Invoking adapter procedures from native Windows Phone 8 applications

fork and edit tutorial (https://github.ibm.com/MFPSamples/DevCenter/tree/master/tutorials/en/foundation/7.0/server-side-development/invoking-adapter-procedures-native-windows-phone-8-applications.html) | report issue (https://github.ibm.com/MFPSamples/DevCenter/issues/new) To create and configure a Windows Phone 8 native project, first follow the Creating your first Native WP8 MobileFirst application (../../hello-world/creating-first-native-windows-phone-8-mobilefirst-application/) tutorial.

This tutorial covers the following topics:

- Initializing WLClient
- Invoking an adapter procedure
- Receiving a procedure response
- Sample application

Initializing WLClient

```
WLClient client = WLClient.getInstance();
```

1. To establish a connection to MobileFirst Server, use the `connect` method by specifying the `MyConnectResponseListener` class instance as a parameter.

   ```
   client.connect(new MyConnectResponseListener(this));
   ```

   The `WLClient` instance tries to connect to the MobileFirst Server instance according to the properties of the `wlclient.properties` file.
   After the connection is established, it invokes one of the methods of the `MyConnectResponseListener` class.
2. Specify that the `MyConnectResponseListener` class implements the `WLResponseListener` interface.

   ```
   public class MyConnectResponseListener : WLResponseListener
   ```

   The `WLResponseListener` interface defines two methods:
   - `public void onSuccess (WLResponse response) { }`
   - `public void onFailure (WLFailResponse response) { }`
3. Use the previous methods to process connection success or connection failure.

Invoking an adapter procedure

After the connection is established with a MobileFirst Server instance, you can use the `WLClient` instance to invoke adapter procedures.

1. Create a `WLProcedureInvocationData` object with the adapter and procedure names.
2. Add the required parameters as an object array and set request options (for example: Invocation Context).
3. Get the existing `WLClient` instance and use it to invoke an adapter procedure.
4. Specify the `MyInvokeListener` class instance as a parameter.

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

Receiving a procedure response

After the procedure invocation is completed, the `WLClient` instance calls one of the methods of the `MyInvokeListener` class.
As before, you must specify that the `MyInvokeListener` class implements the `WLResponseListener` interface.

```
using IBM.Worklight;
namespace InvokingAdapterProceduresWP8{
  public class MyInvokeListener :
WLResponseListener
    { }
{
```

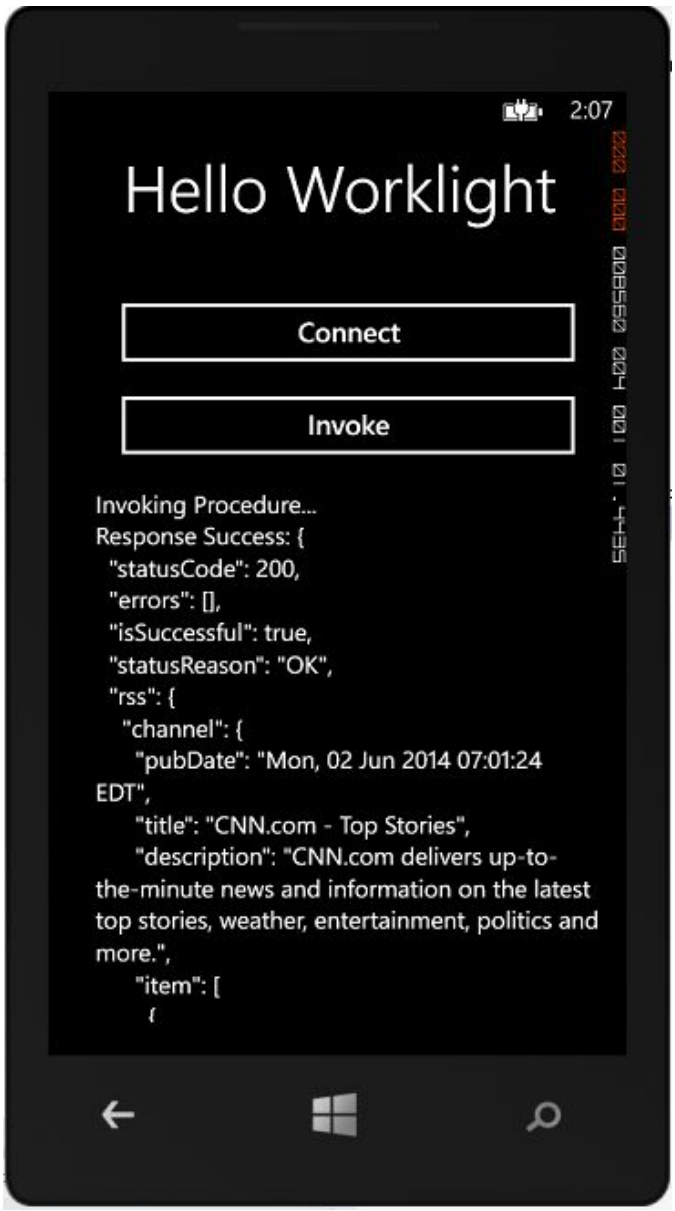
The `onSuccess` and `onFailure` methods are invoked by the `WLClient`. The response object contains the response data. You can use its methods and properties to retrieve the required information.

```
public void onSuccess(WLResponse response)
{
    WLProcedureInvocationResult invocationResponse = ((WLProcedureInvocationResult) response);
>
    JObject items;
    try
    {
        items = invocationResponse.getResponseJSON();
        Deployment.Current.Dispatcher.BeginInvoke(() =>
        {
            myMainPage.AddTextToReceivedTextBlock("Response Success: " + items.ToString());
        });
    }
    catch (JsonReaderException e)
    {
        Deployment.Current.Dispatcher.BeginInvoke(() =><
        {
            myMainPage.AddTextToReceivedTextBlock("JSONException : " + e.Message);
        });
    }
}

public void onFailure(WLFailResponse response)
{
    Deployment.Current.Dispatcher.BeginInvoke(() =>
    {
        myMainPage.AddTextToReceivedTextBlock("Response failed: " + response.ToString());
    });
}
```

Sample application

Click to download (http://public.dhe.ibm.com/software/products/en/MobileFirstPlatform/docs/v700/InvokingAdapterProceduresNativeProject.zip) the Studio project.
Click to download (http://public.dhe.ibm.com/software/products/en/MobileFirstPlatform/docs/v700/InvokingAdapterProceduresWP8Project.zip) the Native project.

The sample contains two projects:
- The `InvokingAdapterProceduresNativeProject.zip` file contains a **MobileFirst Native API** to deploy to MobileFirst Server.
- The `InvokingAdapterProceduresWP8project.zip` file contains a **native Windows Phone 8 application** that uses a MobileFirst native API library to communicate with a MobileFirst Server instance.

Make sure to update the `wlclient.properties` file in `InvokingAdapterProceduresWP8` with the relevant server settings.