Connector Builder Overview

Last Modified on 03/04/2022 6:19 am EST

You can use Connector Builder to create custom integrations to API providers and build your own connector. You can build connectors for REST, SOAP, and OData APIs and even database applications. Publish finished connectors to the Connectors Catalog while automatically generating interactive API Documentation. You can combine services by creating a new hub or mapping your new connector to an existing hub. You can also use the normalized resources that we have already organized into hubs, and leverage the same resources across any other connector that you build.

Anyone building connectors should be familiar with APIs, JavaScript, and JSON, as well as a thorough understanding of:

- API authorization concepts
- Any setup required with the API provider
- The API of the API provider that you are connecting to
- The API provider's API documentation
- The SAP Open Connectors Hub APIs

Before You Begin

Before you begin, here is a sampling of questions you need to know the answers to. You can find most of the information needed in the API provider's documentation.

- Where is the API documentation?
- What kind of API? REST, SOAP, database?

> Note: When creating new connectors, you can import .json and xml (for soap) files; .yaml files are not supported.

- What kind of authentication? OAuth 1.0 or 2.0? Basic? AWS V2 or V4?
- Do you need to create a connected app?
- Does the endpoint support events or bulk?
- Do OAuth 2.0 tokens expire?
- What resources do you want to connect? Accounts, contacts, lists, leads?
- In which Hub should you categorize the connector?
- Have you set up an application to integrate with the API provider? Do you have the authentication information for it?

Custom Connector Checklist

After you build a connector, we recommend that you review the checklist below:

☑ Each GET resources includes OCNQL (the where clause) and pagination.

☑ The connector includes an informative description.

☑ If building an OAuth 2.0 connector in different SAP Open Connectors environments, confirm that the API providers supports all environments.

☑ Check pagination — pageSize and Pagination type — to ensure correct information.

☑ POST and PATCH resource bodies include the correct object data type and models.

☑ Map an authenticated connector instance to a common resource data resource.

☑ For polling events, confirm the correct configuration of dates, using the same format as the API provider. Also ensure that the time zone format matches.

☑ For bulk, confirm the dates are configured using the same format as the API provider.

Connector Conventions

To align your custom connectors with those created by SAP Open Connectors, we recommend that you follow our conventions:

- Resource names: Lowercase the name and use the plural form.
  - Correct: /contacts
  - Incorrect: /contact
- Spaces in paths: Replace spaces with dashes.
  - Correct: /hubs/finance/sales-receipts
  - Incorrect: /hubs/finance/sales_receipts
- Spaces in configuration or parameter names: Use camelCase.
  - Correct: /hubs/finance/sales-receipts/{salesReceiptId}/details
  - Correct: /hubs/crm/contacts/{contactId}/notes
  - Incorrect: /hubs/finance/sales-receipts/{salesreceiptId}/details
  - Incorrect: /hubs/crm/contacts/{contact-Id}/notes
- Descriptions by method:
  - GET — "Search for resources." Or, if no OCNQL query was configured for the endpoint, use "List all resources".
  - GET/{id} — "Retrieve a(n) resource".
  - POST — "Create a(n) resource".
  - PATCH — "Update a(n) resource".
  - DELETE —"Delete a(n) resource"
- `where` parameter descriptions: Following the default "The OCNQL search expression." add an example for the specific connector.

Note: When creating custom VDRs or resources, make sure that the name does not contain the string “events” as it conflicts with the `events` API endpoint. This causes the `GET` and `POST` API calls to fail.
Example:
GET `/events/eventTest` → works.
GET `/events/eventsTest` → does not work.