WSO2 API Manager 4.1.0 Fundamentals - Integration Profile

Developing Unit Test Suite for Artifacts

WSO2 Training

CC by 4.0

Module Objective

At the end of this module, attendees will be able to:

- Understand the basics of WSO2 Micro Integrator's Unit Test Framework.
- Understand how to implement unit tests for integration artifacts.

WSO2 MI Unit Test Framework

- Build unit tests for integration artifacts.
- Build mock services to simulate mock backends.
- Test on either a local server or a remote server.

Developing Unit Tests

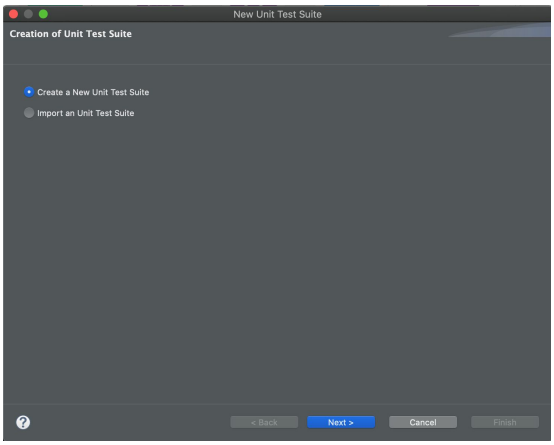
Supported Artifact Types

- Mediation Sequences
- Proxy services
- APIs
- Artifacts with Registry Resources
- Artifacts using Connectors

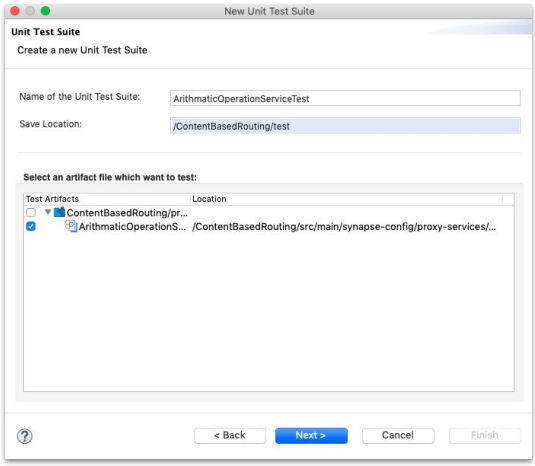
Unit Test Suite

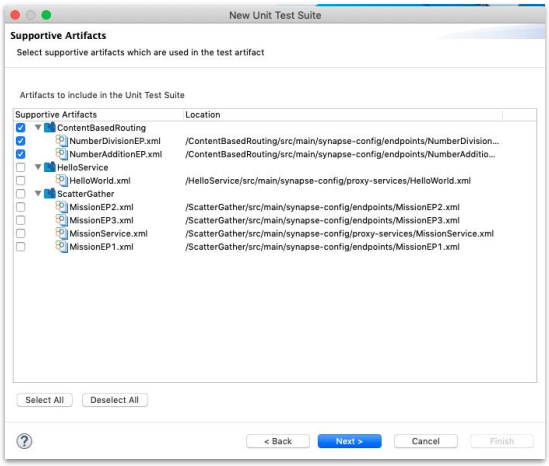
- Allows grouping of unit tests into a single suite.
- Either a sequence, API, or proxy service should be specified per test suite.
- Select the dependent artifacts for a given main artifact.
- Specify mock service (backend).

Unit Test Suite



Unit Test Suite

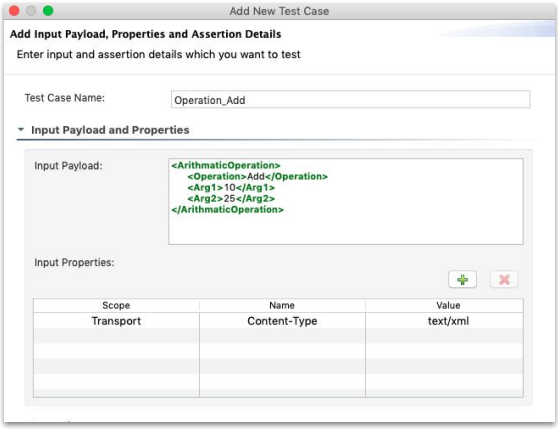


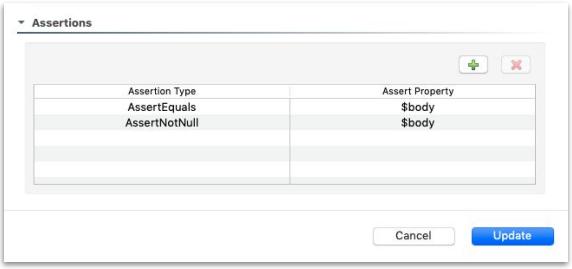


Building a Unit Test Case

- Specify a unique name per test case.
- Input Payload: Should be XML, JSON or Plain Text.
- Input properties: Should be one of axis2, synapse, or transport-scope properties.
- For a sequence test case, you can specify any kind of property.
- For APIs or a proxy service test case, only transport-scope properties are accepted.

Building a Unit Test Case

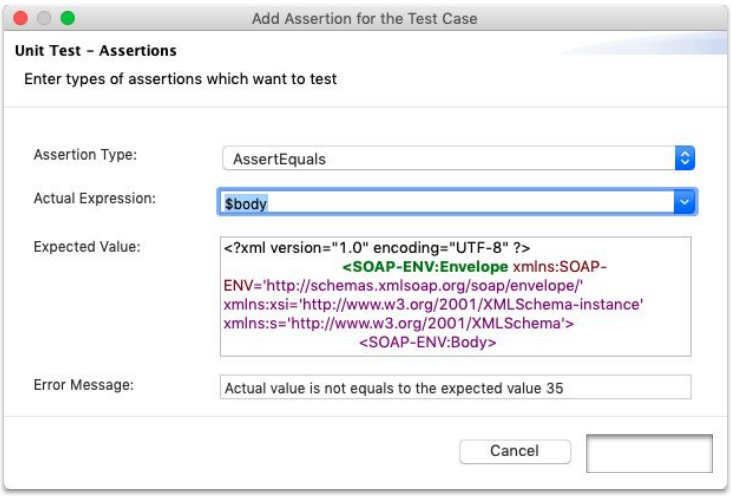




Building a Unit Test Case

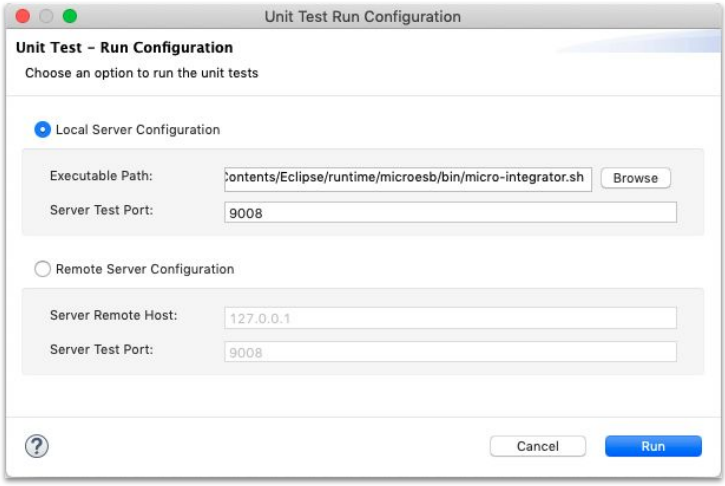
- Request path:
  - Url mapping for the API resource.
  - If the path contains parameters, replace those with values.
- Request method: HTTP method of the resource.
- Assertions: You can match the results using assertions (AssertEquals or AssertNotNull).

Building a Unit Test Case



- Assertion Type:
  - **AssertEquals**
  - **AssertNotNull**
- Assertion Expression:
  - **$body** : asserts the payload.
  - **$ctx:""**: asserts synapse property.
  - **$axis2:"" :** asserts axis2. property
  - **$trp:"":** asserts the transport property.
- Expected Value: The expected response.
- Error Message : To print upon assertion failure.

Running Test Case



- **Local Server**:
  Run test suite using the embedded unit test server in the Micro Integrator.

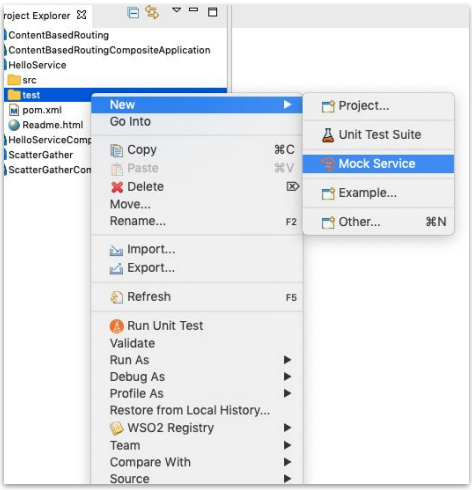
- **Remote Server:**
  Run test suite using a remote unit test server in the Micro Integrator.

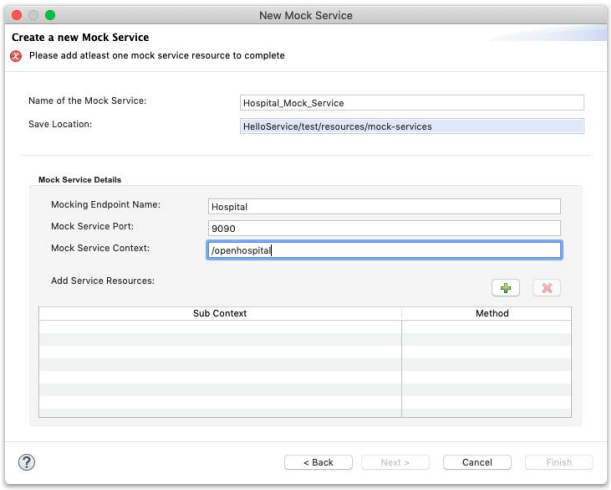
  Be sure to start the Micro Integrator runtime with the **-DsynapseTest** parameter.

Mock Services

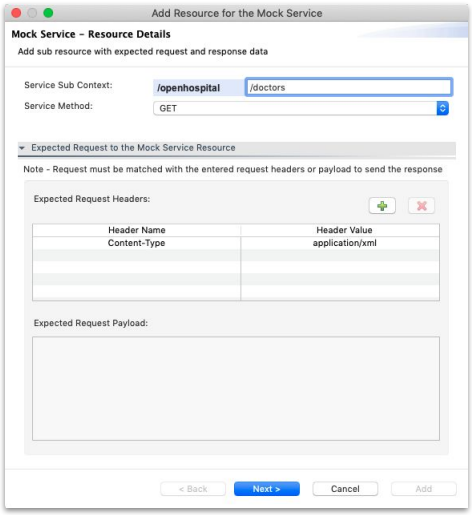
- Mock services allow you to test your integration artifacts without using the actual back-end system.
- Useful when automating the tests as real back-end systems may not be accessible within the continuous integration system.

Building a Mock Service



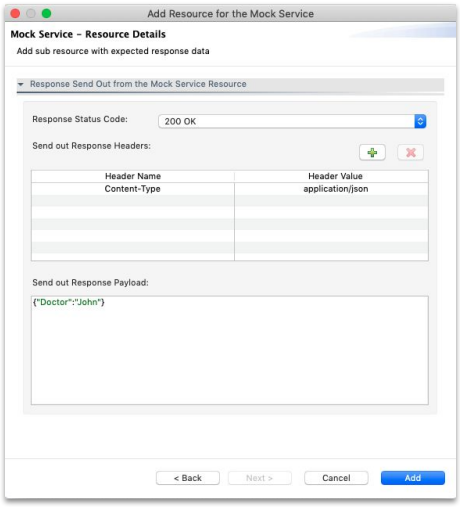


Building a Mock Service



- Mock Service Context : Main url context for mock service. Starts with “/”. Example: /openhostpital
- Service Sub Context: Sub url context under the main URL context for the resource.
- Service method: The HTTP method of the resource.
- Header Name: Expected header name.
- Header Value: Expected header value.
- Expected request payload: Expected request payload for the service.

Building a Mock Service



- Response Status Code: HTTP Status code for the response message.
- Header Name: Name for the given response header.
- Header Value: Value for the specified header.
- Send out response payload: Expected response payload.

Let’s try it out!

Creating a Unit Test Suite