

WSO2con2025

New Experiences of Micro Integrator



Rosen Silva
Technical Lead
WSO2



**Making integration development
faster and smarter than ever!**

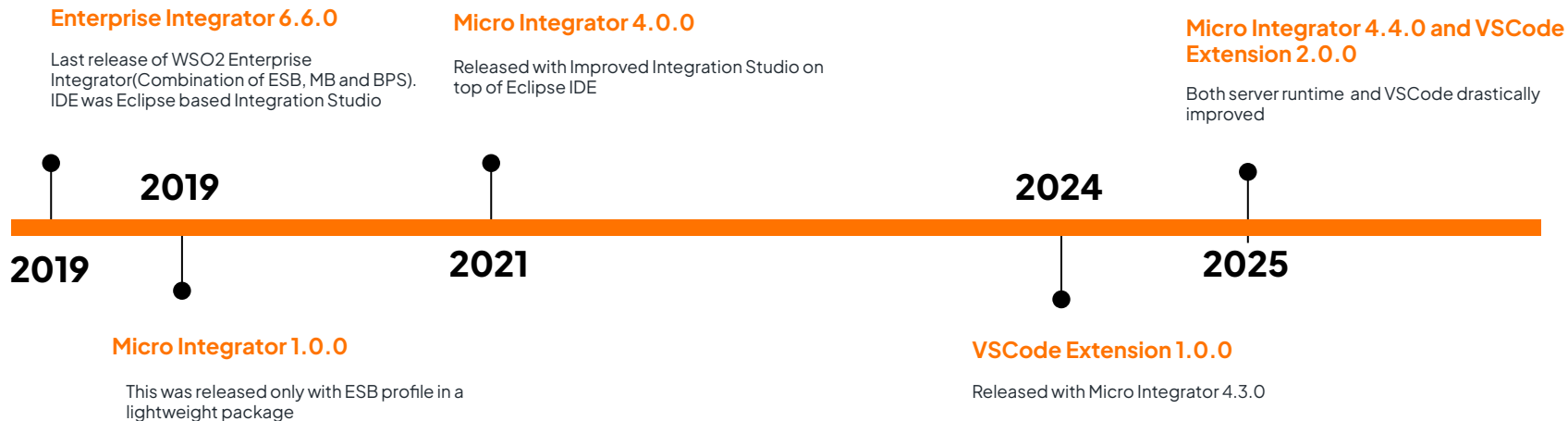


Agenda

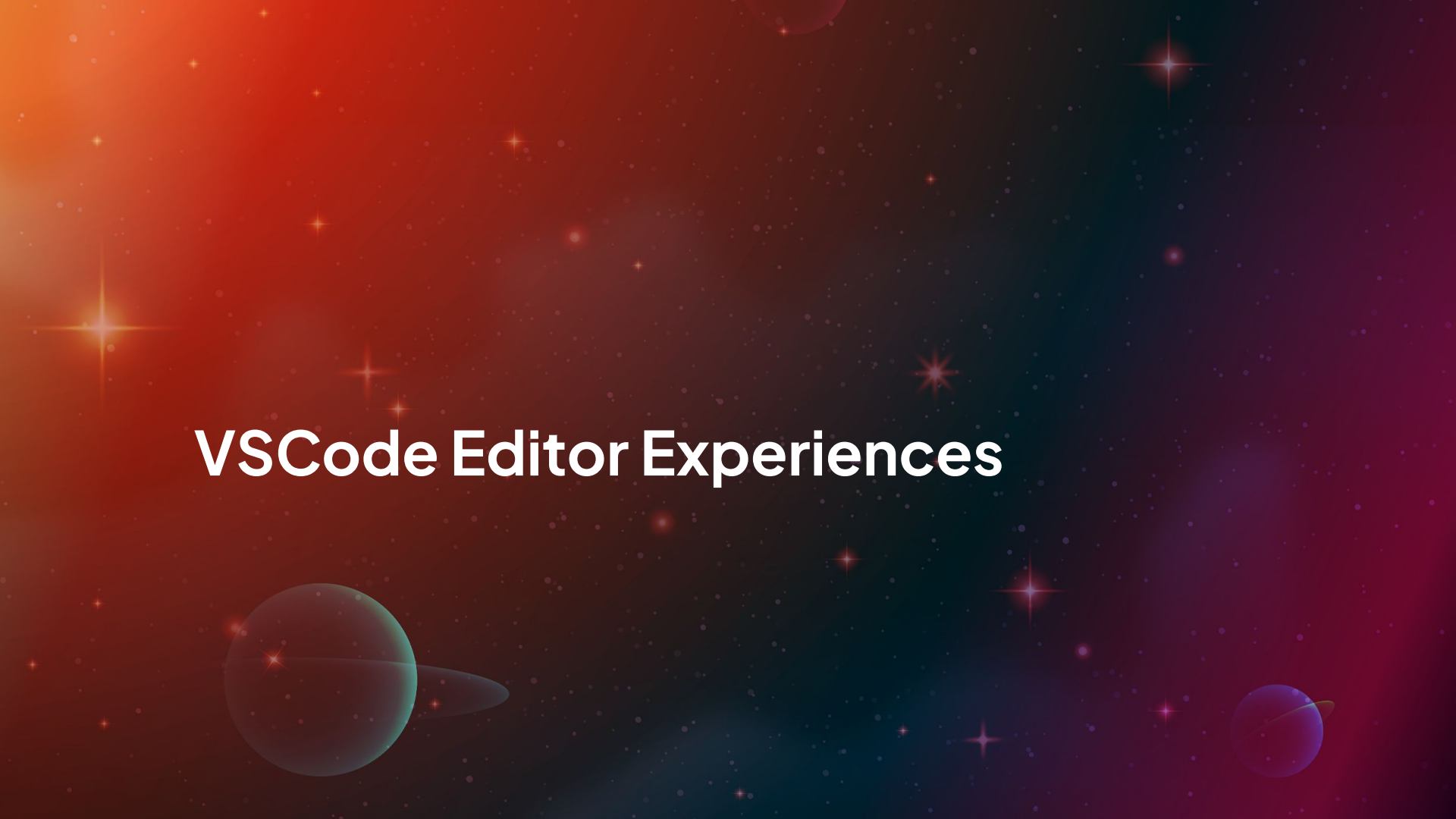
- Evolution of Micro Integrator
- New Expression Language
- Data Transformation and Extensions
- Mediator Try-Out Feature
- New Mediators
- Connector Improvements
- JDK 21 Support and More
- AI Features (Co-pilot and AI Module)
- Future of Micro Integrator



Evolution of Micro Integrator



VSCode Editor Experiences

The background is a vibrant space-themed gradient. It transitions from a bright orange-red on the left to a deep blue and purple on the right. Scattered throughout are numerous small white stars and larger, more prominent four-pointed starburst patterns in various colors like orange, yellow, and white. In the lower-left corner, there is a large, semi-transparent teal planet with a thin ring. In the lower-right corner, there is a smaller, semi-transparent purple planet with a thin ring.



New Expression Language for Micro Integrator

WSO2 Synapse Expressions: Faster, Easier, Secure Integrations

- Easy Payload and Variable Access
 - ⦿ Example: `payload.user, vars.totalPrice`
- Quick Conditional Checks
 - ⦿ Example: `vars.age > 18 ? 'Adult' : 'Child'`
- Filter Data Effortlessly
 - ⦿ Example: `payload.orders[?(@.amount > 100)]`
- Built-in Useful Functions
 - ⦿ Example: `toUpper('wso2'), round(123.456, 2)`
- Secure Secret Access
 - ⦿ Example: `wso2-vault('db-password')`



Synapse Expression Improve Developer Experience

Using XPath and JSONPath in Integration

```
<!-- Extract firstName and lastName from the incoming JSON payload -->  
<property name="firstName" expression="json-eval($.firstName)" scope="default"  
type="STRING"/>  
<property name="lastName" expression="json-eval($.lastName)" scope="default"  
type="STRING"/>
```

```
<!-- Concat firstname and last name to get fullName -->  
<property name="fullName" expression="fn:concat(get-property(firstName))  
,get-property('lastName'))" scope="default" type="STRING"/>
```

New Synapse Expressions

```
<!-- Get full name using new expression to variable -->  
<variable name="fullName" type="STRING" expression="${payload.firstName + payload.lastName}" />
```

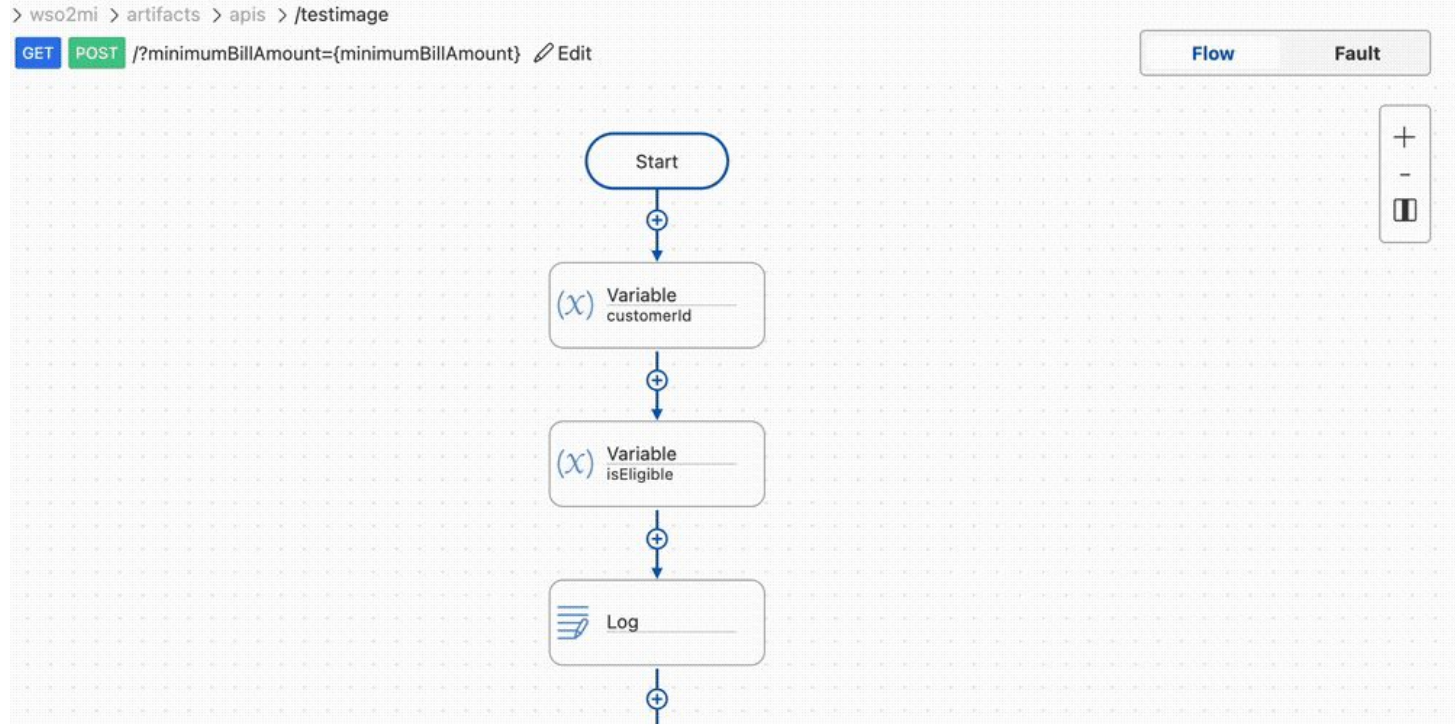


Learn more...

Search: **Synapse Expressions** in <https://mi.docs.wso2.com/>



New Expression Graphical Tooling Support



- 1. Simplicity favors regularity**
- 2. Smaller is faster**
- 3. Good design demands compromise**

- ISA Design Principles -



Data Transformation and Extensions

The background is a vibrant space-themed gradient. It transitions from a bright orange-red on the left to a deep blue and purple on the right. Scattered throughout are numerous small white stars and several larger, more prominent stars with four-pointed diffraction patterns. In the lower-left corner, there is a large, semi-transparent teal planet with a thin ring. In the lower-right corner, there is a smaller, semi-transparent purple planet with a thin ring.

Extend Complex Integration Using

Ballerina Module for MI

- Language built by WSO2
- 100% Tooling Support
- Flexible, Powerful, Beautiful
- Integrations as Code

Java Class Mediator

- Leverage from wider Java ecosystem
- Easily Develop from MI VSCode Plugin

JavaScript or TypeScript

- Runs on Graal JS
- Java like performance. Faster than all previous script engines
- Create using single click from MI VSCode Plugin

Ballerina Module for MI



Mediator Try Out



Try-out Feature



New Mediators

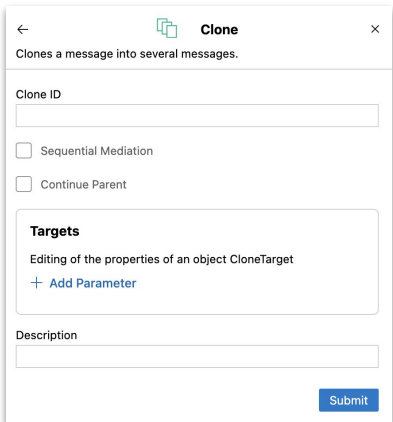


Enhancing WSO2 MI Mediators

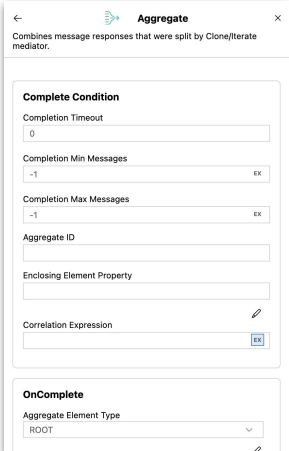
- New Scatter Gather Mediator

- Previously, users need to use two mediators(Clone and Aggregate).
- Having a Call mediator in each clone path is a must.
- Introduce a new single mediator while removing the Call mediator requirement.

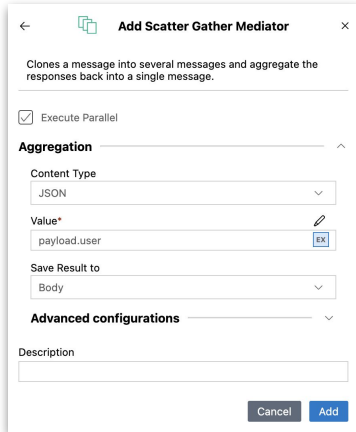
Old



Aggregate



New



Enhancing WSO2 MI Mediators

- New Foreach Mediator

- Currently we have two mediators, Foreach and Iterate to loop over a list.
- Foreach for transformations and Iterate if the user needs to have Call mediators.
- Need to have an Aggregate mediator after Iterate mediator.
- Single Foreach mediator to cater all the above requirements.

Old

The 'Foreach' mediator configuration dialog shows the following fields and options:

- Foreach ID:** A text input field.
- Foreach Expression*:** A text input field with an 'EX' button.
- Sequence:** A section containing a 'Sequence Type' dropdown menu (set to 'Anonymous') and a 'Description' text area.
- Submit:** A blue button at the bottom right.

The dialog also includes a description: 'Splits message based on XPath/JSONPath, processes sequentially, then merges back.'

New

The 'Add Foreach Mediator' configuration dialog shows the following fields and options:

- Collection to iterate*:** A text input field with an 'EX' button.
- Execute Parallel:** A checkbox.
- Save Result to:** A dropdown menu (set to 'Body').
- Advanced:** A section containing a 'Description' text area.
- Buttons:** 'Cancel' and 'Add' buttons at the bottom right.

The dialog also includes a description: 'Loop over a list and then merges back the result.'

Enhancing WSO2 MI Mediators

- New Variable Mediator

- Currently, we need to use the Property mediator to set variables.
- Scopes and other configurations in Property mediator will lead to confusion for new users.

Old

The 'Property' mediator configuration dialog is shown. It has a title bar with a back arrow, a toggle switch, and the title 'Property'. Below the title bar is a description: 'Manipulates message properties by setting and/or removing property values, supporting both constant and dynamically generated values through XPath expressions.' The form contains several fields: 'Property Name*' with a text input 'New Property Name' and an 'EX' icon; 'Property Action*' with a dropdown menu showing 'set'; 'Property Data Type*' with a dropdown menu showing 'STRING'; 'Property Value*' with a text input 'Value' and an 'EX' icon; 'Property Scope*' with a dropdown menu showing 'DEFAULT'; an 'Advanced' section with a chevron icon; 'Value String Pattern' with a text input 'Value String Pattern'; 'Value String Capturing Group' with a text input '0'; and 'Description' with a text input 'Description'. A 'Submit' button is at the bottom right.

New

The 'Add Variable Mediator' configuration dialog is shown. It has a title bar with a back arrow, a toggle switch, and the title 'Add Variable Mediator'. Below the title bar is a description: 'Set or remove a variable to/from the message'. The form contains several fields: 'Action*' with a dropdown menu showing 'set'; 'Name*' with a text input 'user_name'; 'Data Type*' with a dropdown menu showing 'STRING'; 'Value*' with a text input '{payload.user.name}' and an 'EX' icon; and 'Description' with a text input 'Description'. A 'Cancel' button and an 'Add' button are at the bottom right.

Enhancing WSO2 MI Mediators

- Limitations in Payload and Log mediators
 - Need to use an argument list in the Payload Mediator template.
 - Need to add multiple properties to construct a log message.
- Introduce Inline expression support for Payload and Log mediators

Old

The 'Payload Factory' mediator configuration window shows a 'Payload' section with a text area containing a JSON template: `{ "requestID": "${1}", "content": "${2}" }`. Below this is an 'Args' section with a table of arguments:

Arg	Value
1	<code>\${ctx:request}</code>
2	<code>json-eval(\${response...})</code>

. A 'Submit' button is at the bottom.

New

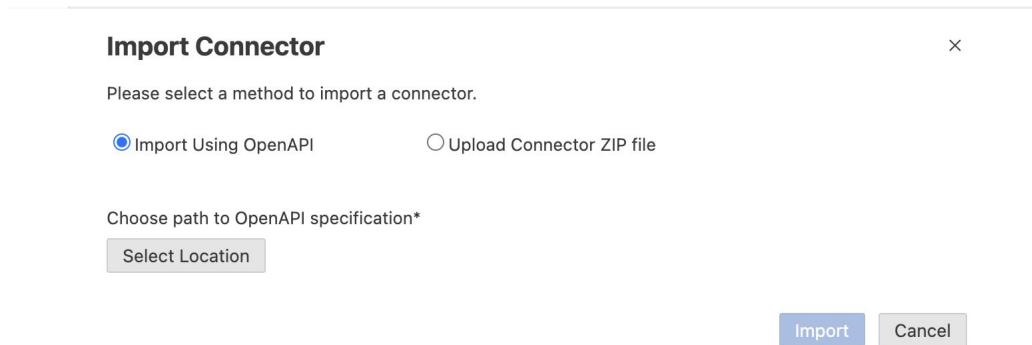
The 'Add Payload Mediator' window shows a 'Payload' section with a text area containing a JSON template: `{ "requestID": "${var.request}", "content": "${payload.response.data}" }`. The 'Add Log Mediator' window shows a 'Message' section with a text area containing a log message: `Processing message ${payload.id} using backend ${var.endpointX}`. Both windows have 'Cancel' and 'Add' buttons at the bottom.

Connector Improvements



Simplify connector creation and connector generation tool

- A new connector plugin to simplify connector development to auto-generate connector using OpenAPI specification.
- Connector generator tool has been integrated with the VSCode extension to generate connector within the project.



The screenshot shows a dialog box titled "Import Connector" with a close button (X) in the top right corner. The dialog contains the instruction "Please select a method to import a connector." and two radio button options: "Import Using OpenAPI" (which is selected) and "Upload Connector ZIP file". Below these options, there is a label "Choose path to OpenAPI specification*" and a button labeled "Select Location". At the bottom right of the dialog, there are two buttons: "Import" and "Cancel".

Improved Connector Usage Experience

- New button to test the connection of the connector in the VSCode extension
- New option to store connector response (payload, headers, attributes) inside a variable.

Add New Connection

Connector: **file**

Connection Name*

FileConnectorConfig

Basic Properties

Working Directory

/home/wso2/file/in

Advanced Properties

Test Connection

Update

Cancel

Edit Sales - operation_get

Connection *

+ Add new connection

SalesCon

Parameters

Year

2024

EX

Quarter

q3

EX

Response

Target*

Sales_Operation_get_1

☒ Replace Message Body

Submit

Introduce new connectors (HTTP connector)

- HTTP connector supports operations for sending HTTP requests.
- During connection creation, we handle certificates, authentication configurations and error handling properties.

HTTP Connector Operations

http:// Http
SNAPSHOT

Select Operation

http:// Get	http:// Post
http:// Put	http:// Delete
http:// Head	http:// Options
http:// Patch	

HTTP Connector New Connection

Add New Connection

Connection Name*
The name for the file connection

Connection Type*
https

Base URL*
Enter the base URL of the service, which serves as the root endpoint for all API requests.

Certificate Properties

Auth Configuration Properties

Error Handling Properties

Add Cancel

Connector dependency management

- Problem

- ⦿ When working with connectors, users has to manually add the libs required by the connector to the libs directory of the MI server.

- Solution

- ⦿ When building an Integration project, the connector and its dependencies are packed into the CAR file so that user does not need to care about manually adding libs to the server.

JDK 21 Support and More

JDK 21 Support

- WSO2 MI is now fully compatible with JDK 21, enabling users to run integrations on the latest Java platform.
- Comprehensive testing and validation ensure smooth runtime operations with JDK 21 without additional configuration.



Environment variable injection for all environment-specific parameters

- Problem

- No unified template to manage external configurations effectively.
- This leads to inconsistent handling and increased complexity across different environments.

- Solution

- Introduced a new unified template to manage external configurations effectively.
- Users can now configure values in multiple ways. The value of a configurable parameter will be resolved in the following order of precedence at runtime:
 - ⊙ Environment Variables
 - ⊙ System Properties
 - ⊙ File Properties

Environment variable injection for all environment-specific parameters

Data Functions Configs

No items found.

[+ Create New Configurable Variable](#)

Property Name* [?]

Property Action* [?]

set

Property Data Type* [?]

STRING

Property Value* [?]

Property Scope* [?]

DEFAULT

Advanced

Description [?]

Data Functions Configs

Add configuration

Important!

After adding the configuration through the form, please add config name and the value to the .env file.

Name*

Type

string

Property Name* [?]

Property Action* [?]

set

Property Data Type* [?]

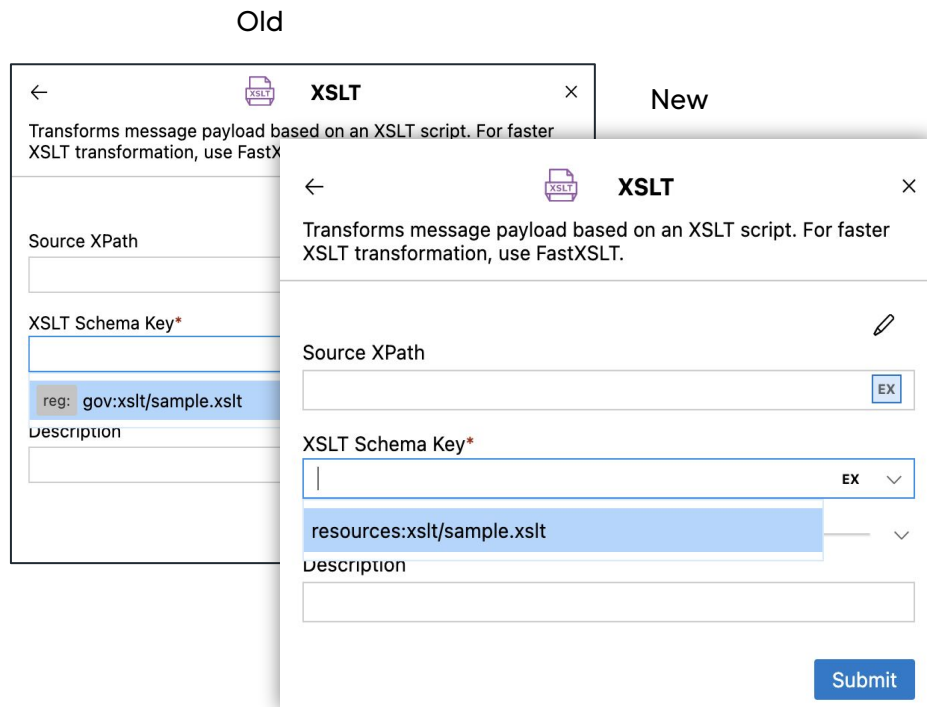
STRING

Property Value* [?]



Project structure gets complexed after adding resources.

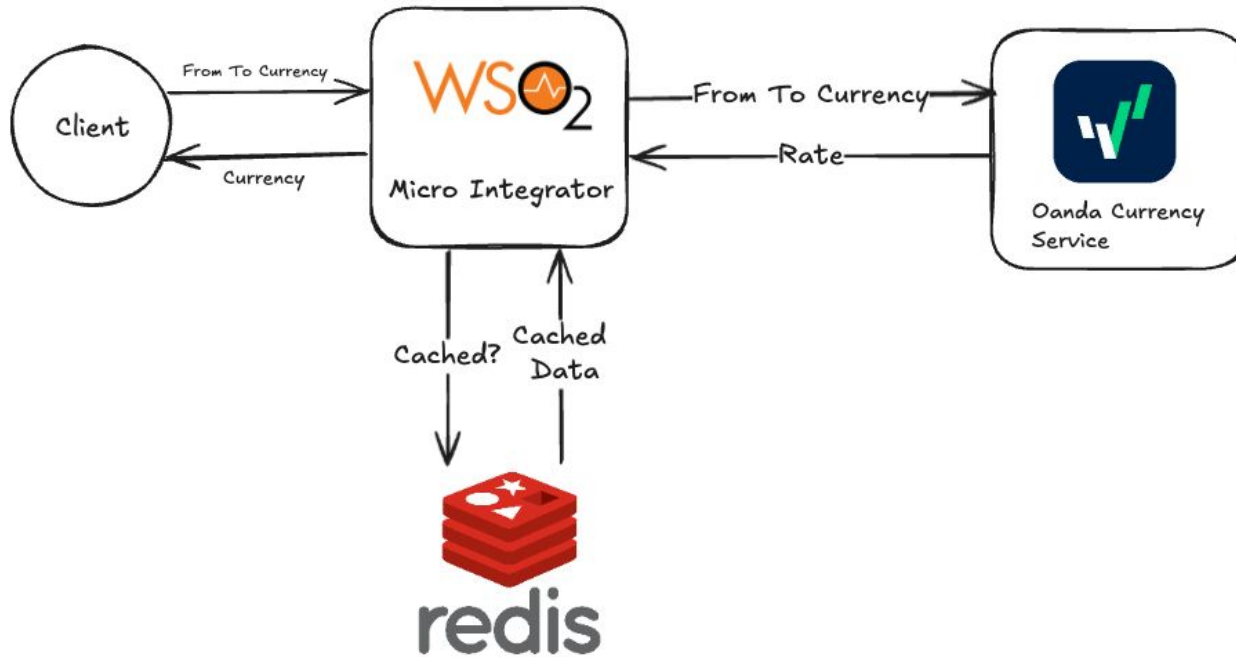
Project structure gets complexed after adding resources.



Co-Pilot Demo



Scenario





Welcome to Micro Integrator. You can open a folder containing a MI project or create a new project.

[Open MI Project](#)[Create New Project](#)

To learn more about how to use Micro Integrator in VS Code, [read our docs](#).

Micro Integrator (MI) for VS Code

A comprehensive integration solution that simplifies your digital transformation journey. Streamlines connectivity among applications, services, data, and cloud using a user-friendly low-code graphical designing experience.

Getting started

Learn about the Micro Integrator Extension in our [Getting Started Guide](#).

Create New Project

Create an empty project.

[Create New Project](#)

Troubleshooting

Experiencing problems? Start with our [Troubleshooting Guide](#).

Explore Samples

Have a look at some examples.



Hello World Service

A simple HTTP service.



API Testing

Unit testing of a REST API artifact.



Content Based Routing

Content-based message routing.



Database Polling

A Task that polls a Database.

[More...](#)



Future of Micro Integrator



Roadmap for Micro Integrator

- GenAI App Building Support
- Seamless integration with API Manager: Discoverability of APIs available in the API Manager from the WSO2 Integrator
- File Processing, ETL, EDI
- Connectors and Protocol Support
 - gRPC, GraphQL, HTTP/2 support in MI
 - Salesforce Marketing, SAP HANA, JD Edwards, Magento Connectors
 - B2B (AS2, AS4) support in MI
- Make runtime further lightweight
- MI on GraalVM



Question Time!



The background is a dark space filled with a vibrant nebula in shades of red, orange, and purple. Scattered throughout are numerous 3D cubes and rectangular prisms, some of which are blue with red faces, while others are solid blue or red. Some cubes appear to be floating, while others are part of larger, more complex geometric structures. A large, dark blue sphere is visible on the left side. The overall aesthetic is futuristic and digital.

Thank you!

WSO2con2025
