Adding the MobileFirst Foundation SDK to iOS Applications

Overview

The MobileFirst Foundation SDK consists of a collection of pods that are available through CocoaPods (http://guides.cocoapods.org) and which you can add to your Xcode project.
The pods correspond to core functions and other functions:

- **IBMMobileFirstPlatformFoundation** - Implements client-to-server connectivity, handles authentication and security aspects, resource requests, and other required core functions.
- **IBMMobileFirstPlatformFoundationJSONStore** - Contains the JSONStore framework. For more information, review the JSONStore for iOS tutorial (../../jsonstore/ios/).
- **IBMMobileFirstPlatformFoundationPush** - Contains the push notification framework. For more information, review the Notifications tutorials (../../../notifications/).
- **IBMMobileFirstPlatformFoundationWatchOS** - Contains support for Apple WatchOS.

In this tutorial you learn how to add the MobileFirst Native SDK by using CocoaPods to a new or existing iOS application. You also learn how to configure the MobileFirst Server to recognize the application.

Prerequisites:

- Xcode and MobileFirst CLI installed on the developer workstation.
- A local or remote instance of MobileFirst Server is running.
- Read the Setting up your MobileFirst development environment (../../../installation-configuration/development/mobilefirst) and Setting up your iOS development environment (../../../installation-configuration/development/ios) tutorials.

Jump to:

Adding the MobileFirst Native SDK

Follow the instructions below to add the MobileFirst Native SDK to a new or existing Xcode project, and to register the application to the MobileFirst Server.

Before you start, make sure that the MobileFirst Server is running.
If using a locally installed server: From a **Command-line** window, navigate to the server's folder and run the command: `./run.sh`.

Creating an application

Create an Xcode project or use an existing one (Swift or Objective-C).

Adding the SDK

1. The MobileFirst Native SDK is provided via CocoaPods.
    - If CocoaPods (http://guides.cocoapods.org) is already installed in your development environment, skip to step 2.
    - If CocoaPods is not installed, install it as follows:
        - Open a **Command-line** window and navigate to the root of the Xcode project.
        - Run the command: `sudo gem install cocoapods` followed by `pod setup`. **Note:** These commands might take several minutes to complete.
2. Run the command: `pod init`. This creates a `Podfile`.
3. Using your favorite code editor, open the `Podfile`.
    - Comment out or delete the contents of the file.
    - Add the following lines and save the changes:

```
use_frameworks!

platform :ios, 8.0
target "Xcode-project-target" do
    pod 'IBMMobileFirstPlatformFoundation'
end
```

        - Replace **Xcode-project-target** with the name of your Xcode project's target.

4. Back in the command-line window, run the commands: `pod install`, followed by `pod update`. These command add the MobileFirst Native SDK files, add the **mfpclient.plist** file, and generate a Pod project.
   Note: The commands might take several minutes to complete.

   > **Important**: From here on, use the `[ProjectName].xcworkspace` file in order to open the project in Xcode. Do **not** use the `[ProjectName].xcodeproj` file. A CocoaPods-based project is managed as a workspace containing the application (the executable) and the library (all project dependencies that are pulled by the CocoaPods manager).

Manually adding the MobileFirst Native SDK

You can also manually add the MobileFirst SDK:

Click for instructions

Registering the application

1. Open a **Command-line** window and navigate to the root of the Xcode project.
2. Run the command:

   ```
   mfpdev app register
   ```

   - If a remote server is used, use the command `mfpdev server add` (../../using-mobilefirst-cli-to-manage-mobilefirst-artifacts/#add-a-new-server-instance) to add it.

   You are asked to provide the application's BundleID. **Important**: The BundleID is **case sensitive**.

The `mfpdev app register` CLI command first connects to the MobileFirst Server to register the application, then generates the **mfpclient.plist** file at the root of the Xcode project, and adds to it the metadata that identifies the MobileFirst Server.

> **Tip:** You can also register applications from the MobileFirst Operations Console:
>
> 1. Load the MobileFirst Operations Console.
> 2. Click the **New** button next to **Applications** to register a new application and follow the on-screen instructions.
> 3. After the application is registered, navigate to the application's **Configuration Files** tab and copy or download the **mfpclient.plist** file. Follow the onscreen instructions to add the file to your project.

Completing the setup process

In Xcode, right-click the project entry, click on **Add Files To [ProjectName]** and select the **mfpclient.plist** file, located at the root of the Xcode project.

Referencing the SDK

Whenever you want to use the MobileFirst Native SDK, make sure that you import the MobileFirst Foundation framework:

Objective-C:

```
#import <IBMMobileFirstPlatformFoundation/IBMMobileFirstPlatformFoundation.h>
```

Swift:

```
import IBMMobileFirstPlatformFoundation
```

Note about iOS 9 and above:

- Starting Xcode 7, Application Transport Security (ATS) (https://developer.apple.com/library/ios/releasenotes/General/WhatsNewIniOS/Articles/iOS9.html#//apple_ref/doc/uid/TP40016198-SW14) is enabled by default. In order to run apps during development, you can disable ATS (read more (http://iosdevtips.co/post/121756573323/ios-9-xcode-7-http-connect-server-error)).

    1. In Xcode, right-click the **[project]/info.plist file → Open As → Source Code**
    2. Paste the following:

```
<key>NSAppTransportSecurity</key>
<dict>
    <key>NSAllowsArbitraryLoads</key>
    <true/>
</dict>
```

Adding Support for Apple watchOS 2

If you are developing for Apple watchOS, the Podfile must contain sections corresponding to the main app and the watchOS extension:

```
# Replace with the name of your watchOS application
xcodeproj 'MyWatchApp'

use_frameworks!

#use the name of the iOS target
target :MyWatchApp do
    platform :ios, 9.0
    pod 'IBMMobileFirstPlatformFoundation'
    end

#use the name of the watch extension target
target :MyWatchApp WatchKit Extension do
    platform :watchos, 2.0
    pod 'IBMMobileFirstPlatformFoundation'
end
```

Verify that the Xcode project is closed and run the `pod install` command.

Updating the MobileFirst Native SDK

To update the MobileFirst Native SDK with the latest release, run the following command from the root folder of the Xcode project in a **Command-line** window:

```
pod update
```

SDK releases can be found in the SDK's CocoaPods repository (https://cocoapods.org/?q=ibm%20mobilefirst).

Generated MobileFirst Native SDK artifacts

mfpclient.plist

Located at the root of the project, this file defines the client-side properties used for registering your iOS app on the MobileFirst server.

Property	Description	Example values
wlServerProtocol	The communication protocol with the MobileFirst Server.	http or https
wlServerHost	The host name of the MobileFirst Server.	192.168.1.63
wlServerPort	The port of the MobileFirst Server.	9080
wlServerContext	The context root path of the application on the MobileFirst Server.	/mfp/
languagePreferences	Sets the default language for client sdk system messages.	en

Bitcode and TLS 1.2

For information about support for Bitcode and TLS 1.2 see the Additional Information (additional-information) page.

Tutorials to follow next

With the MobileFirst Native SDK now integrated, you can now:

- Review the Using the MobileFirst Foundation SDK tutorials (../)
- Review the Adapters development tutorials (../../../adapters/)
- Review the Authentication and security tutorials (../../../authentication-and-security/)
- Review the Notifications tutorials (../../../notifications/)
- Review All Tutorials (../../../all-tutorials)

Last modified on