

WSO2CON2024

Low-Code Integration Tooling



Joseph Fonseca
Senior Director – Head of Developer Tooling
WSO2





Background

Ballerina for VS Code

The screenshot displays the VS Code interface with a Ballerina program in the main editor and its execution flow diagram in the Overview Diagram view.

Explorer: Shows a project named 'SAMPLE' with files: `.devcontainer.json`, `.gitignore`, `Ballerina.toml`, and `main.bal`.

Code Editor: Shows the following Ballerina code in `main.bal`:

```
1
2 import ballerina/http;
3 import ballerina/googleapis/sheets;
4
5 configurable string githubPAT = ?;
6 configurable string repository = "ballerina-platform/bal
7 configurable string sheetsAccessToken = ?;
8 configurable string spreadsheetId = ?;
9 configurable string sheetName = "pull-requests";
10
11 type PR record {
12     string url;
13     string title;
14     string state;
15     string created_at;
16     string updated_at;
17 };
18
19 public function main() returns error? {
20     http:Client github = check new ("https://api.github.
21     map<string> headers = {
22         "Accept": "application/vnd.github.v3+json",
23         "Authorization": "token " + githubPAT
24     };
25     PR[] prs = check github->[repository]/pulls(headers
26
27     sheets:Client gsheets = check new ({auth: {token: sh
28     = check gsheets->appendValue(spreadsheetId, ["Issu
29         {sheetName: sheetName});
30
31     foreach var {url, title, state, created_at, updated_
32         = check gsheets->appendValue(spreadsheetId, [u
33         {sheetName: sheetName});
34     }
35 }
36
```

Overview Diagram: Shows the execution flow for the 'Function main'. The flow starts at 'START', goes through 'new' (creating a 'github' client), then an '=' node (initializing 'headers'). A callout box shows 'headers' as a map with 'Accept' and 'Authorization' keys. The flow then goes to another '=' node (calling 'github->pulls'), followed by a 'new' node (creating a 'gsheets' client). A callout box shows 'gsheets' as a client with an 'auth' field containing a 'token'. The flow then goes to an '=' node (calling 'gsheets->appendValue'), followed by a decision diamond (checking 'pr.title, etc.'). Another '=' node (calling 'gsheets->appendValue') follows, and the flow ends at 'END'.

Developer tooling

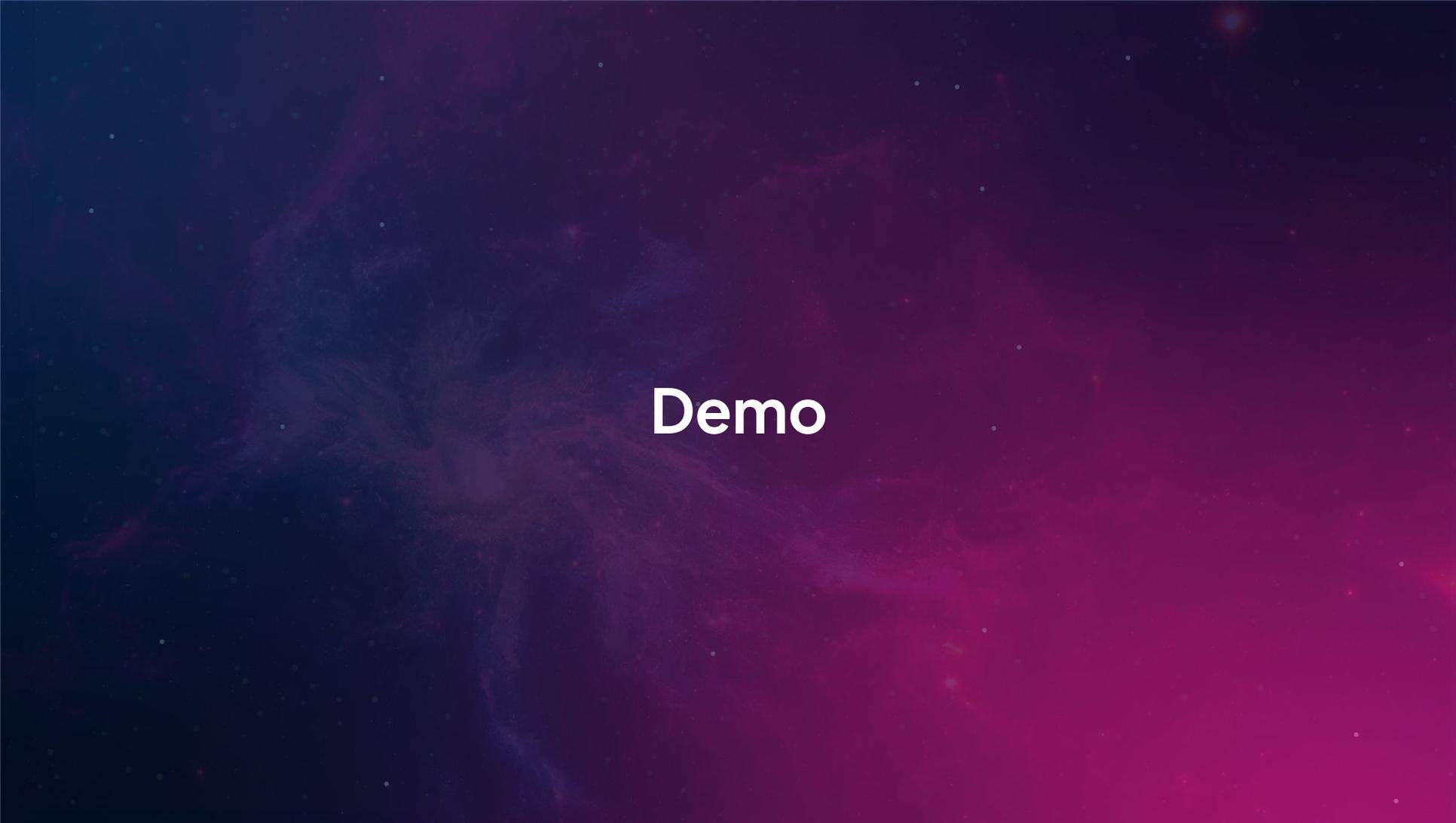
Currently, we maintain a bunch of VS Code extensions and various CLI tools for WSO2 products.

 Ballerina WSO2  wso2.com  12.8K Ballerina Language support, debugging, graphical visualization, AI-based data-... ★★★★★ FREE	 WSO2 Enterprise Integrator WSO2  wso2.com  7.9K VSCode extension for WSO2 Enterprise Integrator ★★★★★ FREE	 Choreo WSO2  wso2.com  1.1K An extension for Signing in to Choreo and managing your projects ★★★★★ FREE	 API Chat WSO2  wso2.com  369 Test APIs with OpenAPI descriptions ★★★★★ FREE	 APK Config Language WSO2  wso2.com  2K APK config language support ★★★★★ FREE	 Siddhi WSO2  wso2.com  657 IntelliSense, Diagnostics and Syntax Highlighting for Siddhi apps ★★★★★ FREE
---	---	--	--	--	--

Introducing Micro Integrator for VS Code

Designed to give a better low-code development experience for WSO2 Micro Integrator



The background of the image is a vibrant, ethereal nebula. It features a gradient of colors from deep, dark blue on the left to a rich, magenta-purple on the right. The nebula's structure is complex, with wispy, filamentary patterns and numerous small, bright white and blue specks scattered throughout, resembling distant stars or dust particles. The overall effect is a sense of vast, cosmic space.

Demo

Key Improvements

Simple Project Structure

- Simplified directory structure
- Self contained project
- Creates a single deployable artifact
- Can contain class mediators

▼ JOKER

▼ src/main

> java

> test

▼ wso2mi

▼ artifacts

> apis

> endpoints

> inbound-endpoints

> local-entries

> message-processors

> message-stores

> proxy-services

> sequences

> tasks

> templates

> resources

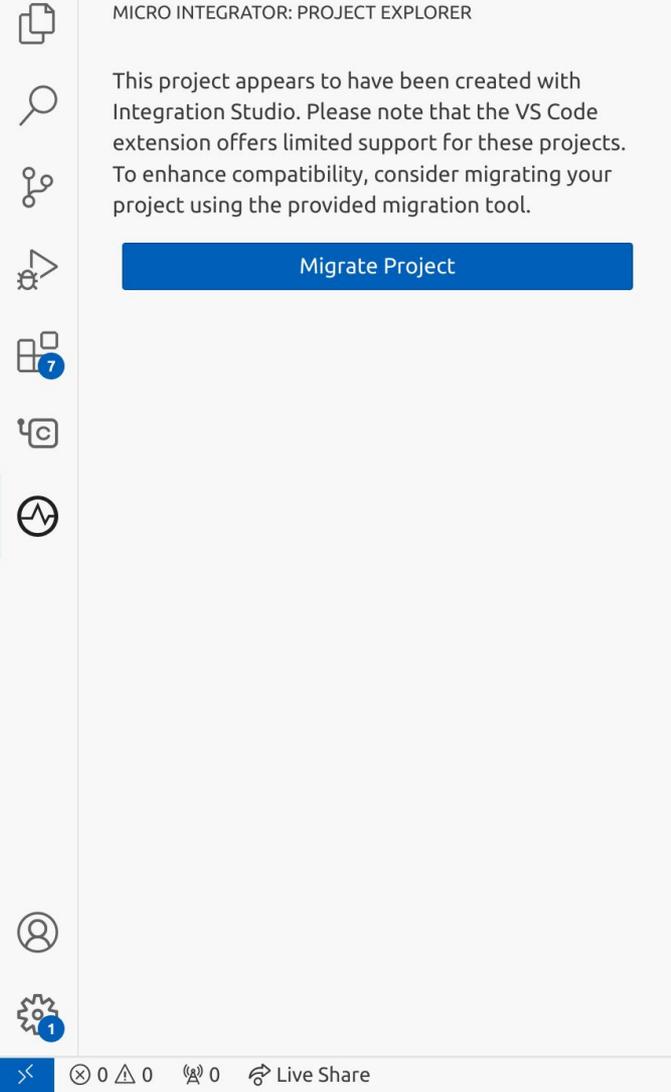
> target

📌 pom.xml



Using with existing projects

1. Migrate the old project to new project structure.
 - a. Preserves git history
2. Continue to use it to edit artifacts with limited support
 - a. No project related features



AI Assistant

- You can try it out with free credits after sign in with the service
- The GA release will enable the addition of your own OpenAI key.

MI Copilot Account Not Found

Please sign in to enable MI Copilot Artificial Intelligence features

Sign In



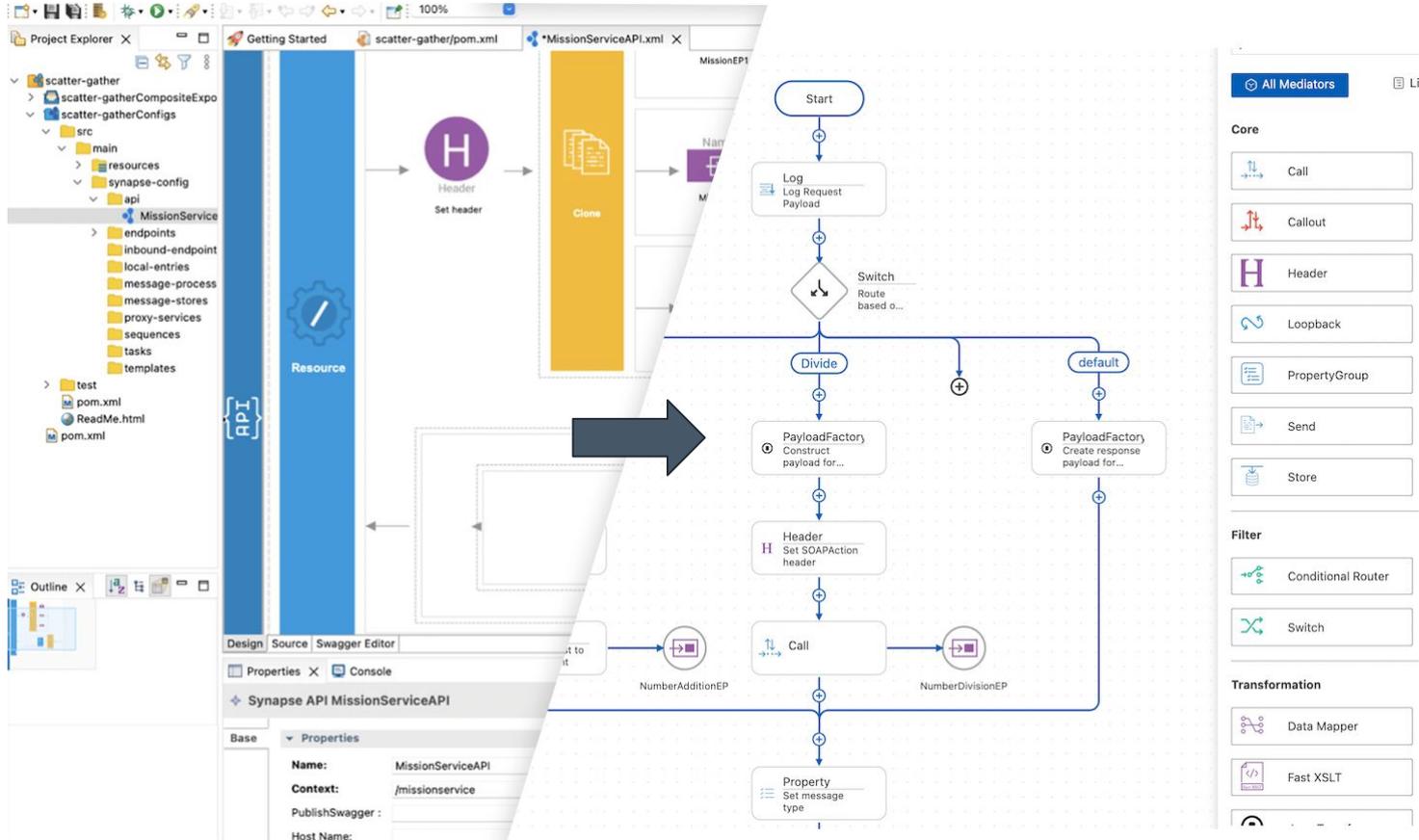
Synapse language Server

- Completions
- Diagnostics
- Go To definition
- Refactoring

```
CalculatorAPI.xml X
src > main > wso2mi > artifacts > apis > CalculatorAPI.xml > api > resource > inSequence
2  <api context="/calculate" name="CalculatorAPI" xmlns="http://
3  <resource methods="POST">
4  <inSequence>
5  <switch description="Route based on the Operati
24 <case regex="Divide">
40 <endpoint key="calculatorDivide" /
41 </call>
42 </case>
43 <default>
44 <payloadFactory description="Create res
45 <format>
46 <Message xmlns="">Unsupported
47 </Message>
48 </format>
49 <args/>
50 </payloadFactory>
51 </default>
52 </switch>
53
54 </i> aggregate
55 </i> bam
56 <fz> bean
57 </resol> builder
58 </api> cache
59 call
call-template
callout
class
clone
conditionalRouter
datamapper
```



Visualize as a Flowchart



Integrated Connector store

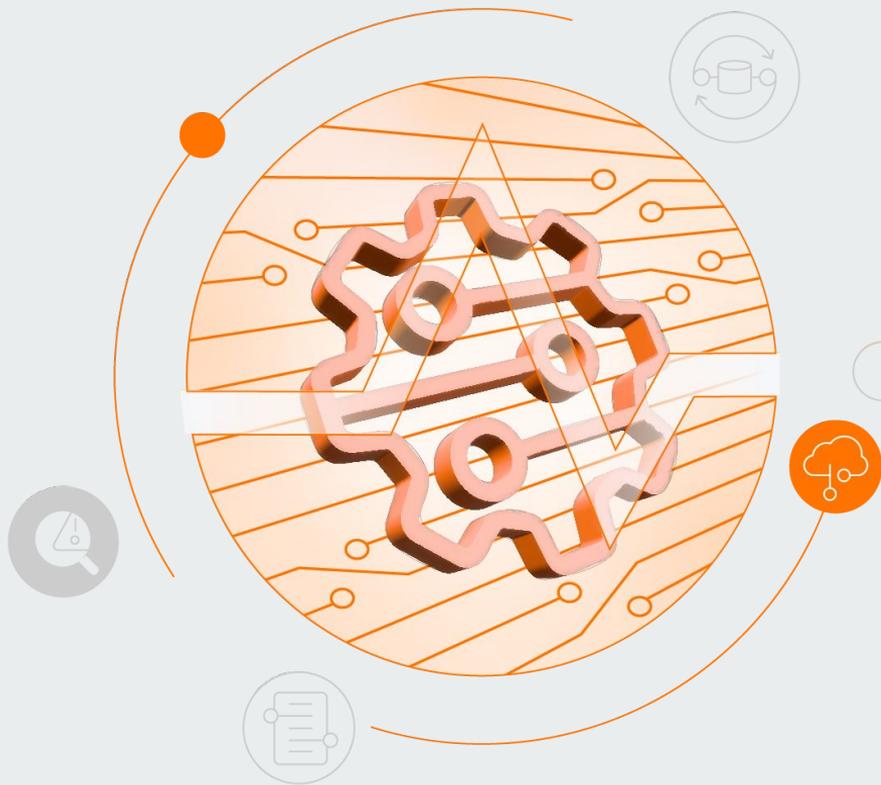
The screenshot displays the Microsoft Azure Logic Apps Designer interface. The main workspace shows a workflow diagram for a resource at `ws02mi/artifacts/apis/calculate` with a `POST` method. The workflow starts with a `Start` node, followed by a `file.copy` connector. A `Switch` node (Route based on the Operation) branches the flow into three paths: `Add`, `Divide`, and `default`. Each path uses a `PayloadFactory` connector to construct a payload, followed by a `Header` connector to set the `SOAPAction` header. The `Add` path uses a `Call` connector to send a request to the `CalculatorAddEP` endpoint. The `Divide` path uses a `Call` connector to send a request to the `CalculatorDivideEP` endpoint. The `default` path also uses a `Call` connector. All paths converge at a `Respond` node.

On the right side, the **Integrated Connector Store** is visible, showing a search bar with the letter 'e' and two tabs: **All Mediators** and **Connectors**. The **Local Connectors** section lists the `file` connector (version 4.0.20). The **Store Connectors** section lists several connectors, including `redis` (2.7.0), `email` (1.1.2), `msazurestorage` (2.0.0), `creatiocrm` (1.0.0), `salesforce_bulkapi_v2` (2.1.0), and `salesforcerest` (2.0.0). The `email` connector is expanded, showing a grid of actions: `list`, `expungeFolder`, `markAsDeleted`, `markAsRead`, `send`, `delete`, `getEmailBody`, and `getEmailAttachment`.

New Datamapper

The screenshot displays the 'Data Mapper View' interface. On the left, the 'DATA MAPPER' section shows two columns: 'input: InputRoot' and 'OutputRoot'. The input fields are 'type: string', 'setup: string', 'punchline: string', and 'id: number'. The output field is 'joke*: string'. Blue lines indicate the mapping from 'setup' and 'punchline' to 'joke*'. A search bar at the top of the mapper section contains the text 'filter input and output fields'. On the right, the 'TS Transform.ts' file is open, showing the following code:

```
1 interface InputRoot {
2   type: string;
3   setup: string;
4   punchline: string;
5   id: number;
6 }
7
8 interface OutputRoot {
9   joke: string;
10 }
11
12 function mapFunction(input: InputRoot): OutputRoot {
13   return {
14     joke: input.setup + input.punchline
15   }
16 }
17
18 // WARNING: Do not edit/remove below function
19 function map_S_InputRoot_S_OutputRoot() {
20   //@ts-ignore
21   return mapFunction(inputInputRoot);
22 }
23
```



Try it out

Developer preview in [VS Code Marketplace](#)
GA release by end of Q2 - 2024

Question Time!





Thank You!