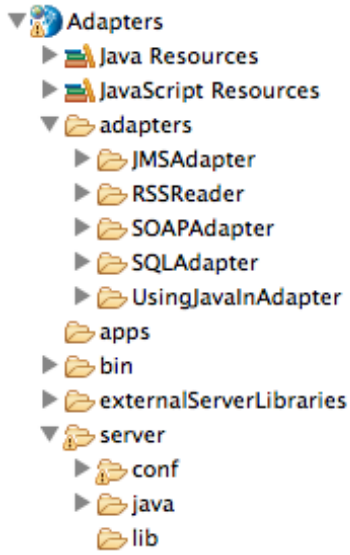
Using Java in adapters

Overview

Adapters are server-side entities implemented in JavaScript.
To learn more about adapters in general, start with the Adapters Overview (../../server-side-development/adapter-framework-overview/) tutorial.

When JavaScript is not sufficient to implement these functions, or if a Java class exists, you can use Java code in your adapter.
The Java code is an extension of the adapter function and not a replacement.

Adding custom Java classes to your project



1. To use an existing Java library, add the JAR file to the **server\lib** folder of your MobileFirst project.
   After the adapter is built and deployed, this JAR file is automatically deployed to MobileFirst Server.
2. To add custom Java code to your project, right-click the **server/java** folder in your Worklight project and add a Java class file. Name it **Calculator.java**.
   Important: The package name must start with either **com**, **org**, or **net**.
3. Add this file to a package. In this sample, the `com.sample.customcode` is used.
   This package name can be interpreted as folders: **java\com\sample\customcode**

4. Add methods to your **Calculator.java** class.
   Here is an example of a static method (does not require a new instance) and of an instance method.

```
package com.sample.customcode;
public class Calculator {
 // Add two integers.
 public static int addTwoIntegers(int first, int second){
  return first + second;
 }
 // Subtract two integers.
 public int subtractTwoIntegers(int first, int second){
  return first - second;
 }
}
```

5. If your Java code has additional dependencies, put the required JAR files in the **server\lib** folder of your MobileFirst project.

Invoking custom Java classes from the adapter

After your custom Java code is created and any required JAR files are added, you can reference it from your MobileFirst Adapter JavaScript as if it were written in Java.

Invoke the static Java method as shown, and use the full class name to reference it directly.

```
function addTwoIntegers(a,b){
 return {
  result: com.sample.customcode.Calculator.addTwoIntegers(a,b
)
 };
}
```

To use the instance method, create a class instance and invoke the instance method from it.

```
function subtractTwoIntegers(a,b){
 var calcInstance = new com.worklight.customcode.Calculator();
 return {
  result : calcInstance.subtractTwoIntegers(a,b)
 };<
}
```

Sample application

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