Invoking adapter procedures from hybrid client applications

Overview

MobileFirst applications can adapt procedures to communicate with any data source without being subjected to same-origin constraints.

Basic API

Adapter procedures can be called from the client-side JavaScript:

```
WL.Client.invokeProcedure(invocationData, options);
```

invocationData

The first step to invoke an adapter procedure in JavaScript is to create an `invocationData` object. The `invocationData` object is used to provide invocation configuration and procedure parameters. It consists of a JSON block of properties.

```
var invocationData = {
  adapter : 'RSSReader',
  procedure : 'getFeedsFiltered'
,
  parameters : []
};
```

The properties are:

- **adapter** (mandatory): A string that contains the name of the adapter as specified in the adapter XML file.
- **procedure** (mandatory): The procedure name as defined in the XML file.
- **parameters** (mandatory): An array of parameters that are passed to the back-end JavaScript™ procedure. Leave this property empty if no parameters are required.

Options

The second step is to define failure and success behaviors in the `options` object.

```
var options = {
  onSuccess : loadFeedsSuccess
,
  onFailure : loadFeedsFailure,
  invocationContext: {}
};
```

The `options` object must be passed for all asynchronous calls to MobileFirst Server.

onSuccess

The `onSuccess` function is to be invoked on successful completion of the asynchronous call.

The `response` typically contains the following properties:

- **`invocationContext`**: The `invocationContext` object that was originally passed in the `options` object, or

undefined if no `invocationContext` object was passed
- **`status`**: The HTTP response status
- **`invocationResult`**: An object that contains the data that is returned by the invoked procedure, and additional information about the procedure invocation.

onFailure

The `onFailure` function is to be invoked on failure.

It includes both server-side and client-side errors (such as server connection failure or timed out calls).
The `response` typically contains the following properties:

- **`invocationContext`**: The `invocationContext` object that was originally passed in the `options` object, or undefined if no `invocationContext` object was passed.
- **`status`**: The HTTP response status
- **`invocationResult`**: An object that contains the data that is returned by the invoked procedure, and additional information about the procedure invocation.

invocationContext

`invocationContext` is an optional parameter. It is an object that is returned to the success and failure handlers.

The `invocationContext` object is used to preserve the context of the calling asynchronous service upon return from the service.

Invocation Results

`invocationResult` is a JSON object that is returned. It contains the data and more information about the procedure invocation.
The object is returned to a corresponding success/failure handler.

```
{
  "errors": [],
  "info": [],
  "warnings": [],
  "isSuccessful": true,
  "responseHeaders": {
    "Cache-Control": "no-cache, must-revalidate, post-check=0, pre-check=0"
  },
  "responseTime": 491,
  "statusCode": 200,
  "statusReason": "OK",
  "totalTime": 592,
  "Items": [{
    "creator": "Jon Fingas",
    "link": "http:\/\/www.engadget.com\/2014\/11\/10\/harvard-used-cameras-to-check-attendance\/?ncid=rss_truncated"
,
    "pubDate": "Mon, 10 Nov 2014 02:21:00 -0500",
    "title": "Harvard used cameras to track attendance without telling students"
  }, {
    "creator": "Jon Fingas",
    "link": "http:\/\/www.engadget.com\/2014\/11\/10\/bmw-ev-charging-street-lights\/?ncid=rss_truncated",
    "pubDate": "Mon, 10 Nov 2014 00:10:00 -0500",
    "title": "BMW's new street lights will charge your electric car"
  }, {
    "creator": "Daniel Cooper",
    "link": "http:\/\/www.engadget.com\/2014\/11\/09\/hwyc-lumia-925\/?ncid=rss_truncated",
    "pubDate": "Sun, 09 Nov 2014 22:43:00 -0500",
    "title": "How would you change Nokia's Lumia 925?"
  }]
}
```


`errors`, `info`, and `warnings` are optional arrays of strings that contain messages.

The `isSuccessful` property is set to `true` if the procedure invocation succeeded (even if no data was retrieved), to `false` otherwise.

The response can contain other metadata such as `responseHeaders`, `responseTime`, `statusCode`, `statusReason`, and `totalTime`.

The rest of the invocation result depends on what was retrieved from the back-end system. In this example, the `Items` element is a JSON representation of the XML code that was received from the back-end, after applying the rules in the XSL file.

```
function loadFeedsSuccess(result){
   WL.Logger.debug("Feed retrieve success");
   if (result.invocationResult.Items.length>0)
      displayFeeds(result.invocationResult.Items)
;
}
```

Sample application

Click to download (http://public.dhe.ibm.com/software/products/en/MobileFirstPlatform/docs/v630/InvokingAdapterProceduresHybridProject.zip) the Studio project.

The attached sample uses the HTTP adapter created in the HTTP Adapter tutorial (../http-adapter-communicating-http-back-end-systems/).

Engadget Reader

New surgical robot makes it easier to perform complicated surgeries (video)
Tue, 01 Apr 2014 08:30:00 -0400

April Fools' 2014 roundup: selfie-taking drones, smartgloves and more
Tue, 01 Apr 2014 07:00:00 -0400

Tesla Model S is 'low hanging fruit' for hackers to remotely track or unlock cars
Tue, 01 Apr 2014 03:41:00 -0400

Verizon FiOS Media Server arrives as Quantum TV DVR that records up to 12 channels at once
Tue, 01 Apr 2014 00:40:00 -0400

Square Market now accepts Bitcoins
Mon, 31 Mar 2014 23:03:00 -0400