Use a Formula as a Resource

Last Modified on 10/27/2021 1:00 pm EDT

On this page

You can expose formulas that have manual triggers as a resource, known as Formula as a Resource, or FaaR. This enables you to use the formula as a synchronous API request. After you update a formula to be used as a resource, you can make API requests to it at `https://api.openconnectors.us2.ext.hana.ondemand.com/elements/api-v2/`. Using a formula as a resource enables you to further remove logic from your applications and also provides the ability to more efficiently chain requests together.

FaaRs enable you to take APIs that require multiple operations to perform an operation — operations that might be simple in other systems — and wrap them in a single request. In this case, the FaaR acts as a canonical resource that performs multi-step processes under the hood.

> NOTE: any formula being used as a resource must include a script step that returns both the status and result. If your FaaR does not include, for example, a step that returns data from a done() method, or else you will receive an error regarding the formula engine.

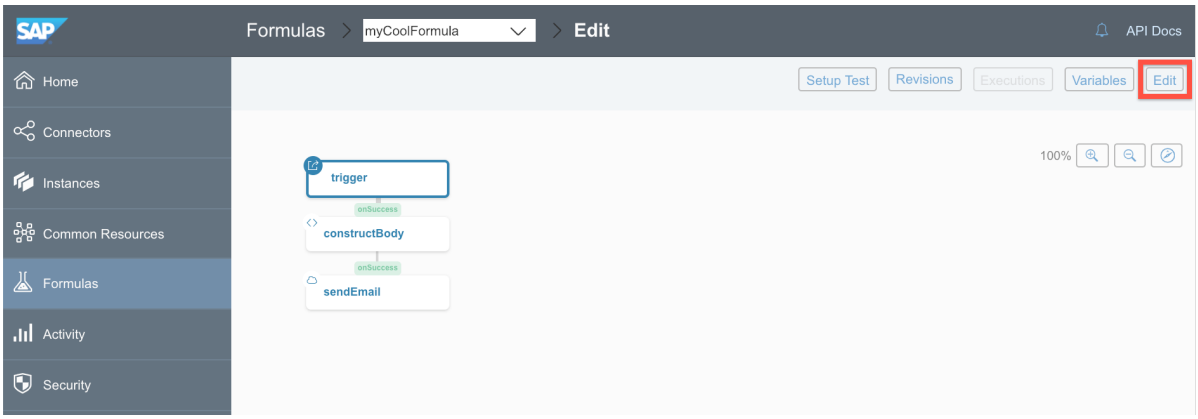
The API requests are synchronous and so a response is required for further processing. To maintain efficiency, SAP Open Connectors imposes a processing time limit of thirty seconds. If the request reaches that limit, the response notifies you.

Note: Using formulas as a resource will result in slower performance compared to counterpart resources in the connector because they use the synchronous call mechanism.

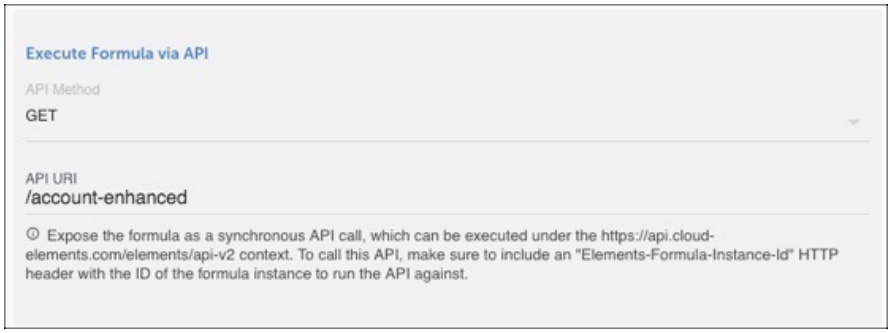
Set up a FaaR

To use a formula as resource:

1. Open the formula template. On the Formulas page, hover over the Formula Card, and then click **Open**.
2. Click **Edit**.



3. Click **Show Advanced**.
4. In the Execute Formula via API section, update the **API Method** and **API URI**.



- In **API Method** select the method used to call the formula, such as GET, POST, PUT, PATCH, or DELETE.
- In **API URI** enter the resource name of the formula, such as `/account-enhanced` .

5. Click **Save**.

After you create a FaaR, you can start making requests or take a look at the API docs, where you can also test your FaaR.

> Note: Formula as a Resource will be slower in performance compared to their counterpart resource in the element as it uses the synchronous call mechanism.

Execute FaaRs

To make an API request to the FaaR, include the formula instance (`elements-formula-instance-id`) in the header in addition to the usual User and Organization (no need to include an connector token) . For example:

```
curl -X GET \
  https://api.openconnectors.us2.ext.hana.ondemand.com/elements/api-v2/formula1 \
  -H 'authorization: User , Organization ' \
  -H 'elements-formula-instance-id: 28683' \
```

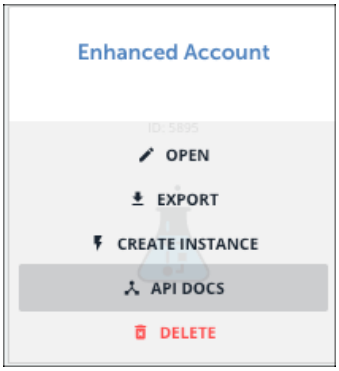
You can find the `elements-formula-instance-id` on a Formula Instance Card under the title.

Note: Formulas used as a resource do not report step-end events. In practice, this means that any of executions using of a formula as a resource using the V3 engine will not report steps or events, even if your formula has debug logging enabled.

Access FaaR API Docs

To access the docs for the FaaR:

1. On the Formulas page, hover over the Formula Card, and then click **API Docs**.



2. On the API Docs page, click **Try it Out**.
3. In **Elements-Formula-Instance-Id** enter a Formula Instance ID, and then execute the API call.

Status Codes

When you create a Formula as a Resource you can specify status codes and descriptions. To include a status code and result in the response to a FaaR request:

1. In the formula, add a JS Script step as the final step.

2. Include a script like the following in the step:

```
done({
 statusCode: xxx
 result: {
  label: 'message'
 }
})
```

3. Save the step.

Property	Description
statusCode	The status code that you want to return in the response, such as `200`, `401`, or `502`. The value must be a valid status code.
result	The body of the response which can be anything related to the status code such as an array of objects, a single object, or text. The example above includes an array containing a key value pair with a label and a message.

Work with Parameters

You can pass parameters into formulas as a resource, and then access the results of the response within the context of the formula.

To include a parameter in a formula as a resource:

1. When you make the request, pass the parameter. An example cURL request with a query parameter to the resource called `formula1`, your endpoint would be:

```
curl -X GET \
  https://api.openconnectors.us2.ext.hana.ondemand.com/elements/api-v2/formula1? \
  -H 'authorization: User , Organization ' \
  -H 'elements-formula-instance-id: 28683' \
```

2. Access data returned in the response to the request by using a script step. The data is available through `trigger.args.request.query`. An example script that passes query data containing an email address to the console would be:

```
var queryObject = trigger.args.request.query;

function getThatEmailAddress(emailAddressObject){
   console.log(emailAddressObject['email']);
}

done(getThatEmailAddress(queryObject))
```

Debugging a Formula as a Resource

FaaRs are executed as synchronous formula executions on the current formula engine, which was designed to set the execution status and report any errors back via the synchronous API call, but not to log execution step values. During development, when the review of the formula is needed (aka the execution logs), the FaaR can be run as an asynchronous execution in order to debug further where the debugLoggingEnabled setting is honored. To trigger an asynchronous execution you can retry an existing synchronous formula execution (e.g. through `PUT /executions/{id}/retries` or click the 'retry' button in the execution logs) or manually trigger the formula (e.g. `POST /formulas/instances/{id}/executions`).