JasperReports Server

Mobile Developer Guide

http://www.jaspersoft.com

Copyright © 2013 Jaspersoft Corporation. All rights reserved. Printed in the U.S.A. Jaspersoft, the Jaspersoft logo, Jaspersoft iReport Designer, JasperReports Library, JasperReports Server, Jaspersoft OLAP, and Jaspersoft ETL are trademarks and/or registered trademarks of Jaspersoft Corporation in the United States and in jurisdictions throughout the world. All other company and product names are or may be trade names or trademarks of their respective owners.

Portions of section **1.2, “Setting Up the Google Android SDK,” on page 6** are modifications based on work created and shared by the Android Open Source Project and used according to terms described in the Creative Commons 2.5 Attribution License. The original page is http://developer.android.com/sdk/installing.html.

This is version 0613-JSP52-03 of the *JasperReports Server Mobile Developer Guide*.

TABLE OF CONTENTS

Chapter 1 Getting Started With Jaspersoft Mobile SDK for Android

JasperReports Server builds on the JasperReports Library as a comprehensive family of Business Intelligence (BI) products, providing robust static and interactive reporting, report server, and data analysis capabilities. These capabilities are available as either individual products, or as part of an integrated end-to-end BI suite utilizing common metadata and providing shared services, such as security, a repository, and scheduling. The BI suite can be deployed in a stand-alone server where users log in to view their data, or the server functionality can be embedded so that users have secure and reliable access to their data within another application.

JasperReports Server functionality can also be accessed from mobile devices. When using the mobile software development kits (SDKs) for the iOS and Android platforms, you can easily embed server functionality in your mobile application and display reports to the user. The mobile SDKs from Jaspersoft use the representational state transfer (REST) API to send and receive data to and from the server.

This preliminary version of this document covers the installation of the Jaspersoft Mobile SDKs and gives a preview of the JasperMobile apps for Android and iOS devices.

This guide is intended for solution architects who want to understand mobile BI and want to design a mobile application using Jaspersoft BI products. It is also intended for programmers who implement the mobile application.

The procedures in this document assume you are familiar with JasperReports Server installation, deployment, and administration. You must have system administrator privileges within JasperReports Server and its application server, as well as read and write access to their files on the host.

This document also assumes you are familiar with the various programming languages and file formats that it uses, including Java, JavaScript, XML, and HTML.

This chapter contains the following sections:

1.1 System Requirements

Before getting started with the Jaspersoft Mobile SDK for Android, take a moment to confirm that your development computer meets the following system requirements.

Supported Operating Systems
Windows XP (32-bit), Windows Vista (32- or 64-bit), or Windows 7 (32- or 64-bit)
Mac OS X 10.5.8 or later (x86 only)
Linux: • Tested on Ubuntu Linux, Lucid Lynx • On Ubuntu Linux, version 8.04 or later is required • GNU C Library (glibc) 2.7 or later is required • 64-bit distributions must be capable of running 32-bit applications (see the Ubuntu Linux installation notes)

The Jaspersoft Mobile SDK for Android can be used with the following integrated development environments (IDEs):

Supported IDEs
Eclipse 3.5 (Galileo) or later
Intellij IDEA 9 (Maia), though 11 has better integration (for more information, see the IDEA documentation)

For development, the following packages are also required:

Development Environment
Java Development Kit (JDK) 5 or 6; the Java Runtime Environment (JRE) is not sufficient
Apache Ant 1.8 or later
Google Android 2.1 (API 7) SDK Platform or later

At runtime, Android apps based on the Jaspersoft Mobile SDK for Android connect to the following servers:

Server Compatibility
JasperReports Server Community Project 4.5.0, after applying a compatibility hotfix available at: http://community.jaspersoft.com/project/mobile-sdk-android/releases
JasperReports Server Community and Commercial releases 4.5.1 or later

1.2 Setting Up the Google Android SDK

If you have not done so already, you need to install and configure the Google Android SDK.

The Google Android SDK uses a modular structure that separates the major parts of the SDK into separately installable components:

- Starter Package
- Android platform versions
- Add-ons
- Tools
- Samples
- Documentation

1.2.1 Installing the Starter Package

The Google Android SDK Starter Package is the only component that you download from the Android developer website:

1. Download the latest Starter Package for your platform from http://developer.android.com/sdk/index.html.
2. Run the installer or unpack the archive file. For more information, see the Android Installing the SDK page.
3. Make a note of the name and location of the Google SDK directory on your system, for example C:\Program Files\Android\android-sdk. This document refers to this location as <GoogleSDK>.

The Google Android SDK Starter Package is not a full development environment. It includes only the core SDK Tools that you can use to download the rest of the SDK components, such as the latest Android platform.

1.2.2 Running the Android SDK Manager

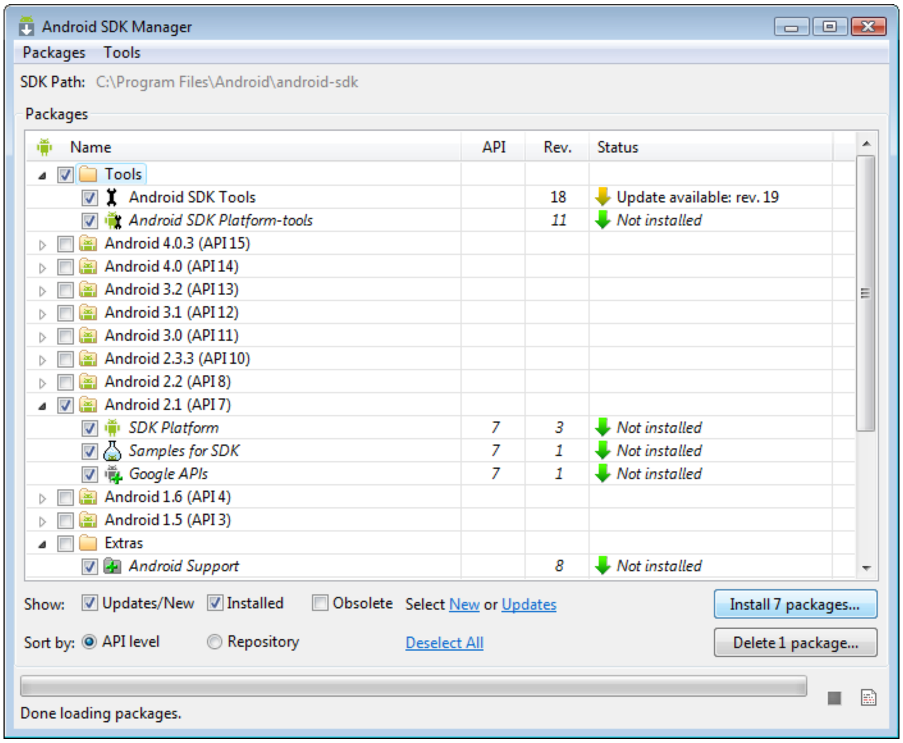
To develop an Android application, you also need to download at least one Android platform version and the associated platform tools. You can add other components and platforms as well, which is highly recommended.

To download essential platform versions and SDK components into your development environment, use the Android SDK and Android virtual device (AVD) Manager.

1. Launch the Android SDK and AVD Manager in one of the following ways:

IDEA:	**Tools > Android > AVD Manager**
Eclipse:	**Window > Android SDK and AVD Manager**
Windows:	`<GoogleSDK>\SDK Manager.exe` (launched automatically after the Google Android SDK installer)
Mac or Linux:	`<GoogleSDK>/tools/android`

2. Use the graphical UI of the Android SDK and AVD Manager to browse the SDK repository and select new or updated components, as shown in the following figure:



SDK Path: C:\Program Files\Android\android-sdk

Packages

Name	API	Rev.	Status
☑ Tools			
☑ Android SDK Tools		18	Update available: rev. 19
☑ *Android SDK Platform-tools*		*11*	*Not installed*
□ Android 4.0.3 (API 15)			
□ Android 4.0 (API 14)			
□ Android 3.2 (API 13)			
□ Android 3.1 (API 12)			
□ Android 3.0 (API 11)			
□ Android 2.3.3 (API 10)			
□ Android 2.2 (API 8)			
☑ Android 2.1 (API 7)			
☑ *SDK Platform*	*7*	*3*	*Not installed*
☑ *Samples for SDK*	*7*	*1*	*Not installed*
☑ *Google APIs*	*7*	*1*	*Not installed*
□ Android 1.6 (API 4)			
□ Android 1.5 (API 3)			
□ Extras			
☑ *Android Support*		*8*	*Not installed*

Show: ☑ Updates/New ☑ Installed □ Obsolete Select New or Updates [Install 7 packages...]

Sort by: ● API level ○ Repository Deselect All [Delete 1 package...]

Done loading packages.

Figure 1-1 The Google Android SDK and AVD Manager's Available Packages Panel

3. Upon first installation, select the components described in the following table:

SDK Component	Description
SDK Tools	If you've just installed the SDK Starter Package, then you already have the latest version of this component. The SDK Tools component is required to develop an Android application. Make sure you keep this up to date.
SDK Platform-tools	This includes more tools that are required for application development. These tools are platform-dependent and typically update only when a new SDK platform is made available, in order to support new features in the platform. These tools are always backward compatible with older platforms, but you must be sure that you have the latest version of these tools when you install a new SDK platform.
Android 2.1 (API 7) SDK Platform	You need to download at least one platform into your environment, and this is the minimum API level that the Jaspersoft Mobile Android SDK is compatible with. You can download and install any other platforms as well, so that you can test your application on the full range of Android platform versions that your application will support.

4. Install the selected packages. The SDK Manager downloads the components and installs them in your SDK environment.
5. You should periodically run the SDK Manager to look for and install updates to the components you use.

For more information about using the Android SDK and AVD Manager, see the Android Adding SDK Components page.

1.3 Setting Up the Jaspersoft Mobile SDK for Android

Once you've installed and set up the basic configuration of SDK components and platforms, you're ready to download the Jaspersoft Mobile SDK for Android.

1. Download the Mobile SDK from the Jaspersoft Community website: http://community.jaspersoft.com/project/mobile-sdk-android
2. The archive includes two base packages that you can include in your project:
    - `com.jaspersoft.android.sdk.client` – Provides a REST client to simplify the work with JasperReports Server REST API, a mapping of returned data to the object model, a wrapper class to work more conveniently with input controls, and an improved asynchronous tasks mechanism. You can include this package in your project as a JAR file.
    - `com.jaspersoft.android.sdk.ui` – Provides a set of predefined Android adapters and layouts that help with handling data through REST and simplify UI creation. You can include this package in your project as an Android Library Project.
3. The most convenient way is using the Android Maven Plugin to manage and build your Android project. In this case you can include Jaspersoft's Maven artifacts in your .pom file just as a dependencies:

```
<dependency>
  <groupId>com.jaspersoft.android.sdk</groupId>
  <artifactId>js-android-sdk-client</artifactId>
  <version>1.0</version>
</dependency>

<dependency>
  <groupId>com.jaspersoft.android.sdk</groupId>
  <artifactId>js-android-sdk-ui</artifactId>
  <version>1.0</version>
  <apklib>
</dependency>
```

4. These maven dependencies are in the JasperForge Maven repository. You can add the following definition to your pom file to use it:

```
<repositories>
  <repository>
    <id>Jaspersoft Artifactory</id>
<url>http://jaspersoft.artifactoryonline.com/jaspersoft/jaspersoft-mobile-sdk/</url>
    <snapshots><enabled>false</enabled></snapshots>
    <releases><enabled>true</enabled></releases>
  </repository>
</repositories>
```

For more information about the Jaspersoft Mobile SDK for Android, see the project pages on the community website. For more information about using Maven, see Android Application Development with Maven.

1.4 Structure of the Mobile SDK for Android

The Jaspersoft Mobile SDK for Android is a set of Java classes and UI components to easily connect and consume the services provided by JasperReports Server using the REST API. The Mobile SDK uses the version of the REST API available in JasperReports Server 4.5.1 or later. The following documentation is available on the Mobile SDK release page:

- *JasperReports Server Web Services Guide*
- Javadoc for the Mobile SDK packages

The Mobile SDK is divided into a client package and a user interface (UI) package.

1.4.1 Mobile SDK Client Package

The Mobile SDK client package includes all the classes to interact with JasperReports Server through its REST API. The following diagram shows the relationship between the classes of the client package.

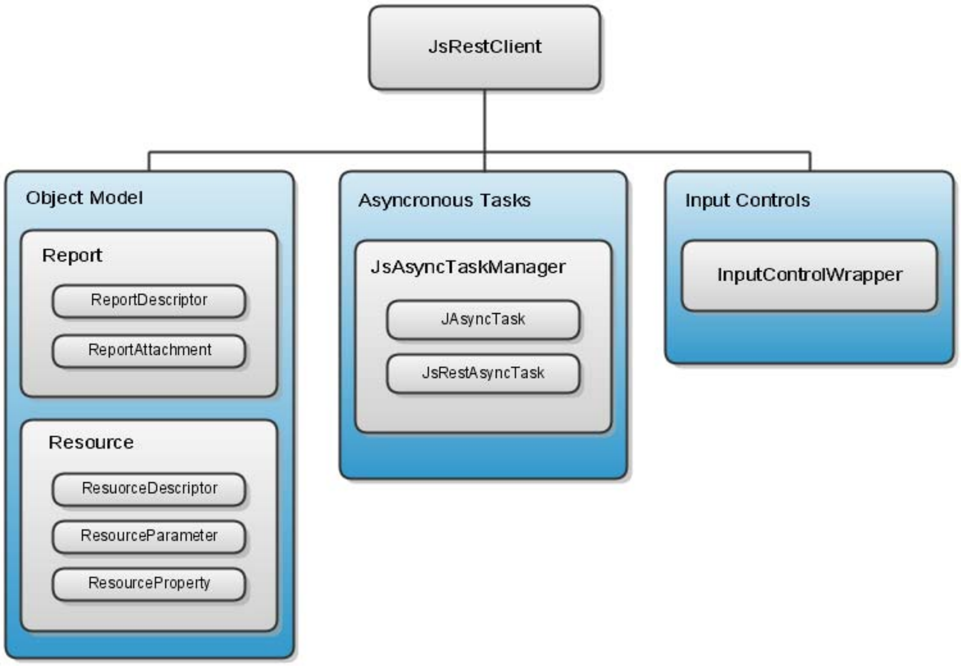


Figure 1-2 Client Package of the Jaspersoft Mobile SDK for Android

The components of the client package are:

- `JsRestClient` – Provides a set of convenient methods to interact with the JasperReports Server REST API and performs the mapping of the returned data to the object model.
- Object model – A set of classes that represent objects in the server. These classes make it simpler to work with REST data, which can be returned in JSON or XML format.
- Asynchronous tasks mechanism – Performs expensive operations in background mode and publishes the results to the UI thread without having to manipulate threads or handlers.
- Input controls wrapper – Simplifies the manipulation of input controls, independent of their type and UI appearance.

1.4.2 Mobile SDK UI Package

The following diagram shows the components of the Mobile SDK UI package. These predefined UI components can be reused in the creation of the UI for your client app:

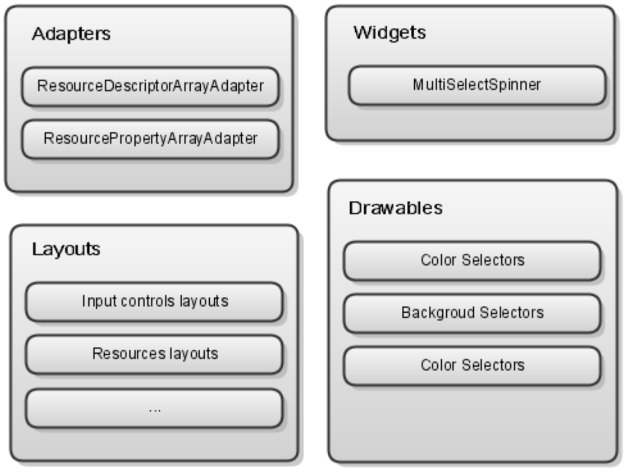


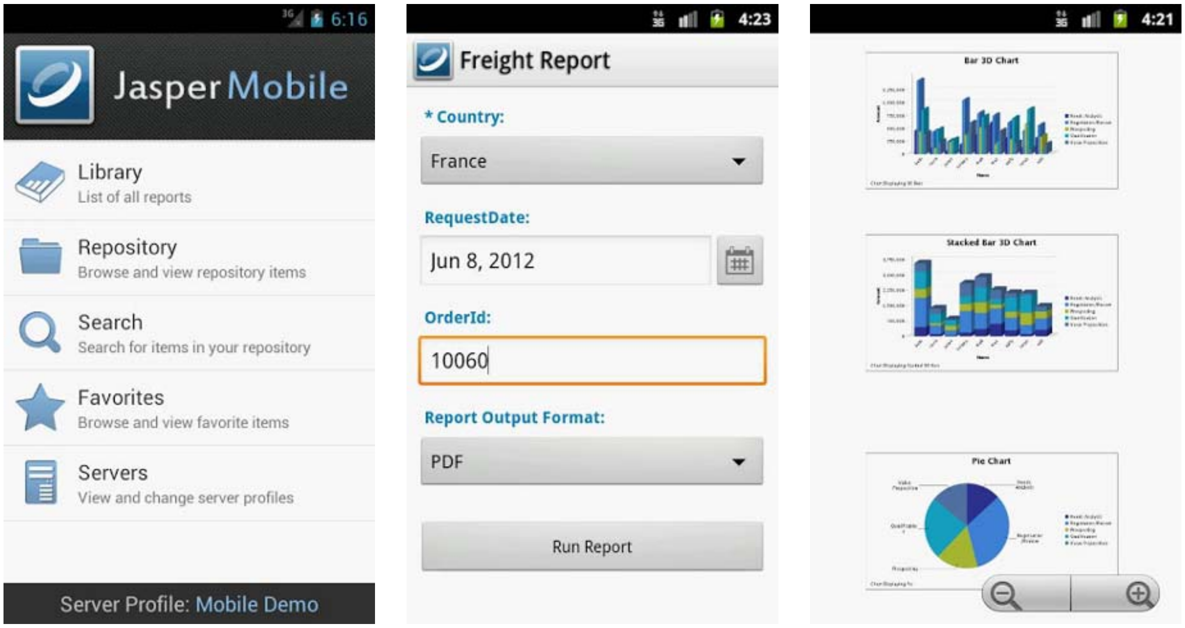
Figure 1-3 UI Packages of the Jaspersoft Mobile SDK for Android

1.5 JasperMobile App for Android

Jaspersoft has published the JasperMobile app for Android that was developed with the Jaspersoft Mobile SDK. You can install the app from the Google Play website: http://play.google.com/store/apps/details?id=Jaspesoft%20BI.

The JasperMobile app for Android is a client that lets you access one or more JasperReports Server instances from your Android device. This app uses both the client and UI packages of the Mobile SDK to implement the following features:

- Repository Browser – Explore the repository of JasperReports Server and view all the resource properties. A built-in viewer lets you display image resources and the source code of text files like JRXML, resource bundle and CSS.
- Repository Search – Search for resources by name in the whole repository.
- Input Controls – All the JasperReports Server input controls are rendered using native components. The large set of input controls include boolean checks, text and numeric fields, date and time selectors, single and multiple choice from lists based on queries or static items.
- Running reports – Jaspersoft Mobile allows you to easily execute reports. The application introspects the report unit, identifies the input controls and then renders them in a table view based on the report parameters. The support for cascading input controls allows you to dynamically populate list items based on end user selection.
- Favorites – When browsing the repository, any resource can be easily flagged as a 'Favorite' and then displayed in the special Favorites view.



Server Profile: Mobile Demo

* Country: France

RequestDate: Jun 8, 2012

OrderId: 10060

Report Output Format: PDF

Figure 1-4 Home and Report interaction Screens of the JasperMobile App For Android

Jaspersoft has released the source code to the JasperMobile app as open source. You can download this code for a complete example of how to use the Mobile SDK and as a basis for developing your own apps. The source code is available from the JasperMobile for Android project page on Jaspersoft's Community website (http://community.jaspersoft.com/project/mobile-sdk-android).

CHAPTER 2 GETTING STARTED WITH JASPERSOFT MOBILE SDK FOR iOS

The Jaspersoft Mobile SDK for iOS provides a library of classes to write iPhone, iPad, and iPod Touch apps that access JasperReports Server and display reports. It allows you create reporting apps or to seamlessly embed reporting features into your own mobile apps. Jaspersoft also provides the JasperMobile app for iOS, a client app that can browse the repository, display reports, and store a list of favorite reports.

Use the Mobile SDK and JasperMobile app to put advanced and interactive reporting at your customers' fingertips, wherever and whenever they need it.

This chapter documents version 1.5 of the Jaspersoft Mobile SDK for iOS.

This chapter contains the following sections:

- **System Requirements**
- **Downloading the Mobile SDK**
- **Choosing your Installation**
- **Installing by Linking**
- **Using Your Own Version of RestKit Library**
- **Installing by Adding the Compiled Library**
- **Structure of the Mobile SDK for iOS**
- **JasperMobile App for iOS**

2.1 System Requirements

Before getting started with the Jaspersoft Mobile SDK for iOS, take a moment to confirm that your development computer meets the following system requirements.

Supported Operating Systems
Mac OS X

As with all iOS libraries, the Jaspersoft Mobile SDK for iOS is supported only in Xcode:

Supported IDE
Xcode

For development, the following packages are also required:

Development Environment
RestKit library (versions 0.10.1 - 0.10.3)

At runtime, iOS apps based on the Jaspersoft Mobile SDK for iOS can connect to the following servers:

Server Compatibility
JasperReports Server Community and Commercial releases 4.5.1 or later

2.2 Downloading the Mobile SDK

Download the Mobile SDK from the Jaspersoft Community website: http://community.jaspersoft.com/project/mobile-sdk-ios.

Unzip the downloaded file into a folder, and open this folder in Finder.

The following documentation is also available on the download page (http://community.jaspersoft.com/project/mobile-sdk-ios/releases):

- *JasperReports Server Web Services Guide*
- Standalone installation guide for the Mobile SDK packages (same information as presented here)

2.3 Choosing your Installation

There are three ways to install the Jaspersoft Mobile SDK for iOS in Xcode:

- Recommended – Linking to it as a static library (iOS) or framework (OS X).
- Alternate – Adding the compiled library to your project.
- Not recommended – Adding it as source code to your project.

Installation by static library linking is slightly longer, but it is the recommended way to install the Mobile SDK because of the following benefits:

- It avoids classname collisions (when different classes have the same name).
- It is independent of ARC (automatic reference counting) that was introduced in Xcode 4.2.
- Re-installing or updating the Mobile SDK is simple.

If you don’t want to link the Jaspersoft Mobile SDK you can install it as a compiled library. The benefits of this approach are:

- All your project files are in one place, without the need to add them separately.
- The installation is simpler.
- Re-installing or updating the Mobile SDK is simple.

However, when installed by adding the compiled library, you may have the following issues:

- You will not have direct access to SDK library source code, therefore you cannot modify it or debug in runtime.
- Class name collisions are possible because Objective-C does not have namespaces.
- The SDK was written for an iOS version earlier than 4.2 and does not use ARC (automatic reference counting).

In previous versions of Jaspersoft Mobile SDK for iOS, the installation instructions suggested copying the SDK classes and all of the dependent libraries as source code directly into your Xcode project. This created complications when upgrading to newer versions, and it is no longer recommended. This chapter documents the first two installation methods.

2.4 Installing by Linking

Jaspersoft recommends installing by linking to the Mobile SDK as a static library (iOS) or Framework (OS X). It requires a few more steps than including the compiled library in your project, but the advantages make it worth the effort. For more information, see section **2.3, "Choosing your Installation," on page 14**. The following procedure gives all of the steps:

1. Copy the unzipped Mobile SDK to a permanent location.
2. Add a cross-project reference by dragging the jaspersoft-sdk.xcodeproj file from the permanent location into your open application project in Xcode:

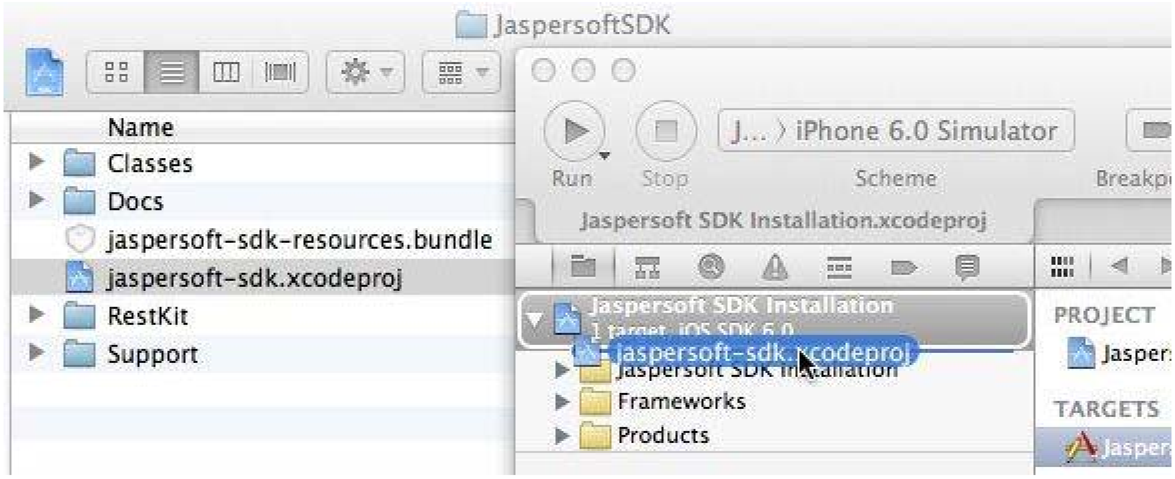


Figure 2-1 Adding a Cross-Project Reference to the Jaspersoft Mobile SDK

3. Delete any pre-existing references or copies of jaspersoft-sdk-resources.bundle in your project.
4. Add the SDK resource bundle reference by dragging the jaspersoft-sdk-resources.bundle file into the Resources directory of your open application project in Xcode. You can drop it into any directory, but Resources makes sense in this context.

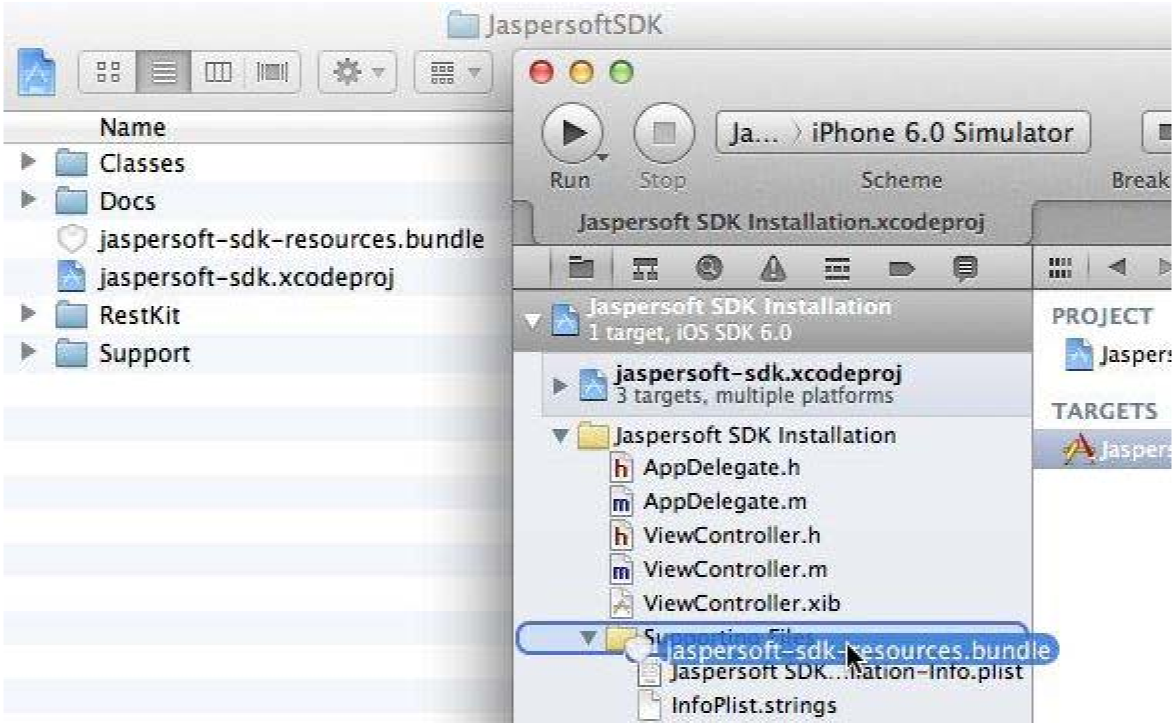
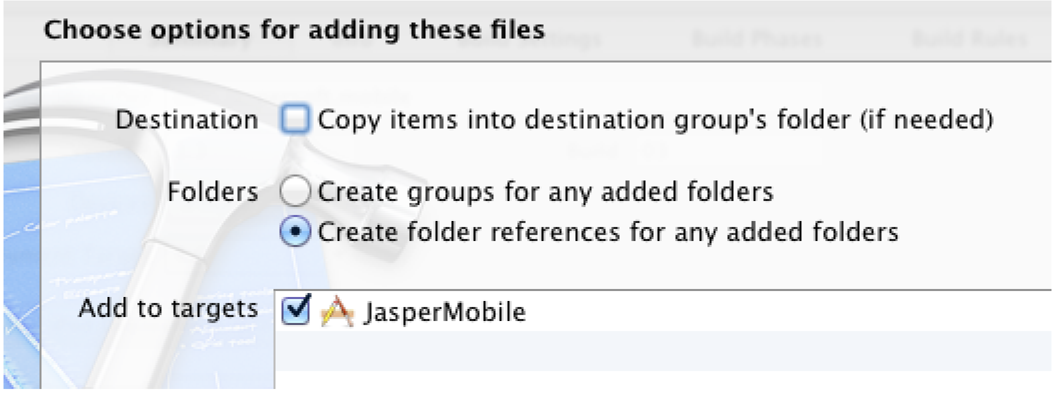


Figure 2-2 Adding the Jaspersoft Mobile SDK Resource Bundle

5. When the options dialog appears, leave the **Destination** box clear. Under **Folders**, select **Create folder references for any added folders**, and make sure there is a check beside your project in the **Add to targets** list:



Choose options for adding these files

Destination □ Copy items into destination group's folder (if needed)

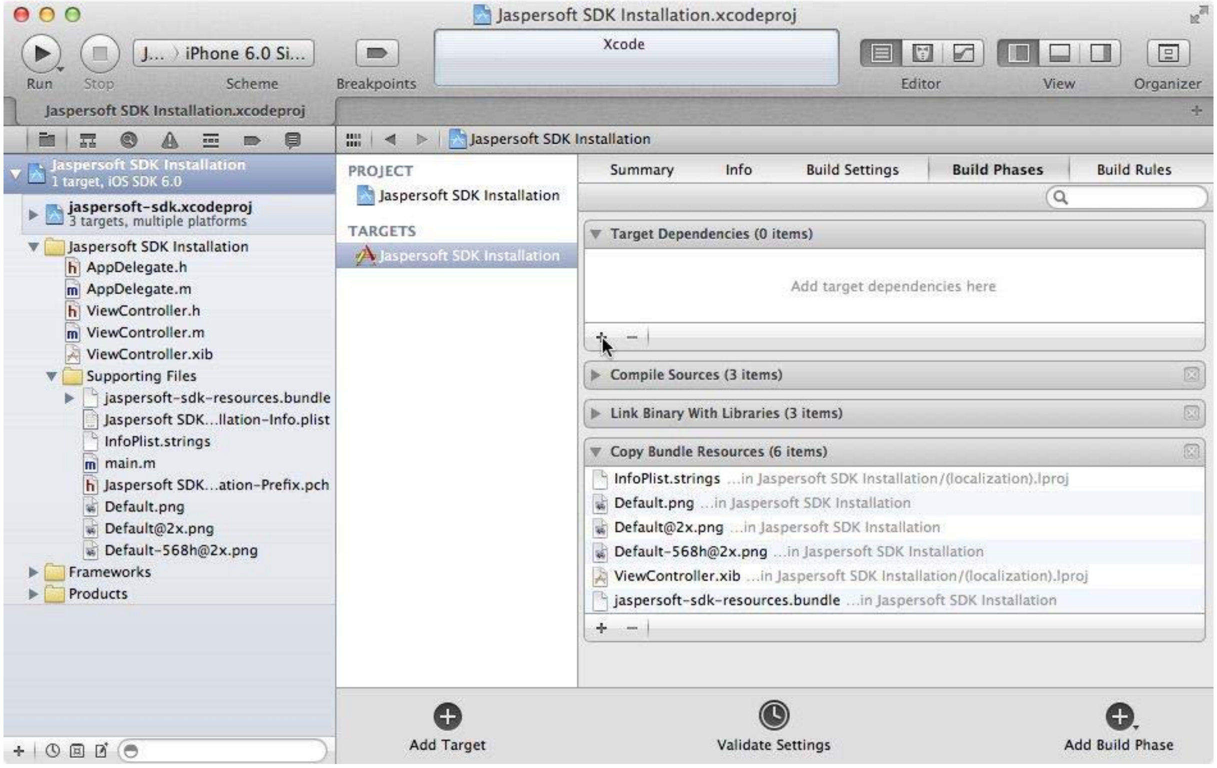
Folders ○ Create groups for any added folders
● Create folder references for any added folders

Add to targets ☑ JasperMobile

Figure 2-3 Options for Adding the Jaspersoft Mobile SDK Resource Bundle

6. Open the target settings editor, select the target that you want the Mobile SDK to link with, and select its **Build Phases** tab.

7. In the **Target Dependencies** list, click **+** to add a dependency on the jaspersoft-sdk (iOS) or jaspersoft-sdk-framework (OS X) aggregate target.



Jaspersoft SDK Installation.xcodeproj
Run
Stop
J... › iPhone 6.0 Si...
Scheme
Breakpoints
Xcode
Editor
View
Organizer
Jaspersoft SDK Installation.xcodeproj
Jaspersoft SDK Installation
Jaspersoft SDK Installation
1 target, iOS SDK 6.0
jaspersoft-sdk.xcodeproj
3 targets, multiple platforms
Jaspersoft SDK Installation
AppDelegate.h
AppDelegate.m
ViewController.h
ViewController.m
ViewController.xib
Supporting Files
jaspersoft-sdk-resources.bundle
Jaspersoft SDK...llation-Info.plist
InfoPlist.strings
main.m
Jaspersoft SDK...ation-Prefix.pch
Default.png
Default@2x.png
Default-568h@2x.png
Frameworks
Products
PROJECT
Jaspersoft SDK Installation
TARGETS
Jaspersoft SDK Installation
Summary
Info
Build Settings
Build Phases
Build Rules
Target Dependencies (0 items)
Add target dependencies here
Compile Sources (3 items)
Link Binary With Libraries (3 items)
Copy Bundle Resources (6 items)
InfoPlist.strings ...in Jaspersoft SDK Installation/(localization).lproj
Default.png ...in Jaspersoft SDK Installation
Default@2x.png ...in Jaspersoft SDK Installation
Default-568h@2x.png ...in Jaspersoft SDK Installation
ViewController.xib ...in Jaspersoft SDK Installation/(localization).lproj
jaspersoft-sdk-resources.bundle ...in Jaspersoft SDK Installation
Add Target
Validate Settings
Add Build Phase

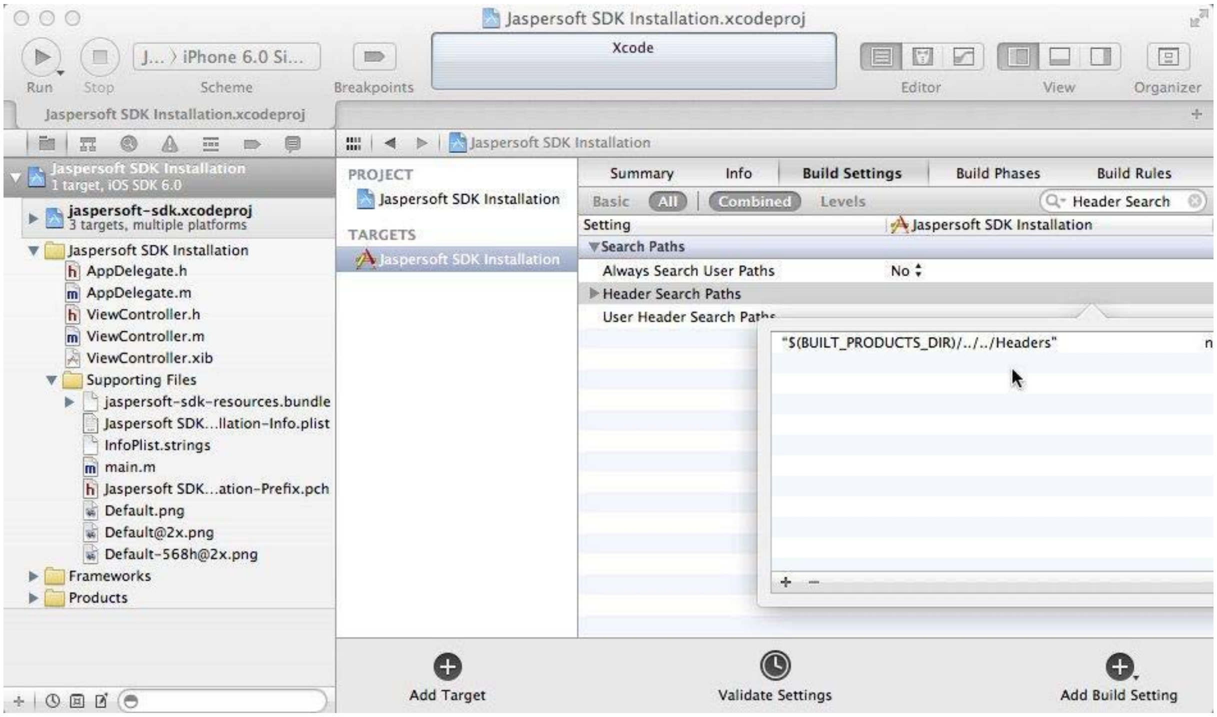
Figure 2-4 Adding Target Dependencies

8. In the **Link Binary With Libraries** list, add a link to the following frameworks and libraries according to your platform:

iOS Targets	OS X Targets
libjaspersoft-sdk.a (normally shows up in red)	jaspersoft-sdk.framework (normally shows up in red)
CFNetwork.framework	
CoreData.framework	CoreData.framework
Security.framework	Security.framework
MobileCoreServices.framework	CoreServices.framework
SystemConfiguration.framework	SystemConfiguration.framework
libxml2.dylib	libxml2.dylib
QuartzCore.framework	

9. On the **Build Settings** tab, expand **Search Paths > Header Search Paths**, and double-click on the value column at the top level. If it’s not already present, add the following path:

```
$(BUILT_PRODUCTS_DIR)/../../Headers
```



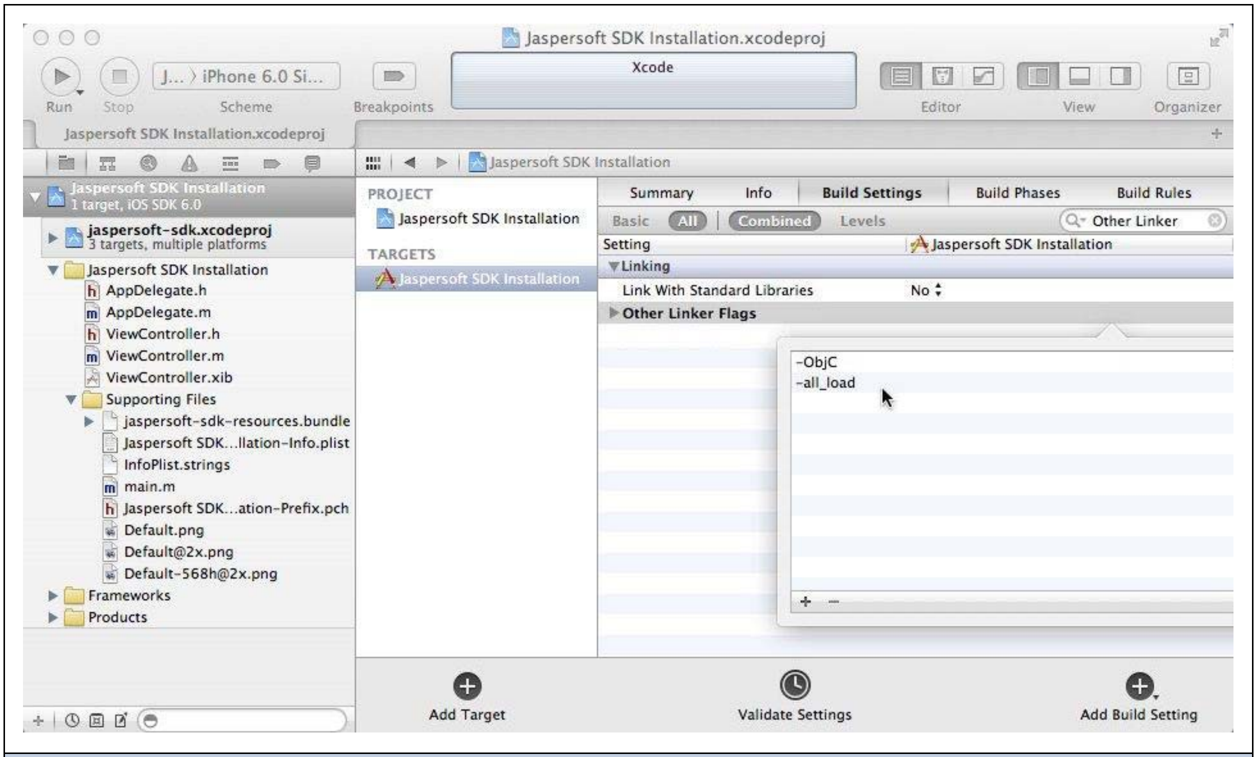
Search Paths
Always Search User Paths No
Header Search Paths
User Header Search Paths
"$(BUILT_PRODUCTS_DIR)/../../Headers"

Figure 2-5 Adding Header Search Paths

10. On the **Build Settings** tab, expand **Linking > Other Linker Flags**, and if not already present, add the following flags:

```
-ObjC -all_load
```

Make sure these are properly separated into 2 flags.



Linking

Link With Standard Libraries No

Other Linker Flags

-ObjC
-all_load

Figure 2-6 Adding Other Linker Flags

11. You have now completed the installation of the Jaspersoft Mobile SDK for iOS into your application. To verify the installation, open up your App Delegate and add an import of the `JaspersoftSDK` header (already included in the JasperMobile app for iOS):

    ```
    #import <jaspersoft-sdk/JaspersoftSDK.h>
    ```

12. Build your project and verify the build output. The project should build without any issues, demonstrating that the Jaspersoft Mobile SDK library was linked successfully.

2.5 Using Your Own Version of RestKit Library

If you have already installed the RestKit library, you can disable it from being linked by the Jaspersoft Mobile SDK:

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

The Jaspersoft Mobile SDK for iOS will now build without including Restkit directly into your library file.

2.6 Installing by Adding the Compiled Library

In this type of installation, you compile the Mobile SDK and include the binary in your iOS app’s compiled code. For a list of advantages and disadvantages of this installation method, see section **2.3, “Choosing your Installation,” on page 14**.

1. Download and unzip the Jaspersoft Mobile SDK for iOS, as described in section **2.2, “Downloading the Mobile SDK,” on page 14**.
2. Open the jaspersoft-sdk.xcodeproj file in Xcode.

3. Click the scheme and select **jaspersoft-sdk-universal** as the current scheme. The type of device doesn't matter, it can be **iOS Device** or **iPhone/iPad Simulator**.

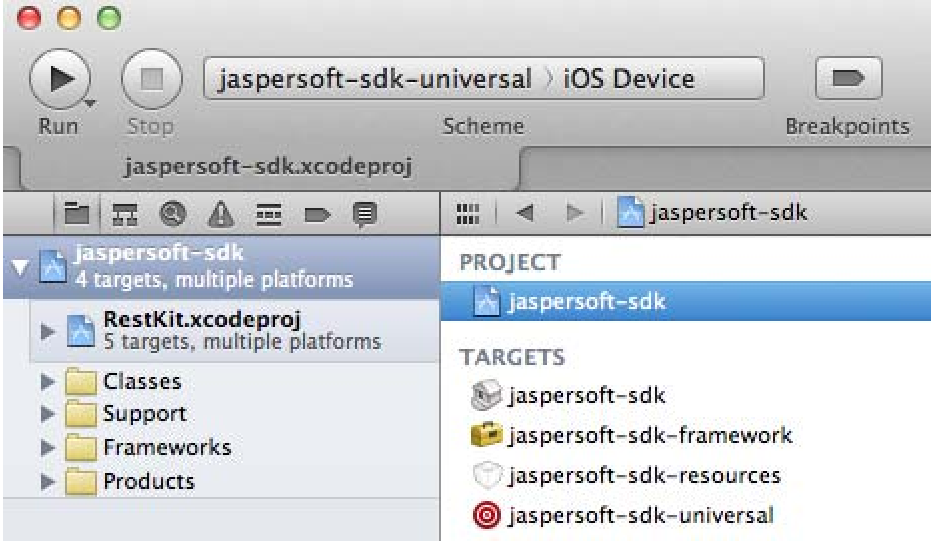


Figure 2-7 Selecting the Universal Scheme

4. Build the Jaspersoft Mobile SDK project.
5. Open your application project in Xcode.
6. In Finder, open the JaspersoftSDK folder and drag the jaspersoft-sdk-lib folder into your own project.

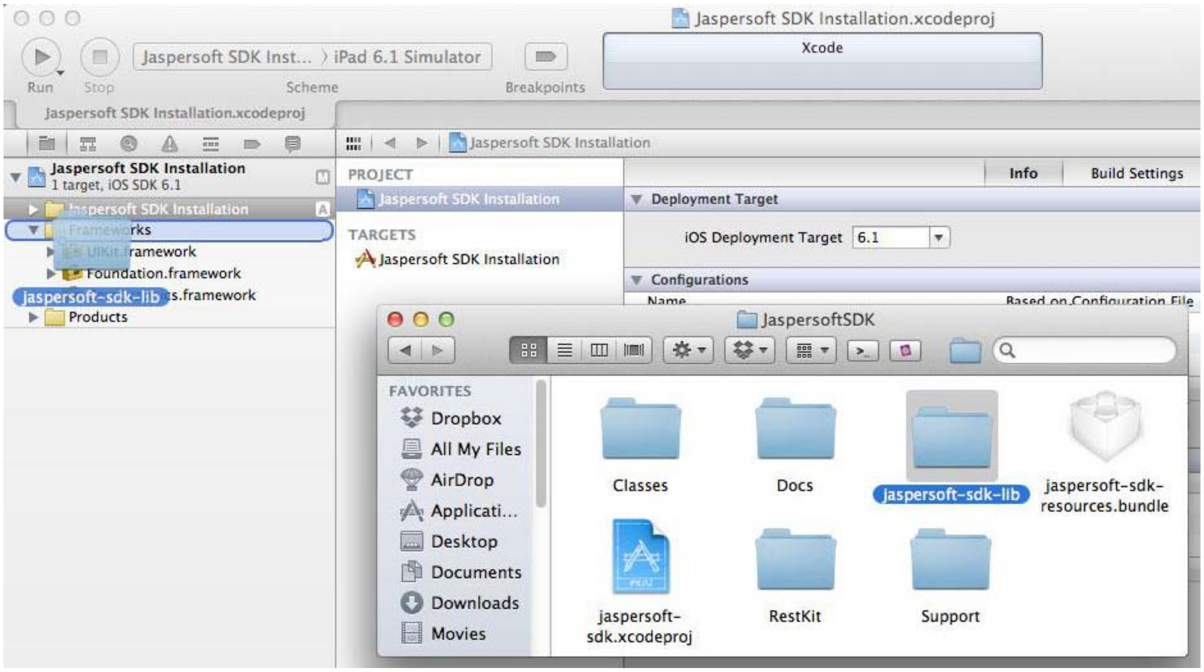
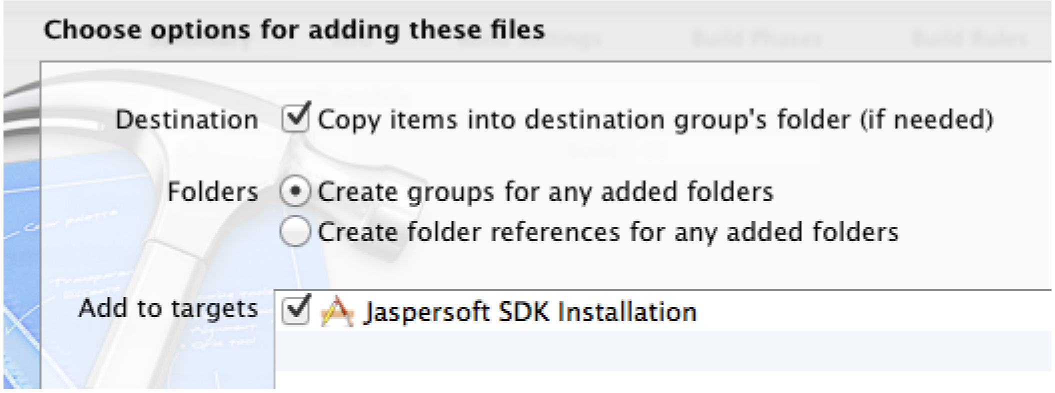


Figure 2-8 Adding the Compiled Library to Your Project

7. When the options dialog appears, leave the **Destination** box checked to copy items. Under **Folders**, select **Create groups for any added folders**, and make sure there is a check beside your project in the **Add to targets** list.



Choose options for adding these files

Destination ☑ Copy items into destination group's folder (if needed)

Folders ● Create groups for any added folders
○ Create folder references for any added folders

Add to targets ☑ Jaspersoft SDK Installation

Figure 2-9 Options for Adding the Compiled Library

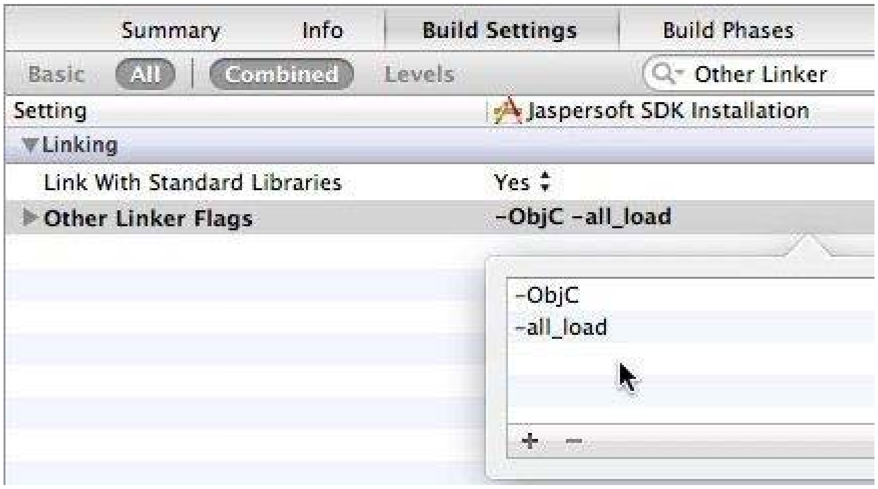
8. Open the target settings editor, and select the **Build Phases** tab.
9. In the **Link Binary With Libraries** list, add a link to the following frameworks and libraries according to your platform:

iOS Targets	OS X Targets
CFNetwork.framework	
CoreData.framework	CoreData.framework
Security.framework	Security.framework
MobileCoreServices.framework	CoreServices.framework
SystemConfiguration.framework	SystemConfiguration.framework
libxml2.dylib	libxml2.dylib
QuartzCore.framework	

10. On the **Build Settings** tab, expand **Linking > Other Linker Flags**, and if not already present, add the following flags:

    `-ObjC -all_load`

    Make sure these are properly separated into 2 flags.



Setting	Jaspersoft SDK Installation
▼Linking	
Link With Standard Libraries	Yes
▶Other Linker Flags	-ObjC -all_load

-ObjC
-all_load

Figure 2-10 Adding Other Linker Flags

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

    ```
    #import "JaspersoftSDK.h"
    ```

12. Build your project and verify the build output. The project should build without any issues, demonstrating that the Jaspersoft Mobile SDK library was added successfully.

2.7 Structure of the Mobile SDK for iOS

The Jaspersoft Mobile SDK for iOS is a set of classes and UI components to easily connect and consume the services provided by JasperReports Server using the REST API. The Mobile SDK uses the version of the REST API available in JasperReports Server 4.5.1 or later.

The following diagram shows the relationship between the classes.

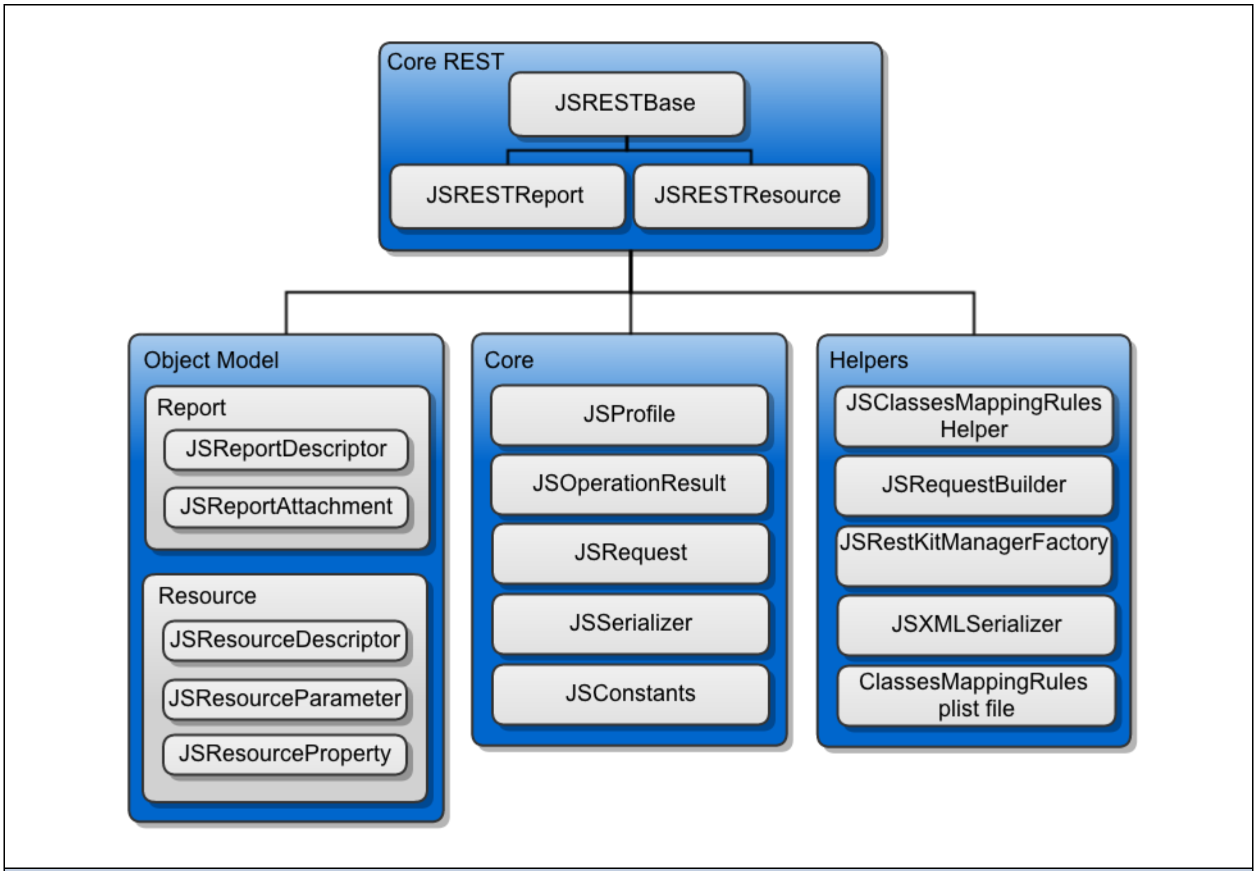


Figure 2-11 Components of the Jaspersoft Mobile SDK for iOS

2.8 JasperMobile App for iOS

Jaspersoft has published the JasperMobile app for iOS that was developed with the Jaspersoft Mobile SDK. You can install the app from the iTunes Store: https://itunes.apple.com/us/app/jaspermobile/id467317446?l=it&ls=1&mt=8.

The JasperMobile app for iOS is a client that lets you access one or more JasperReports Server instances from your iPhone, iPad, or iPod Touch. This app uses both the client and UI packages of the Mobile SDK to implement the following features:

- Repository Browser – Explore the repository of JasperReports Server and view all the resource properties. New icons provide visual cues about resource types (folders, reports, and more).
- Repository Search – Search for resources by name in the whole repository.
- Input Controls – All the JasperReports Server input controls are rendered using native components. The large set of input controls include boolean checks, text and numeric fields, date and time selectors, single and multiple choice from lists based on queries or static items.
- Running reports – Jaspersoft Mobile allows you to easily execute reports. The application introspects the report unit, identifies the input controls and then renders them in a table view based on the report parameters. The support for cascading input controls allows you to dynamically populate list items based on end user selection.
- Favorites – When browsing the repository, any resource can be easily flagged as a 'Favorite' and then displayed in the special Favorites view.

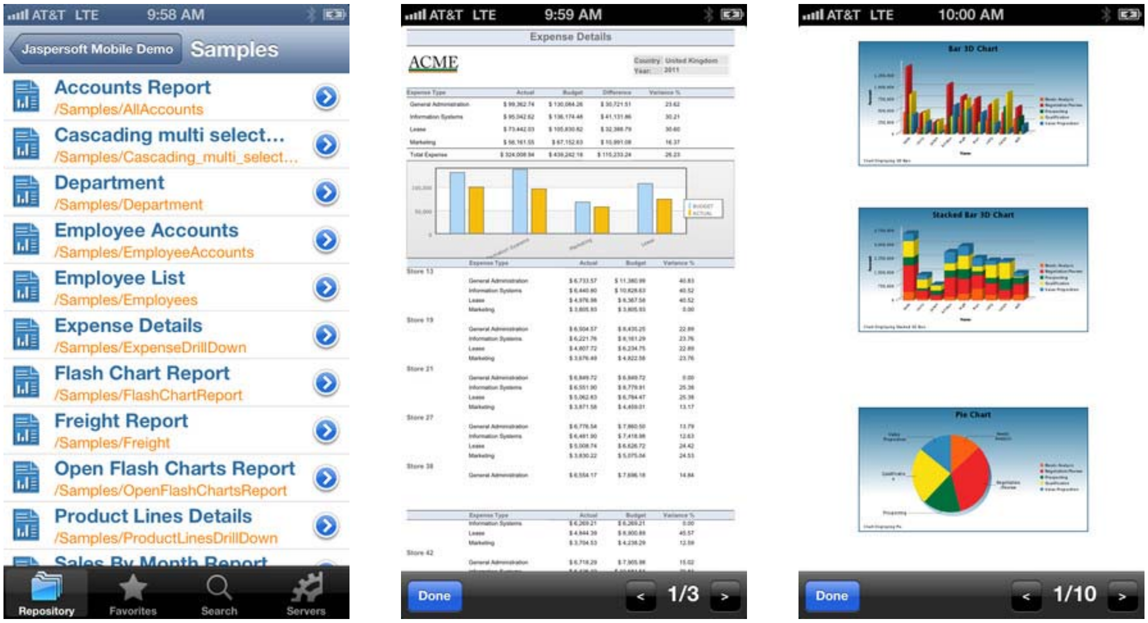


Figure 2-12 Repository and Charts Displaying in the JasperMobile App for iOS