Adding the MobileFirst Platform Foundation SDK to Android Applications

Overview

The MobileFirst Platform Foundation SDK consists of a collection of dependencies, available through Maven Central (http://search.maven.org/), that you can add to your Android Studio project. The dependencies correspond to core functions and other functions:

- **IBMMobileFirstPlatformFoundation** - Implements client/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 (../../using-the-mfpf-sdk/jsonstore-android/).
- **IBMMobileFirstPlatformFoundationPush** - Contains the Push Notifications framework. For more information, review the Notifications tutorials (../../notifications/).

In this tutorial you will learn how to add the MobileFirst Native SDK using Gradle to either a new or existing Android Studio 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.

> For instruction to manually add the SDK files to a project, visit the user documentation (http://www-01.ibm.com/support/knowledgecenter/SSHS8R_8.0.0/wl_welcome.html).

Prerequisites:

- Android Studio and MobileFirst Developer CLI installed on the developer workstation.
- MobileFirst Server to run locally, or a remotely running MobileFirst Server
- Make sure you have read the Setting up your MobileFirst development environment (../../setting-up-your-development-environment/mobilefirst-development-environment) and Setting up your Android development environment (../../setting-up-your-development-environment/android-development-environment) tutorials.

Jump to:

Adding the MobileFirst Native SDK

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

Before starting, make sure the MobileFirst Server is running.
If using a locally installed server: From a **Command-line** window, navigate to the server's **scripts** folder and run the command: `./start.sh` in Mac and Linux or `start.cmd` in 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 below `apply plugin: 'com.android.application'`:

```
repositories{
    //jcenter()
    maven {
        url "http://visustar.francelab.fr.ibm.com:8081/nexus/content/repositories/mobile-s/"
    }
}
```

3. Add the following inside `android`:

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

4. Add the following lines inside `dependencies`:

```
compile group: 'com.ibm.mobile.foundation',
name: 'ibmmobilefirstplatformfoundation',
version: '8.0.Beta1-SNAPSHOT',
ext: 'aar',
transitive: true
```

   Or:

```
compile 'com.ibm.mobile.foundation:ibmmobilefirstplatformfoundation:8.0.Beta1-SNAPSHOT'
```

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.

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
```

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 adding to it the metadata that identifies the MobileFirst Server.

> **Tip:** The application registration can also be performed from the MobileFirst Operations Console:
> 1. Open your browser of choice and load the MobileFirst Operations Console using the address `http://localhost:9080/mfpconsole/`. You can also open the console from the **Command-line** using the CLI command `mfpdev server console`.
> 2. Click on the "Create new" button next to "Applications" to create a new application and 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.

Creating an WLClient instance

Before using any MobileFirst-supplied APIs, first create a `WLClient` instance in the `onCreate` method:

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

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 at the **./app/src/main/assets/** folder of the Android Studio project, this file contains server connectivity properties and is user-editable:

- `wlServerProtocol` – The communication protocol to MobileFirst Server. Either `HTTP` or `HTTPS`.
- `wlServerHost` – The hostname of the MobileFirst Server instance.
- `wlServerPort` – 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

Tutorials to follow next

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

- 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)