

Android

The purpose of this demonstration is to make you experience an end-to-end flow where IBM MobileFirst Platform Foundation SDK for Android is integrated into an Android project and used to retrieve data by using a MobileFirst adapter.

To learn more about creating projects and applications, using adapters and lots more, visit the Native Android Development (../) landing page.

Prerequisite: Make sure that you have installed the following software:

- MobileFirst Platform command line tool (download (file:///home/travis/build/MFPSamples/DevCenter/_site/downloads))
 - Android Studio
-

1. Create a MobileFirst back-end project and adapter.

- Create a back-end project in a location of your choice.

```
[code lang="shell"]
mfp create MyProject
cd MyProject
[/code]
```

- Add an HTTP adapter to the project.

```
[code lang="shell"]
mfp add adapter MyAdapter -t http
[/code]
```

2. Deploy artifacts to the MobileFirst Server.

- Start the MobileFirst Server and deploy the adapter.

```
[code lang="shell"]
mfp start
mfp push
[/code]
```

3. Create an Android project in Android Studio.

4. Add the MobileFirst Android SDK to the Android Studio project

- In **Project > Gradle scripts**, select **build.gradle (Module: app)**.
- After apply plugin: 'com.android.application', add the following line:

```
[code lang="xml"]repositories{
jcenter()
}[/code]
```

- Inside `android`, add the following lines:

```
[code lang="xml"]packagingOptions {
pickFirst 'META-INF/ASL2.0'
pickFirst 'META-INF/LICENSE'
pickFirst 'META-INF/NOTICE'
}[/code]
```

- Inside `dependencies`, add the following lines:

```
[code lang="xml"]compile group: 'com.ibm.mobile.foundation',
name: 'ibmmobilefirstplatformfoundation',
version: '7.1.0.0',
ext: 'aar',
transitive: true[/code]
```

- Add the following permissions to the `AndroidManifest.xml` file:

```
[code lang="xml"]
<uses-permission android:name="android.permission.INTERNET"/>
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE"/>
<uses-permission android:name="android.permission.GET_TASKS" />
[/code]
```

- Add the MobileFirst UI activity:

```
[code lang="xml"]<activity android:name="com.worklight.wlclient.ui.UIActivity" />[/code]
```

- In Terminal, navigate to the root of the Android Studio project and add the required configuration files by running this command:

```
[code lang="xml"]
mfp push[/code]
```

- **Implement MobileFirst adapter invocation.**

- **Main Activity class**

Make sure that your `MainActivity` class extends the `Activity` class:

```
[code lang="java"]
public class MainActivity extends Activity {
...
}[/code]
```

Add the following `import` statements:

```
[code lang="java"]import com.worklight.wlclient.api.*;
import android.util.Log;
import java.net.URI;
import java.net.URISyntaxException;[/code]
```

Add the following lines to the `onCreate` method:

```
[code lang="java"]
```

```

super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);

final WLClient client = WLClient.createInstance(this);

client.connect(new WLResponseListener() {

    @Override
    public void onSuccess(WLResponse wlResponse) {
        URI adapterPath = null;
        try {
            adapterPath = new URI("/adapters/MyAdapter/getFeed");
        } catch (URISyntaxException e) {
            e.printStackTrace();
        }
        WLResourceRequest request = new
        WLResourceRequest(adapterPath,WLResourceRequest.GET);
        request.send(new MyInvokeListener());
    }

    @Override
    public void onFailure(WLFailResponse wIFailResponse) {
        Log.i("MFPMMyProject","Failed connecting to the MobileFirst Server: " +
        wIFailResponse.getErrorMsg());
    }
});[/code]

```

■ MyInvokeListener class

Add a new MyInvokeListener class.

Add the following import statements:

```

[code lang="java"]import com.worklight.wlclient.api.*;
import android.util.Log;[/code]

```

Paste the following lines:

```

[code lang="java"]
public class MyInvokeListener implements WLResponseListener {

    @Override
    public void onSuccess(WLResponse wlResponse) {
        Log.i("MFPMMyProject","Adapter invocation response: " +
        wlResponse.getResponseJSON());
    }

    @Override
    public void onFailure(WLFailResponse wIFailResponse) {
        Log.i("MFPMMyProject", "Adapter invocation response: " +

```

```
wlFailResponse.getErrMsg());
}
}[/code]
```

◦ Final configurations

- Create an Android Virtual Device (AVD).

◦ Click Run.

Review the LogCat view for the data retrieved by the adapter request.



(<https://developer.ibm.com/mobilefirstplatform/wp-content/uploads/sites/32/2015/04/Screenshot-2015-04-14-at-17.31.24.png>)