Adding the MobileFirst Foundation SDK to Android Applications

Overview

The MobileFirst Foundation SDK consists of a collection of dependencies that are available through Maven Central (http://search.maven.org/), and which you can add to your Android Studio project. The dependencies 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 Andoid tutorial (../../jsonstore/android/).
- **IBMMobileFirstPlatformFoundationPush** - Contains the Push Notifications framework. For more information, review the Notifications tutorials (../../../notifications/).

In this tutorial, you learn how to add the MobileFirst Native SDK by using Gradle to a new or existing Android Studio application. You also learn how to configure the MobileFirst Server to recognize the application, and to find information about the MobileFirst configuration files that are added to the project.

Prerequisites:

- Android Studio 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 Android development environment (../../../installation-configuration/development/android) tutorials.

Jump to:

Adding the MobileFirst Native SDK

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

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

Creating an Android application

Create an Android Studio project or use an existing one.

Adding the SDK

1. In **Android → Gradle Scripts**, select the **build.gradle (Module: app)** file.
2. Add the following lines after `apply plugin: 'com.android.application'`:

```
repositories{
    jcenter()
}
```

3. Add the following line inside the `android` section:

```
packagingOptions {
    pickFirst 'META-INF/ASL2.0'
    pickFirst 'META-INF/LICENSE'
    pickFirst 'META-INF/NOTICE'
}
```

4. Add the following lines inside the `dependencies` section:

```
compile group: 'com.ibm.mobile.foundation',
name: 'ibmmobilefirstplatformfoundation',
version: '8.0.+',
ext: 'aar',
transitive: true
```

   Or on a single line:

```
compile 'com.ibm.mobile.foundation:ibmmobilefirstplatformfoundation:8.0.+'
```

5. In **Android → app → manifests**, open the `AndroidManifest.xml` file. Add the following permissions above the **application** element:

```
<uses-permission android:name="android.permission.INTERNET"/>
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE"/>
```

6. Add the MobileFirst UI activity next to the existing **activity** element:

```
<activity android:name="com.worklight.wlclient.ui.UIActivity" />
```

> If a Gradle Sync request appears, accept it.

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 Android Studio 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.

The `mfpdev app register` CLI command first connects to the MobileFirst Server to register the application, followed by generating the **mfpclient.properties** file in the **[project root]/app/src/main/assets/** folder of the Android Studio project, and to add 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.properties** file. Follow the onscreen instructions to add the file to your project.

Creating a WLClient instance

Before using any MobileFirst APIs, create a `WLClient` instance:

```
WLClient.createInstance(this);
```

Note: Creating a `WLClient` instance should happen only once in the entire application lifecycle. It is recommended to use the Android Application class to do it.

Updating the MobileFirst Native SDK

To update the MobileFirst Native SDK with the latest release, find the release version number and update the `version` property accordingly in the **build.gradle** file.
See step 4 above.

SDK releases can be found in the SDK's JCenter repository (https://bintray.com/bintray/jcenter/com.ibm.mobile.foundation%3Aibmmobilefirstplatformfoundation/view#).

Generated MobileFirst Native SDK artifacts

mfpclient.properties

Located in the **./app/src/main/assets/** folder of the Android Studio project, this file defines the client-side properties used for registering your Android 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

Support for Javadoc and Android Service

For information about support for Javadoc and Android Service 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)