Build Formula Templates

Last Modified on 05/17/2021 2:18 pm EDT

Formula templates enable you to build a single template for a formula that you can reuse with different connectors and values. After you build a formula template, you can then create a formula instance where you replace the variables with actual instances and values. This approach helps you build efficient and reusable formulas.

Build a Formula Template

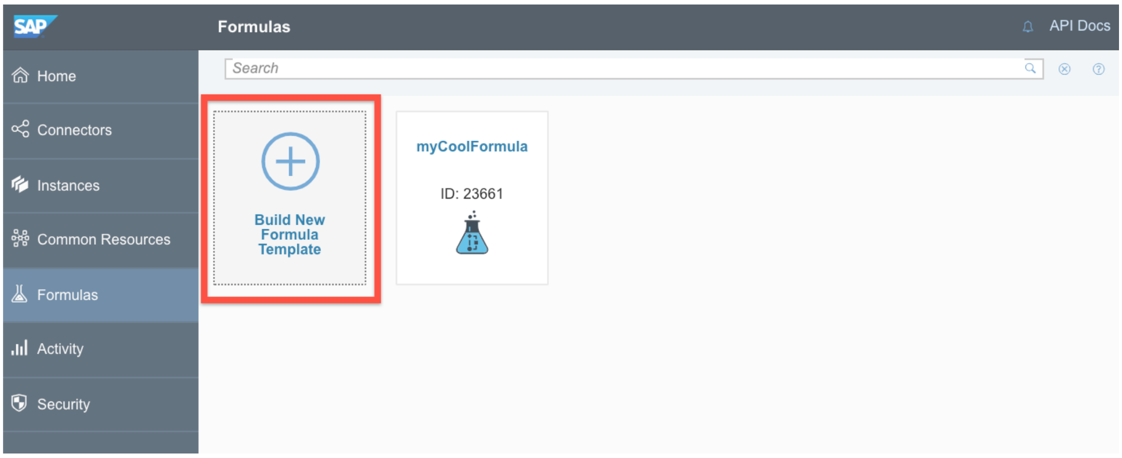
Formula templates include a trigger that kicks off the formula, steps that execute as a result of the trigger, and variables to represent input required to run an instance of the formula. You can build formulas that use triggers that kick off when something happens to a connector instance, you can schedule triggers, or you can manually kick off a trigger. See Formula Triggers for more information about each kind of trigger.

As you build a formula, you also build context consisting of triggers, variables, steps, and values created by steps. You can refer to that context in later steps in the formula. Consider each part as a building block that has a name and information that you can refer to. For example, in a formula that sends an email notification for a new contact, you build the body of the email in one step. In a later step when you actually send the message, refer to the step where you built the body.

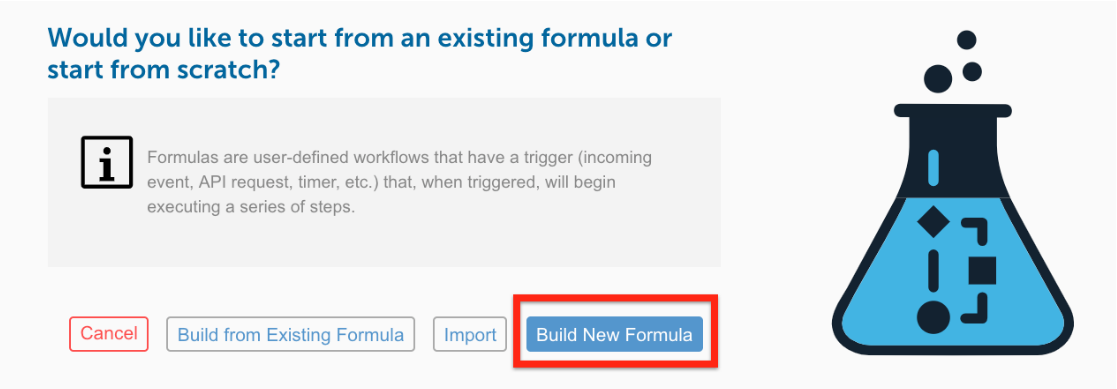
> **Note:** Use these steps to get started building a formula template. Consider using some of the example formulas to get a deeper understanding of how to build a formula template.

Create a Formula Template in the UI

1. Open the Formulas page.
2. Click **Build New Formula Template**.



3. Click **Build New Formula**.

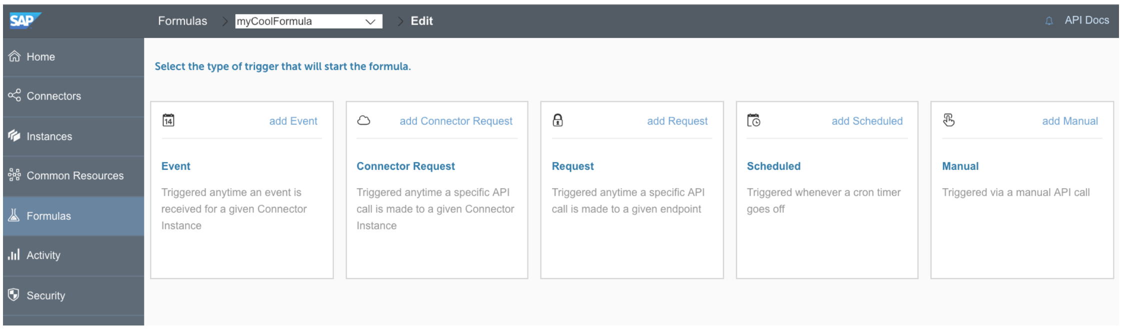


Would you like to start from an existing formula or start from scratch?

Formulas are user-defined workflows that have a trigger (incoming event, API request, timer, etc.) that, when triggered, will begin executing a series of steps.

4. Enter a name for your formula, and then click **Create**.

   SAP Open Connectors provides you a list of triggers. For details about each trigger, see Formula Triggers.



Select the type of trigger that will start the formula.

Event
Triggered anytime an event is received for a given Connector Instance

Connector Request
Triggered anytime a specific API call is made to a given Connector Instance

Request
Triggered anytime a specific API call is made to a given endpoint

Scheduled
Triggered whenever a cron timer goes off

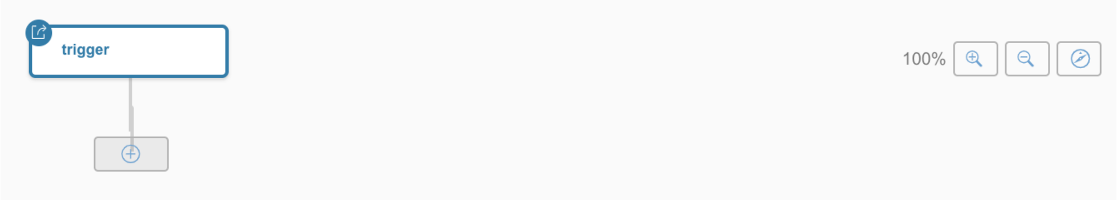
Manual
Triggered via a manual API call

5. Select your trigger.

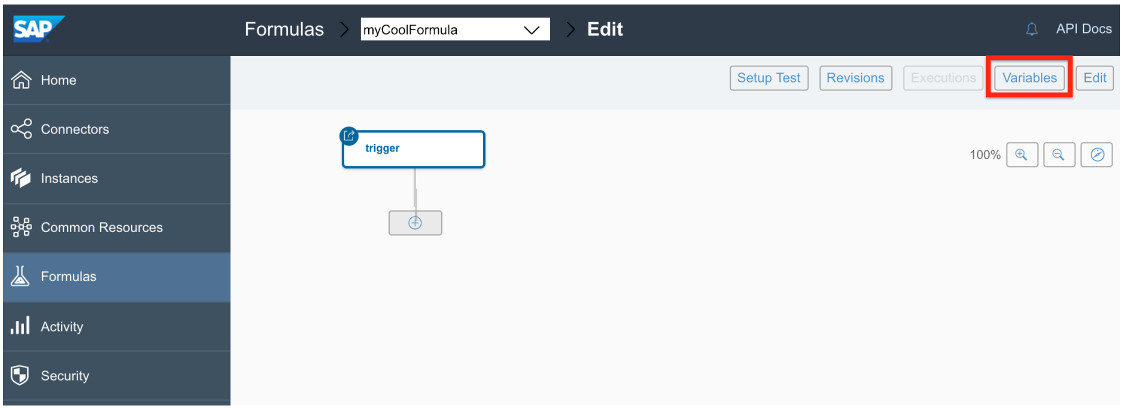
   > **Note:** Deleting a trigger removes its steps from the canvas. After deleting a trigger, you will be prompted to pick a new trigger, at which point you can add back the steps by selecting them from the Add From Existing section.

   - Choose **Event** for a formula triggered by an event configured on a connector instance.
   - Choose **Connector Request** for a formula triggered when a specific request is made to a connector instance.
   - Choose **Scheduled** for a formula to occur at a specific time or regular interval.
   - Chose **Manual** to trigger the formula with an API call to `POST /formulas/instances/:id/executions`.
6. Complete the trigger properties, which vary based on the selected trigger.
   - For **Event** provide a Connector Instance Variable.
   - For **Formula Request** provide a Connector Instance Variable, an API method, and API endpoint.
   - For **Scheduled** provide a Cron schedule.
   - For **Manual** you do not need to provide any more properties.
7. Click **Save**.

   SAP Open Connectors shows the first trigger node in the formula visualization.



8. Add any variables that you will need to run the formula instance.
   1. Click **Variables**.



2. Select the type of variable, and then enter a name.

   Takes note of the Formula Step Variable Name, which is how you will refer to the variable throughout the formula.

3. Click **Save**.

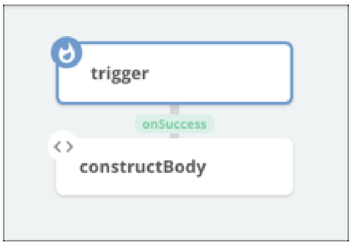
9. Add your first step. In the formula visualization, click ⊕ .

10. Select the type of step that you want to add.

11. Complete the step properties, and then click **Save**. For details on each step, see a Connector Instance Variable.

> **Note:** Click **Add From Existing** to find a step to copy.

SAP Open Connectors shows the first trigger node and the step that you just created in the formula visualization. The path from the trigger to the step defaults to On Success, but you can add failure steps later.



12. To add another step, click the step, and then select one of the following:

    - **Insert Above** adds a new step prior to the current step with an On Success path connecting the two.
    - **Add On Success** adds a step for the formula to continue to if the current step succeeds.
    - **Add On Failure** adds a step for the formula to continue to if the current step fails.

13. Continue adding steps until you complete the formula template.

14. Add a description by clicking **Edit**. See Add a Description to a Formula Template.

15. After you complete the formula, test it.

> **Note:** SAP Open Connectors saves your formula template in the background as you add each component.

Build a Formula Template via API

You can build a formula with an API request to the `POST /formulas` endpoint. Your formula JSON includes the triggers, steps and variables that make up your formula. The `configuration` object contains your variables, while the `triggers` and `steps` objects contain their relevant information. See the sample file below:

```
{
  "name":"Formula Name",
  "description":"Formula Description.",
  "configuration":[
    {
      "name":"instanceName",
      "type":"elementInstance",
      "key":"instanceName"
    },
    {
      "name":"instanceName2",
      "type":"elementInstance",
      "key":"instanceName2"
    }     {
      "name":"valueName",
      "type":"value",
      "key":"valueName"
    }
  ],
  "triggers":[
    {
      "type":"event",
      "properties":{
        "elementInstanceId":"${instanceName}"
      },
      "onSuccess":[
        "step1"
      ]
    }
  ],
  "steps":[
    {
      "name":"step1",
      "type":"filter",
      "properties":{
        "body":"done(trigger.event.eventType === 'UPDATED' && trigger.event.objectType === 'Conta
ct');"
      },
      "onSuccess":[
        "step2"
      ]
    },
    {
      "name":"step2",
      "type":"script",
      "properties":{
        "body":"done({\n  \"subject\": \"CRM Event Occurred\",\n  \"to\": \"recipient@cloud-eleme
nts.com\",\n  \"from\": \"sender@cloud-elements.com\",\n  \"message\": `${trigger.event.objectTyp
e} with ID ${trigger.event.objectId} was ${trigger.event.eventType}`\n});"
      },
      "onSuccess":[
        "step3"
      ]
    },
    {
      "name":"step3",
      "type":"elementRequest",
      "properties":{
        "method":"POST",
        "elementInstanceId":"${instanceName2}",
        "api":"/messages",
        "body":"${steps.step2}"
      }
    }
  ]
}
```

Test Formula Templates

You can test a formula template as you build it. When you test a formula template, you can either use an existing formula instance or build a new formula instance.

> ☑ **Tip:** If you plan to use the formula instance that you build while testing formulas only for testing, consider appending the name test to it.

To test a formula template:

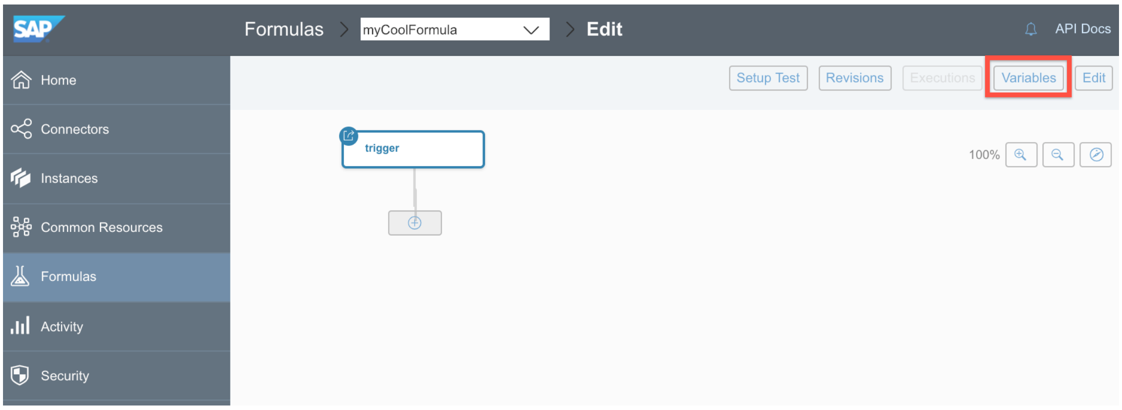
1. At the top of the formula visualization, click **Try It Out**.
2. Click **Select Instance**.
3. Either choose an existing formula instance (and skip down to the Select Trigger step) or click **Add New Instance**.
   1. Enter a name for the new formula instance.
   2. For each variable, click the variable and add a connector instance or enter a value.
   3. Click **Create Instance**.
   4. Select the instance that you just built.
4. Click **Select Trigger**.
5. Provide trigger information, depending on the type of trigger.

Create Variables

You can create two types of variables: **Connector Instance** variables — replaced by connector instances in the formula instance — and **Value** variables — replaced by values entered in the formula instance. When you create a variable, note the Formula Step Variable Name which is how you refer to the variable in the formula context. See Formula Variables for more about variables in the formula context.

To create a formula variable:

1. Open the formula template. On the Formulas page, hover over the formula card, and then click **Open**.
2. Click **Variables**.



3. Select the type of variable to create.
4. Enter a name for the variable. You can include spaces, but we remove the spaces to create the **Formula Step Variable Name** (identified as `key` in the JSON), which is how you refer to the variable in the formula context.
5. Click **Save**.

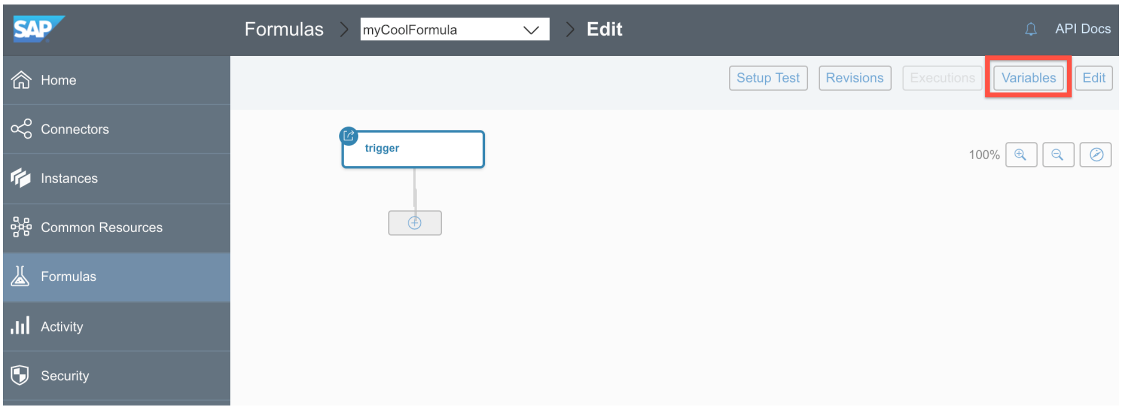
Edit and Delete Variables

You can edit the name of a variable or remove a variable from a formula template.

> ℹ **Note:** You cannot delete a formula variable if an instance of the formula exists.

To edit or delete a formula variable:

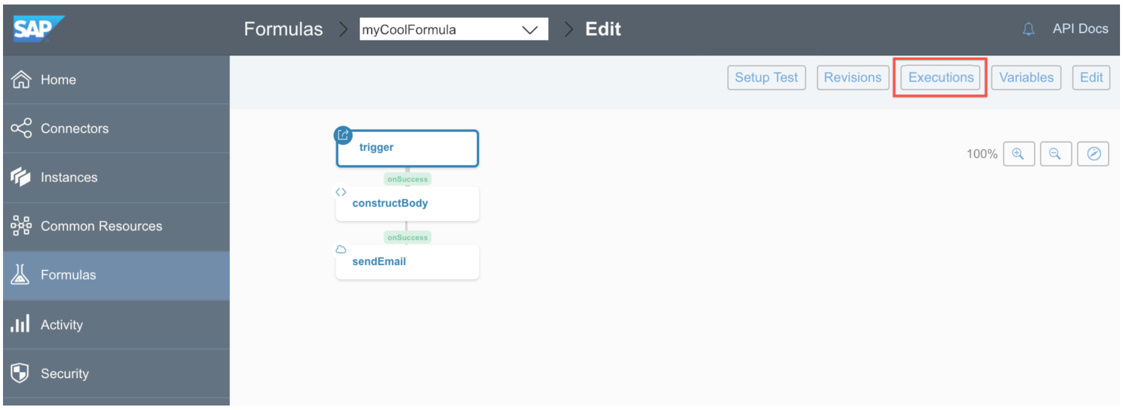
1. Open the formula template. On the Formulas page, hover over the formula card, and then click Open.
2. Click **Variables**.



3. Click **Edit Variable**.
4. Select the variable to edit or delete, and then:
   - Edit the name, and then click **Save**. Or,
   - Click **Delete**.

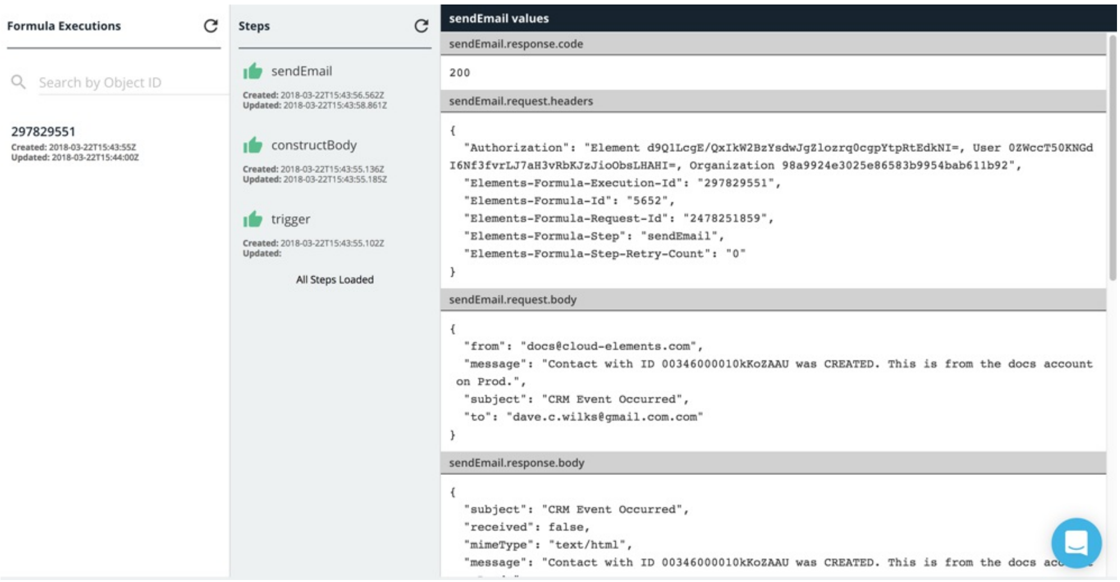
Review and Replay Executions

You can troubleshoot formula by reviewing executions.



The executions appear in three columns from left to right:

- Formula Executions — the list of executions of the selected formula template. We identify failed executions in red.
- Steps —The steps in the formula and an icon showing their success 👍 or failure 👎.
- Execution Values —the request and response details associated with the selected step.



Formula Executions

Search by Object ID

297829551
Created: 2018-03-22T15:43:55Z
Updated: 2018-03-22T15:44:00Z

Steps

sendEmail
Created: 2018-03-22T15:43:56.562Z
Updated: 2018-03-22T15:43:58.861Z

constructBody
Created: 2018-03-22T15:43:55.136Z
Updated: 2018-03-22T15:43:55.185Z

trigger
Created: 2018-03-22T15:43:55.102Z
Updated:

All Steps Loaded

sendEmail values

sendEmail.response.code

```
200
```

sendEmail.request.headers

```
{
  "Authorization": "Element d9Q1LcgE/QxIkW2BzYsdwJgZlozrq0cgpYtpRtEdkNI=, User 0ZWccT50KNGd
I6Nf3fvrLJ7aH3vRbKJzJioObsLHAHI=, Organization 98a9924e3025e86583b9954bab611b92",
  "Elements-Formula-Execution-Id": "297829551",
  "Elements-Formula-Id": "5652",
  "Elements-Formula-Request-Id": "2478251859",
  "Elements-Formula-Step": "sendEmail",
  "Elements-Formula-Step-Retry-Count": "0"
}
```

sendEmail.request.body

```
{
  "from": "docs@cloud-elements.com",
  "message": "Contact with ID 00346000010kKoZAAU was CREATED. This is from the docs account
 on Prod.",
  "subject": "CRM Event Occurred",
  "to": "dave.c.wilks@gmail.com.com"
}
```

sendEmail.response.body

```
{
  "subject": "CRM Event Occurred",
  "received": false,
  "mimeType": "text/html",
  "message": "Contact with ID 00346000010kKoZAAU was CREATED. This is from the docs acc
```

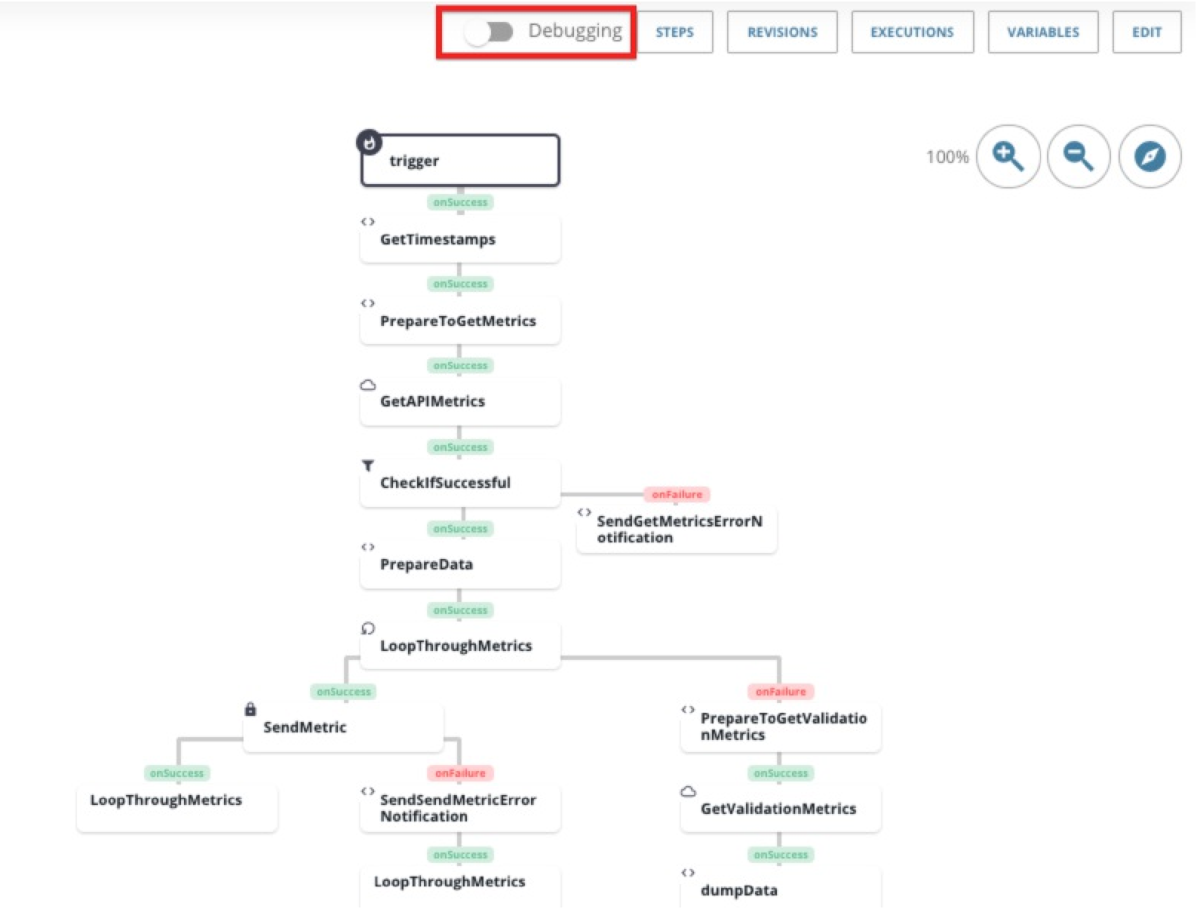
> ℹ **Note:** To review executions, you must have a formula instance. Otherwise, the Executions button is unavailable.

To review executions:

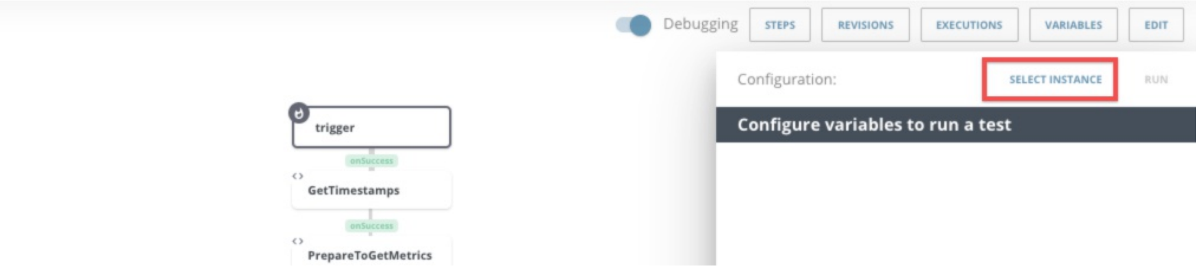
1. On the Formulas page, hover over the formula card, and then click **Open**.
2. Click **Executions**.
3. Click the execution to review.
4. Click the step to review.

To replay executions:

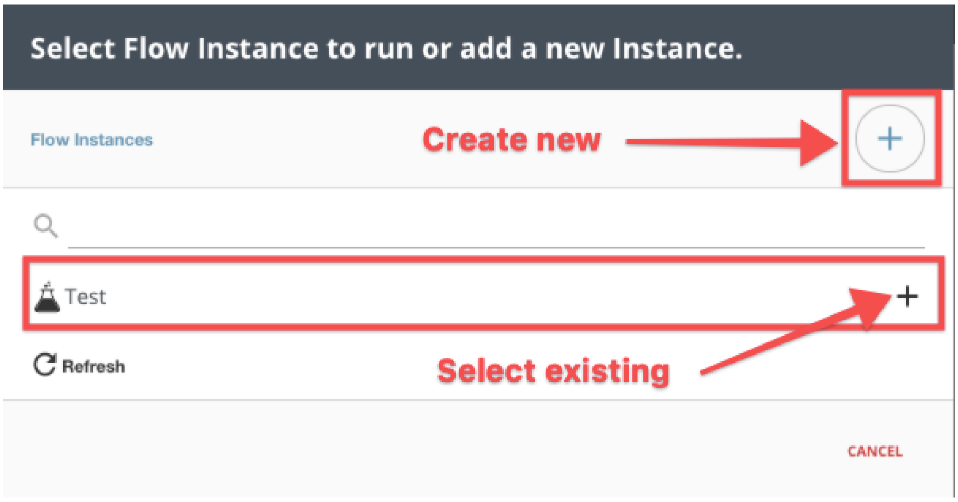
1. On the Formulas page, hover over the formula card, and then click **Open**.
2. Enable debugging by clicking the **Debugging** toggle.



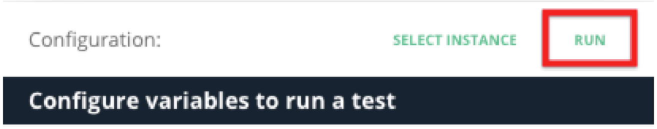
3. Click **Select Instance**.



4. Either select an existing formula instance by clicking its corresponding plus sign button, or click the Create Formula Instance button to create a new instance and define its variables.



5. After selecting an instance, click Run.



Configure variables to run a test

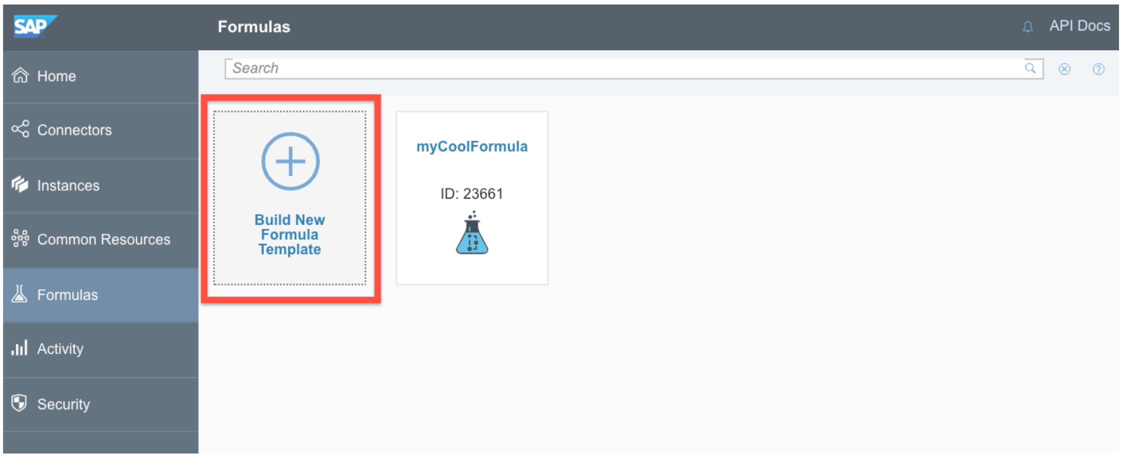
As the execution runs, you can click individual steps in order to see the step's execution value.

Copy a Formula Template

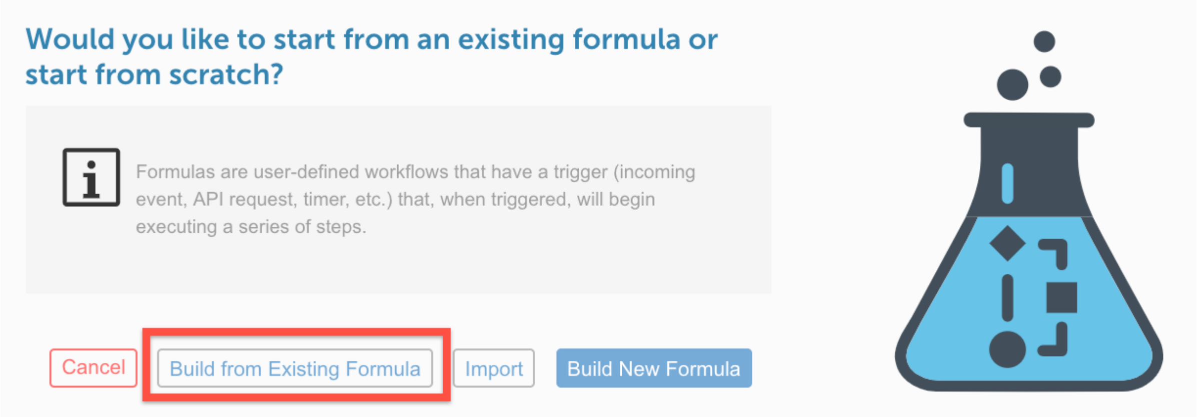
You can build a formula template based on an existing template or from a SAP Open Connectors sample formula, resulting in a copy of the template.

To build a copy of a formula template:

1. Click **Formulas**, and then on the Formulas page, click **Build New Formula**.



2. Click **Build From Existing Formula**.



Would you like to start from an existing formula or start from scratch?

Formulas are user-defined workflows that have a trigger (incoming event, API request, timer, etc.) that, when triggered, will begin executing a series of steps.

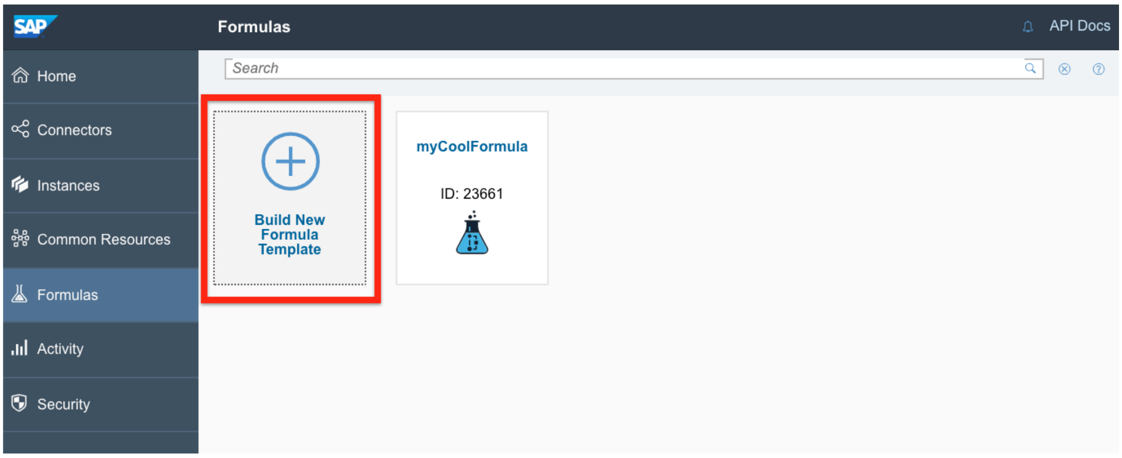
   - To use one of your existing formula templates, select the template.
   - To use a SAP Open Connectors formula template, click CE Sample Formulas, and then select the formula.
3. Enter a name for your formula, and then click **Create**.

Import and Export Formula Templates

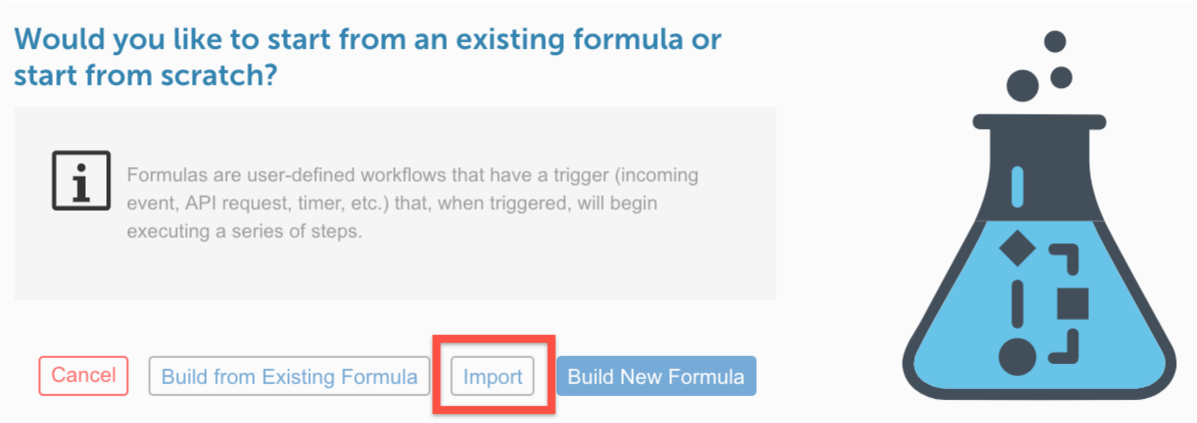
You can export a formula template as a JSON file and import other templates from their JSON source file.

To import a formula template:

1. Click **Formulas**, and then on the Formulas page, click **Build New Formula**.



2. Click **Import**.


Would you like to start from an existing formula or start from scratch?

Formulas are user-defined workflows that have a trigger (incoming event, API request, timer, etc.) that, when triggered, will begin executing a series of steps.

3. Select the JSON file to import.
4. Optionally rename the file, and then click **Create**.

To export a formula:

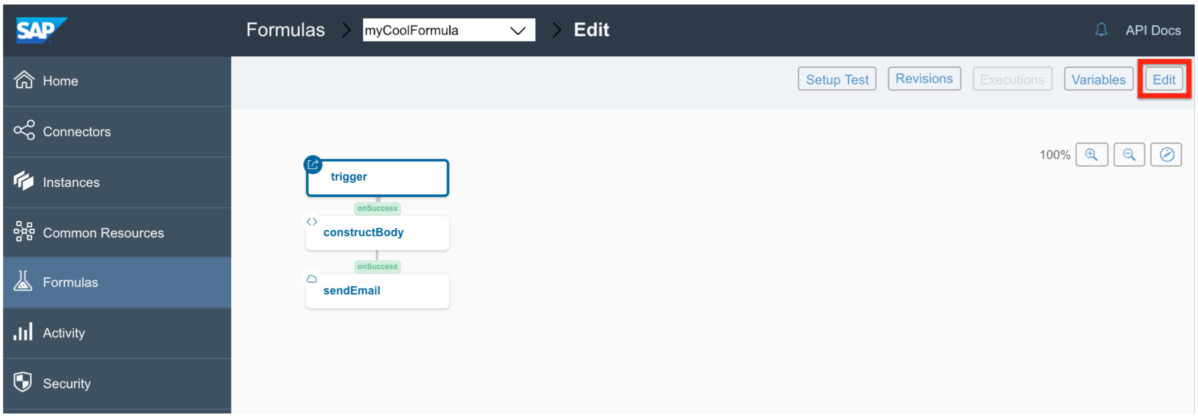
1. Navigate to the Formulas page.
2. Hover over the connector card, and then click **Export**.
3. Save the file.

Parallel Executions

To help formula instances execute as efficiently as possible, they take advantage of multithreaded executions. That is, multiple executions of the same formula instance can make requests at the same time. Some API providers prevent multiple requests from the same account at the same time. If this happens, you can change a formula template or an individual formula instance to execute one step at a time. This makes the formula less efficient and results in an increase in the execution time.

To change a formula from the default multithreaded execution to single threaded:

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. Switch **Single Threaded Execution** on.
5. Click **Save**.

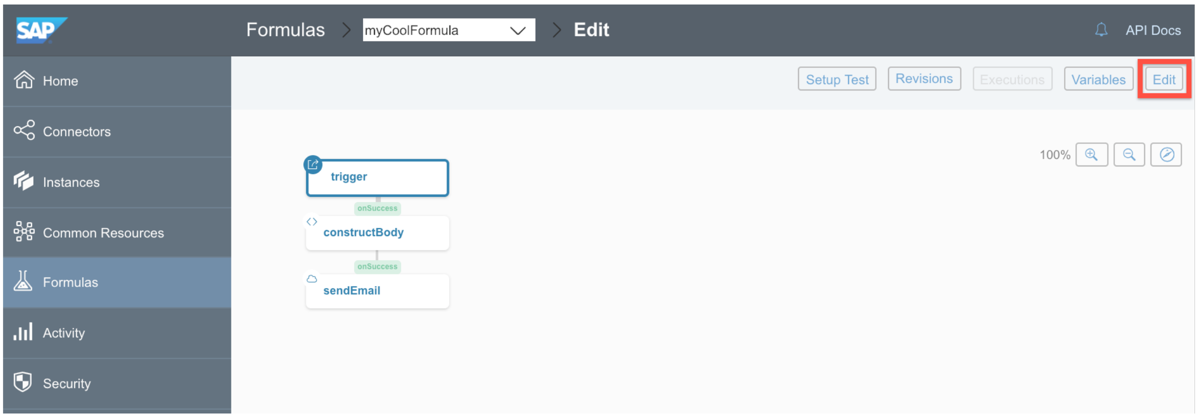
Alternatively, in the formula template or formula instance JSON, update `singleThreaded` to `true` :

```
{
"singleThreaded": true
}
```

Rename a Formula Template

To rename a formula template:

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



3. Rename the formula template.
4. Click **Save**.

Add a Description to a Formula Template

Descriptions provide more information about a formula template. Use the information to describe the formula template use case and anything another user might need when creating a formula instance. See Formula Readmes and Descriptions for additional information.

To add a description to a formula template:

1. Open the formula template and click the space beneath **Formula Description**.
2. Add your formula description and click **Save**.

Delete a Formula Template

You can delete a formula template if the template has no instances associated with it. If the formula template includes instances, delete those first.

To delete a formula template:

1. Navigate to the Formulas page.
2. Hover over the Formula Template card, and then click **Delete**.
3. Confirm the deletion.

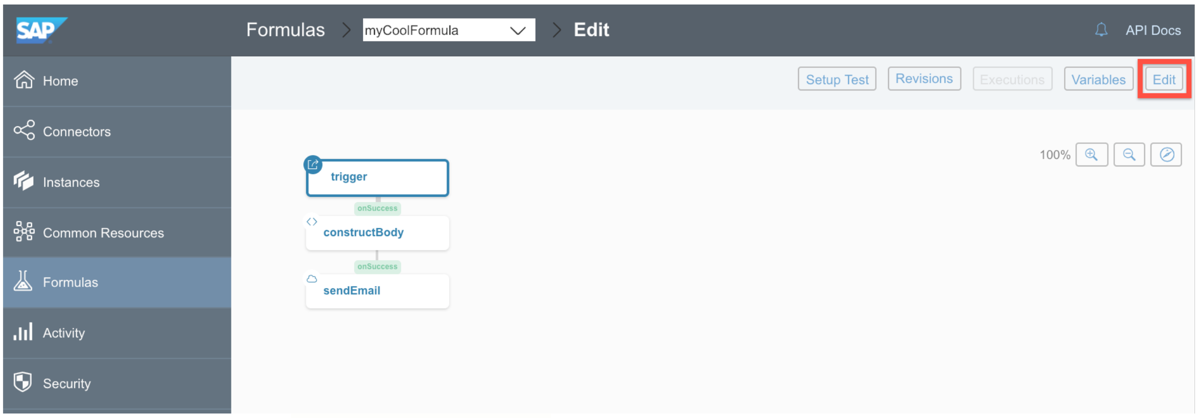
Deactivate a Formula Template

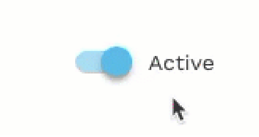
You can deactivate a formula template to prevent any executions of formula instances of the template.

> **Note:** You can also deactivate a specific formula instance. See Deactivate a Formula Instance.

To deactivate a formula:

1. Open the formula template. From the Formulas page, hover over the formula template card, and then click **Open**.
2. Click **Edit**.





3. Switch the **Active** slider off.
4. Click **Save**.

Access Formula Information

Each formula template and formula instance exposes information about itself through the `info` object.

```
{
  "info":{
   "formulaId":"123",
   "formulaName":"name",
   "formulaInstanceId":"123",
   "formulaInstanceName":"name",
   "formulaExecutionId":"1234"
  }
}
```

Formula information contributes to the formula context. You can access the information with `info.property`, such as `info.formulaInstanceId` or `info.formulaName`.