

Windows 8.1 Universal and Windows 10 UWP end-to-end demonstration

fork and edit tutorial (<https://github.ibm.com/MFPSamples/DevCenter/tree/master/tutorials/en/foundation/8.0/quick-start/windows-8-10/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" Visual 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:

- Configured Visual Studio 2013/5
- 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 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 on the "New" button next to **Applications** and select the desired *platform*, *identifier* and *version* values.



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



3. Editing application logic

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

```

IWorklightClient _newClient = WorklightClient.CreateInstance();

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

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

WorklightResponse resp= return Task.Run<WorklightResponse> (() => {
    rr.send();
});

if (resp.success) {
    Debug.WriteLine("Success: " + resp.ResponseText);
} else {
    Debug.WriteLine("Failure: " + resp.error);
}

```

4. Creating an adapter

1. Click on the "New" button next to **Adapters**

- Select the **Actions → Download sample** option. Download the **Java** adapter sample.

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

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

```
mpfdev adapter build
```

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

The screenshot shows the MobileFirst Operations Console interface. The top navigation bar includes 'Analytics Console', 'Hello, admin', and an information icon. The left sidebar contains a menu with 'Dashboard', 'Runtimes', 'mfp', 'Applications' (with a 'New' button), 'Adapters' (with a 'New' button), 'Settings', 'Devices', and 'Error Log'. The main content area is titled 'Create a new Adapter' and includes a 'Deploy Adapter' button. Below the title, it states: 'Adapters are used to securely connect back-end systems to client applications and cloud services. Adapters are built as Maven projects and can be written in JavaScript or Java.' A section titled 'Follow these steps to create an adapter' lists five steps: 1. Set up your development environment, 2. Create, 3. Develop, 4. Build, and 5. Deploy. Step 2 is currently active. Under step 2, it says 'There are three ways to start developing adapter projects:' and provides buttons for 'Console', 'CLI', and 'Maven'. Below these buttons is a text input field with the placeholder 'Start with one of the packaged sample projects'.

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.
2. Press the **Run App** button.

The adapter response is then printed in the Visual Studio 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 symbols.
'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 loading symbols.
'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 Code' is enabled.
'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 optimized and the debugger option 'Just My Code' is enabled.
'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 Code' is enabled.
'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 is optimized and the debugger option 'Just My Code' is enabled.
'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.Serialization.Primitives.dll'. Skipped loading symbols.
'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 Code' is enabled.
'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\ResourceRequestWin8.Windows\bin\x64\Del
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 Server-side development tutorials ([../adapters/](#))
- Review the Authentication and security tutorials ([../authentication-and-security/](#))
- Review the Notifications tutorials ([../notifications/](#))
- Review All Tutorials ([../all-tutorials](#))