

# Android end-to-end demonstration

fork and edit tutorial (<https://github.ibm.com/MFPSamples/DevCenter/tree/master/tutorials/en/foundation/8.0/quick-start/android/index.md>)

| 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 an application and an adapter are registered using the MobileFirst Operations Console, an "skeleton" Android Studio project is downloaded and edited to call the adapter, and the result is printed to the log - verifying a successful connection with the MobileFirst Server.

### Prerequisites:

- Android Studio
- MobileFirst Developer CLI (download  
([file:///home/travis/build/MFPSamples/DevCenter/\\_site/downloads](file:///home/travis/build/MFPSamples/DevCenter/_site/downloads)))
- *Optional*. Stand-alone MobileFirst Server (download  
([file:///home/travis/build/MFPSamples/DevCenter/\\_site/downloads](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 **scripts** folder and run the command: `./start.sh` in Mac and Linux or `start.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 on the "New" button next to **Applications** and select the desired *platform*, *identifier* and *version* values.



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



### 3. Editing application logic

1. Open the Android Studio project and import the project.
2. Select the **app/java/com/mfp/sample/MainActivity.java** file and:
  - Add the following imports:

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

- Paste the following code snippet, inside the `protected void onCreate()` function:

```

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

WLResourceRequest request = new WLResourceRequest(adapterPath, WLResourceRequest.GET);
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

1. Click on the "New" button next to **Adapters** and download the **Java** adapter sample.

If Maven and MobileFirst CLI are not installed, follow the on-screen **Setting up your environment** instructions to install.

Alternatively, download this prepared .adapter artifact and deploy it from the MobileFirst Operations Console using the **Actions → Deploy adapter** action.

MobileFirst Operations Console

Home > mfp > Create a new Adapter

### Create a new Adapter

It seems like you don't have any adapters, lets get started [Deploy Adapter](#)

**Follow these steps to set up an adapter** [Hide guide](#)

- 1 Setting up your environment
- 2 Start with a sample adapter
 

[Console](#)
[CLI](#)
[Maven](#)

CONSOLE Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aenean euismod bibendum laoreet. Proin gravida dolor sit amet lacus accumsan et viverra justo commodo. Proin sodales pulvinar tempor.
- 3 In your IDE of choice, edit the adapter code - REST end points and adapter descriptor
- 4 Build and package
- 5 Upload adapter



- Review the Authentication and security tutorials ([../authentication-and-security/](#))
- Review the Notifications tutorials ([../notifications/](#))
- Review All Tutorials ([../all-tutorials](#))