

# Web app end-to-end demonstration

## Overview

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

1. A sample application that is pre-bundled with the MobileFirst client SDK is registered and downloaded from the MobileFirst Operations Console.
2. A 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:

- A modern web browser
- *Optional.* MobileFirst 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

Make sure you have created a Mobile Foundation instance (.../bluemix/using-mobile-foundation), or If using the MobileFirst Foundation Development Kit (.../installation-configuration/development/mobilefirst), navigate to the server's folder and run the command: `./run.sh` in Mac and Linux or `run.cmd` in Windows.

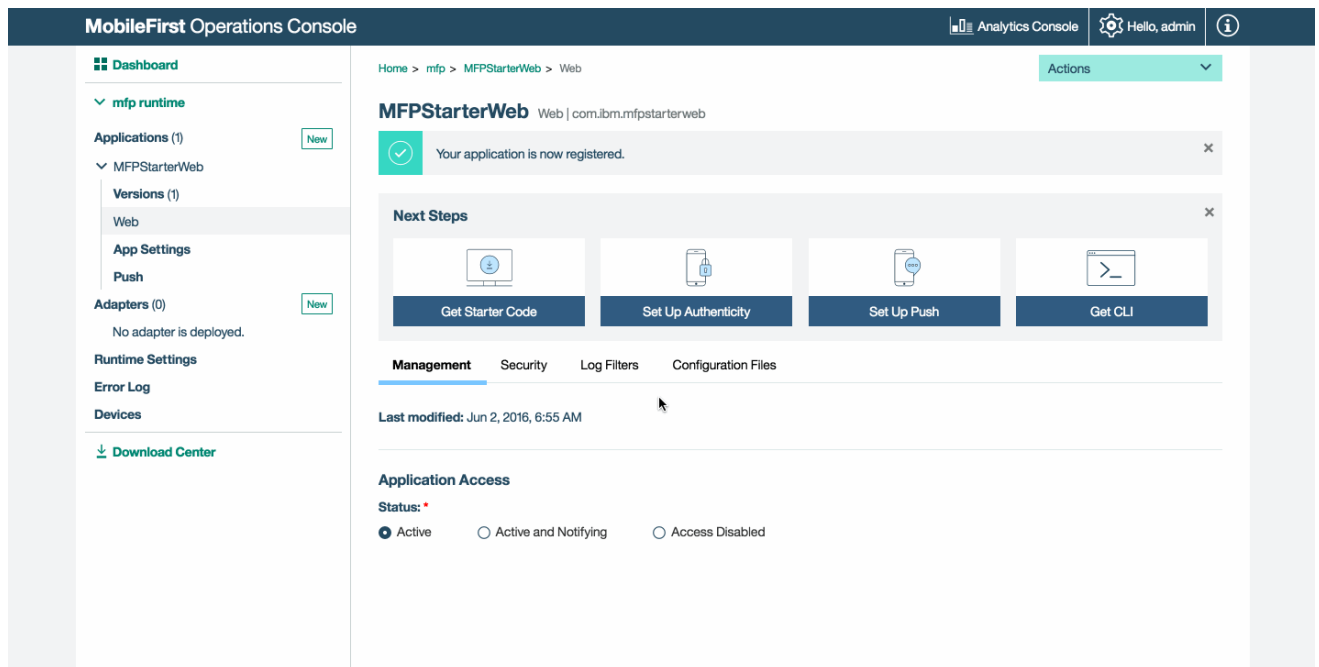
## 2. Creating and registering an application

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 **Web** platform
  - Enter **com.ibm.mfpstarterweb** as the **application identifier**
  - Click on **Register application**

The screenshot shows the MobileFirst Operations Console interface. On the left is a sidebar with navigation links: Dashboard, mfp runtime, Applications (0) [New], Adapters (0) [New], Runtime Settings, Error Log, Devices, and Download Center. The main content area is titled 'Home > mfp > Register Application'. It contains a 'Register Application' form with the following fields: 'Application Name' (MFPStarterWeb), 'Optional display name of the application', 'Choose Platform' (Web selected), and 'Application ID' (com.ibm.mfpstarterweb). A 'Register application' button is at the bottom. A note at the bottom states: '\* When you add the SDK to your application without the MobileFirst Command Line Interface, follow the instructions in the Configurations Files tab of your new app.'

2. Click on the **Get Starter Code** tile and select to download the Web sample application.



### 3. Editing application logic

1. Open the project in your code editor of choice.
2. Select the **client/js/index.js** file and paste the following code snippet, replacing the existing `WLAuthorizationManager.obtainAccessToken()` function:

```
WLAuthorizationManager.obtainAccessToken()
.then(
  function(accessToken) {
    titleText.innerHTML = "Yay!";
    statusText.innerHTML = "Connected to MobileFirst Server";

    var resourceRequest = new WLResourceRequest(
      "/adapters/javaAdapter/resource/greet/",
      WLResourceRequest.GET
    );

    resourceRequest.setQueryParameter("name", "world");
    resourceRequest.send().then(
      function(response) {
        // Will display "Hello world" in an alert dialog.
        alert("Success: " + response.responseText);
      },
      function(response) {
        alert("Failure: " + JSON.stringify(response));
      }
    );
  },
  function(error) {
    titleText.innerHTML = "Bummer...";
    statusText.innerHTML = "Failed to connect to MobileFirst Server";
  }
);
```

### 4. Deploy 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**.

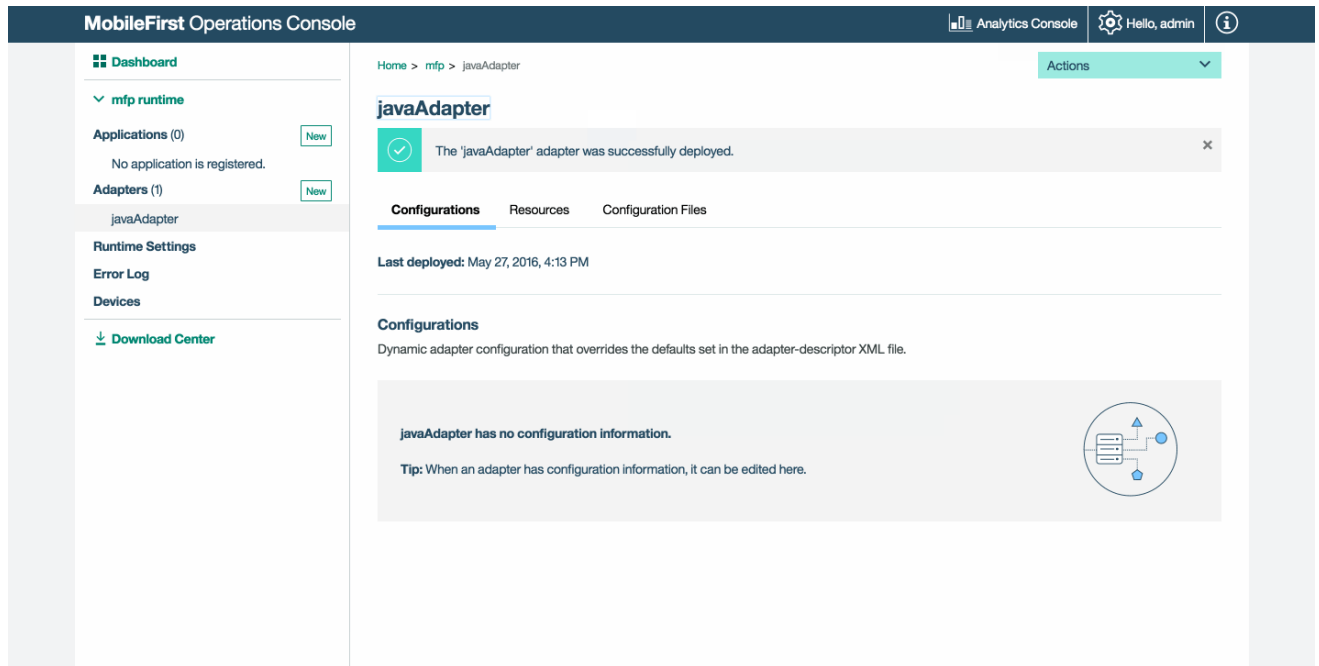
1. Select the **Actions → Download sample** option. Download the "Hello World" **Java** adapter sample.

If Maven and MobileFirst 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.



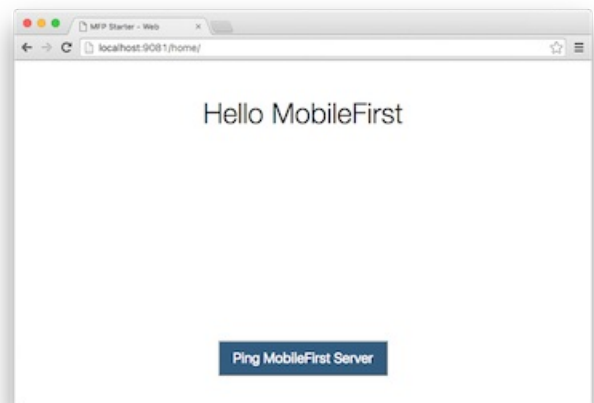
## 5. Testing the application

1. From a **Command-line** window, navigate to the **[project root] → node-server** folder.
2. Run the command: `npm start` to install required Node.js configuration and start the Node.js server.
3. Open the **[project root] → node-server → server.js** file and edit the **host** and **port** variables with the correct values for your MobileFirst Server.
  - If using a local MobileFirst Server, the values are typically **http, localhost** and **9080**.
  - If using a remote MobileFirst Server (on Bluemix), the values are typically **https, your-server-address** and **443**.

For example:

```
var host = 'https://mobilefoundation-xxxx.mybluemix.net'; // The Mobile Foundation server address
var port = 9081; // The local port number to use
var mfpURL = host + ':443'; // The Mobile Foundation server port number
```

4. In your browser, visit the URL: `http://localhost:9081/home` (`http://localhost:9081/home`).



## Secure Origins Policy

When using Chrome during development, the browser may not allow an application to load if using both HTTP and a host that **is not** "localhost". This is due to the Secure Origins Policy implemented and used by default in this browser.

To overcome this, you can start the Chrome browser with the following flag:

```
--unsafely-treat-insecure-origin-as-secure="http://replace-with-ip-address-or-host:port-number" --user-data-dir=/test-to-new-user-profile/myprofile
```

- Replace "test-to-new-user-profile/myprofile" with the location of a folder that will act as a new Chrome user profile for the flag to work.

## 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 displayed in an alert.

## 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 Foundation (../application-development/) tutorials
- Review the Adapters development (../adapters/) tutorials
- Review the Authentication and security tutorials (../authentication-and-security/)
- Review All Tutorials (../all-tutorials)

*Last modified on*

IBM	Social	Site
Legal notices (file:///home/travis/build/MFPSamples/DevCenter/https://www.facebook.com/ibmmobiledev/)	Facebook	RSS feed
Privacy (http://www.ibm.com/privacy/us/en/)	Twitter (https://twitter.com/ibmmobiledev)	Open issue (https://github.com/MobileFirst-Platform-Developer-Center/DevCenter/issues/new)
Terms of use (file:///home/travis/build/MFPSamples/DevCenter/C3/A/Make-usu97Q/)	YouTube (https://www.youtube.com/channel/UC3A/Make-usu97Q/)	Contribute (https://github.com/MobileFirst-Platform-Developer-Center/DevCenter/blob/master/contributing.m
Third party notice (file:///home/travis/build/MFPSamples/DevCenter/site/third-party-notice/)	GitHub (https://github.com/MobileFirst-Platform-Developer-Center)	Report abuse (https://www.ibm.com/developerworks/commu