SOTI

MobiControl SDK for iOS Integration Guide (v13.4.5)

November 2022

SOTI

Table of Contents

Introduction

The MobiControl SDK for iOS (herein referred to “the SDK”) features an easy yet powerful method of incorporating features of the MobiControl iOS App (found in the App Store) such as Remote View, File Sync, and Content Library into your own enterprise or App Store applications.

This flexibility allows your organization to leverage the MobiControl technology and features within your own Apps to reduce the cost of developing such functionality yourself while maintaining consistent company branding and deployment options for your App. An SDK-enabled application is intended for mobile end users and requires the use of a MobiControl *backend* server.

The key features of the SDK are as follows:

- Remote View
- Obtaining Device Information
- File Sync
- Content Library
- Location Services
- Application Catalog
- Jailbreak Detection

This document serves as a guide for the integration of the SDK by referencing common scenarios, external tools, development documentation and other resources. MobiControl system administrators deploying an SDK-enabled application, and developers integrating the SDK are suitable audiences for this guide.

Integrating the SDK into your Application

Minimum Requirements

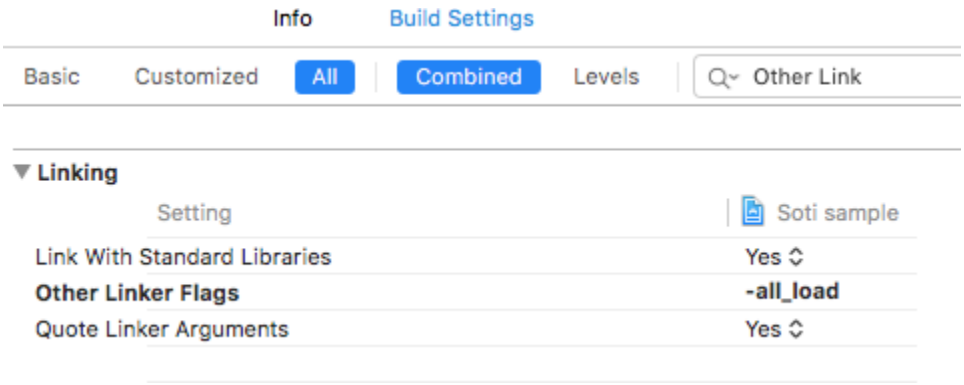
The following highlights the ecosystem requirements for developing and deploying an SDK-enabled application.

- MobiControl *backend* v13.0 and greater
- SDK-enabled applications can be deployed on devices running iOS 9 and greater
- SDK-enabled applications must be able to communicate to the MobiControl backend over TCP 443 and 5494, and to mc-enroll.soti.net over TCP 443
- Membership in one of the Apple Developer Programs

Setting up your Xcode Project

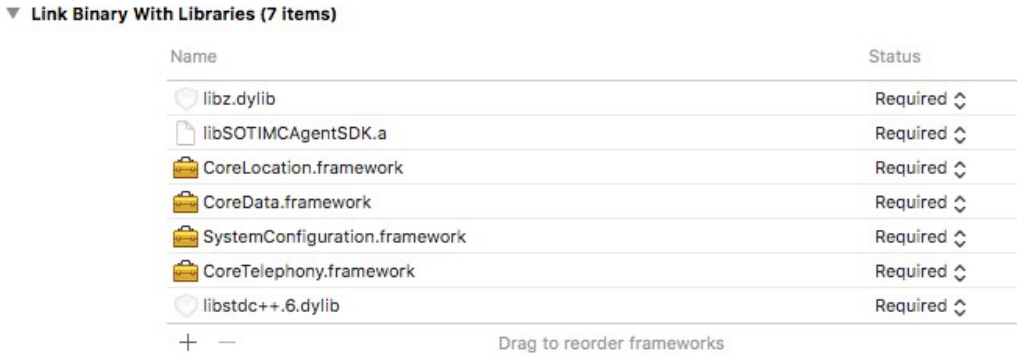
The first step of integrating the SDK is to configure your Xcode project with a link to the SDK, and configure any other required parameters for building the project. The following assumes you have an existing project with default settings.

1. Set **Build Settings > Linking > Other Linker Flags** to **–all_load** or **force_load.** This is required due to unexpected Xcode behavior that prevents -ObjC from loading object files from static libraries that contain only categories and no classes.



2. Configure **Build Settings > Apple LLVM 8.1 – Language C++ > C++ Language Dialect** and **C++ Standard Library** to **Compiler Default.**

3. Set **General > Linked Frameworks and Libraries** to include the following required frameworks, and link the SDK binary:
    a. libstdc++.6.dylib
    b. libz.dylib
    c. CoreLocation.framework
    d. CoreData.framework
    e. SystemConfiguration.framework
    f. CoreTelephony.framework
    g. libSOTIMCAgentSDK.a



4. Add Files: SDKResources.bundle
5. Copy Build Resources > Build Phases

Establishing a Connection to the MobiControl Deployment Server

Each SDK-enabled application will connect to one MobiControl environment at a time. This connection is persistent and is established over TCP 5494. At the time of compilation however, the SDK-enabled App will not know the address of the deployment server it will communicate with. Instead it will rely on an iOS feature called App Config to provide the necessary connection properties.

App Config works only when the device is enrolled to MobiControl, and when the SDK-enabled application is installed and “Managed” by MobiControl. Refer to “Deploying your Application” for more details.

In your Xcode project, continue your SDK integration by including the “AgentSDK.h” header and “enrolling” the SDK to the deployment server using the `connectToDSUsingAppConfig` method. This method will read and store the connection properties obtained through App Config and attempt the first connection. Register a delegate to receive callbacks for connections by using `connectionStatusUpdate`.

The SDK is considered “enrolled” after its first connection (`kConnectionStatusOK`) and association with the corresponding device record in MobiControl.

NOTE:	If from the MobiControl web console, the installation status of the SDK-enabled application on the target device is anything other than “Managed” the SDK will not be able to retrieve the connection parameters from App Config.

The SDK will automatically make subsequent connections when the SDK-enabled App is in the foreground. `connectToDS` and `disconnect` can be used to manually connect and disconnect the SDK’s connection to the deployment server as required.

Server Authentication

The SDK will perform validation of the MobiControl deployment server it is connecting to by verifying the deployment server certificate presented in the connection. Certificate validity is determined by verifying the certificate’s expiration date, hostname, and performing chain validation against the device’s list of known roots.

An iOS device does not have inherit trust of the standard MobiControl Root generated at time of installation and will always fail this validation. To work around this problem the MobiControl Root must be included in the App Config connection properties as described in “Deploying your Application” > “App Configuration ”.

A validation failure during the initial connection will result in `connectionStatusUpdate` returning `kConnectionStatusUnverified`. At this point you can choose to allow the user to proceed with the connection and save this response for subsequent connections using `setUserResponseToUnverifiedDeploymentServer`.

A validation failure during subsequent connections will result in `connectionStatusUpdate` returning `kConnectionStatusUntrusted` and the connection will be closed immediately.

Client Authentication

As part of the App Config provided during the App installation, MobiControl will generate a time sensitive one-time-use authentication token. The SDK will use this token to authenticate to the MobiControl deployment server for the first time which will in response issue a client authentication certificate for use in subsequent connections.

In the event the authentication token or the client authentication certificate has expired `connectionStatusUpdate` will return one of several constants identifying the type of authentication required by the deployment server. The authentication type will correspond to the authentication requirement of the Add Device Rule the device was originally enrolled under, or the default Add Device Rule if the original is not found or disabled.

NOTE:	The "AD" authentication type is actually "LDAP" as referred to in the MobiControl web console.

You can handle this authentication requirement by presenting the user with UI that requires them to provide their respective credentials and call `authenticateClient`, or by asking the user to re-install the App from the App Catalog. Re-installation will re-issue a new authentication token.

NOTE:	MobiControl will automatically renew client authentication certificates before their expiration if the application was opened within the renewal window.

Connections that have not successfully performed client authentication are maintained, however the MobiControl Deployment Server will not provide the SDK-enabled App with information that is considered sensitive. For example, File Sync, and Content Library content will not be distributed, while features such as Remote View will continue to function.

Integrating Features of the SDK

Included in the SDK is comprehensive documentation for the integration of the SDK features. Refer to the "Documentation" directory for this documentation, and MobiControl's online help documentation for the corresponding server configuration and behavioral descriptions.

There are several features notably absent from the SDK documentation; Remote View, File Sync, Location Services, and Jailbreak detection. These features are available immediately upon a successful connection to the MobiControl deployment server and do not have any additional configurations within the SDK, although there may be corresponding server-side configurations.

If you only wish to leverage the Remote View feature of the SDK, your application must not call `AgentSdkCore:setAppCapability`. If you wish to integrate any other features of the SDK, your application must call `AgentSdkCore:setAppCapability` with the parameter `kAPP_CAPABILITIES_ALL`. Please see the MobiControlConnectionManager.m in the Sample Application for an example of how to do so. Note that the call in the Sample Application is purposefully commented out and only present for illustrative purposes, because the Sample Application does not leverage any features of the SDK other than Remote View.

iTunes App Store Review Guidelines

If you are developing an App that is destined for the App Store you should be cognizant of Apple's App Store Review Guidelines and ensure that your application, and the features of the SDK you implement meet Apple's expectations. For example, the SDK includes functionality such as the App Catalog that is only suitable for enterprise applications.

Sample Application

Accompanying the SDK is a sample application that embeds the SDK. It establishes a connection to a MobiControl deployment server, which allows the MobiControl administrator to Remote View into the application. The sample XCode project can be found in the folder "Sample App". In order to build this application:

- Copy either the provided 'SDK\libSOTIMCAgentSDK.a' or 'SDK\libSOTIMCAgentSDK_Development.a' to 'Sample SDK-enabled App/SDK/libs/libSOTIMCAgentSDK.a'. See "Testing your application" for the differences between the two libraries. The sample application expects the file libSOTIMCAgentSDK.a in the path 'Sample SDK-enabled App/SDK/libs/' so, if you choose to use 'libSOTIMCAgentSDK_Development.a', you will need to rename the file to 'libSOTIMCAgentSDK.a' after copying it.
- Copy the provided 'SDK\SDKResources.bundle' to path 'Sample SDK-enabled App/SDK/libs/'
- Copy the provided 'SDK\Header' files into path 'Sample SDK-enabled App/SDK/h/'
- Change the bundle identifier in the sample application project to an appropriate one for your company (eg. com.yourcompanyname.sampleapplication).
- Either allow XCode to "Automatically manage signing" or set the provisioning profiles in the sample application project manually.
- Build the application to ensure it is free of errors.
- Archive the application so that it can be deployed using MobiControl.

REMINDER: You must archive the application and distribute it through MobiControl to retrieve the App Config connection properties for end to end functionality. See "Deploying your Application" for more information.

Testing your Application

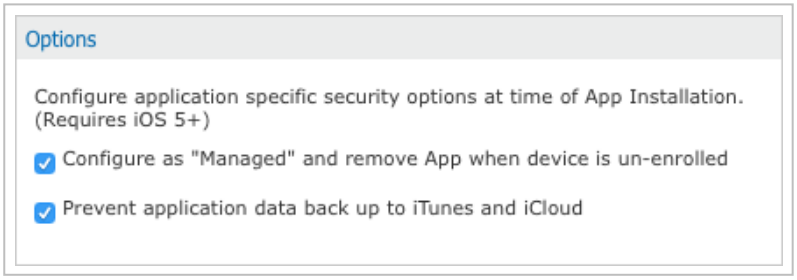
The SDK includes a library ('libSOTIMCAgentSDK_Development.a') that supports i386 and x86_64 architectures needed to run your App in the iOS simulator that accompanies Xcode. Alternatively, you can run the App on a physical device connected through Xcode instead of the iOS simulator. In this case you may use the library 'libSOTIMCAgentSDK.a', which supports all architectures except i386 and x86_64 and is suitable for your production builds.

In both cases however, the SDK's configuration is dependent on settings provided to it through the MobiControl App Catalog. This deployment mechanism is not supported by Xcode and therefore support of the iOS simulator and installation of the app through Xcode are limited to testing other areas of your application only. Any dependency placed on the SDK should be configurable in code to allow testing of other functionality before end-to-end integration tests. To complete end-to-end testing including the SDK please refer to "Deploying your Application".

Deploying your Application

An SDK-enabled application can be deployed via App Store or Enterprise distribution methods, but in both cases MobiControl must initiate the installation of the application on the device in order to configure it with the necessary settings and to link the App with the respective device record in MobiControl. These necessary settings are referred to as "SDK configuration."

You can create or leverage an existing iOS App Catalog Rule to deploy your SDK-enabled App Store or Enterprise App. Simply add a new catalog entry in the respective rule that contains your application. Under **App Catalog Rule > SDK-enabled App Entry > Advanced** ensure that the **Configure as "Managed" and remove App when device is un-enrolled** setting is checked.



Finally, there are two methods to configure your application with the SDK configuration.

Automatic Configuration

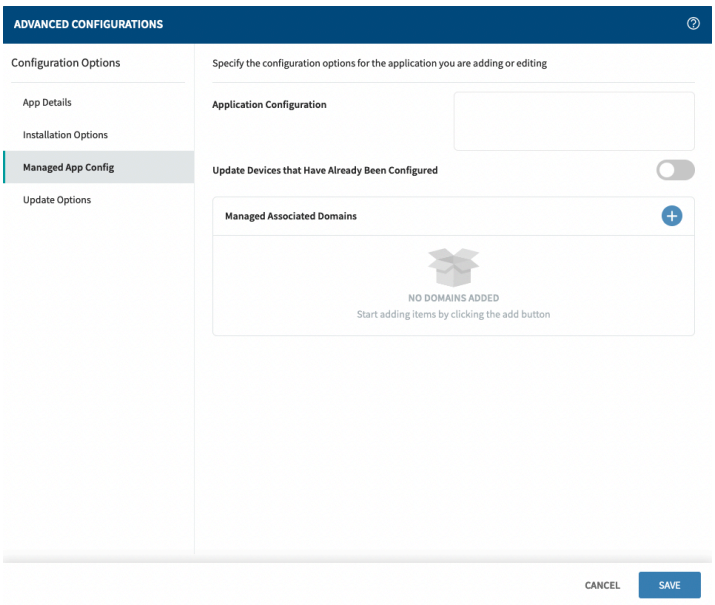
If your SDK-enabled application does not require any configuration settings of its own, MobiControl can automatically configure your application with the SDK configuration (by

sending all the settings specified in “App Configuration” section). To configure MobiControl to do so:

- Identify your SDK-enabled application’s bundle identifier. It should be something like com.companyname.applicationname. For instance, for the MobiControl agent, it’s net.soti.mobicontrol.
- Configure your MobiControl instance by making the following change to the file <MobiControl installation path>\MCDeplSvr.exe.config:
    - Search for “soti.mdm” configuration by searching for the line containing “<soti.mdm”
    - Append the following parameter: “agentApplicationIds="net.soti.mobicontrol,<your SDK-enabled application’s bundle identifier>"”.
        - In our example above, you would’ve ended up with the following line: “<soti.mdm agentApplicationIds="net.soti.mobicontrol,com.companyname.applicationname">”. Note that there are no spaces within the agentApplicationIds.
        - Do not delete any pre-existing parameters of the soti.mdm configuration. For example, if the parameter “authorizationMode="0"” existed already, you would end up with the following line: “<soti.mdm authorizationMode="0" agentApplicationIds="net.soti.mobicontrol,com.companyname.applicationname">”
    - Save the file
- Launch the MobiControl Administration Utility and restart the Deployment Server and the Management Service and ensure they start successfully.

Manual Configuration

If your SDK-enabled application requires its own configuration settings in addition to the SDK configuration, you must manually specify the SDK configuration (in addition to your application’s own configuration settings) in the **Configuration Command** under **App Catalog Rule > SDK-enabled App Entry > Application Configuration**. Refer to “App Configuration ” for the App Config syntax and contents required for the SDK.



App Configuration

The SDK configuration's connection settings can be specified in one of two ways:

- Specify the MobiControl Deployment Server address(es), any root certificates required to trust the MobiControl Deployment Server, the Add Devices Rule Tag and the Sitename.
- Specify the Enrollment ID to automatically discover all the connection settings mentioned above.

The SDK configuration must be delivered using iOS's App Config feature. App Config requires the configuration to be specified in the Plist format. The following key/values must be specified in the App Config to successfully configure your SDK-enabled application:

Key	Description
DEVICE_DI	String representing the iOS device's UDID. Specify only the macro %DeviceIdentifier%. *Required.*
AUTHTOKEN	String representing the authentication token. Customize the suffix of the %AuthToken_...% macro to configure the validity of the token. For example, %AuthToken_1m% represents 1 minute, %AuthToken_1h% represents 1 hour, %AuthToken_1d% represents 1 day. SOTI recommends a validity not exceeding 1 day. *Optional.*

ENROLLMENTID	String representing the Enrollment ID of the preferred Add Device Rule. Enrollment ID will be used to discover other connection parameters noted below. *Optional when the preferred key* DS_ADDRESSES *is specified.*
DS_MC_ROOT_CERTS	Array of base 64 encoded MobiControl Root certificates or base 64 encoded Root certificates that issued the certificate bound to the deployment server in MobiControl Administration Utility. *Not required if the Deployment Server Certificate is trusted by the device (ie: a commercial certificate). Otherwise, optional when ENROLLMENTID is specified, but preferred over* ENROLLMENTID*.*
DS_ADDRESSES	Array of deployment server hostnames. This key takes precedence over ENROLLMENTID if also specified. *Optional when ENROLLMENTID is optional when ENROLLMENTID is specified, but preferred over* ENROLLMENTID.
RULETAG	String in the form of a GUID representing the Add Devices Rule Tag of the Add Devices Rule that was used to enroll the device. The Add Devices Rule Tag can be found in the summary page of the Add Devices Rule in the web console. *Optional when ENROLLMENTID is specified, but preferred over* ENROLLMENTID*.*
SITENAME	String representing the Site Name of the MobiControl deployment. The Site Name can be found in the web console under **All Devices > Servers > Global Settings > Site Name.** The value is typically “MobiControl” unless customized. *Optional when ENROLLMENTID is specified, but preferred over* ENROLLMENTID*.*

Examples

The following is an example of the configuration you would provide in the App Catalog's "Application Configuration" section if you are using "Manual Configuration" and using the Enrollment ID to automatically discover all the connection settings.

```
<dict>
    <key>DEVICE_DI</key>
    <string>%DeviceIdentifier%</string>
    <key>AUTH_TOKEN</key>
    <string>%AuthToken_1h%</string>
    <key>ENROLLMENTID</key>
    <string>XXXYYYNN</string>
    <key>MyAppConfigurationKey1</key>
    <string>MyAppConfiguration1Value</string>
    <key>MyAppConfigurationKey2</key>
    <array>
        <string>MyAppConfiguration2Value1</string>
        <string>MyAppConfiguration2Value2</string>
    </array>
</dict>
```

The following is an example of the configuration you would provide in the App Policy "Application Configuration" section if you are using "Manual Configuration" and specifying all the connection settings.

```
<dict>
    <key>DEVICE_DI</key>
    <string>%DeviceIdentifier%</string>
    <key>AUTH_TOKEN</key>
    <string>%AuthToken_1h%</string>
    <key>SITENAME</key>
    <string>MobiControl</string>
    <key>DS_ADDRESSES</key>
    <array>
        <string>ds1.yourcompany.com</string>
        <string>ds2.yourcompany.com</string>
    </array>
    <key>DS_MC_ROOT_CERTS</key>
    <array>
        <string>308202C730820230A0D ... truncated for brevity</string>
        <string>1020208065DD00304DD ... truncated for brevity</string>
    </array>
    <key>MyAppConfigurationKey1</key>
    <string>MyAppConfiguration1Value</string>
    <key>MyAppConfigurationKey2</key>
    <array>
        <string>MyAppConfiguration2Value1</string>
        <string>MyAppConfiguration2Value2</string>
    </array>
</dict>
```

Changes from SDK v12

Bitcode Capability

With SDK v13, your SDK-enabled application can be built with Bitcode enabled. To enable Bitcode, set **Build Settings -> Build Options -> Enable Bitcode** to **Yes**.

OpenSSL Version

To ensure your app is more secure, SDK v13 now uses OpenSSL version 1.1.1o. If your app leverages OpenSSL, you will need to update your application to use the same version.

JSONKit Deprecation

SDK v13 is no longer dependent on JSONKit. If your app or any of its libraries leverage JSONKit, they no longer have to be in sync with the one included in the SDK.

Setting Application Capability

SDK v13.1.0+ now requires your application to explicitly specify which features of the SDK it wishes to leverage. Please see Integrating SDK Features for information on how to do so.

Glossary

The following terms and abbreviations are used throughout this document and are listed here as a reference:

- **AD –** Active Directory
- **CA** – Certificate Authority
- **Client Certificate** – Credential used to prove the identity of the requesting party
- **DS** – Deployment Service (Responsible for Agent-based communication)
- **DSE –** DS Extensions (Responsible for Web-based device communication)
- **EMM** – Enterprise Mobility Management
- **FQDN –** Fully Qualified Domain Name
- **HTTP(s) –** Hypertext Transfer Protocol (secure)
- **LDAP(s) –** Lightweight Directory Access Protocol (secure)
- **MDM** – Mobile Device Management
- **MS** – Management Service (Responsible for MobiControl Web Console)
- **PKI –** Public Key Infrastructure
- **Plist –** Property List
- **SDK –** Software Developer Kit
- **SCEP –** Simple Certificate Enrollment Protocol
- **Server Certificate –** Credential used to prove the identify of a server