Joystick - API - v2

All request responses (api, sql server, mongo) are returned in an object named APIResponse with the following fields:

TestName = Scenario Name in SpecFlow feature file

RequestType = Type of request (HttpRequest-GET, HttpRequest-PUT, HttpRequest-POST, SqlServerRequest-JSON, MongoDBRequest-JSON)

Environment = Environment SpecFlow Scenario was run against

RequestBody = the content of the request (JSON, SQL, Mongo)

ResponseCode = response of request

ResponseBody = a json formatted string of the response data

RequestDurationMilliseconds = the duration of the request in milliseconds

Given steps

api url

Sets the url variable that will be used by the api request

Note: Configuration code will need to be added to the APIFramework_Steps.cs file to allow for this.

Examples:

```
Given api url https://test.com/test

# Adding an entry from the Resource file
# Example: TestRun_Resources.URL = https://test.com
Given api url %testrun_resources.url%/test
# url is now: https://test.com/test
```

path

Sets a path entry (/value) at the end of url provided in the api url step

Examples:

```
# Set the url for the api request
Given api url https://test.com/test
# add a path value to the end of the url
And path test2
# request url now has test2 added to the path: https://test.com/test/test2

# add a path variable to the end of the url
Given var %test% as test2 type string
And path %test%
# request url now has test2 added to the path: https://test.com/test/test2
```

param

Sets a query parameter at the end of the url provided in the api url step

Examples:

```
# Set the url for the api request
Given api url https://test.com/test
# add a query parameter value to the end of the url
And param id value test2
# request url now has ?id=test2 added to the path:
#https://test.com/test?id=test2

# add a query parameter variable to the end of the url
Given var %test% as test2 type string
And param id value %test%
# request url now has ?id=test2 added to the path:
#https://test.com/test?id=test2

# add a query parameter from a previous request, must be prefixed with
#response.
And param id value response.id
```

var

Sets a named variable for the life of the SpecFlow Scenario. Variable names must be surrounded by %, for example %variablename%

Examples:

```
# variable names must be surrounded with % example: %testvalue%
# data types available are string, int, and date

# assigning a string value
Given var %stringValue% as 'ABCDEFGH' type string

# assigning a int value
Given var %intValue% as 2 type int

# assigning specific date
Given var %dateValue% as 05/01/2009 14:57:32 type date

# assigning the current date
Given var %dateValue% as now type date

# assigning a variable from the previous api response using path.
# must prefix the value with 'response.'
# example json previous api response: {'id': 0,  'name': 'string' }
Given var %testid% is equal to response.id type string
Then print %testid% debug
# Debug Window should show: %testid% = 0
```

request json

Sets a json string that will be used in the API request.

Tokens in the json string will be replaced with variable data if the token and variable are named the same.

Examples:

```
# Set the url for the api request
Given api url https://test.com/test

# Set the request json
And request json { 'id': 1061, 'category': {'id': 0,  'name': 'string' }}
```

```
# Replacing tokens in json string with variables
Given api url https://test.com/test
And var %testid% as 1061 type string
And request json { 'id': %testid%, 'category': {'id': 0,  'name': 'string'
}}
# after token/variable replacement of %testid%, json will be: { 'id': 1061,
'category': #{'id': 0,  'name': #'string' }}
```

request read json file <file name>

Loads the json string that will be used in the API request from a file on the file system or source control.

Tokens added to the file will be replaced with variable data if the token and variable are named the same. (see request json above)

Examples:

```
# Set the url for the api request
Given api url https://test.com/test
# Set the json file to be used with the request
And request read json file ..\testfile.json
```

request sqlserver <sql statement>

Sets the SQL query statement for the *method sqlserver* step request.

Note: all SQL server responses are serialized to json with a root node of **sqlresult** in the following format

```
{
"sqlresult" : [
     {
      <first row of data = 0>
      <second row of data = 1>
     }
]}
```

Asserting responses are done through the existing assert json response steps using the following format

sqlresult[row number].ColumnName

Examples:

```
# Set a var for test case and perform sql server request.
Given var %test% as test type string
And request sqlserver SELECT Test FROM TestTable WHERE id = %test%
And sqlserver connection string <Data Source ...>
When method sqlserver
Then assert json response sqlresult[0].Test is equal to test
And assert json response sqlresult[0].Test contains test

# assert using variable
Then assert json response sqlresult[0].Test is equal to %test%
And assert json response sqlresult[0].Test contains %test%
```

request read sqlserver file <file name> (Future enhancement)

Loads the json string that will be used in the method sqlserver request from a file on the file system.

Tokens added to the file system file will be replaced with variable data if the token and variable are named the same. (see request json above)

sql server connection string

Sets the connection string that will be used in the request of the method sqlserver step

```
Given sqlserver connection string <Data Source ...>
When method sqlserver
```

When steps

method

Sets a request method for the request and executes the request

Http requests available:

- GET
- POST
- PUT

Sql Server requests available:

- SQLServer

MongoDB requests available (Future Enhancement)

- MongoDB

```
# Set the url for the api request
Given api url https://test.com/test
# Set the json file to be used with the request
And request read json file ..\testfile.json

# Send the POST request
When method POST

# Send a GET request
When method GET

# Send a PUT request
When method PUT

# Send a SQL Server request
Given request sqlserver SELECT TOP 1 * FROM TestTable
When method sqlserver
Then assert json response sqlresult[0].Test is equal to test

#Send a MongoDb request
Given request mongodb from file .\mongodb_query.txt
When method mongodb
Then assert json response id is equal to Test
```

add <product name> cookie to method

Adds an authentication cookie header by product name to the request, sets a request method for the request and executes the request

<u>Products available</u>:

- GPS

<u>Http requests available</u>:

- GET
- POST
- PUT

Examples:

```
# Set the url for the api request
Given api url https://test.com/test
# Set the json file to be used with the request
And request read json file ..\testfile.json

# Send the POST request
When add TestProduct cookie method POST
```

add bearer <access_token> to method

Adds an authentication bearer header to the request, sets a request method for the request and executes the request

Examples:

```
Given var %bearer% is equal to response.access_token type string
When add bearer %bearer% to method GET
```

<u>Then steps</u>

assert json response <response> is equal to <value>

All request responses are returned as json responses. The response json is added to the APIResponse object’s ResponseBody field. The assert json response steps allow verification of the json responses.

Note the *assert is equal to* steps perform an exact match assert.

Examples:

```
# assert response is equal to value
Then assert json response id is equal to Test

# assert response is equal to a variable
Then assert json response id is equal to %test%

# assert response using dotted xpath in json is equal to a variable
Then assert json response list[0].field1 is equal to %test%

Given request sqlserver 'SELECT TOP 1 * FROM TestTable'
When method sqlserver
# assert a sql server response is equal.
Then assert json response sqlresult[0].Test is equal to test
```

assert json response <response> contains <value>

All request responses are returned as json responses. The response json is added to the APIResponse object’s ResponseBody field. The assert json response step allows verification of the json response.

Examples:

```
# assert response contains the value Test
Then assert json response id contains Test

# assert response contains the value in %test% variable
Then assert json response id contains %test%

# assert response using dotted xpath in json contains a variable
Then assert json response list[0].field1 contains %test%

Given request sqlserver 'SELECT TOP 1 * FROM TestTable'
When method sqlserver
```

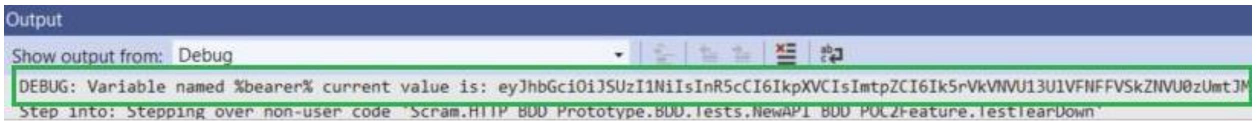
```
# assert a sql server response contains the value Test
Then assert json response sqlresult[0].Test contains Test
```

print var <variable name> debug

Prints a variable to the Visual Studio debug console

Example:

```
Given var %bearer% is equal to response.access_token type string
Then print %bearer% debug
```



```
DEBUG: Variable named %bearer% current value is: eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCIsImtpZCI6Ik5rVkVNVU13U1VFNFFVSkZNVU0zUmtJM
Step into: Stepping over non-user code 'Scram.HTTP BDD Prototype.BDD.Tests.NewAPI BDD POC2Feature.TestTearDown'
```

print response debug

Prints the response object to the Visual Studio debug console

Example:

```
Then print response debug
```

SpecFlow Error - Steps are bound but still purple after APIFramework_Steps file import

When importing the new framework, SpecFlow will most likely show the new steps in intellisense but when the steps are chosen the steps are purple and Specflow reports that the steps are already bound with execution errors.

Delete the existing SpecFlow .cache files to regenerate the linking to repair.

Steps:

1. Exit Visual Studio.
2. Open Windows Explorer.
3. In the address bar, type %TEMP% and hit Enter to go to your temp folder.
4. Find the files whose names start with "specflow-stepmap-YourProjectName"

with a .cache extension.

5. Delete those files.
6. Start Visual Studio again.