**ServiceAutoTest**

**Purpose**

To compare responses from two different endpoints for SOAP and REST services

**Inputs**

Each test case folder will need the following:

* config.json file (refer to the JSON and XML test cases
  + REST Test Cases
    - EndPoint1: string
    - Endpoint2: string
    - Headers: list of key value pairs strings to pass into the header of the request
    - CallType: “REST”
    - HttpMethod: rest action, options are “GET”, “PUT”, ”POST”, ”DELETE”, ”PATCH”
  + SOAP Test Cases
    - EndPoint1: string
    - Endpoint2: string
    - Headers: list of key value pair strings to pass into the header of the request
    - CallType: “SOAP”
    - SOAPAction: string
* payload.txt file
  + Contains the raw text for request (xml/json)
* Example test cases are available in the docs folder (XmlTestCase, RestTestCase)

**Outputs**

Once a test case is processed, the following files will be produced:

* yyyy-MM-dd\_logfile.txt
* endpoint1.resp
* endpoint2.resp
* Archive folder with a zip of the files used in the last run
* reportResults.html containing the compare between the endpoint1.resp and endpoint2.resp
* When processing json, two additional files will be produced: endpoint1.response.json and endpoint2.response.json. These are the original json responses. These two files are converted into xml as endpoint1.resp and endpoint2.resp and compared

**Response Comparison**

The xml is compared using an old but effective library from Microsoft called XMLDiffPatch. The options currently configured in the BokServiceAutoTest console app are XmlDiffOptions.IgnoreChildOrder, XmlDiffOptions.IgnorePrefixes, XmlDiffOptions.IgnoreWhitespace, XmlDiffOptions.IgnoreComments.

Below is a list of XmlDiffOptions and an explanation of what they do.

<https://documentation.help/Microsoft-XML-Diff/xmldiffpatch_xmldiffoptions_ref_1tsz.htm>

**App Configuration**

|  |  |  |
| --- | --- | --- |
| **appsettings.json** | | |
| **Attribute Name** | **Attribute Type** | **Attribute Description** |
| TestCaseBaseFolder | String | This is the directory that holds all of the test case folders. For example: "C:\\Test\\BokServiceAutoTest" |
| ConfigFileName | String | Name of the config file to look for in each test case folder. For example:  "config.json" |
| PayloadFileName | String | Name of the payload file to look for in each test case folder. For example: "payload.txt", |
| ReportFileName | String | Name of the generated report file in each test case folder. For example:  "reportResults.html" |
| PollingIntervalSeconds | Int | This is a bit of weird one and more useful if we were processing folders or files from an input directory and removing them but it’s necessary to trigger the batching for this code and handy should we convert this from a console to a service. It’s essentially reloading the batches but does not interrupt existing batches being processed. So in short it’s probably optimal to set at 3 seconds. Anyway, an example value is 3. |
| DegreesOfParallelism | Int | The maximum number of test cases to process in parallel. If any test cases finish before the PollingIntervalSeconds has elaspsed, other unprocessed test cases will start until reaching the number specified in DegreesOfParallelism. For example: 5 |
| ArchiveTestCaseRuns | Boolean | When set to true, the files in the test case folder except for logs older than 24 hours will be zipped in an “Archive” folder within the test case folder as yyyyMMddHHmmss.zip using the date created attribute of the endpoint1.resp and the current date and time should endpoint1.resp not exist. An example value is true or false |