Using Java in JavaScript Adapters

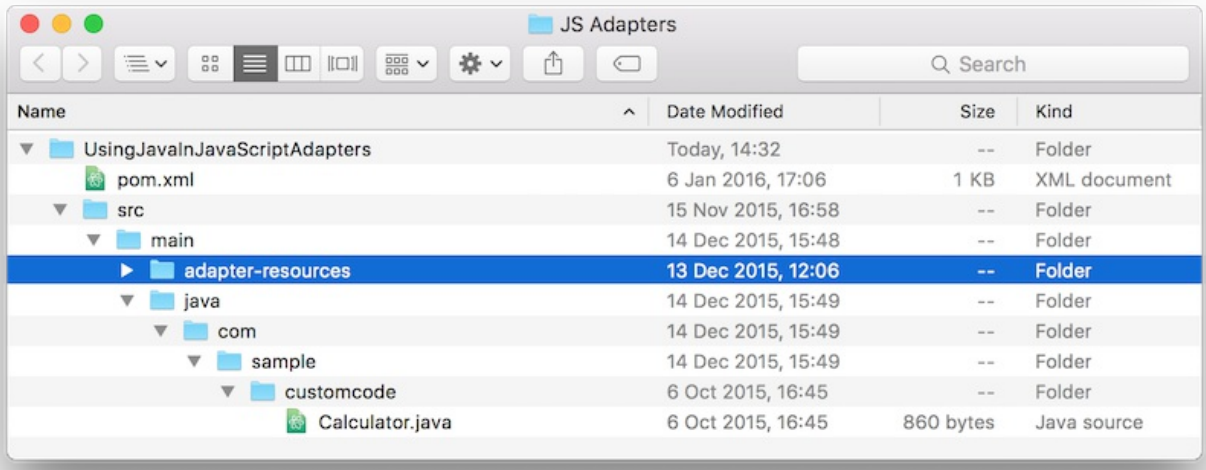
fork and edit tutorial (https://github.ibm.com/MFPSamples/DevCenter/tree/master/tutorials/en/foundation/8.0/adapters/javascript-adapters/using-java-in-javascript-adapters/index.md) | report issue (https://github.ibm.com/MFPSamples/DevCenter/issues/new)

Overview

When JavaScript is not sufficient to implement required functionality, or if a Java class already exists, you can use Java code as an extension for the JavaScript adapter.

Prerequisite: Make sure to read the JavaScript Adapters (../) tutorial first.

Adding custom Java classes



To use an existing Java library, add the JAR file as a dependency in your project. For more information on how to add a dependency, see the Dependencies section in the Creating Java and JavaScript Adapters (../../creating-adapters/#dependencies) tutorial.

To add custom Java code to your project, add a folder named **java** to the **src**/**main** folder in your adapter project and put your package in it. The sample in this tutorial uses a `com.sample.customcode` package and a Java class file named `Calculator.java`.

> **Important:** The package name must start with either `com`, `org`, or `net`.

Add methods to your Java class.
Here are an examples of a static method (that does not require a new instance) and an instance method:

```
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;
  }
}
```

Invoking custom Java classes from the adapter

After your custom Java code is created and any required JAR files are added, you can call it from the JavaScript code:

- 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.sample.customcode.Calculator();
    return {
      result : calcInstance.subtractTwoIntegers(a,b)
    };
  }
  ```

Sample

Click to download (https://github.com/MobileFirst-Platform-Developer-Center/Adapters/tree/release80) the Maven project.

Sample usage

- Use either Maven or MobileFirst Developer CLI to build and deploy the adapter (../../creating-adapters/).