Adding the MobileFirst Platform Foundation SDK to iOS Applications

Overview

The MobileFirst Platform Foundation SDK provides a set of API methods enabling a developer to implement various MobileFirst features, such as: authentication and security mechanisms, notifications, resource requests, collecting analytics data and more.

> For a complete list of MobileFirst SDK abilities visit the user documentation TODO: missing link

In this tutorial you will learn how to add the MobileFirst Native SDK using CocoaPods to either a new or existing iOS application. You will also learn how to configure the MobileFirst Server to recognize the application, as well as find information about the MobileFirst configuration files that are added to the project.

Pre-requisites: Xcode and MobileFirst CLI installed on the developer workstation.
Make sure you have read through the Setting up your development environment (../../setting-up-your-development-environment) tutorials.

Jump to:

Adding the MobileFirst Native SDK

Before starting, make sure the MobileFirst Server is running.
From **Terminal** run the command:

```
mfpdev server start
```

Follow the below instructions to manually add the MobileFirst Native SDK to either a new or existing Xcode project.

1. Create an Xcode project or use an existing one.
2. Open **Terminal** and navigate to the root of the Xcode project.
3. Run the command:

   ```
   mfpdev app register
   ```

   The `mfpdev app register` CLI command first connects to the MobileFirst Server and registers the application, followed by generating the `mfpclient.plist` file at the root of the Xcode project.

> The application registration can also be done from the MobileFirst Operations Console:
> 1. Open your browser of choice and load the MobileFirst Operations Console using the address `http://localhost:10080/mfpconsole/`. You can also open the console from **Terminal** using the CLI command `mfpdev server console`.
> 2. Click on the "Create new" button next to "Applications" to create a new application. Follow the on-screen instructions.
> 3. After successfully registering your application you can optionally download a "skeleton" Android Studio project pre-bundled with the MobileFirst Native SDK.

4. Run the command:

```
mfpdev app pull
```

   The `mfpdev app pull` CLI command registers the application in the MobileFirst Server, followed by creates a **mobilefirst** folder at the root of the Xcode project populating it with the `application-descriptor.json` file.

   These files are further explained in the Generated MobileFirst Native SDK artifacts section below.

   > **Tip:** Learn more about the various CLI commands in the Introduction to MobileFirst CLI tutorial TODO: missing link

5. The MobileFirst Native SDK is provided via CocoaPods. If CocoaPods (http://guides.cocoapods.org) is not installed in your development environment, install it as follows:
   - Open **Terminal** and navigate to the root of the Xcode project
   - Run the command: `sudo gem install cocoapods`
   - Run the command: `pod setup`
     Note: This command may take several minutes to complete.

6. Run the command: `pod init`. This creates a `Podfile`.
7. Using your favorite editor, open the `Podfile` file.
8. Comment out or remove the contents of the file.
9. Add the following lines and save the changes:

```
source 'https://github.com/CocoaPods/Specs.git'
pod 'IBMMobileFirstPlatformFoundation'
```

10. Run the command: `pod install`. This command adds the MobileFirst Native SDK, generates the Pod project, and integrates it with the Xcode project.
    Note: This command may 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).

11. Open the Xcode project by double-clicking the **.xcworkspace** file.
12. Right-click the project and select **Add Files To [ProjectName]**, select the `mfpclient.plist`, located in the root folder of the Xcode project.

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

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

Note about Swift:

> Because Swift is designed to be compatible with Objective-C you can use the MobileFirst SDK from within an iOS Swift project, too. Create a Swift project and follow the same steps, as described at the beginning of the tutorial, to integrate the MobileFirst Native SDK. Use `import IBMMobileFirstPlatformFoundation` in any class that needs to use the SDK.

Note about iOS 9:

> If you are developing for iOS9, consider disabling ATS (http://iosdevtips.co/post/121756573323/ios-9-xcode-7-http-connect-server-error) in the application's `info.plist` to be able to test locally without security restrictions.

Generated MobileFirst Native SDK artifacts

Two MobileFirst-related artifacts are available in the Xcode project after it has been integrated with the MobileFirst Native SDK: the `mfpclient.plist` and the `application-descriptor.json` file.

mfpclient.plist

Located at the root of the project, this file contains server configuration properties and is user-editable:

- `protocol` – The communication protocol to MobileFirst Server. Either `HTTP` or `HTTPS`.
- `host` – The hostname of the MobileFirst Server instance.
- `port` – The port of the MobileFirst Server instance.
- `wlServerContext` – The context root path of the application on the MobileFirst Server instance.
- `languagePreference` - Sets the default language for client sdk system messages

application-descriptor.json

Located in the **<xcode-project-root-directory>/mobilefirst** folder, this file contains application configuration settings such as its `bundleId`version and is user-editable.

If edited, be sure to update the MobileFirst Server by running the CLI command: `mfpdev app push`.
The file can also be updated by pulling from the server its latest variation by running the CLI command: `mfpdev app pull`.

The file can be edited via the MobileFirst Operations Console.

```
{
    "applicationKey": {
        "bundleId": "com.sampleone.bankApp",
        "version": "1.0",
        "clientPlatform":"ios"
    },

    ...
    ...
    ...
 }
```

Tutorials to follow next

Now that the application is integrated with the MobileFirst Native SDK you can continue reading tutorials for Native iOS development (../../ios-tutorials/) to learn more about authentication and security, server-side development, notifications, and more.