Resource Request from Native iOS Objective-C Applications

RENAMING

Overview

MobileFirst applications can access resources using the `WLResourceRequest` REST API.
The REST API works with all adapters and external resources LINK TO using-mobilefirst-server-authenticate-external-resources.
This tutorial explains how to use the `WLResourceRequest` API with an HTTP adapter.

To create and configure an iOS native project, first follow the Adding the MobileFirst Platform Foundation SDK to iOS Applications (../../adding-the-mfpf-sdk/adding-the-mfpf-sdk-to-ios-applications) tutorial.

Calling an adapter procedure

The `WLResourceRequest` class handles resource requests to MobileFirst adapters or external resources.

1. Define the URL of the resource:

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

   - For JavaScript adapters, use `/adapters/{AdapterName}/{procedureName}`
   - For Java adapters, use `/adapters/{AdapterName}/{path}`
   - To access resources outside of the project, use the full URL
2. Create a `WLResourceRequest` object and choose the HTTP method (GET, POST, etc):

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

3. Add the required parameters:

   - In JavaScript adapters, which use ordered nameless parameters, pass an array of parameters with the name `params`:

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

   - In Java adapters or external resources, use the `setQueryParameter` method for each parameter:

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

4. Call the procedure by using the `sendWithCompletionHandler` method.
   Supply a completion handler to manage the retrieved data:

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

Use the `response` and `error` objects to get the data that is retrieved from the adapter.
The `response` object contains the response data and you can use its methods and properties to retrieve the required information.

> There are also other signatures for the `send` method, which are not covered in this tutorial. Those signatures enable you to set parameters in the body instead of the query and provide more granular management of the retrieved data (such as non-text responses, PDF, etc). You can use the `sendWithDelegate` method and provide a delegate that conforms to both the `NSURLConnectionDataDelegate` and `NSURLConnectionDelegate` protocols.
> See the user documentation to learn more about `WLResourceRequest`.

Sample application

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 `mfpclient.plist` file in **NativeiOSInvoking** with the relevant server settings.

SCREENSHOT