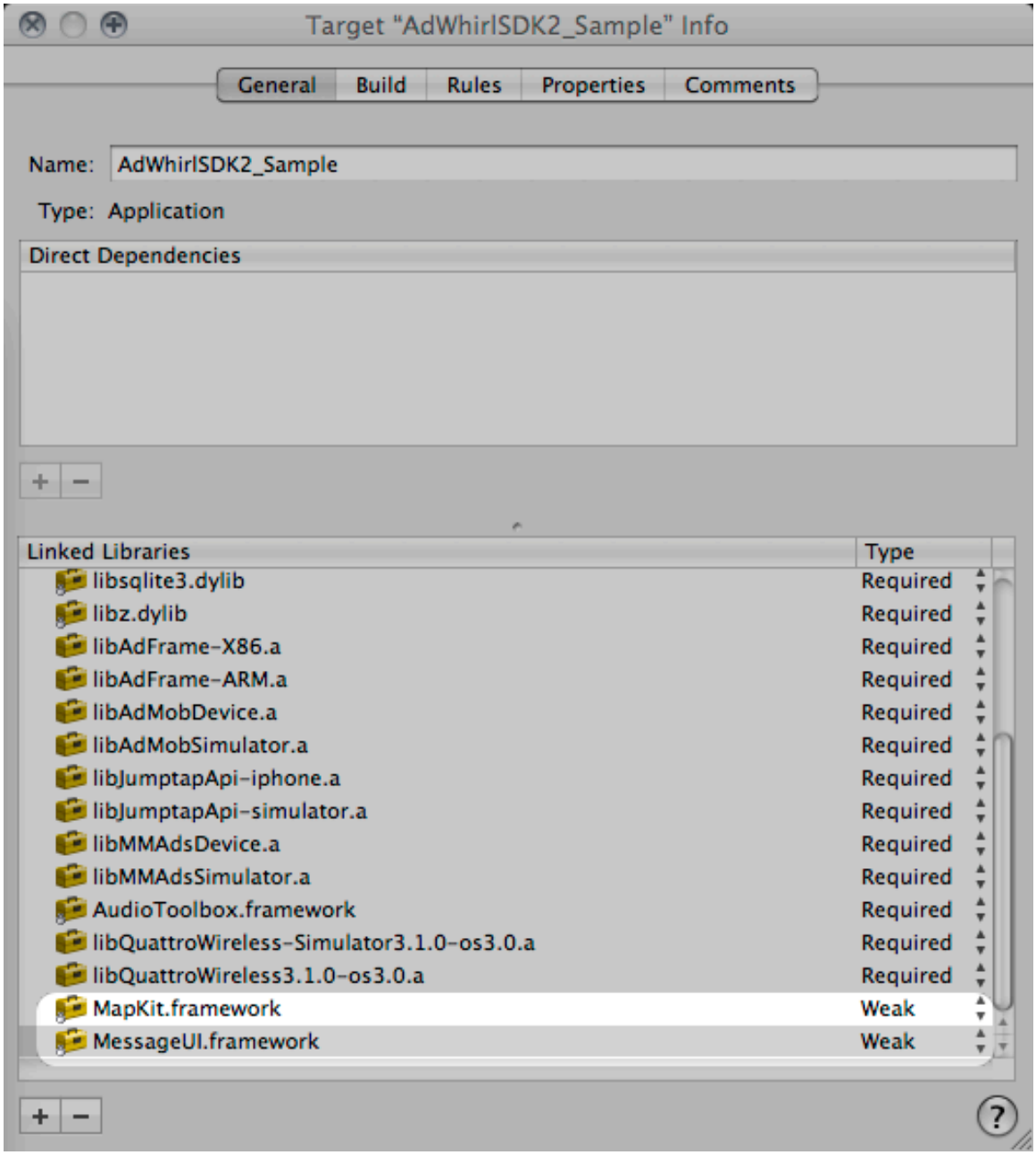
3. Add the supporting frameworks required by the supported ad networks:

- `CoreLocation.framework`
- `AddressBook.framework`
- `AudioToolbox.framework`
- `MapKit.framework`
- `MediaPlayer.framework`
- `MessageUI.framework`
- `QuartzCore.framework`
- `SystemConfiguration.framework`
- `libsqlite3.dylib`
- `libz.dylib`

The frameworks do not affect the size of your executable unless they are used by ad network libraries that you integrate.



Some frameworks need to be weak-linked. You can weak-link a framework by right-clicking your application's target, select "Get Info." Under the General tab, in the list of Linked Libraries, change the Type from "Required" to "Weak".



4. Add the ad network libraries. All of the ad network libraries are optional. You do not have to integrate an ad network if you don't want to run the ads, or don't have access to the SDK.

    a. AdMob

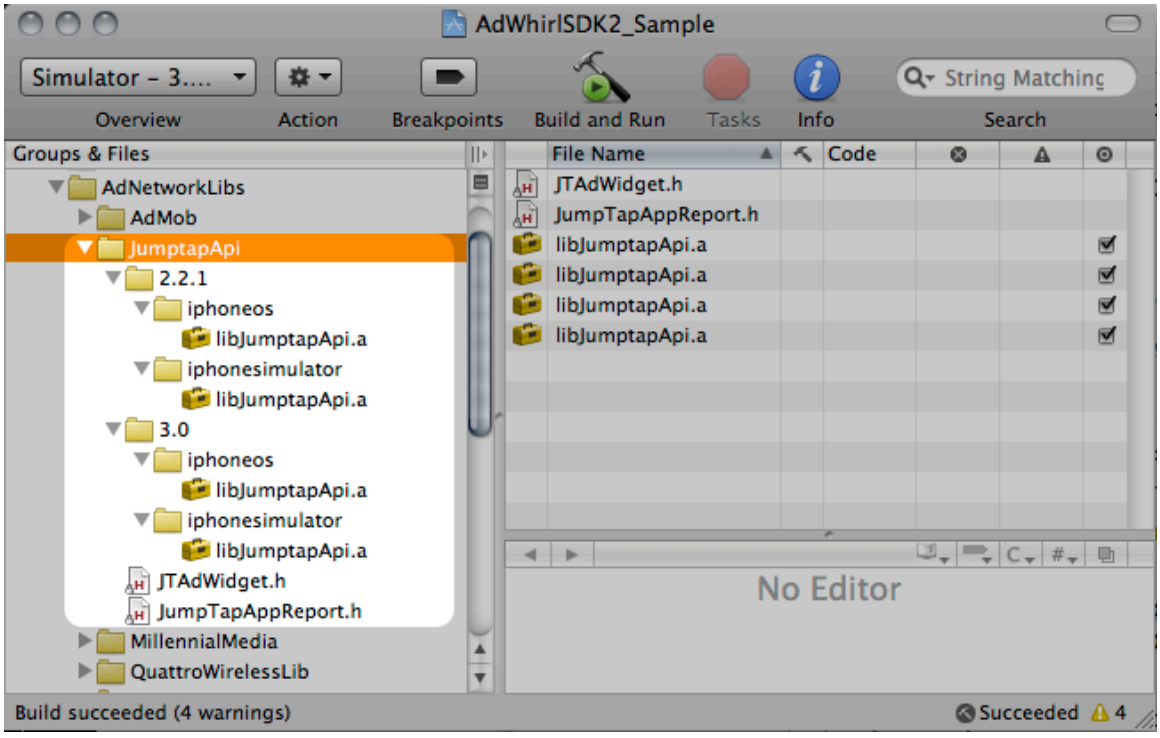
    i. Download the AdMob SDK from http://www.admob.com . As of version 20091119, AdMob provides separate binaries for iPhone OS 2.2.1 and 3.0
    ii. Drag the `AdMob` folder into your Xcode project (containing `AdMobDelegateProtocol.h`, `AdMobView.h`, `libAdMobSimulator.a`, `libAdMobDevice.a`).

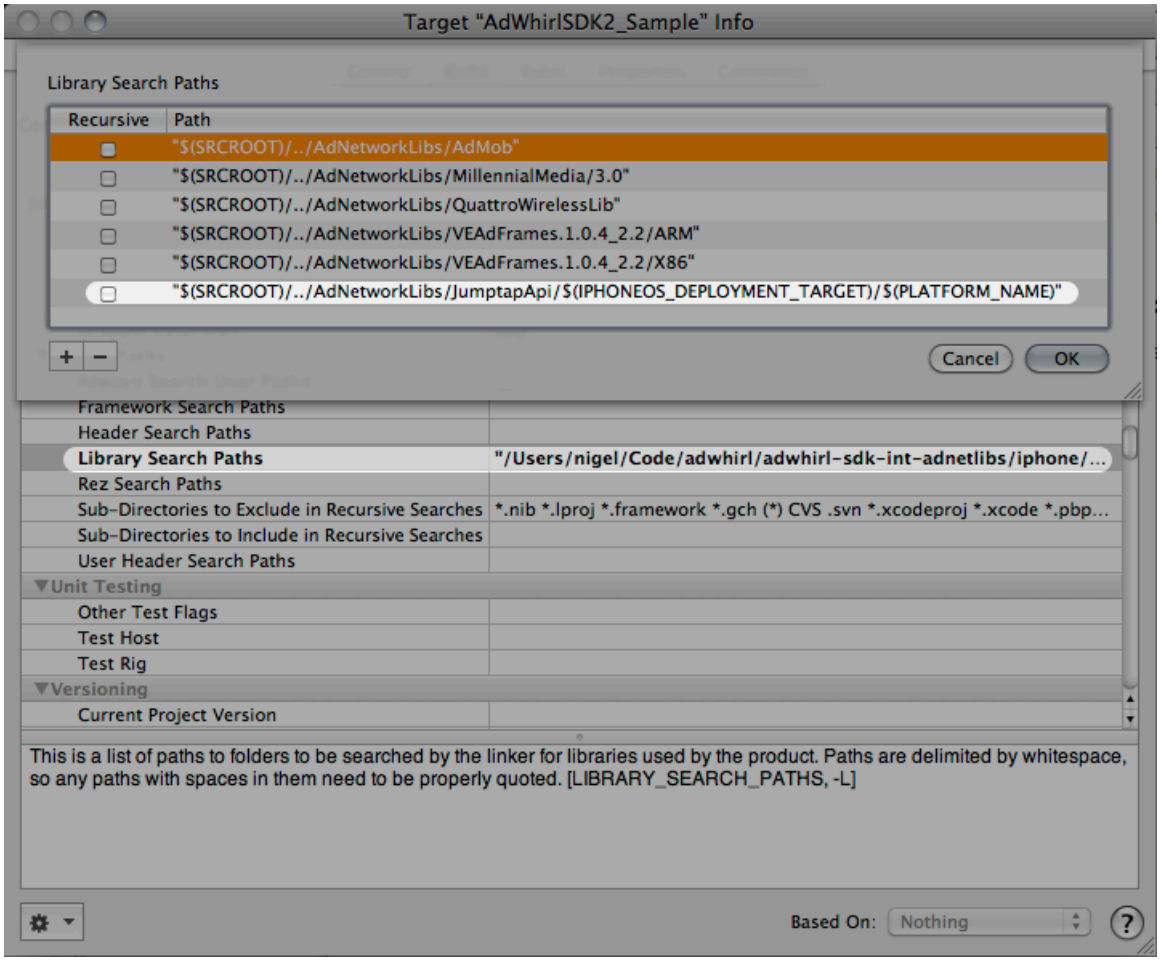
b. JumpTap

i. Download the JumpTap SDK from https://support.jumptap.com/index.php/IPhone_Library_Integration#Downloading_the_library
As of version 1.0.3 (24 Nov 2009) JumpTap provides separate libraries for 3.0 and 2.2.1

ii. Drag the `JumpTap` folder into your Xcode project (containing the folders `2.2.1` and `3.0`, and the files `JTAdWidget.h` and `JumpTapAppReport.h`).

iii. Right click your primary target -> Get Info -> Build tab. Double click on the value for "Library Search Paths". Remove all the JumptapApi paths except one, and change it to "$(SRCROOT)/JumptapApi/$(IPHONEOS_DEPLOYMENT_TARGET)/$(PLATFORM_NAME)/" . The path components before "/JumptapApi" may differ in your environment.



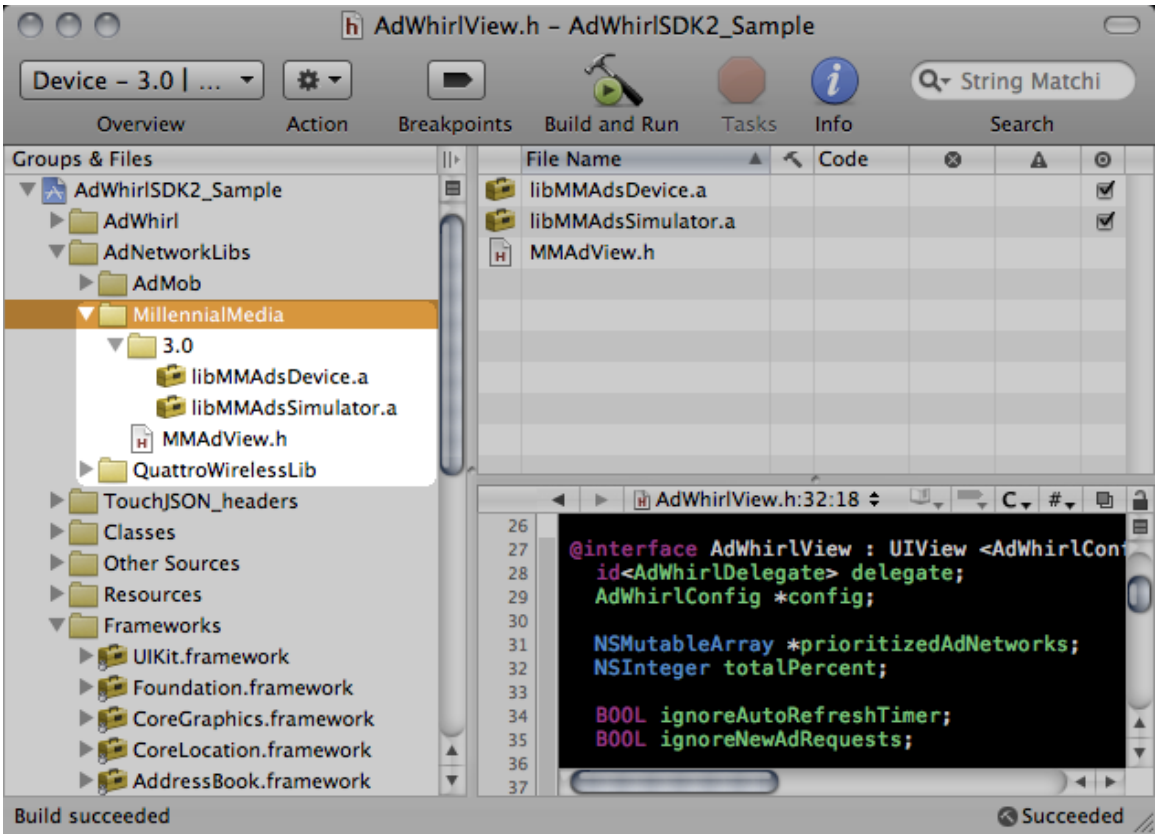


c. Millennial Media

i. Download the Millenial Media SDK from http://developer.millennialmedia.com/. This SDK provides separate binaries for both iPhone OS 2.2.1 and iPhone OS 3.0

ii. Drag the `MillennialMedia` folder into your Xcode project (containing the folders `2.2.1` and `3.0`, and the file `MMAdView.h`).

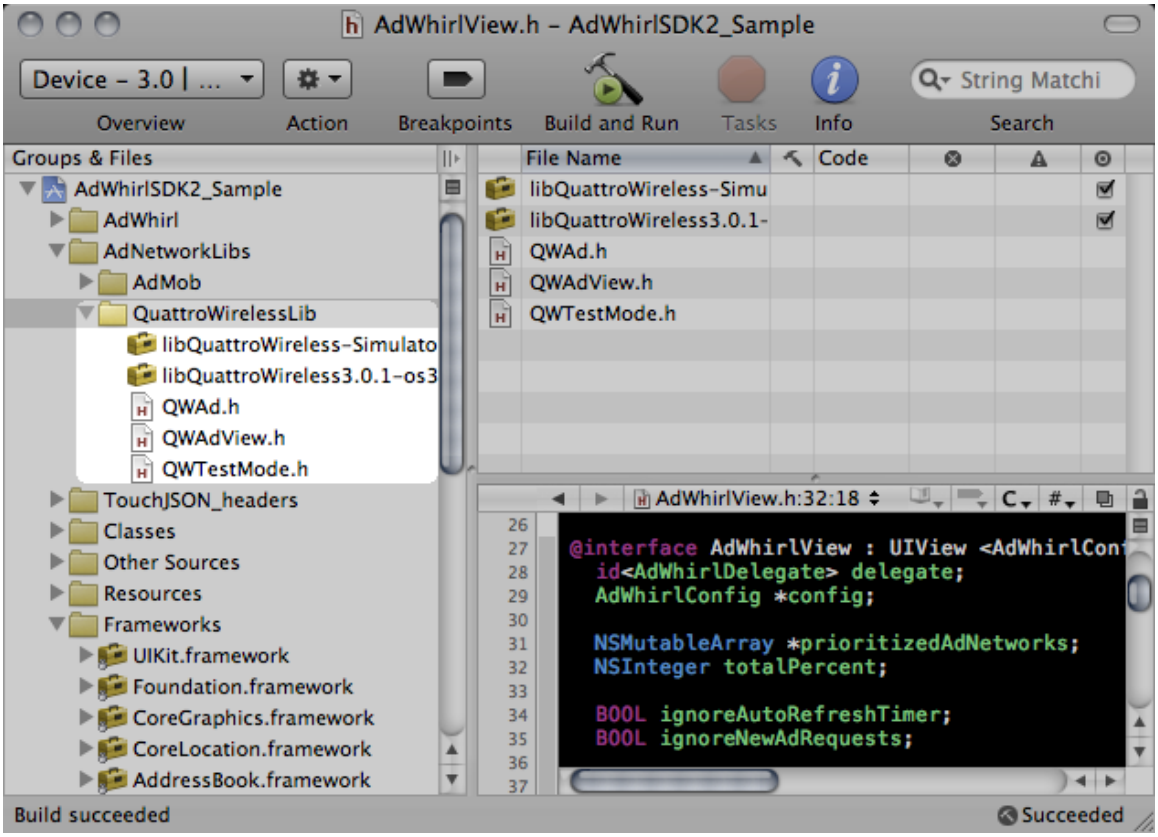
iii. Remove the `3.0` folder.



d. Quattro Wireless

i. Download the Quattro Wireless SDK from http://www.quattrowireless.com. This SDK provides one binary for both iPhone OS 2.2.1 and iPhone OS 3.0 support

ii. Drag the `QuattroWirelessLib` folder into your Xcode project (containing `libQuattroWireless-Simulator3.1.0-os3.0.a`, `libQuattroWireless3.1.0-os3.0.a`, `QWTestMode.h`, `QWAd.h`, `QWAdView.h`. Exact names of `.a` files may be different depending on the version you download).

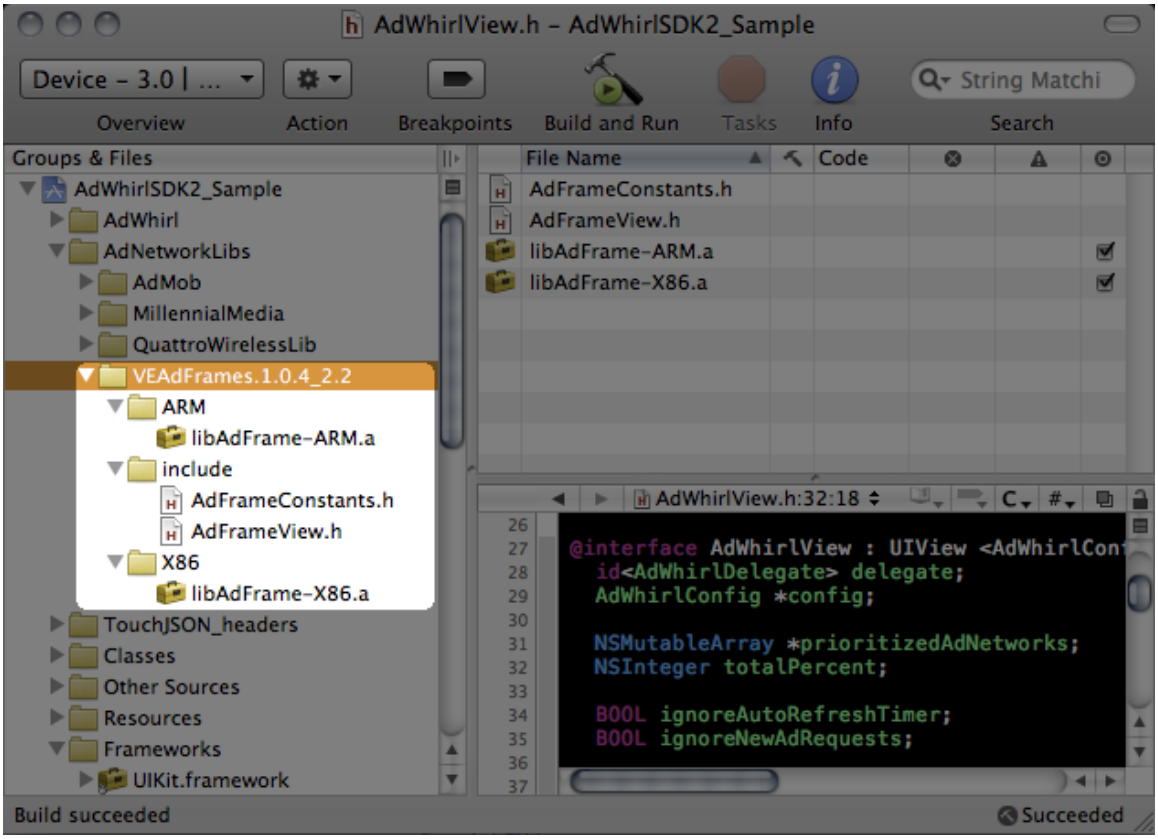


e. VideoEgg

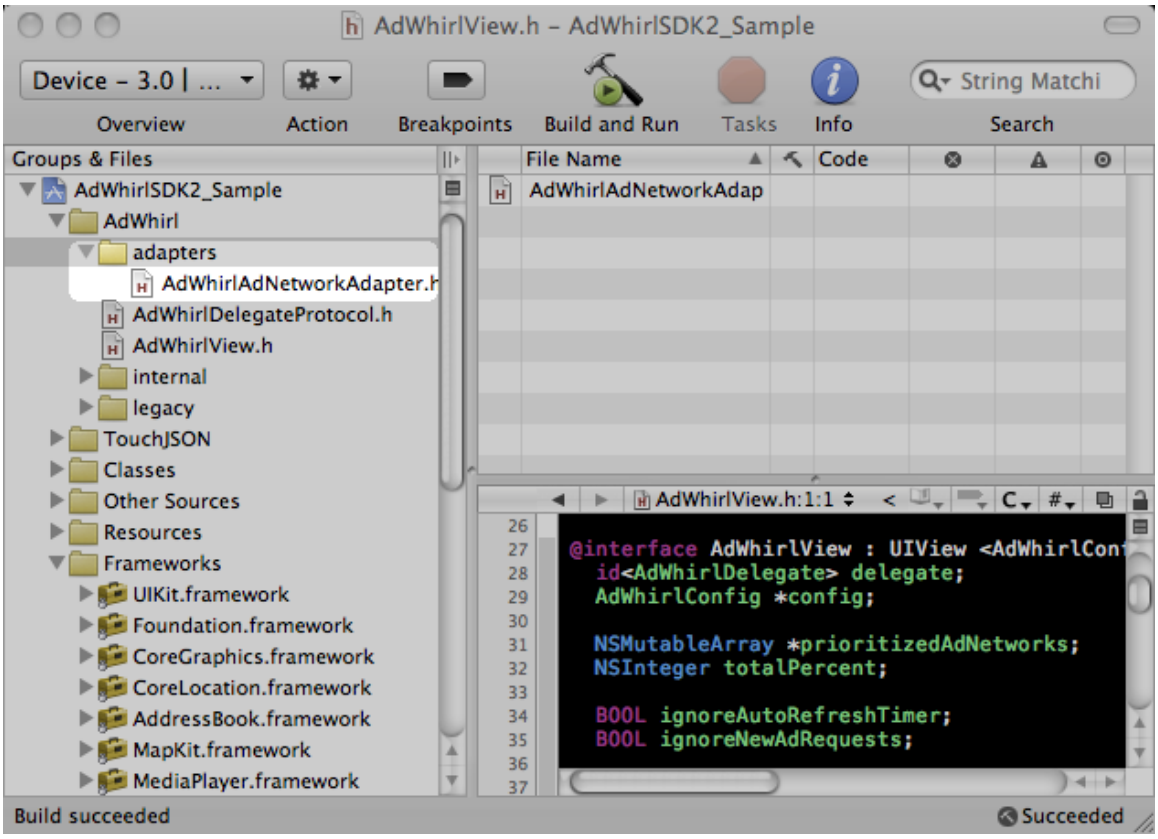
i. Download the VideoEgg SDK from http://www.videoegg.com/. This SDK provides one binary for both iPhone OS 2.2.1 and iPhone OS 3.0 support

ii. Drag the appropriate `VEAdFrames` folder under the `lib` folder into your project (containing the folders `ARM`, `X86`, and `include`).

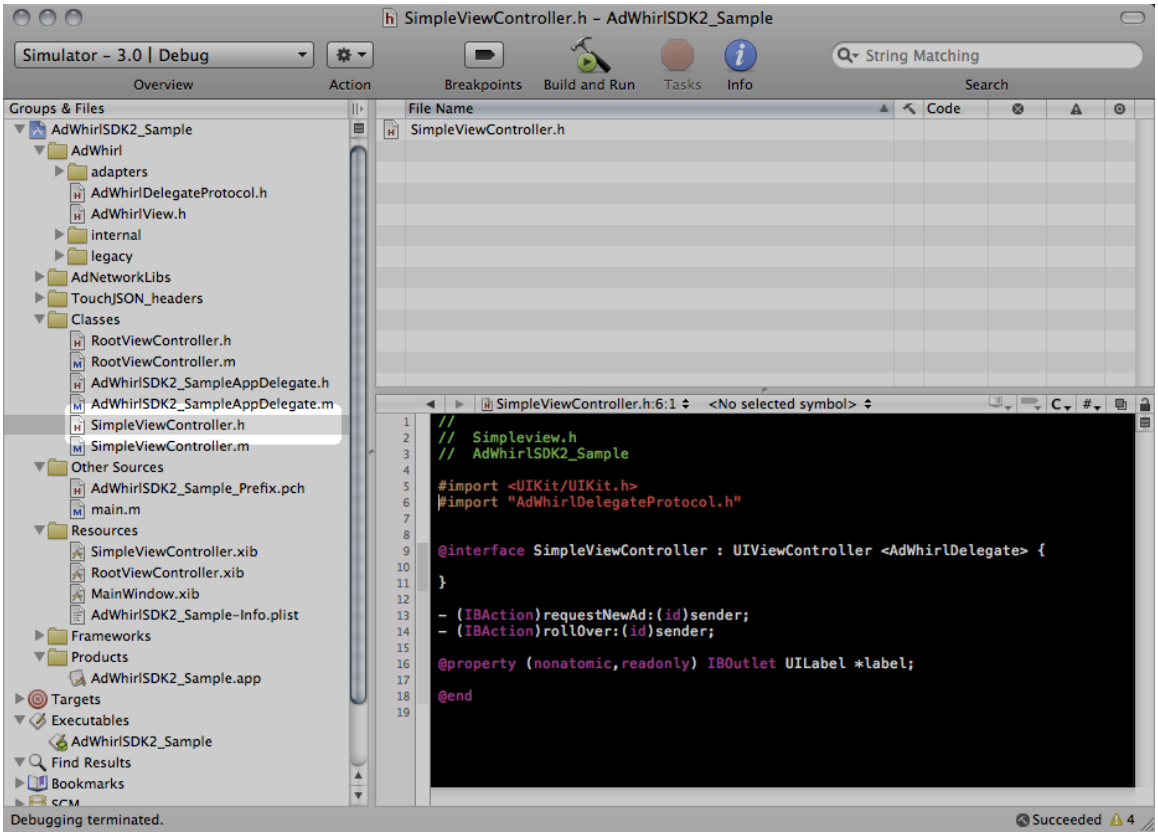
iii. To avoid build errors, remove `AdFrameView.o` from under the `ARM` and `X86` folders. Rename `libAdFrame.a` in each directory to include the architecture in the file name, such as `libAdFrame-ARM.a` and `libAdFrame-X86.a`.


AdWhirlView.h – AdWhirlSDK2_Sample
Device – 3.0 | ...
Overview
Action
Breakpoints
Build and Run
Tasks
Info
String Matchi
Search
Groups & Files
AdWhirlSDK2_Sample
AdWhirl
AdNetworkLibs
AdMob
MillennialMedia
QuattroWirelessLib
VEAdFrames.1.0.4_2.2
ARM
libAdFrame-ARM.a
include
AdFrameConstants.h
AdFrameView.h
X86
libAdFrame-X86.a
TouchJSON_headers
Classes
Other Sources
Resources
Frameworks
UIKit.framework
File Name
Code
AdFrameConstants.h
AdFrameView.h
libAdFrame-ARM.a
libAdFrame-X86.a
AdWhirlView.h:32:18
@interface AdWhirlView : UIView <AdWhirlCon
id<AdWhirlDelegate> delegate;
AdWhirlConfig *config;
NSMutableArray *prioritizedAdNetworks;
NSInteger totalPercent;
BOOL ignoreAutoRefreshTimer;
BOOL ignoreNewAdRequests;
Build succeeded
Succeeded

5. Remove the ad network adapters that you don't use. Under the `adapters/` group, you must remove the ad network adapters for ad networks that you did not integrate in step 4, or your project will not compile. In the extreme case, you can remove all files under `adapters/` except the `AdWhirlAdNetworkAdapter.h` file, in which case your app can only run custom ads and generic notifications.

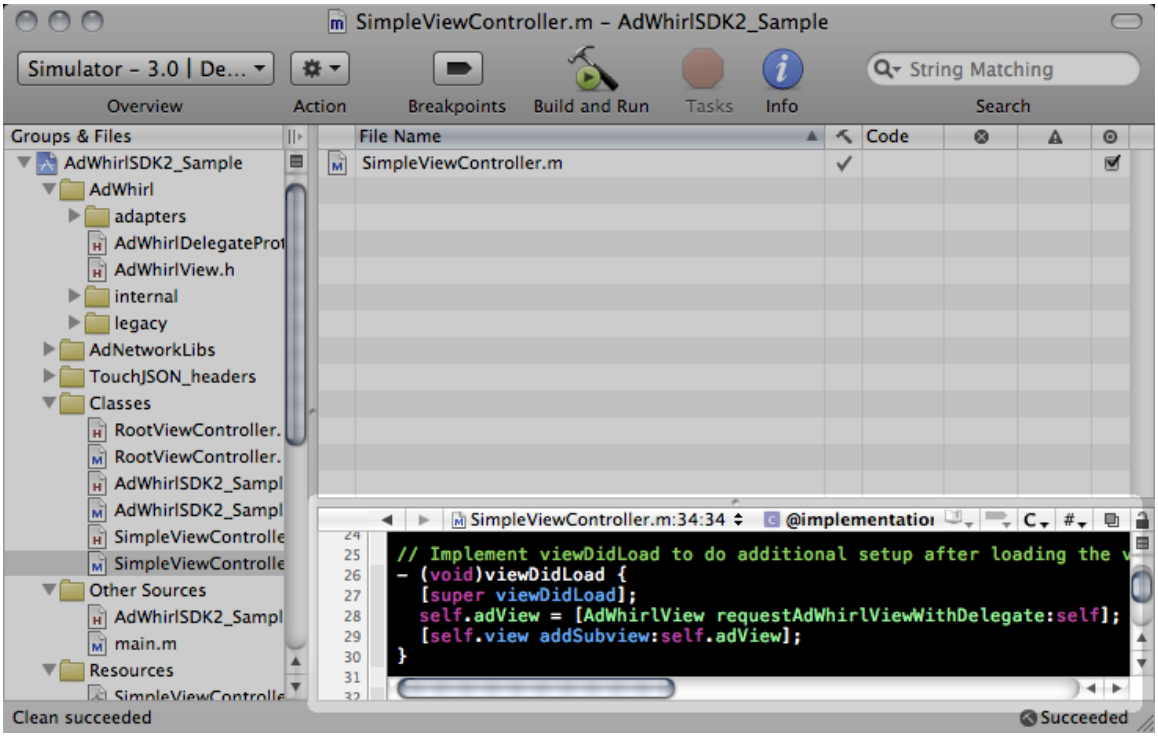


6. Implement the required `AdWhirlDelegate` methods in your code. If you have integrated with AdWhirl before and have implemented `AdRollerDelegate`, you do not have to change the code. This open source SDK is a drop-in replacement of ARRoller APIs and is backwards-compatible. At a minimum, you must implement the `adWhirlApplicationKey` method to return your AdWhirl application key. See the comments in `AdWhirlDelegateProtocol.h` for optional methods to implement.



7. Request an `AdWhirlView` in your code and add it to your view hierarchy. For example:

```
 AdWhirlView *awView = [AdWhirlView
requestAdWhirlViewWithDelegate:self];
  [self.view addSubview:awView];
```



8. To avoid build warnings (it is useful to have "Treat Warnings as Errors" turned on), you can uncheck static libraries from the target that are not the same architecture as those you are building.