

# Quick Start demonstration

fork and edit tutorial (<https://github.com/MobileFirst-Platform-Developer-Center/DevCenter/#fork-destination-box>) | report issue (<https://github.com/MobileFirst-Platform-Developer-Center/DevCenter/issues/new>)

The purpose of this demonstration is to make you experience an end-to-end flow where the MobileFirst Platform Foundation SDK for Windows Phone 8 (Silverlight) is integrated into a Visual Studio project and used to retrieve data by using a MobileFirst adapter.

To learn more about creating projects and applications, using adapters, and lots more, visit the Native Windows Phone 8 Development (../) landing page.

**Prerequisite:** Make sure that you have installed the following software:

- MobileFirst Platform command line tool (download ([file:///home/travis/build/MFPSamples/DevCenter/\\_site/downloads](file:///home/travis/build/MFPSamples/DevCenter/_site/downloads)))
  - Visual Studio 2013
- 

## 1. Create a MobileFirst project and adapter.

- Create a new project and Windows Phone 8 framework/server-side application entity.

```
mfp create MyProject
cd MyProject
mfp add api MyWindowsPhone8 -e windowsphone8
```

- Add an HTTP adapter to the project.

```
mfp add adapter MyAdapter -t http
```

## 2. Deploy artifacts to the MobileFirst Server.

- Start the MobileFirst Server and deploy the server-side application entity and adapter.

```
mfp start
mfp push
```

## 3. Create a Visual Studio Windows Phone 8 Silverlight project.

## 4. Add a reference to the following libraries in your project.

- `worklight-windowsphone8.dll`
- `Newtonsoft.Json.dll`

## 5. Implement the MobileFirst adapter invocation.

- The following code invokes an adapter:

```
WLProcedureInvocationData invocationData = new WLProcedureInvocationData("MyAdapter", "getStories");
invocationData.setParameters(new Object[]{});
String myContextObject = "InvokingAdapterProceduresWP8";
WLRequestOptions options = new WLRequestOptions();
options.setInvocationContext(myContextObject);
WLClient.getInstance().invokeProcedure(invocationData, new MyInvokeListener(this), options);
```

## 6. Final configurations

- Copy the `wlclient.properties` file to the root of the native project.
- In Visual Studio, open the Properties window of `wlclient.properties` and set the **Copy to Output Directory** option to **Copy always**.
- Supply the server IP address to the `wlServerHost` property in `wlclient.properties`.
- Add the following capabilities to the `WMAAppManifest.xml` file:

`ID_CAP_NETWORKING`

`ID_CAP_IDENTITY_DEVICE`

## 7. Click Run.

Review the Visual Studio console for the data retrieved by the adapter request.

