

Invoking adapter procedures from native Android applications

Overview

To create and configure an Android native project, first follow the "Creating your first Native Android MobileFirst application (../hello-world/creating-first-native-android-mobilefirst-application/)" tutorial.

Initializing WLClient

1. Create an instance of the `WLClient` class.

The `WLClient` instance requires a reference to the activity in which it is running.

```
WLClient client = WLClient.createInstance(context);
```

2. To establish a connection to the MobileFirst Server instance, use the `connect` method by specifying the `MyConnectListener` class instance as a parameter.

The `WLClient` instance tries to connect to the MobileFirst Server according to the properties of the **wlclient.properties** file.

After the connection is established, it invokes one of the methods of the `MyConnectListener` class.

3. Specify that the `MyConnectListener` class implements the `WLResponseListener` interface.

```
public class MyConnectListener implements WLResponseListener {
```

The `WLResponseListener` interface defines two methods:

- `public void onSuccess (WLResponse response) { }`
- `public void onFailure (WLFailResponse response) { }`

4. Use these methods to process connection success or connection failure.

Invoking an adapter procedure

After the connection is established with a MobileFirst Server, you can use the `WLClient` instance to invoke adapter procedures.

1. Create a `WLProcedureInvocationData` object with the adapter and procedure names.

```
String adapterName = "RSSReader";
String procedureName = "getStoriesFiltered";
WLProcedureInvocationData invocationData =
    new WLProcedureInvocationData(adapterName, procedureName);
```

2. Add the required parameters as an object array and set request options (for example: `timeout`).

```
Object[] parameters = new Object[] { "world" };
invocationData.setParameters(parameters);
WLRequestOptions options = new WLRequestOptions();
options.setTimeout(30000);
```

3. Get the existing `WLClient` instance and use it to invoke an adapter procedure.

Specify the `MyInvokeListener` class instance as a parameter.

You learn how to define this class instance in the next section.

```
WLClient client = WLClient.getInstance();
client.invokeProcedure(invocationData, new MyInvokeListener(), options);
```

Receiving a procedure response

After the procedure invocation is completed, the `WLClient` instance calls one of the methods of the `MyInvokeListener` class.

1. Specify that the `MyInvokeListener` class implements the `WLResponseListener` interface.

```
public class MyInvokeListener implements WLResponseListener {
```

The `WLClient` instance invokes the `onSuccess` and `onFailure` methods.

If the procedure invocation is successful, the `onSuccess` method of `MyInvokeListener` is invoked.

2. Use that method to get the data that is retrieved from the adapter. The `response` object contains the response data. You can use its methods and properties to retrieve the required information.

```
public void onSuccess(WLResponse response) {
    String responseText = response.getResponseText();
    AndroidNativeApp.updateTextView("Adapter Procedure Invoked Successfully\n" +
    responseText);
}
public void onFailure(WLFailResponse response) {
    String responseText = response.getResponseText();
    AndroidNativeApp.updateTextView("Failed to Invoke Adapter Procedure\n" + responseText);
}
```

Sample application

Click to download

(<http://public.dhe.ibm.com/software/products/en/MobileFirstPlatform/docs/v630/InvokingAdapterProceduresNativeProject.zip>) the Studio project.

Click to download

(<http://public.dhe.ibm.com/software/products/en/MobileFirstPlatform/docs/v630/InvokingAdapterProceduresAndroidProject.zip>) the Native project.

The sample is made up of two projects:

- The **InvokingAdapterProceduresNativeProject.zip** file contains a MobileFirst Native API to deploy to your MobileFirst Server instance.
- The **InvokingAdapterProceduresAndroidProject.zip** file contains a native Android application that uses a MobileFirst native API library to communicate with MobileFirst Server.

Make sure to update the **wlclient.properties** file in the native Android project with the required server settings.

