Resource request from native iOS Objective-C applications

Overview

To create and configure an iOS native project, first follow the “Configuring a native iOS application with the MobileFirst Platform SDK (../../hello-world/configuring-a-native-ios-application-with-the-mfp-sdk/)” tutorial.

MobileFirst applications can access resources using the `WLResourceRequest` REST API.
This tutorial explains how to use the `WLResourceRequest` API with an HTTP adapter.

Initializing WLClient

1. Access the `WLClient` functionality by using
   `[WLClient sharedInstance]` anywhere in your application.
2. Initiate the connection to the server by using the `wlConnectWithDelegate` method.
   For most actions, you must specify a delegate object, such as a `MyConnectListener` instance in the following example:

   ```
   MyConnectListener *connectListener = [[MyConnectListener alloc] initWithController:self];
   [[WLClient sharedInstance] wlConnectWithDelegate:connectListener];
   ```

   Note: Remember to `#import` when using the MobileFirst SDK.

3. Create a delegate to be used in the `wlConnectWithDelegate` method and receive the response from the MobileFirst Server instance. Name the class `MyConnectListener`.
   The header file must specify that it implements the `WLDelegate` protocol.

   ```
   @interface MyConnectListener : NSObject <WLDelegate> {
      @private
      ViewController *vc;
   }
   <p>
   ```

   The `WLDelegate` protocol specifies that the class implements the following methods:
   - The `onSuccess` method: `(WLResponse *)response`
   - The `onFailure` method: `(WLFailResponse *)response`

   After `wlConnectWithDelegate` finishes, the `onSuccess` method or the `onFailure` method of the supplied `MyConnectListener` instance is invoked.
   In both cases, the response object is sent as an argument.

4. Use this object to operate data that is retrieved from the server.

```
-(void)onSuccess:(WLResponse *)response{
    NSLog(@"\nConnection Success: %@", response);
    NSString *resultText = @"Connection success. ";
    if ([response responseText] != nil){
        resultText = [resultText stringByAppendingString:[response responseText]]
;
    }
    [vc updateView:resultText];
}
-(void)onFailure:(WLFailResponse *)response{
    NSString *resultText = @"Connection failure. ";
    if ([response responseText] != nil){
        resultText = [resultText stringByAppendingString:[response responseText]]
;
    }
    [vc updateView:resultText];
}
```

Invoking an adapter procedure

After the connection is established with a MobileFirst Server instance, you can use the `WLResourceRequest` class to invoke adapter procedures or call any REST resources.

1. Define the URI of the resource. For a JavaScript HTTP adapter:
   `/adapters/{AdapterName}/{ProcedureName}`

   ```
   NSURL* url = [NSURL URLWithString:@"/adapters/RSSReader/getFeed"];
   ```

2. Create a `WLResourceRequest` object and choose the HTTP method (GET, POST, etc).

   ```
   WLResourceRequest* request = [WLResourceRequest requestWithURL:url method:WLHttpMethod
   Get];
   ```

3. Add the required parameters.
   - For JavaScript-based adapters, use the `params` parameter name to set an array of parameters.

     ```
     [request setQueryParameterValue:@"['MobileFirst_Platform']" forName:@"params"];
     ```

   - For Java adapters or other resources, you can use `setQueryParameterValue` for each parameter.

     ```
     [request setQueryParameterValue:@"value1" forName:@"param1"];
     [request setQueryParameterValue:@"value2" forName:@"param2"];
     ```

4. Trigger the request with a call to the `sendWithCompletionHandler` method.

Specify a `completionHandler` instance.

```
[request sendWithCompletionHandler:^(WLResponse *response, NSError *error) {
    NSString* resultText;
    if(error != nil){
        resultText = @"Invocation failure.";
        resultText = [resultText stringByAppendingString: error.description];
    }
    else{
        resultText = @"Invocation success.";
        resultText = [resultText stringByAppendingString:response.responseText];
    }
    [self updateView:resultText];
}];
```

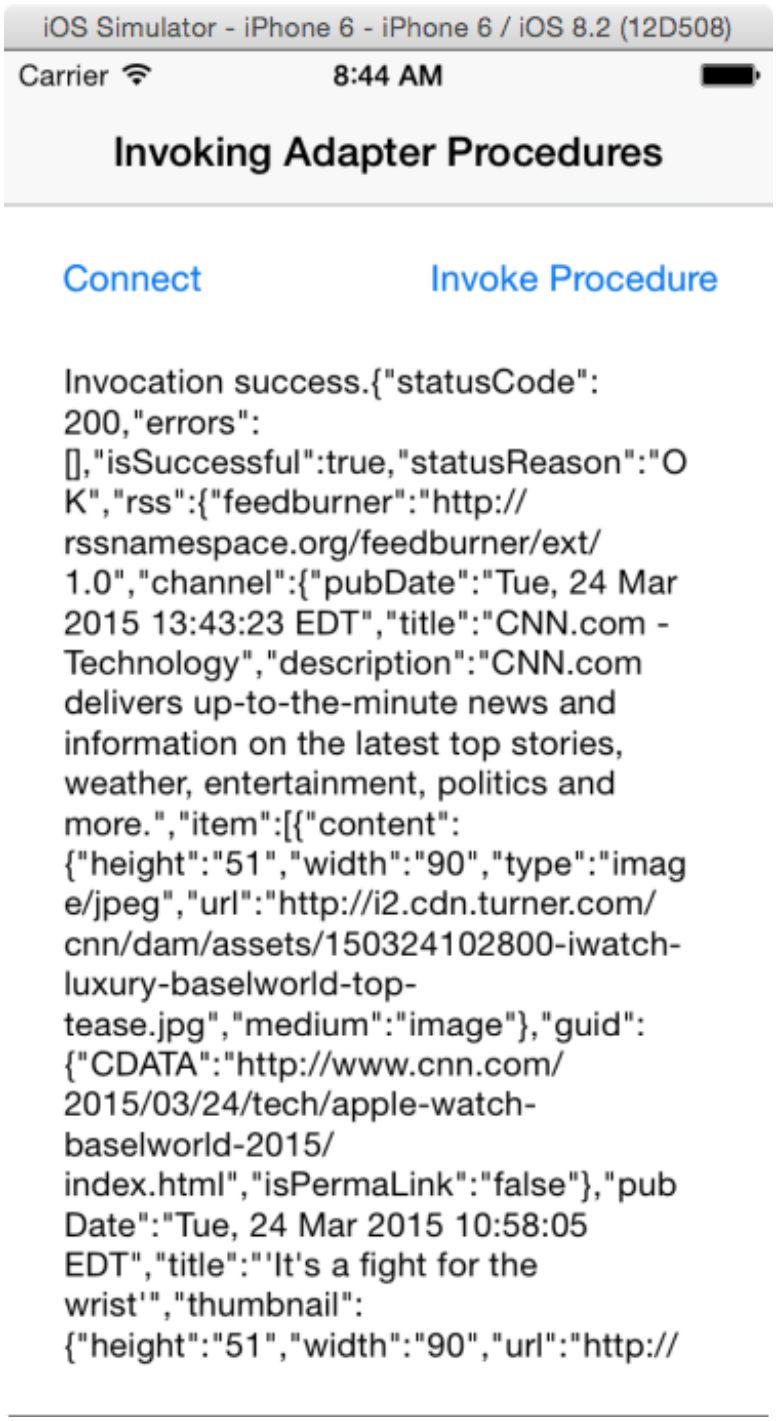
Other signatures, which are not covered in this tutorial, exist for the `send` method. Those signatures enable you to set parameters in the body instead of the query, or to handle the response with a delegate instead of a completion handler. See the user documentation to learn more.

Sample and result

Click to download (https://github.com/MobileFirst-Platform-Developer-Center/InvokingAdapterProcedures) the MobileFirst project.

Click to download (https://github.com/MobileFirst-Platform-Developer-Center/InvokingAdapterProceduresObjC) the Native project.

- The `InvokingAdapterProcedures` project contains a **MobileFirst native API** which you can deploy to your MobileFirst Server instance and required to deploy to the server.
- The `InvokingAdapterProceduresObjC` project contains a **native iOS application** that uses a MobileFirst native API library to communicate with the MobileFirst Server instance.
- Make sure to update the `worklight.plist` file in **NativeiOSInvoking** with the relevant server settings.


iOS Simulator - iPhone 6 - iPhone 6 / iOS 8.2 (12D508)
Carrier
8:44 AM
Invoking Adapter Procedures
Connect
Invoke Procedure
Invocation success.{"statusCode":
200,"errors":
[],"isSuccessful":true,"statusReason":"O
K","rss":{"feedburner":"http://
rssnamespace.org/feedburner/ext/
1.0","channel":{"pubDate":"Tue, 24 Mar
2015 13:43:23 EDT","title":"CNN.com -
Technology","description":"CNN.com
delivers up-to-the-minute news and
information on the latest top stories,
weather, entertainment, politics and
more.","item":[{"content":
{"height":"51","width":"90","type":"imag
e/jpeg","url":"http://i2.cdn.turner.com/
cnn/dam/assets/150324102800-iwatch-
luxury-baselworld-top-
tease.jpg","medium":"image"},"guid":
{"CDATA":"http://www.cnn.com/
2015/03/24/tech/apple-watch-
baselworld-2015/
index.html","isPermaLink":"false"},"pub
Date":"Tue, 24 Mar 2015 10:58:05
EDT","title":"'It's a fight for the
wrist'","thumbnail":
{"height":"51","width":"90","url":"http://