Android Quick Start demonstration

fork and edit tutorial (https://github.ibm.com/MFPSamples/DevCenter/tree/master/tutorials/en/foundation/7.0/quick-start/android-quick-start.html) | report issue (https://github.ibm.com/MFPSamples/DevCenter/issues/new)

Overview

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

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

Required installed:

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

---

1. **Create a MobileFirst project and adapter**

   - **Create a new project and Android framework/server-side application entity**

     ```
     mfp create MyProject
     cd MyProject
     mfp add api MyAndroidFramework -e android
     ```

   - **Add a HTTP adapter to the project**

     ```
     mfp add adapter MyAdapter -t http
     ```

2. **Deploy artifacts to the MobileFirst Server**

   - **Start the MobileFirst Server and deploy the server-side application entity and adapter**

     ```
     mfp start
     # Wait until a browser window is opened, displaying the MobileFirst Console
     mfp deploy
     ```

3. **Create an Android project**

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

   - From **project-folder-location** > **MyProject** > **apps** > **MyAndroidFramework**, copy the following files: `worklight-android.jar`, `uicandroid.jar`, `bcprov.jar` and `android-async-http.jar`

- Open the **Project** view and navigate to the **app\libs** folder. Paste the copied files
- Right-click on any of the added `.jar` files and select **Add as library** to add all libraries
- Create an **assets** folder under **src\main** and paste into it the `wlclient.properties` file
- Add the following permissions to the `AndroidManifest.xml` file:

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

- Add the MobileFirst UI activity in the `AndroidManifest.xml` file:

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

5. **Implement MobileFirst adapter invocation**

- **Main Activity class** Add imports:

```
import com.worklight.wlclient.api.*;
import android.util.Log;
```

Add the following to `onCreate`:

```
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 = new URI("/adapters/MyAdapter/getFeed");
      WLResourceRequest request = new WLResourceRequest(adapterPath,WLResource
Request.GET);
      request.send(new MyInvokeListener());
   }

   @Override
   public void onFailure(WLFailResponse wlFailResponse) {
      Log.i("MFPMyProject","Failed connecting to the MobileFirst Server: " + wlFailRespons
e.getErrorMsg());
   }
});
```

- **`MyInvokeListener` class** Add a new `MyInvokeListener` class Add imports:

```
import com.worklight.wlclient.api.*;
import android.util.Log;
```

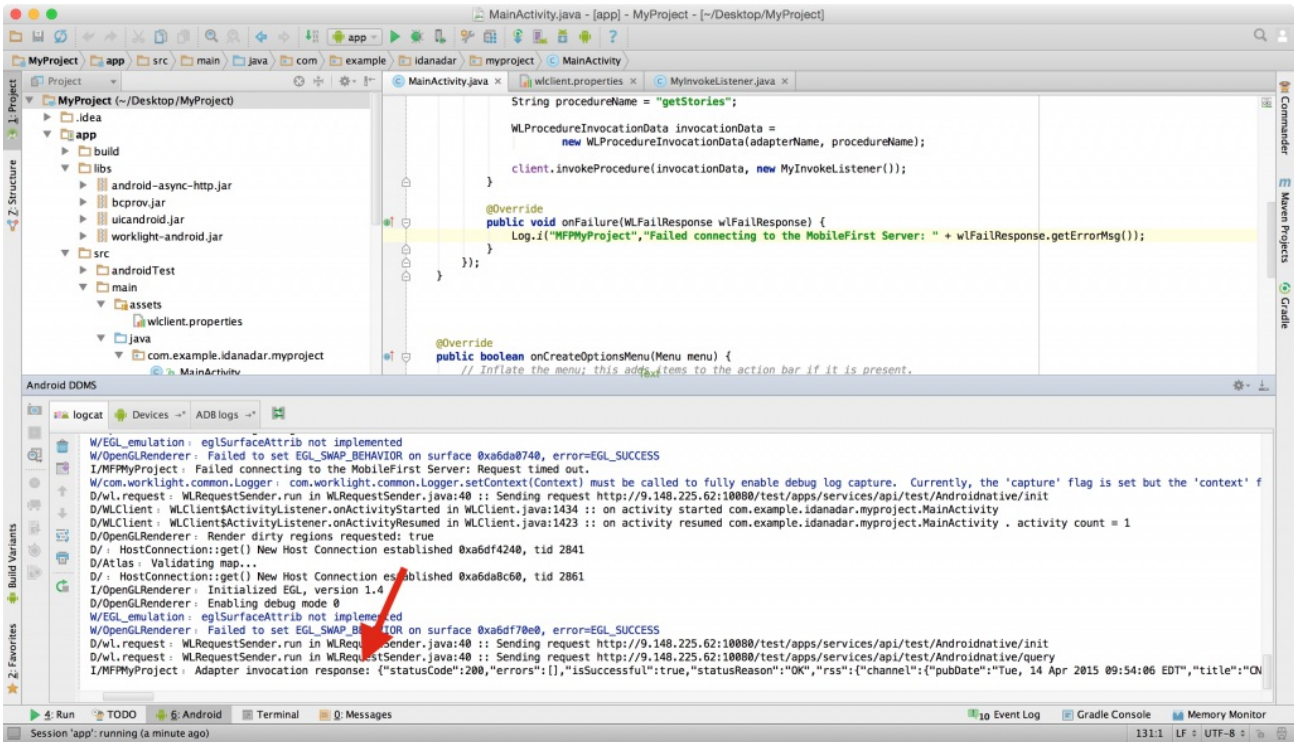
Paste the following:

```
public class MyInvokeListener implements WLResponseListener {

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

   @Override
   public void onFailure(WLFailResponse wlFailResponse) {
      Log.i("MFPMyProject", "Adapter invocation response: " + wlFailResponse.getErrorMs
g());
   }
}
```

6. **Final configurations**
   - Supply the machine's IP address for the `host` property in `wlclient.properties`
   - Create an AVD

7. **Click Run**

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



```
            String procedureName = "getStories";

            WLProcedureInvocationData invocationData =
                    new WLProcedureInvocationData(adapterName, procedureName);

            client.invokeProcedure(invocationData, new MyInvokeListener());
        }

        @Override
        public void onFailure(WLFailResponse wlFailResponse) {
            Log.i("MFPMyProject","Failed connecting to the MobileFirst Server: " + wlFailResponse.getErrorMsg());
        }
    });
}



@Override
public boolean onCreateOptionsMenu(Menu menu) {
    // Inflate the menu; this adds items to the action bar if it is present.
```

```
W/EGL_emulation: eglSurfaceAttrib not implemented
W/OpenGLRenderer: Failed to set EGL_SWAP_BEHAVIOR on surface 0xa6da0740, error=EGL_SUCCESS
I/MFPMyProject: Failed connecting to the MobileFirst Server: Request timed out.
W/com.worklight.common.Logger: com.worklight.common.Logger.setContext(Context) must be called to fully enable debug log capture.  Currently, the 'capture' flag is set but the 'context' f
D/wl.request: WLRequestSender.run in WLRequestSender.java:40 :: Sending request http://9.148.225.62:10080/test/apps/services/api/test/Androidnative/init
D/WLClient: WLClient$ActivityListener.onActivityStarted in WLClient.java:1434 :: on activity started com.example.idanadar.myproject.MainActivity
D/WLClient: WLClient$ActivityListener.onActivityResumed in WLClient.java:1423 :: on activity resumed com.example.idanadar.myproject.MainActivity . activity count = 1
D/OpenGLRenderer: Render dirty regions requested: true
D/: HostConnection::get() New Host Connection established 0xa6df4240, tid 2841
D/Atlas: Validating map...
D/: HostConnection::get() New Host Connection established 0xa6da8c60, tid 2861
I/OpenGLRenderer: Initialized EGL, version 1.4
D/OpenGLRenderer: Enabling debug mode 0
W/EGL_emulation: eglSurfaceAttrib not implemented
W/OpenGLRenderer: Failed to set EGL_SWAP_BEHAVIOR on surface 0xa6df70e0, error=EGL_SUCCESS
D/wl.request: WLRequestSender.run in WLRequestSender.java:40 :: Sending request http://9.148.225.62:10080/test/apps/services/api/test/Androidnative/init
D/wl.request: WLRequestSender.run in WLRequestSender.java:40 :: Sending request http://9.148.225.62:10080/test/apps/services/api/test/Androidnative/query
I/MFPMyProject: Adapter invocation response: {"statusCode":200,"errors":[],"isSuccessful":true,"statusReason":"OK","rss":{"channel":{"pubDate":"Tue, 14 Apr 2015 09:54:06 EDT","title":"CN
```