

# Android end-to-end demonstration

## Overview

The purpose of this demonstration is to experience an end-to-end flow where an application is quickly created using the MobileFirst Operations Console and connectivity is verified with the MobileFirst Server.

### Prerequisites:

- Configured Android Studio
- 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

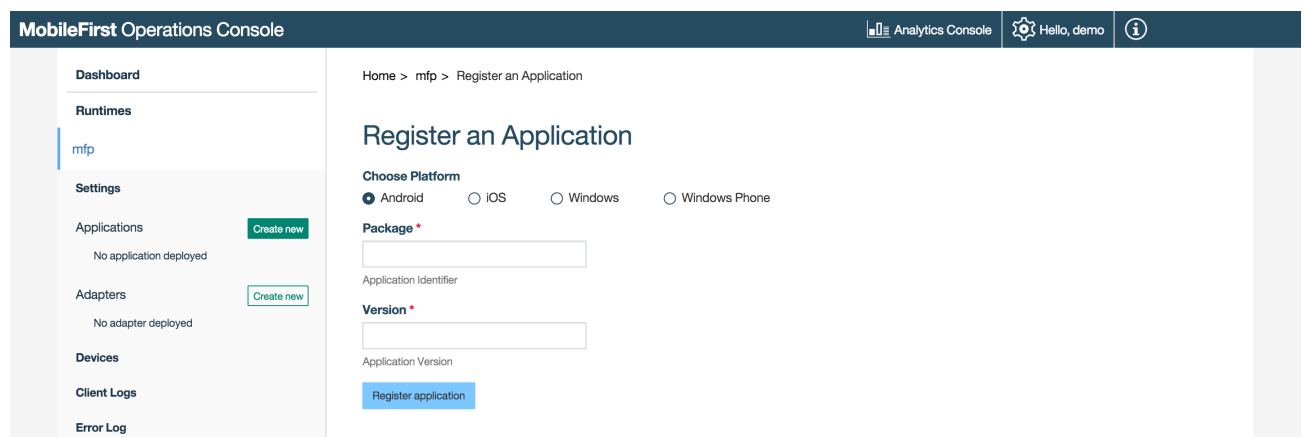
If a remote server was already set-up, skip this step.

From a **Command-line** window, navigate to the server's **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 "Create 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.
2. Select the **app/java/com.mfp.sample/MainActivity.java** file and paste the following code snippets:

- Imports:

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

- In `protected void onCreate()`:

```

WLClient client = 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) {
        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 "Create new" button next to **Adapters** and download the **Java** adapter sample.

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

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



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

```
mfpdev adapter build
```

- When the build finishes, run the command:

```
mfpdev adapter deploy
```

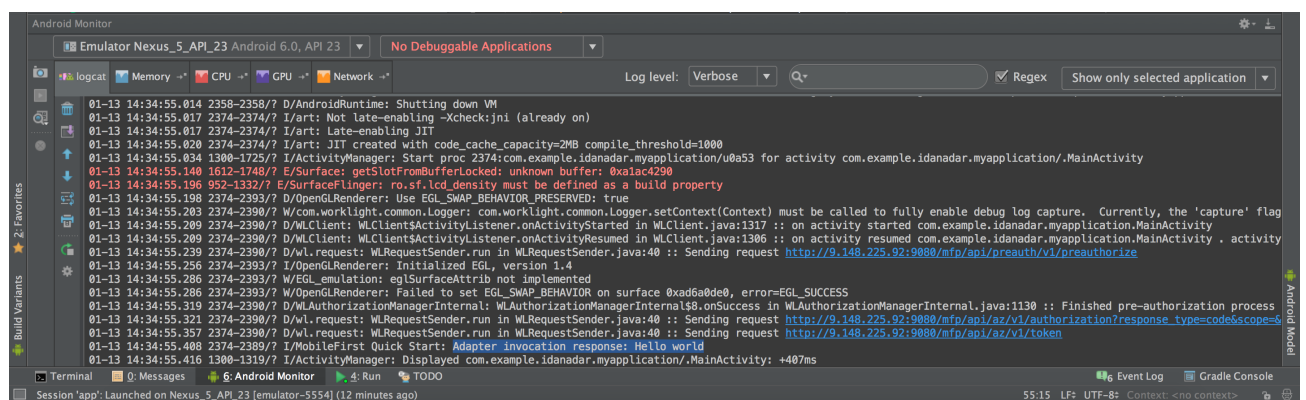
If using a remote MobileFirst Server, run the command:

```
mfpdev adapter deploy Replace-with-remote-server-name
```

## 5. Testing the application

- In Android Studio, click on the **Run App** button.

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



## Next steps

- Review the Client-side development tutorials (../client-side-development/)
- Review the Server-side development tutorials (../server-side-development/)
- Review the Authentication and security tutorials (../authentication-and-security/)

- [Review All Tutorials \(../../all-tutorials\)](#)