

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 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 ping 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

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.cmd`.

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**

MobileFirst Operations Console

Home > mfp > Register Application

Register Application

Application Name
MFPStarterCSharpWindows ✓

Optional display name of the application

Choose Platform *
☐ Android ☐ iOS ☒ Windows ☐ Web

Choose Windows platform
☐ Windows 8.1 ☐ Windows Phone 8.1 ☒ Windows 10 UWP

Package Identity Name *
com.ibm.mfpstarterwindows ✓

Application identifier

Version *
1.0

The version information as found in the Package.appxmanifest file of the Visual Studio project, in the format Major.Minor.Build.Revision

[Register application](#)

* 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.

- Click on the **Get Starter Code** tile and select to download the Windows 8.1 or Windows 10 sample application.

MobileFirst Operations Console

Home > mfp > MFPStarterCSharpWindows > Windows 10 UWP 1.0

MFPStarterCSharpWindows

Windows 10 UWP v 1.0 | com.ibm.mfpstarterwindows

✓ Your application is now registered.

Next Steps

- [Get Starter Code](#)
- [Set Up Authenticity](#)
- [Set Up Push](#)
- [Get CLI](#)

Management | Authenticity | Security | Log Filters | Configuration Files

Last modified: May 28, 2016, 9:20 AM

Application Access

Status: *

☒ Active ☐ Active and Notifying ☐ Access Disabled

3. Editing application logic

- Open the Visual Studio project.
- Select the solution's **MainPage.xaml.cs** file and paste the following code snippet into the `GetAccessToken()` method:

```

try
{
    IWorklightClient _newClient = WorklightClient.CreateInstance();
    accessToken = await _newClient.AuthorizationManager.ObtainAccessToken("");
    if (accessToken.IsValidToken && accessToken.Value != null && accessToken.Value != "")
    {
        System.Diagnostics.Debug.WriteLine("Received the following access token value: " + accessToken.Value);
        titleTextBlock.Text = "Yay!";
        statusTextBlock.Text = "Connected to MobileFirst Server";

        Uri adapterPath = new Uri("/adapters/javaAdapter/resource/greet", UriKind.Relative);
        WorklightResourceRequest request = _newClient.ResourceRequest(adapterPath, "GET", "");
        request.SetQueryParameter("name", "world");
        WorklightResponse response = await request.Send();

        System.Diagnostics.Debug.WriteLine("Success: " + response.ResponseText);
    }
}
catch (Exception e)
{
    titleTextBlock.Text = "Uh-oh";
    statusTextBlock.Text = "Client failed to connect to MobileFirst Server";
    System.Diagnostics.Debug.WriteLine("An error occurred: '{0}'", e);
}

```

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.

5. Testing the application

1. In Visual Studio, select the **mfpclient.resw** file and edit the **protocol**, **host** and **port** properties 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**.

Alternatively, if you have installed the MobileFirst CLI, then navigate to the project root folder and run the command `mfpdev app register`. If a remote MobileFirst Server is used, run the command `mfpdev server add` (`../../application-development/using-mobilefirst-cli-to-manage-mobilefirst-artifacts/#add-a-new-server-instance`) to add the server, followed by for example: `mfpdev app register myBluemixServer`.

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 Output console.

```
Output
Show output from: Debug
/adapters/JavaAdapter/users/world
'ResourceRequestWin8.Windows.exe' (CLR v4.0.30319: Immersive Application Domain): Loaded 'C:\WINDOWS\Microsoft.Net\assembly\GAC_MSIL\System.Net.Requests\v4.0.4.0.0_b03f5f7f11d50a3a\System.Net.Requests.dll'. Skipped loading
'ResourceRequestWin8.Windows.exe' (CLR v4.0.30319: Immersive Application Domain): Loaded 'C:\WINDOWS\Microsoft.Net\assembly\GAC_MSIL\System.Net.Primitives\v4.0.4.0.0_b03f5f7f11d50a3a\System.Net.Primitives.dll'. Skipped load
'ResourceRequestWin8.Windows.exe' (CLR v4.0.30319: Immersive Application Domain): Loaded 'C:\WINDOWS\Microsoft.Net\assembly\GAC_MSIL\System.Net.Http\v4.0.4.0.0_b03f5f7f11d50a3a\System.Net.Http.dll'. Skipped loading symbols
'ResourceRequestWin8.Windows.exe' (CLR v4.0.30319: Immersive Application Domain): Loaded 'C:\WINDOWS\system32\WinMetadata\Windows.Foundation.winmd'. Skipped loading symbols. Module is optimized and the debugger option 'Just My C
'ResourceRequestWin8.Windows.exe' (CLR v4.0.30319: Immersive Application Domain): Loaded 'C:\WINDOWS\Microsoft.Net\assembly\GAC_MSIL\System.Runtime.Extensions\v4.0.4.0.0_b03f5f7f11d50a3a\System.Runtime.Extensions.dll'. Skipped loading symbols
'ResourceRequestWin8.Windows.exe' (CLR v4.0.30319: Immersive Application Domain): Loaded 'C:\WINDOWS\Microsoft.Net\assembly\GAC_MSIL\System.Threading\v4.0.4.0.0_b03f5f7f11d50a3a\System.Threading.dll'. Skipped loading symbols
'ResourceRequestWin8.Windows.exe' (CLR v4.0.30319: Immersive Application Domain): Loaded 'C:\WINDOWS\Microsoft.Net\assembly\GAC_MSIL\System.IO\v4.0.4.0.0_b03f5f7f11d50a3a\System.IO.dll'. Skipped loading symbols. Module is opti
'ResourceRequestWin8.Windows.exe' (CLR v4.0.30319: Immersive Application Domain): Loaded 'C:\WINDOWS\system32\WinMetadata\Windows.Security.winmd'. Skipped loading symbols. Module is optimized and the debugger option 'Just My C
'ResourceRequestWin8.Windows.exe' (CLR v4.0.30319: Immersive Application Domain): Loaded 'C:\WINDOWS\Microsoft.Net\assembly\GAC_MSIL\System.Globalization\v4.0.4.0.0_b03f5f7f11d50a3a\System.Globalization.dll'. Skipped loading symbols
'ResourceRequestWin8.Windows.exe' (CLR v4.0.30319: Immersive Application Domain): Loaded 'C:\WINDOWS\Microsoft.Net\assembly\GAC_MSIL\System.Linq\v4.0.4.0.0_b03f5f7f11d50a3a\System.Linq.dll'. Skipped loading symbols. Module
'ResourceRequestWin8.Windows.exe' (CLR v4.0.30319: Immersive Application Domain): Loaded 'C:\WINDOWS\Microsoft.Net\assembly\GAC_MSIL\System.Reflection\v4.0.4.0.0_b03f5f7f11d50a3a\System.Reflection.dll'. Skipped loading symbols
'ResourceRequestWin8.Windows.exe' (CLR v4.0.30319: Immersive Application Domain): Loaded 'C:\WINDOWS\Microsoft.Net\assembly\GAC_MSIL\System.Runtime.Serialization.Primitives\v4.0.4.0.0_b03f5f7f11d50a3a\System.Runtime.Seriali
'ResourceRequestWin8.Windows.exe' (CLR v4.0.30319: Immersive Application Domain): Loaded 'C:\WINDOWS\system32\WinMetadata\Windows.System.winmd'. Skipped loading symbols. Module is optimized and the debugger option 'Just My C
'ResourceRequestWin8.Windows.exe' (CLR v4.0.30319: Immersive Application Domain): Loaded 'C:\WINDOWS\Microsoft.Net\assembly\GAC_MSIL\System.Text.Encoding\v4.0.4.0.0_b03f5f7f11d50a3a\System.Text.Encoding.dll'. Skipped loading symbols
'ResourceRequestWin8.Windows.exe' (CLR v4.0.30319: Immersive Application Domain): Loaded 'C:\Users\worklight\Documents\Visual Studio 2015\Projects\ResourceRequestWin8\ResourceRequestWin8.Windows\bin\x64\Del
'ResourceRequestWin8.Windows.exe' (CLR v4.0.30319: Immersive Application Domain): Loaded 'C:\WINDOWS\Microsoft.Net\assembly\GAC_MSIL\System.Text.Encoding.Extensions\v4.0.4.0.0_b03f5f7f11d50a3a\System.Text.Encoding.Extensions
Adapter invocation response:Hello world
```

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 the Notifications tutorials (`../../notifications/`)
- Review All Tutorials (`../../all-tutorials`)

