Windows 8.1 Universal and Windows 10 UWP 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:

- Configured Visual Studio 2013/5
- *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

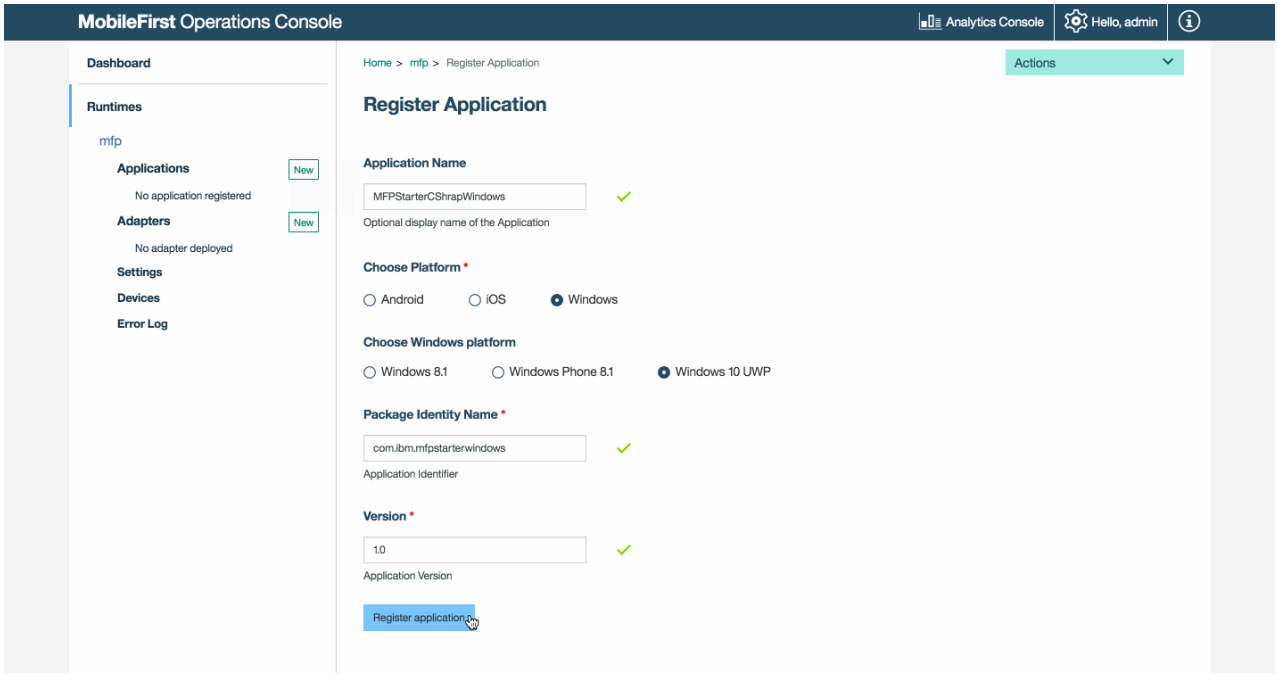
> If a remote server was already set-up, skip this step.

From a **Command-line** window, navigate to the server's folder and run the command: `run.bat`.

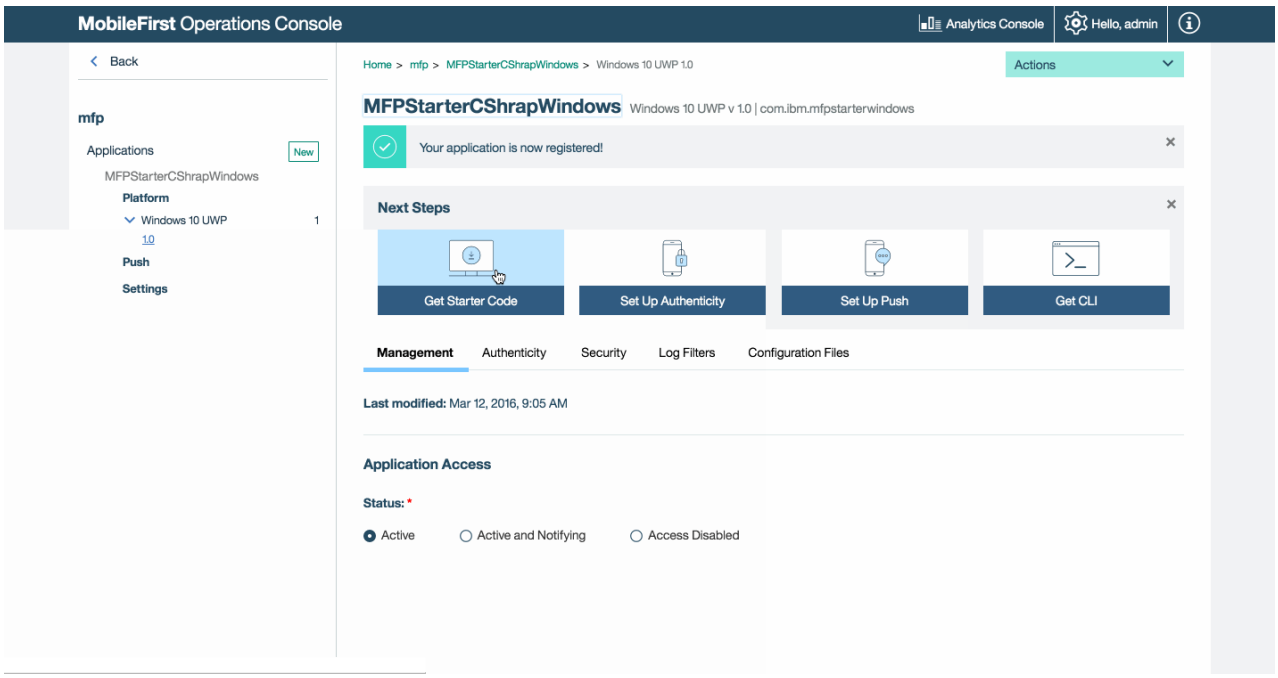
2. Creating an application

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 a **Windows** platform
    - Enter **MFPStarterCSharp.Windows** as the **application identifier** for Windows, or **MFPStarterCSharp.WindowsPhone** for Windows Phone
    - 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 Windows 8.1 or Windows 10 mobile app scaffold.



3. Editing application logic

1. Open the Visual Studio project.
2. Select the solution's **MainPage.xaml.cs** file and paste the following code snippet:

```
WorklightAccessToken accessToken = await Worklight.WorklightClient.CreateInstance().Authorizati
onManager.ObtainAccessToken("");

if(accessToken.IsValidToken && accessToken.Value != null && accessToken.Value != "")
{
   try
   {
     IWorklightClient _newClient = WorklightClient.CreateInstance();

     StringBuilder uriBuilder = new StringBuilder().Append("/adapters/JavaAdapter/users/world");

     WorklightResourceRequest rr = _newClient.ResourceRequest(uriBuilder.ToString(), "GET");

     WorklightResponse resp= await rr.send();

     if (resp.success)
     {
        Debug.WriteLine("Success: " + resp.ResponseText);
     } else
     {
        Debug.WriteLine("Failure: " + resp.error);
     }
   }catch(Exception e)
   {
     Debug.WriteLine(e.StackTrace);
   }
}
```

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**.

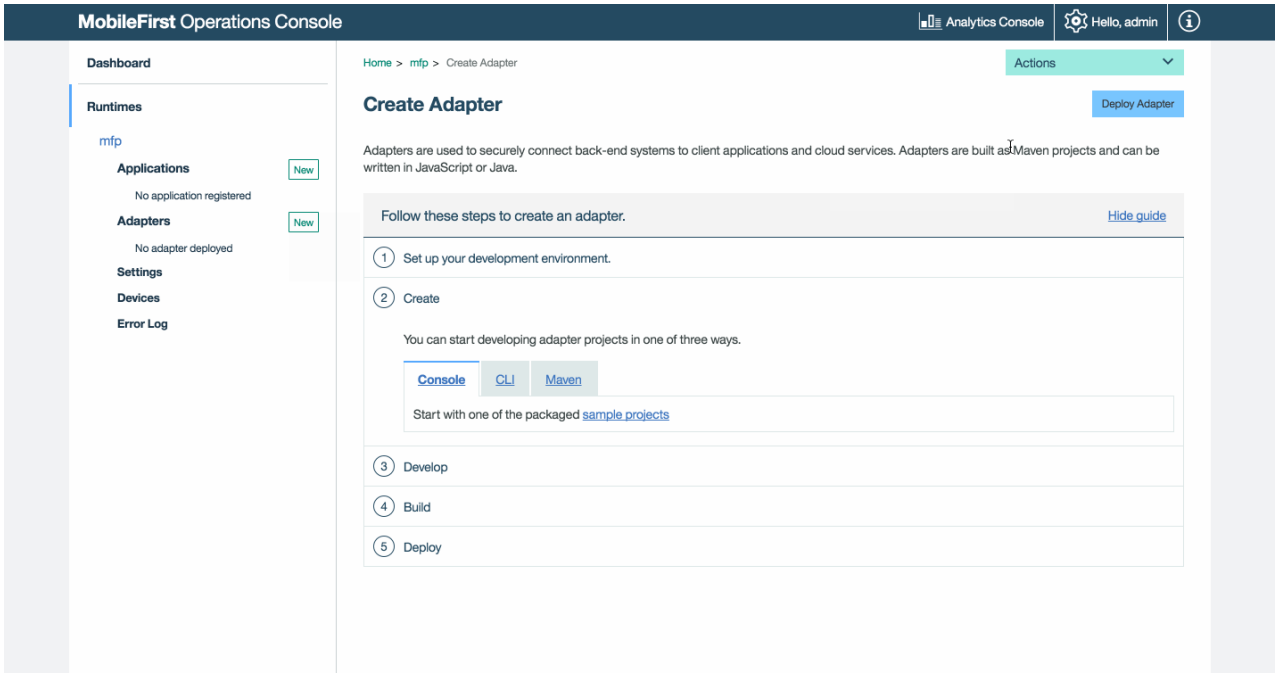
1. Select the **Actions → Download sample** option. Download the "Hello World" **Java** adapter sample.

   > If Maven and MobileFirst 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

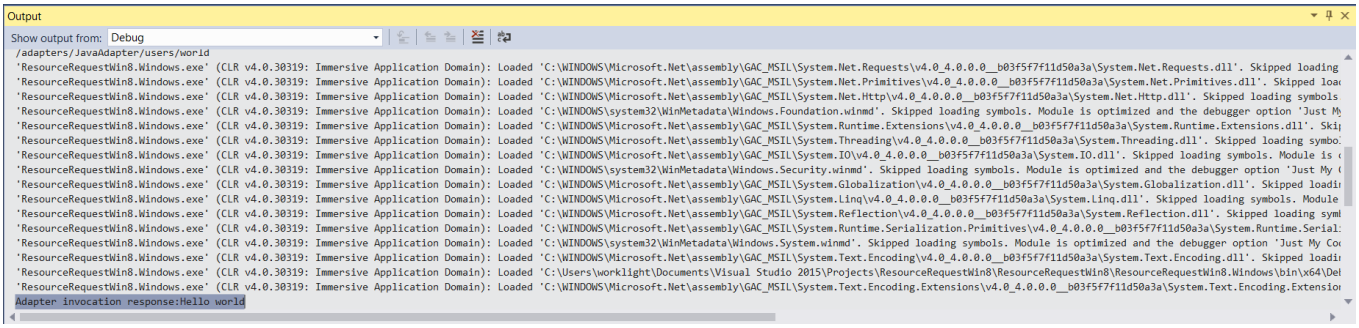
1. In Visual Studio, select the **mfpclient.resw** file and edit the **host** property with the IP address of the MobileFirst Server.

   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. Press 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 Visual Studio's Outpout console.



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)