Adding the MobileFirst Foundation SDK to Cordova Applications

Overview

In this tutorial, you learn how to add the MobileFirst SDK to a new or existing Cordova application that has been created with Apache Cordova, Ionic, or another thirdy-party tool. You also learn how to configure the MobileFirst Server to recognize the application, and to find information about the MobileFirst configuration files that are changed in the project.

The MobileFirst Cordova SDK is provided as a set of Cordova plug-ins, and is registered at NPM (https://www.npmjs.com/package/cordova-plugin-mfp).
Available plug-ins are:

- **cordova-plugin-mfp** - The core SDK plug-in
- **cordova-plugin-mfp-push** - Provides push notifications support
- **cordova-plugin-mfp-jsonstore** - Provides JSONStore support
- **cordova-plugin-mfp-fips** - *Android only*. Provides FIPS support
- **cordova-plugin-mfp-encrypt-utils** - *iOS only*. Provides support for encryption and decryption

Support level: The platform versions supported by the MobileFirst plug-ins, at **minimum**, are **cordova-ios@4.0.1**, **cordova-android@5.1.1** and **cordova-windows@4.2.0**.

Jump to:

- Cordova SDK components
- Adding the MobileFirst Cordova SDK
- Updating the MobileFirst Cordova SDK
- Generated MobileFirst Cordova SDK artifacts
- Tutorials to follow next

Cordova SDK components

cordova-plugin-mfp

The cordova-plugin-mfp plug-in is the core MobileFirst plug-in for Cordova, and is required. If you install any of the other MobileFirst plug-ins, the cordova-plugin-mfp plug-in is automatically installed, too, if not already installed.

> The following Cordova plug-ins are installed as dependencies of cordova-plugin-mfp:
>
> - cordova-plugin-device
> - cordova-plugin-dialogs
> - cordova-plugin-globalization
> - cordova-plugin-okhttp

cordova-plugin-mfp-jsonstore

The cordova-plugin-mfp-jsonstore plug-in enables your app to use JSONstore. For more information about JSONstore, see the JSONStore tutorial (../../using-the-mfpf-sdk/jsonstore/cordova/).

cordova-plugin-mfp-push

The cordova-plugin-mfp-push plug-in provides permissions that are necessary to use push notification from the MobileFirst Server for Android applications. Additional setup for using push notification is required. For more information about push notification, see the Push notifications tutorial (../../notifications/push-notifications-overview/).

cordova-plugin-mfp-fips

The cordova-plugin-mfp-fips plug-in provides FIPS 140-2 support for the Android platform. For more information, see FIPS 140-2 support (http://www.ibm.com/support/knowledgecenter/en/SSHS8R_8.0.0/com.ibm.worklight.admin.doc/admin/c_using_FIPS_140-2_support.html?view=kc#c_using_FIPS_140-2_support).

cordova-plugin-mfp-encrypt-utils

The cordova-plugin-mfp-encrypt-utils plug-in provides iOS OpenSSL frameworks for encryption for Cordova applications with the iOS platform. For more information, see Enabling OpenSSL for Cordova iOS (additional-information).

Prerequisites:

- Apache Cordova CLI 6.x (https://www.npmjs.com/package/cordova) 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 (../../setting-up-your-development-environment/mobilefirst-development-environment) and Setting up your Cordova development environment (../../setting-up-your-development-environment/cordova-development-environment) tutorials.

Adding the MobileFirst Cordova SDK

Follow the instructions below to add the MobileFirst Cordova SDK to a new or existing Cordova project, and register it in 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`.

> **Note:** If you are adding the SDK to an existing Cordova application, the plug-in overwrites the `MainActivity.java` file for Android and `Main.m` file for iOS.

Adding the SDK

Consider creating the project by using the MobileFirst Cordova **application template**. The template adds the necessary MobileFirst-specific plug-in entries to the Cordova project's **config.xml** file, and provides a MobileFirst-specific, ready-to-use, **index.js** file that is adjusted for MobileFirst application development.

New Application

1. Create a Cordova project:

   ```
   cordova create Hello com.example.helloworld HelloWorld --template cordova-template-mfp
   ```

   - "Hello" is the folder name of the application.
   - "com.example.helloworld" is the ID of the application.
   - "HelloWorld" is the Name of the application.
   - --template modifies the application with MobileFirst-specific additions.

   > The templated **index.js** enables you to use additional MobileFirst features as such Multilingual application translation (../../using-the-mfpf-sdk/translation) and initialization options (see the user documentation for more information).

2. Navigate to the root of the Cordova project: `cd myapp`
3. Add one or more supported platforms to the Cordova project by using the Cordova CLI command: `cordova platform add ios|android|windows`. For example:

   ```
   cordova platform add ios
   ```

> **Note:** Because the application was configured using the MobileFirst template, the MobileFirst core Cordova plug-in is added automatically as the platform is added in step 3.

Existing Application

1. Navigate to the root of your existing Cordova project and add the MobileFirst core Cordova plug-in:

```
cordova plugin add cordova-plugin-mfp
```

2. Navigate to the **www\js** folder and select the **index.js** file.
3. Add the following function:

```
function wlCommonInit() {

}
```

The MobileFirst API methods are available after the MobileFirst client SDK has been loaded. The `wlCommonInit` function is then called.
Use this function to call the various MobileFirst API methods.

Registering the application

1. Open a **Command-line** window and navigate to the root of the Cordova project.
2. Register the application to MobileFirst Server:

```
mfpdev app register
```

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

The `mfpdev app register` CLI command first connects to the MobileFirst Server to register the application, then updates the **config.xml** file at the root of the Cordova project with metadata that identifies the MobileFirst Server.

Each platform is registered as an application in 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.

Using the SDK

The MobileFirst API methods are available after the MobileFirst client SDK has been loaded. The `wlCommonInit` function is then called.
Use this function to call the various MobileFirst API methods.

Updating the MobileFirst Cordova SDK

To update the MobileFirst Cordova SDK with the latest release, remove the **cordova-plugin-mfp** plug-in: run the `cordova plugin remove cordova-plugin-mfp` command and then run the `cordova plugin add cordova-plugin-mfp` command to add it again.

SDK releases can be found in the SDK's NPM repository (https://www.npmjs.com/package/cordova-plugin-mfp).

Generated MobileFirst Cordova SDK artifacts

config.xml

The Cordova configuration file is a mandatory XML file that contains application metadata, and is stored in the root directory of the app.
After the MobileFirst Cordova SDK is added to the project, the Cordova-generated **config.xml** file receives a set of new elements that are identified with the namespace `mfp:`. The added elements contain information related to MobileFirst features and the MobileFirst Server.

example of MobileFirst settings added to the config.xml file:

```
<?xml version='1.0'encoding='utf-8'?>
<widget id="..." xmlns:mfp="http://www.ibm.com/mobilefirst/cordova-plugin-mfp">
   <mfp:android>
      <mfp:sdkChecksum>3563350808</mfp:sdkChecksum>
      <mfp:appChecksum>0</mfp:appChecksum>
      <mfp:security>
         <mfp:testWebResourcesChecksum enabled="false" ignoreFileExtensions="png, jpg, jpeg, gif, mp4, mp3" />
      </mfp:security>
   </mfp:android>
   <mfp:windows>
      <mfp:sdkChecksum>3563350808</mfp:sdkChecksum>
     <mfp:windows10>
       <mfp:sdkChecksum>...</mfp:sdkChecksum>
       <mfp:security>
         <mfp:testWebResourcesChecksum/>
       </mfp:security>
   </mfp:windows>
   <mfp:platformVersion>8.0.0.00-20151214-1255</mfp:platformVersion>
   <mfp:clientCustomInit enabled="false" />
   <mfp:server runtime="mfp" url="http://10.0.0.1:9080" />
   <mfp:directUpdateAuthenticityPublicKey>the-key</mfp:directUpdateAuthenticityPublicKey>
   <mfp:languagePreferences>en</mfp:languagePreferences>
</widget>
```

Click for full list of config.xml properties

Editing MobileFirst settings in the config.xml file

You can use the MobileFirst CLI to edit the above settings by running the command:

```
mfpdev app config
```

Tutorials to follow next

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

- Review the Using the MobileFirst Foundation SDK tutorials (../../using-the-mfpf-sdk/)
- 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)