Android end-to-end demonstration

Overview

The purpose of this demonstration is to experience an end-to-end flow:

1. A scaffold application - an application that is pre-bundled with the MobileFirst client SDK, is registered and downloaded from the MobileFirst Operations Console.
2. An new or provided adapter is deployed to the MobileFirst Operations Console.
3. The application logic is changed to make a resource request.

End result:

- Successfully pinging the MobileFirst Server.
- Successfully retrieving data using a MobileFirst Adapter.

Prerequisites:

- Android Studio
- *Optional*. MobileFirst Developer CLI (download (file:////home/travis/build/MFPSamples/DevCenter/_site/downloads))
- *Optional*. Stand-alone MobileFirst Server (download (file:////home/travis/build/MFPSamples/DevCenter/_site/downloads))

1. Starting the MobileFirst Server

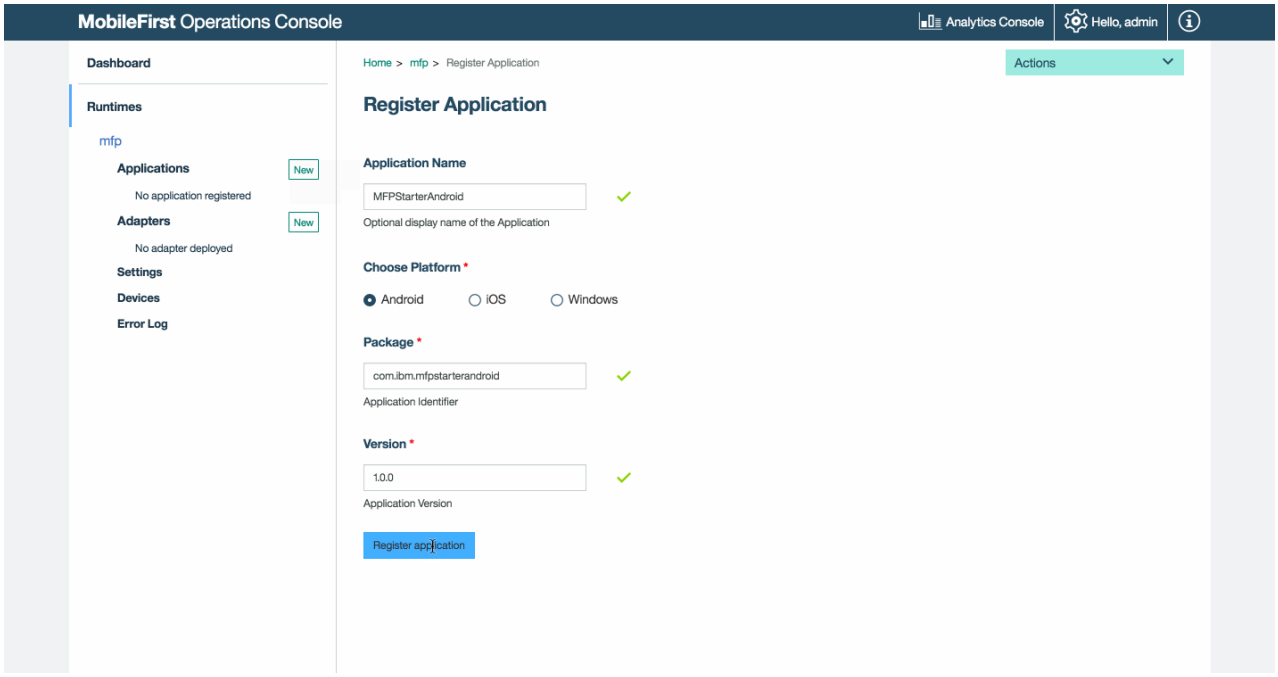
> If a remote server was already set-up, skip this step.

From a **Command-line** window, navigate to the server's folder and run the command: `./run.sh` in Mac and Linux or `run.cmd` in Windows.

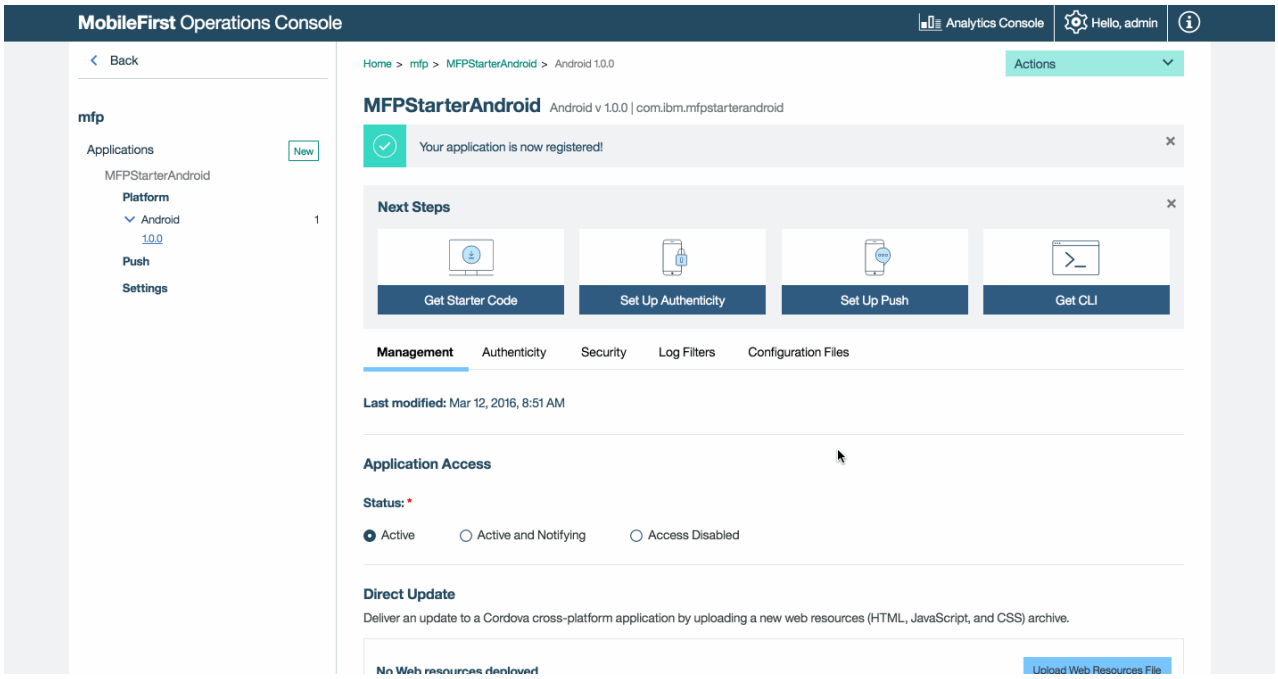
2. Creating an application

In a browser window, open the MobileFirst Operations Console by loading the URL: `http://your-server-host:server-port/mfpconsole`. If running locally, use: http://localhost:9080/mfpconsole (http://localhost:9080/mfpconsole). The username/password are *admin/admin*.

1. Click the **New** button next to **Applications**
    - Select the **Android** platform
    - Enter **com.ibm.mfpstarterandroid** as the **application identifier**
    - Enter **1.0.0** as the **version** value
    - Click on **Register application**



2. Click on the **Get Starter Code** tile and select to download the Android application scaffold.



3. Editing application logic

1. Open the Android Studio project and import the project.
2. Select the **app/java/com.ibm.mfpstarterandroid/ServerConnectActivity.java** file and:

- Add the following imports:

```
import java.net.URI;
import java.net.URISyntaxException;
import android.util.Log;
```

- Paste the following code snippet, replacing the `void onSuccess()` function:

```
public void onSuccess(AccessToken token) {
    System.out.println("Received the following access token value: " + token);
    runOnUiThread(new Runnable() {
        @Override
        public void run() {
            titleLabel.setText("Yay!");
            connectionStatusLabel.setText("Connected to MobileFirst Server");
        }
    });

    URI adapterPath = null;
    try {
        adapterPath = new URI("/adapters/javaAdapter/users/world");
    } catch (URISyntaxException e) {
        e.printStackTrace();
    }

    WLResourceRequest request = new WLResourceRequest(adapterPath, WLResourceRequest.G
ET);
    request.send(new WLResponseListener() {
        @Override
        public void onSuccess(WLResponse wlResponse) {
            // Will print "Hello world" in LogCat.
            Log.i("MobileFirst Quick Start", "Success: " + wlResponse.getResponseText());
        }

        @Override
        public void onFailure(WLFailResponse wlFailResponse) {
            Log.i("MobileFirst Quick Start", "Failure: " + wlFailResponse.getErrorMsg());
        }
    });
}
```

4. Creating an adapter

Download this prepared .adapter artifact (../javaAdapter.adapter) and deploy it from the MobileFirst Operations Console using the **Actions → Deploy adapter** action.

Alternatively, click the **New** button next to **Adapters**.

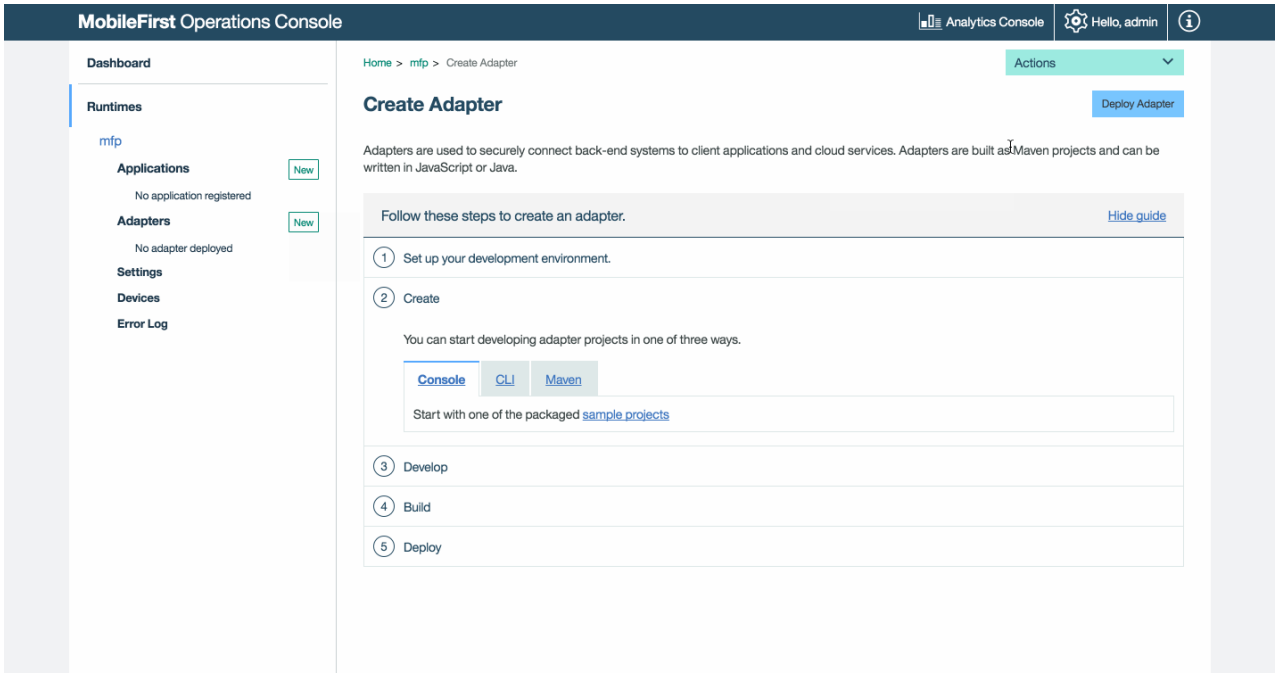
1. Select the **Actions → Download sample** option. Download the "Hello World" **Java** adapter sample.

   > If Maven and MobileFirst Developer CLI are not installed, follow the on-screen **Set up your development environment** instructions.

2. From a **Command-line** window, navigate to the adapter's Maven project root folder and run the command:

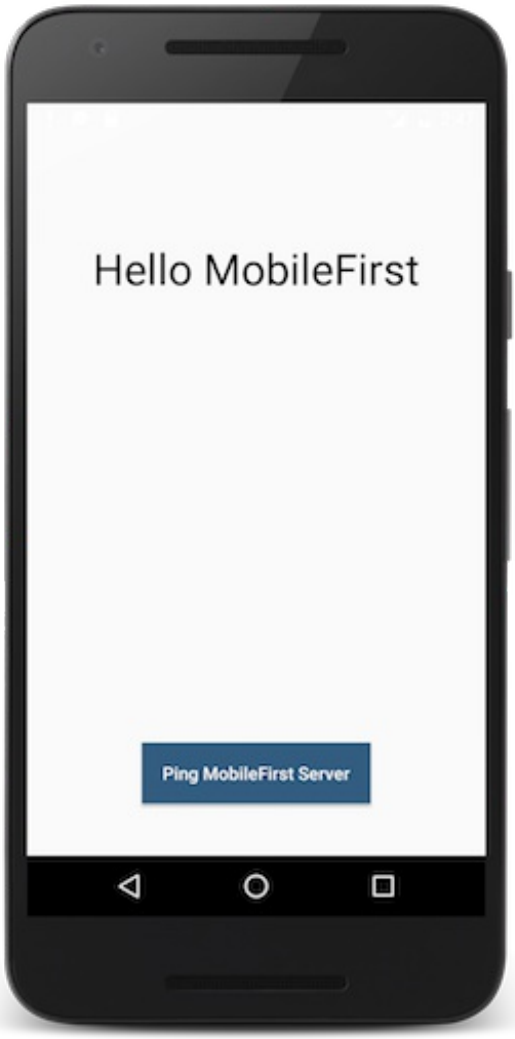
   ```
   mfpdev adapter build
   ```

3. When the build finishes, deploy it from the MobileFirst Operations Console using the **Actions → Deploy adapter** action. The adapter can be found in the **[adapter]/target** folder.



5. Testing the application

1. In Android Studio, select the



[project]/app/src/main/assets/mfpclient.properties file and edit the **host** property with the IP address of the MobileFirst Server.

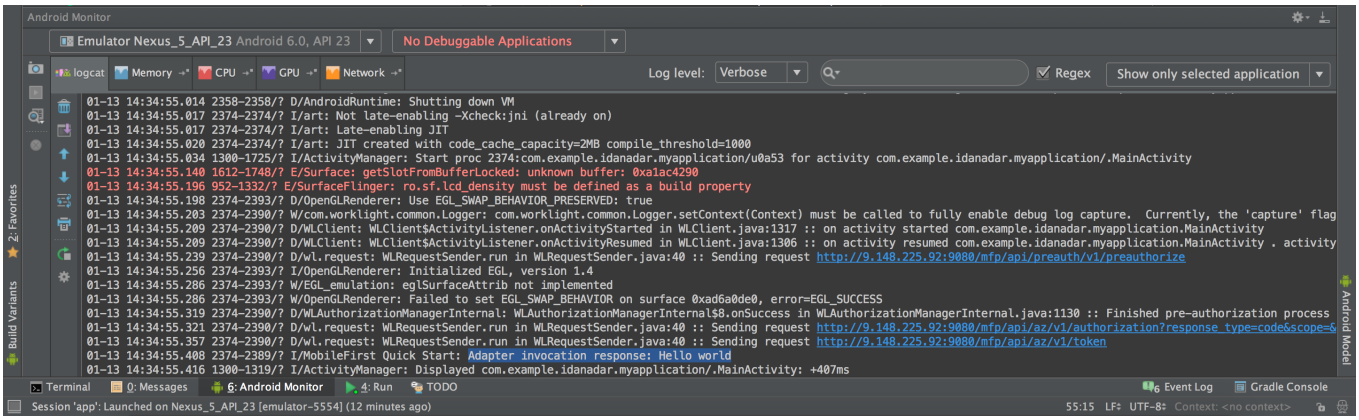
Alternatively, if you have installed the MobileFirst Develper CLI then navigate to the project root folder and run the command `mfpdev app register`. If a remote server is used instead of a local server, first use the command `mfpdev server add` to add it.

2. Click on the **Run App** button.

Results

- Clicking the **Ping MobileFirst Server** button will display **Connected to MobileFirst Server**.
- If the application was able to connect to the MobileFirst Server, a resource request call using the deployed Java adapter will take place.

The adapter response is then printed in Android Studio's LogCat view.



Next steps

Learn more on using adapters in applications, and how to integrate additional services such as Push Notifications, using the MobileFirst security framework and more:

- Review the Using the MobileFirst Platform Foundation (../../using-the-mfpf-sdk/) tutorials
- Review the Adapters development (../../adapters/) tutorials
- Review the Authentication and security tutorials (../../authentication-and-security/)
- Review the Notifications tutorials (../../notifications/)
- Review All Tutorials (../../all-tutorials)