Jaspersoft Mobile SDK Install Guide

This guide describes 2 ways you can install the Jaspersoft Mobile SDK for iOS. You could add the source files directly to the project, as well, but that method is not covered here, and it is not recommended for compatibility with future versions.

Here are the 2 ways we recommend:
1) **Linking as Static Library or Framework** (section I below) - this is the recommended way.
Benefits of this approach:

- you can easily update/replace the SDK
- resolve class name collisions (when different classes have same name)
- it's ARC independent

2) **Adding the Compiled Library** (section III below) - if you don't want to have the SDK library source inside your application, or don't want to do the additional linking, you can use this type of installation.
Benefits of this approach:

- your project will have all required files in one place, without the need to add them separately
- installation is easier than linking as a static library or adding sources directly
- easy to re-install / update SDK - you just need to replace existing library with a new one and thats it

Some drawbacks:

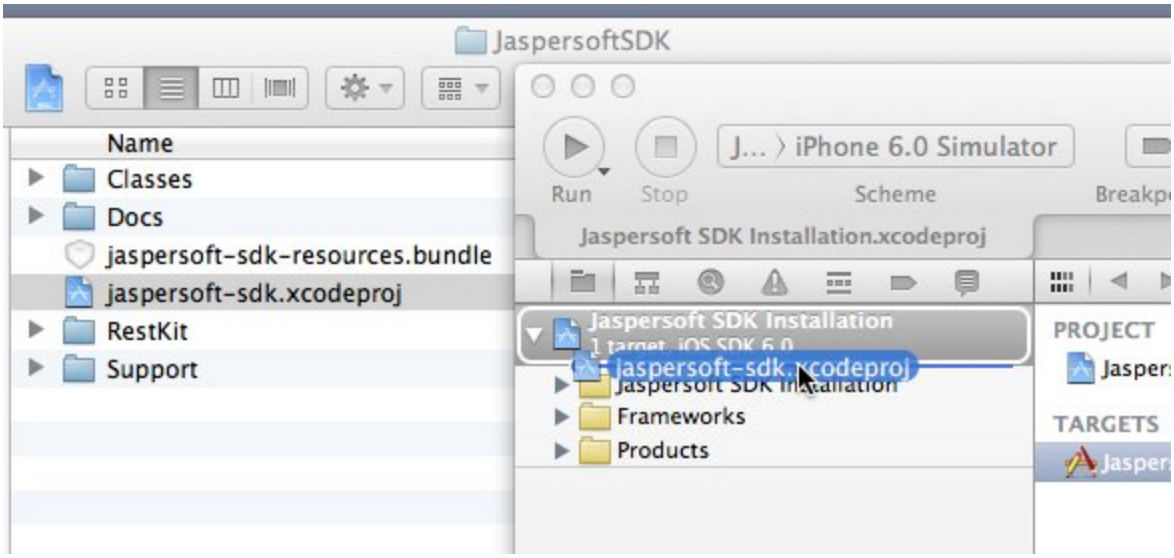
- you will not have direct access to SDK library source code, so you cannot modify it or debug in runtime
- possible class name collisions (as Objective-C has no namespace)
- problems with non-ARC code

I. Linking as Static Library (iOS) or Framework (OS X)

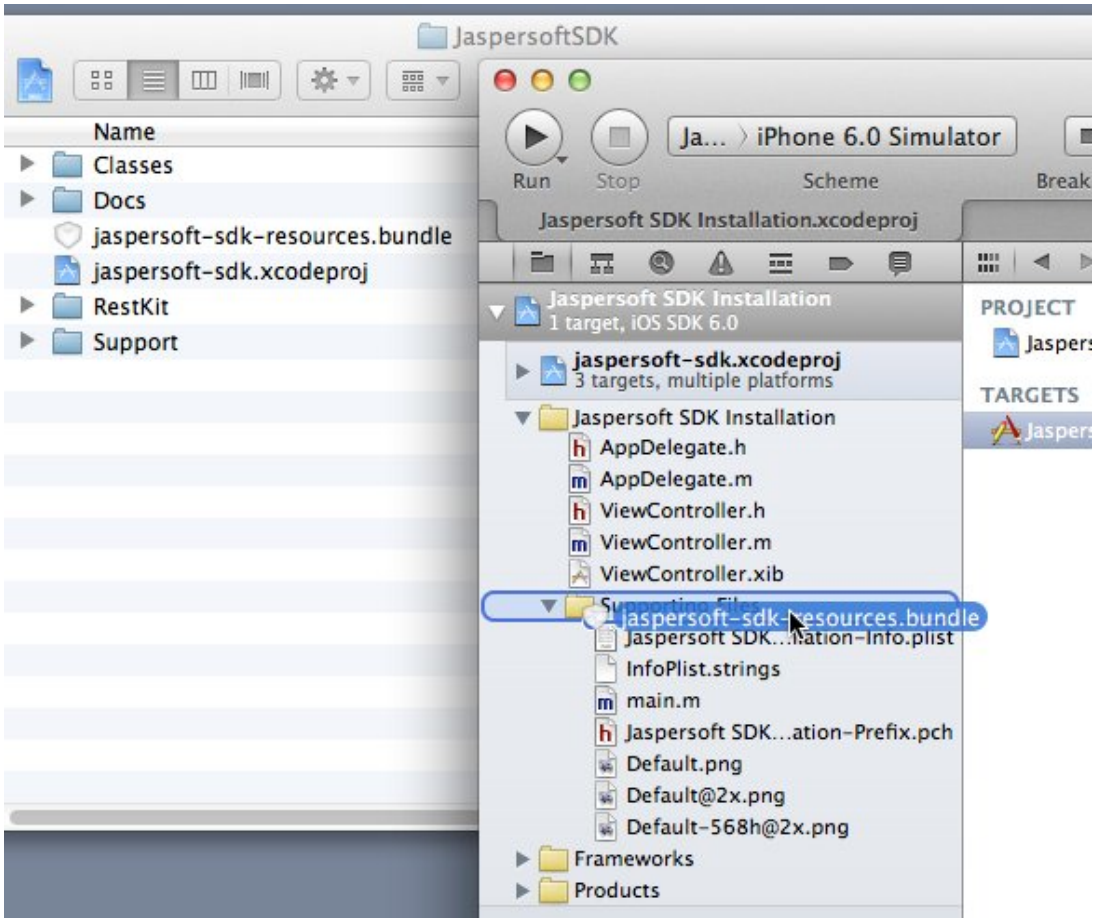
Note: The Jaspersoft Mobile SDK depends on the **RestKit** library (see NOTICES.txt for license details). Installation via linking will add **RestKit** inside the SDK automatically. You can remove it from the SDK if you have already installed it directly in your project (see section II below for instructions).

To install via **linking**:

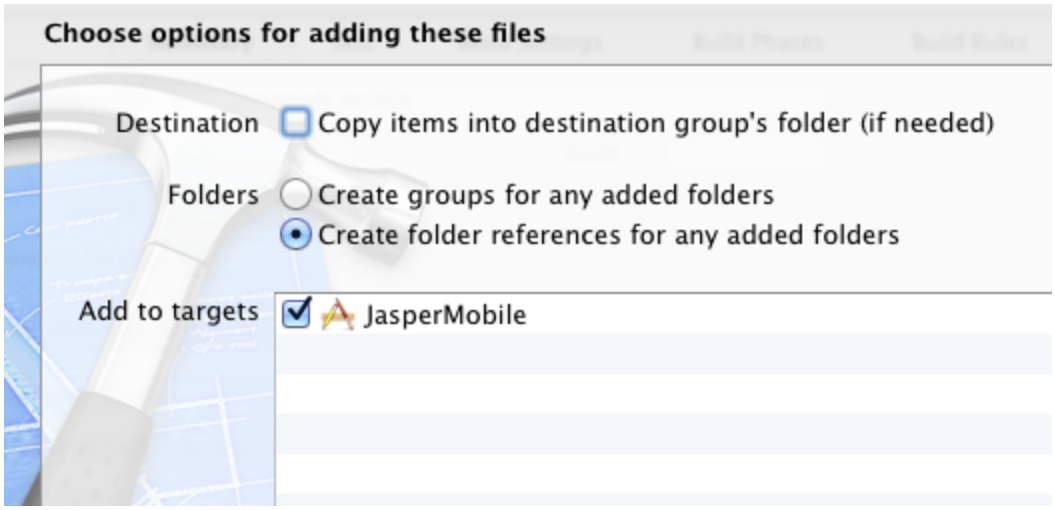
1. Add a cross-project reference by dragging the **jaspersoft-sdk.xcodeproj** file into your open application project in Xcode.



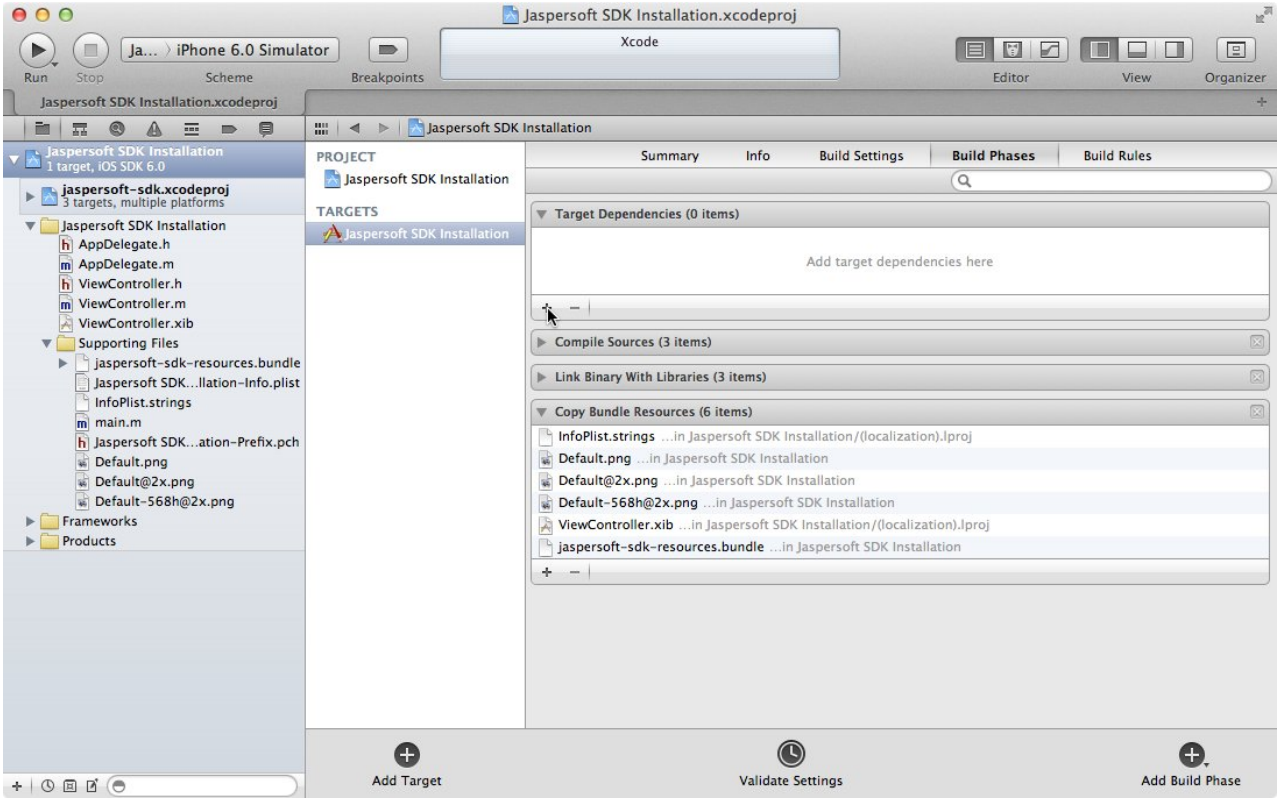
2. Delete any pre-existing references or copies of **jaspersoft-sdj-resources.bundle** in your project.
3. Add an SDK resource bundle reference by dragging the **jaspersoft-sdk-resources.bundle** file into your 'Resources' directory (it could be any directory, but Resources makes sense in this context) in your open application project in Xcode.



4. When the options dialog comes up, leave ‘Destination’ checkbox unchecked (do not copy items...). For ‘Folders’ select the ‘Create folder reference..’ radio button, and make sure your project is checked in the ‘Add to targets’ list:

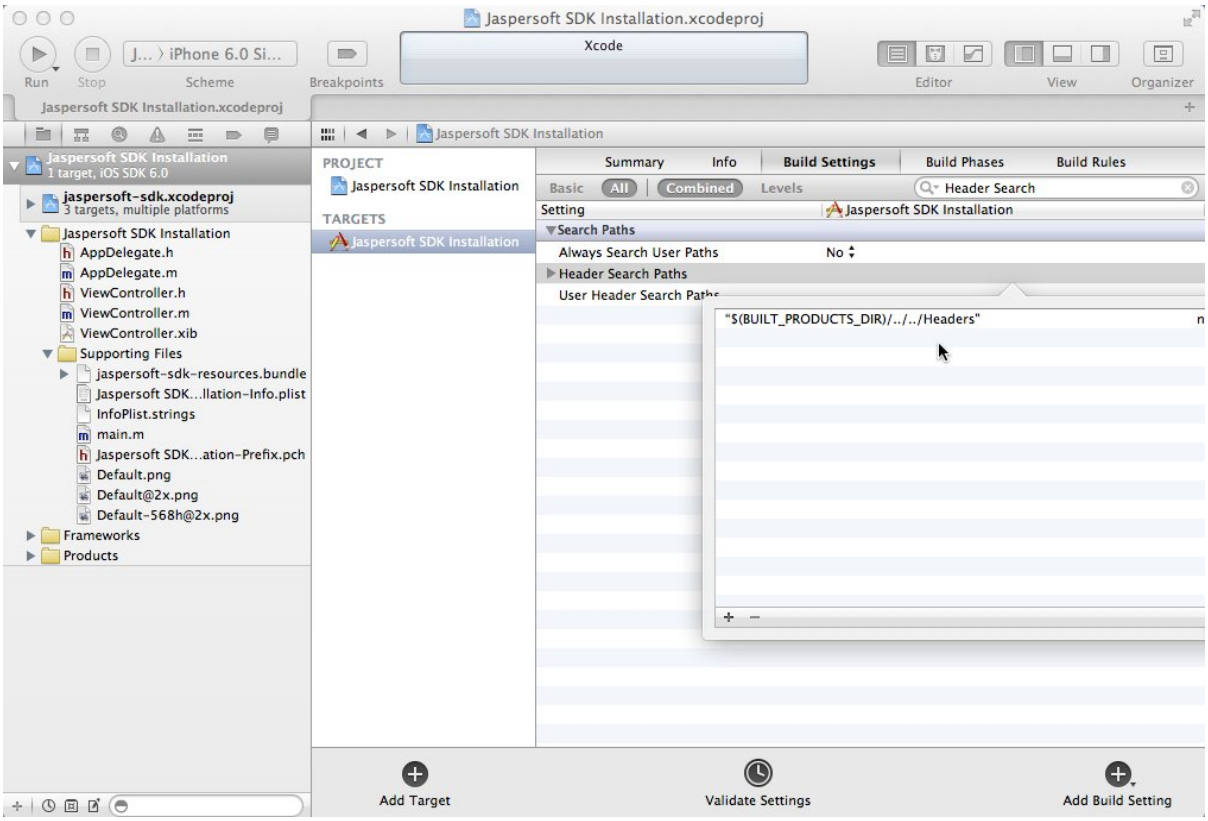


5. Open target settings editor -> **Build Phases** for the target you want to link library into
6. In the tab **Target Dependencies** tab add dependency on the **jaspersoft-sdk** (on iOS) or **jaspersoft-sdk-framework** (on OS X) aggregate target.

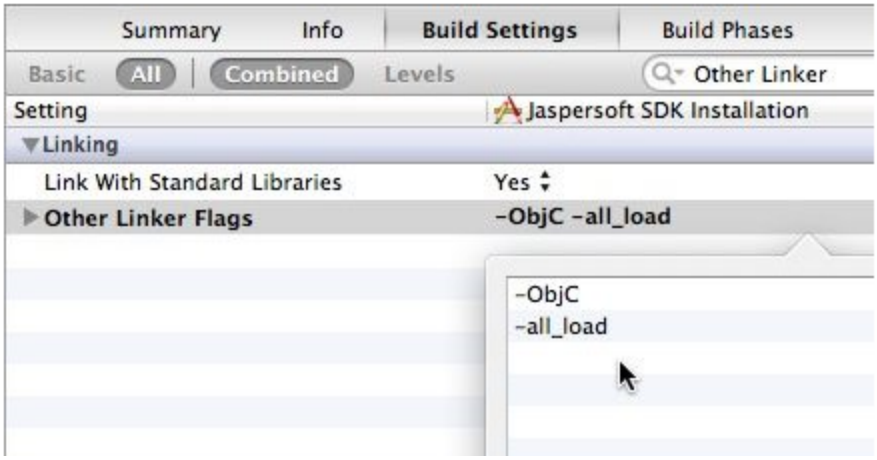


7. In the tab **Link Binary With Libraries** link against next frameworks and libraries:
    - **libjaspersoft-sdk.a on iOS or jaspersoft-sdk.framework on OS X** (don't worry that it shows up in red)
    - **CFNetwork.framework on iOS**
    - **CoreData.framework**
    - **Security.framework**
    - **MobileCoreServices.framework on iOS or CoreServices.framework on OS X**
    - **SystemConfiguration.framework**
    - **libxml2.dylib**
    - **QuartzCore.framework on iOS**

8. Open **Build Settings -> Search Paths -> Header Search Paths** and double-click on the value column in the top level. Add **"$(BUILT_PRODUCTS_DIR)/../../Headers",** if it's not already there**.**



9. Open **Build Settings -> Linking -> Other Linker Flags,** and add "**-ObjC**" and "**-all_load**"**,** if they are not already there (you can copy/paste the following string, just make sure it gets properly separated into 2 flags: -ObjC -all_load)**.**



10. You have now completed installation of the **Jaspersoft Mobile SDK for iOS** into your application.
11. To verify the installation, open up your App Delegate and add an import of the **JaspersoftSDK** header:

    #import <jaspersoft-sdk/JaspersoftSDK.h>

    Note: If you are building the JasperMobile app for iOS, it's already there.
12. Build your project and verify build output. Project should build without any issues.

II. Disabling Inclusion of RestKit library inside SDK

If you have already installed the **RestKit** library, you can disable including it again for the JS Mobile SDK.

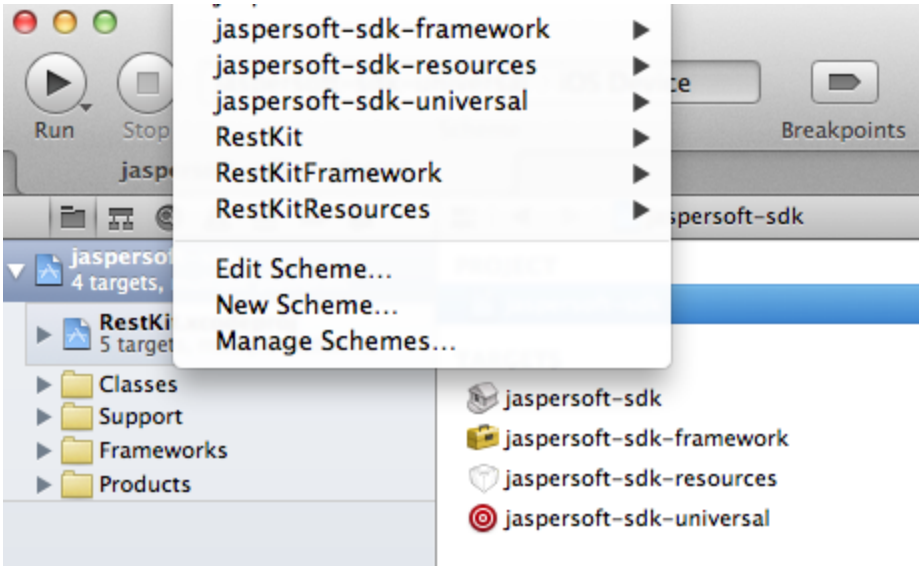
To disable including **RestKit** library for SDK:

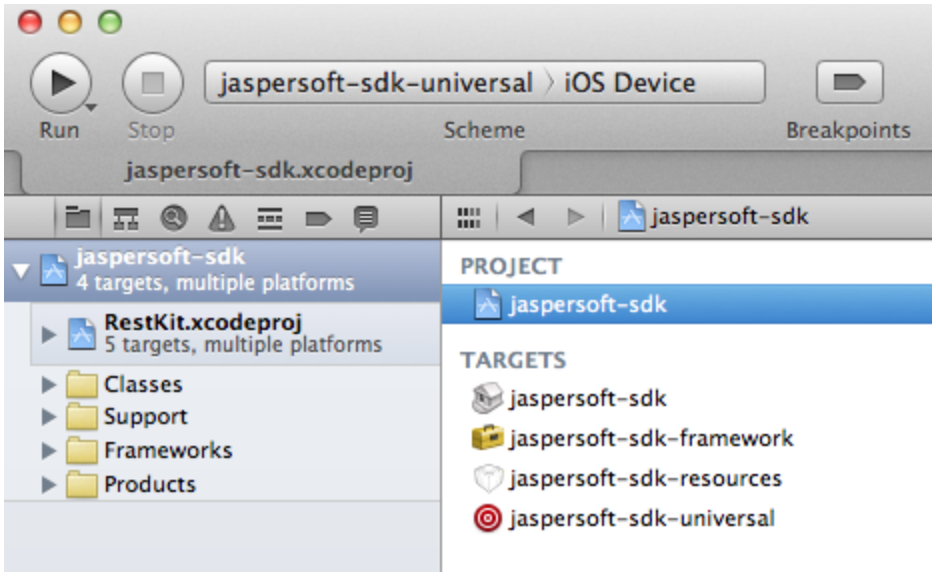
1. Open **jaspersoft-sdk** target settings
2. Delete reference to file **RestKit.xcodeproj** in your open application project
3. Confirm deletion

This will build the SDK without including the **RestKit** directly into your library file.

III. Adding the Compiled Library

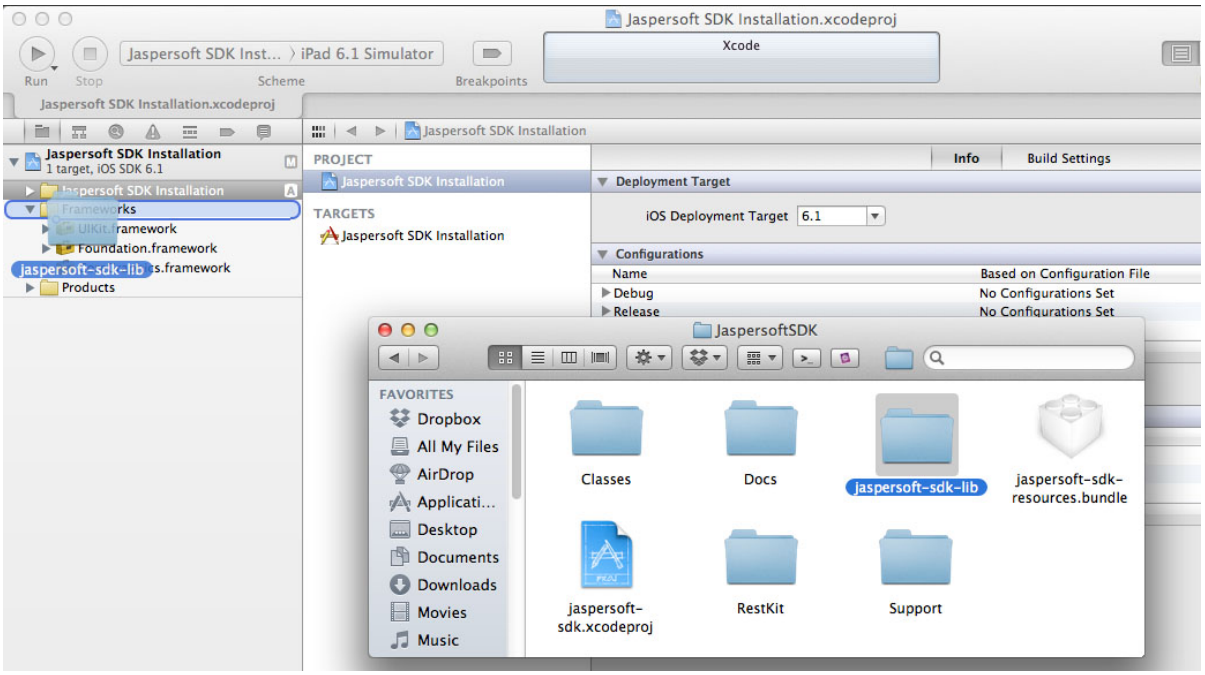
1. Open in Xcode **jaspersoft-sdk.xcodeproj** project file
2. Select **jaspersoft-sdk-universal** as current scheme (type of device doesn't matter, it can be iOS Device or iPhone / iPad simulator)



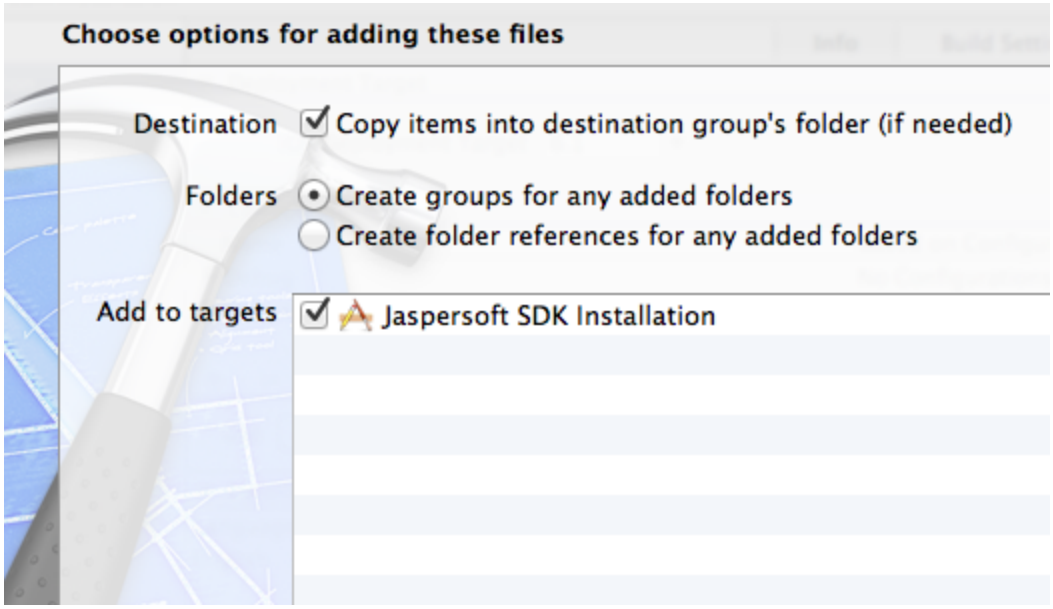


3. Build project (**⌘B** keybind**)**

4. Open SDK project location directory. Find the **jaspersoft-sdk-lib** folder, and drag it into your project.

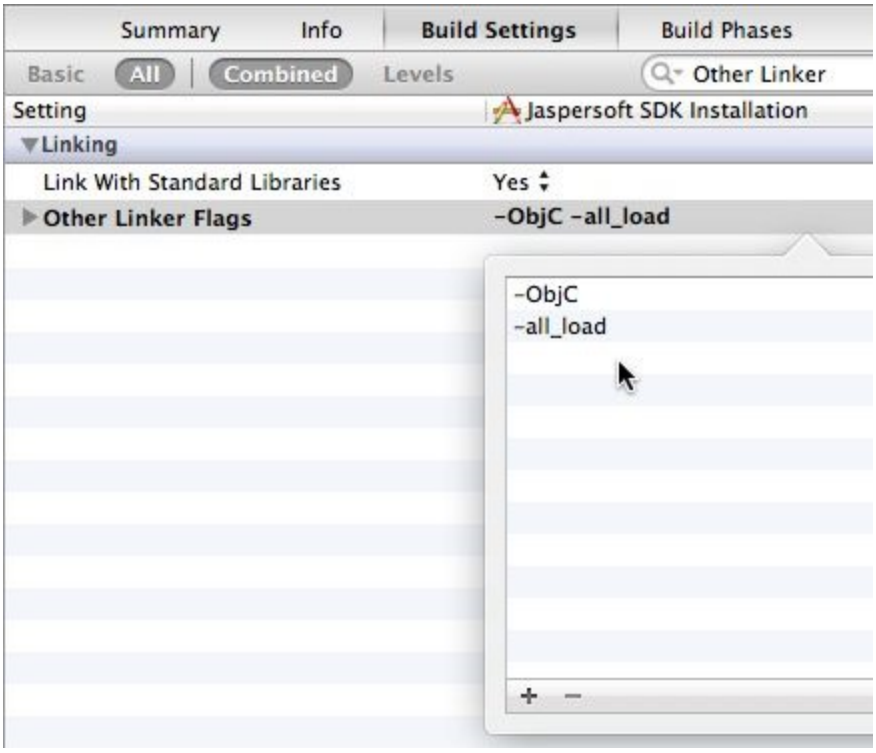


5. When the options dialog comes up, leave 'Destination' checkbox checked (do not copy items...). For 'Folders' select the 'Create folder reference..' radio button, and make sure your project is checked in the 'Add to targets' list:



6. Open target settings editor -> **Build Phases**

7. In the tab **Link Binary With Libraries** link against next frameworks and libraries:
    - **CFNetwork.framework on iOS**
    - **CoreData.framework**
    - **Security.framework**
    - **MobileCoreServices.framework on iOS or CoreServices.framework on OS X**
    - **SystemConfiguration.framework**
    - **libxml2.dylib**
    - **QuartzCore.framework on iOS**
8. Open **Build Settings -> Linking -> Other Linker Flags,** and add "**-ObjC**" and "**-all_load**"**,** if they are not already there (you can copy/paste the following string, just make sure it gets properly separated into 2 flags: -ObjC -all_load)**.**



9. You have now completed installation of the Jaspersoft Mobile SDK for iOS into your application.
10. To verify the installation, open up your App Delegate and add an import of the JaspersoftSDK header:

    #import "JaspersoftSDK.h"

    Note: If you are building the JasperMobile app for iOS, it's already there.
11. Build your project and verify build output. Project should build without any issues.