Resource Request from Native iOS Swift Applications

Overview

MobileFirst applications can access resources using the `WLResourceRequest` REST API. The REST API works with all adapters and 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.

WLResourceRequest

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

1. To call a resource, create a `WLResourceRequest` object and specify the path to the adapter and the HTTP method(GET, POST, etc):

   ```
   let request = WLResourceRequest(URL: NSURL(string: "/adapters/RSSReader/getFeed"), method:
   WLHttpMethodGet)
   ```

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

     ```
     request.setQueryParameterValue("['param1', 'param2']", 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")
     ```

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

```
request.sendWithCompletionHandler { (WLResponse response, NSError error) -> Void in
    var resultText = ""
    if(error != nil){
        resultText = "Successfully called the resource"
        resultText += error.description
    }
    else if(response != nil){
        resultText = "Failed to call the resource"
        resultText += response.responseText
    }
    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/ResourceRequestSwift) the Native project.

- The ResourceRequestSwift project contains a native iOS Swift application that uses a MobileFirst native SDK to communicate with the MobileFirst Server instance.
- Make sure to update the mfpclient.plist file in the native iOS project with the relevant server settings.

SCREENSHOT