11/05/2021

Convert API requests to suit your system

Bosbec Workflow Builder

English

Bosbec AB
Färögatan 33
164 51 Kista

Phone: +46 (8) 22 25 25
E-Mail: info@bosbec.se
www.bosbec.se

Table of Contents

Introduction 3
Build your request URL 4
    Obtain your API token........ 4
    Create the web URL 4
Configure the workflow 5
    Trigger 6
    Get query parameter 7
    Request setup 8
    Request to the endpoint 9
Further reading 11

Introduction

An API, Application Programming Interface, allows two systems to communicate with each other and serves as a vital building block in modern applications.

The API serves as a set of rules that defines how a software program can request information from other software, and even though APIs are getting standardized in terms of accessibility (HTTP and REST), the use case can still differ in how a system makes the request. This guide covers how you can build your own API converter and use a simple web URL request initiate a more advanced REST request.

In this scenario a system will send a web URL request containing a query parameter to a Bosbec Workflow and the workflow will convert this to a REST request suitable for the endpoint.

Prerequisites:

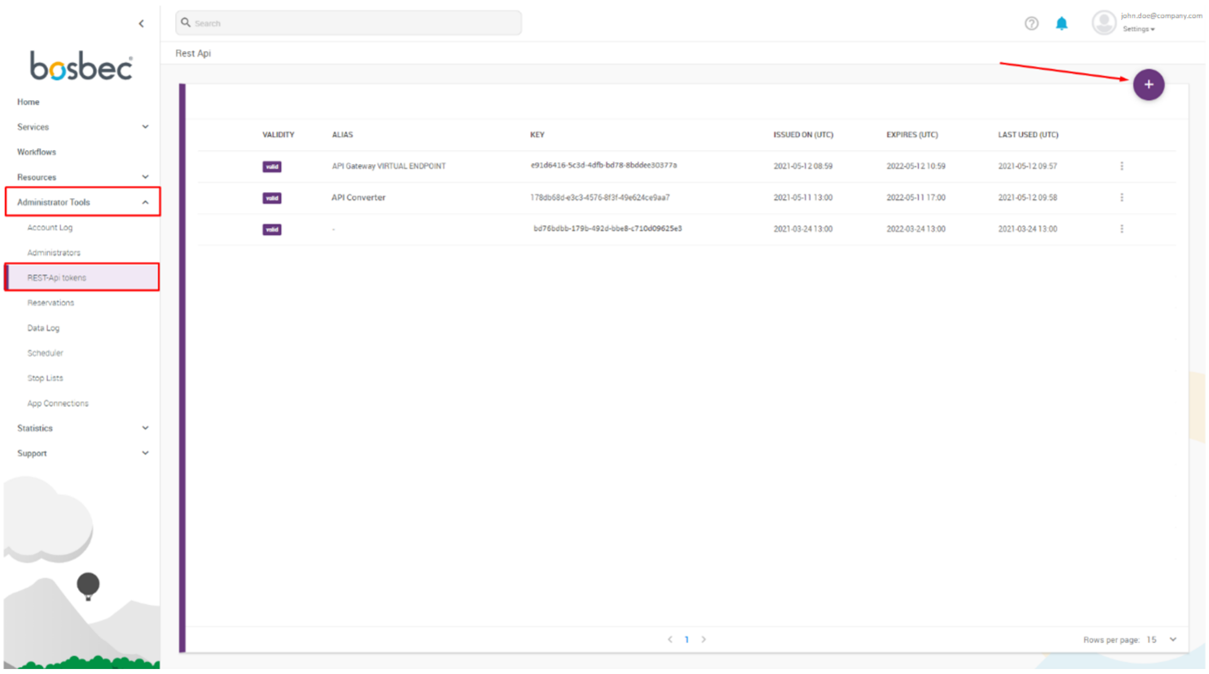
- You are logged in as an administrator at www.bosbec.io
    - *Permissions must be Moderator or higher to be able to create workflow.*
- You have created an Authentication token in the administrator interface.
    - *This token is used to request the workflow in this example, although it is possible to create this solution with public settings for the "Incoming HTTP trigger". Learn more about this on our help page, "Incoming data".*

Build your request URL

The Workflow Builder will react on an incoming web URL request to the HTTP IN-system. To enable the communication between your system and the workflow unique, it is required to have an API token. You can make your web URL adaptable to domains and paths, whichever is suitable to your system, but in this example we will use an API token.

Obtain your API token

Obtain your API token in the administrative interface. In the navigation menu to your left, click on "Administrator Tools", "REST Api-tokens" and the plus icon to your left to create a new token. See the image below.



VALIDITY	ALIAS	KEY	ISSUED ON (UTC)	EXPIRES (UTC)	LAST USED (UTC)
valid	API Gateway VIRTUAL ENDPOINT	e91d6416-5c3d-4dfb-bd78-8bddee30377a	2021-05-12 08:59	2022-05-12 10:59	2021-05-12 09:57
valid	API Converter	178db68d-e3c3-4576-8f3f-49e624ce9aa7	2021-05-11 13:00	2022-05-11 17:00	2021-05-12 09:58
valid	-	bd76bdbb-179b-492d-bbe8-c710d09625e3	2021-03-24 13:00	2022-03-24 13:00	2021-03-24 13:00

Image 1: Obtain an API token in the administrative interface, click the plus icon to create a new API token.

Create the web URL

Now you can create your web URL to use from your system which will call the workflow. The web URL is in the format:

```
https://in.bosbec.io/2/[TOKEN]
```

In this example we will put a query parameter (city=Stockholm) at the end of the URL, so our web URL is:

```
https://in.bosbec.io/2/178db68d-e3c3-4576-8f3f-
49e624ce9aa7?city=Stockholm
```

Configure the workflow

So far, we have obtained an API token and a web URL used to send information to and initialize the workflow. The next step then becomes quite natural to build the workflow. The workflow consists of one trigger and four jobs. In this section we will go through the configurations of each job to make this API converter ready to use.

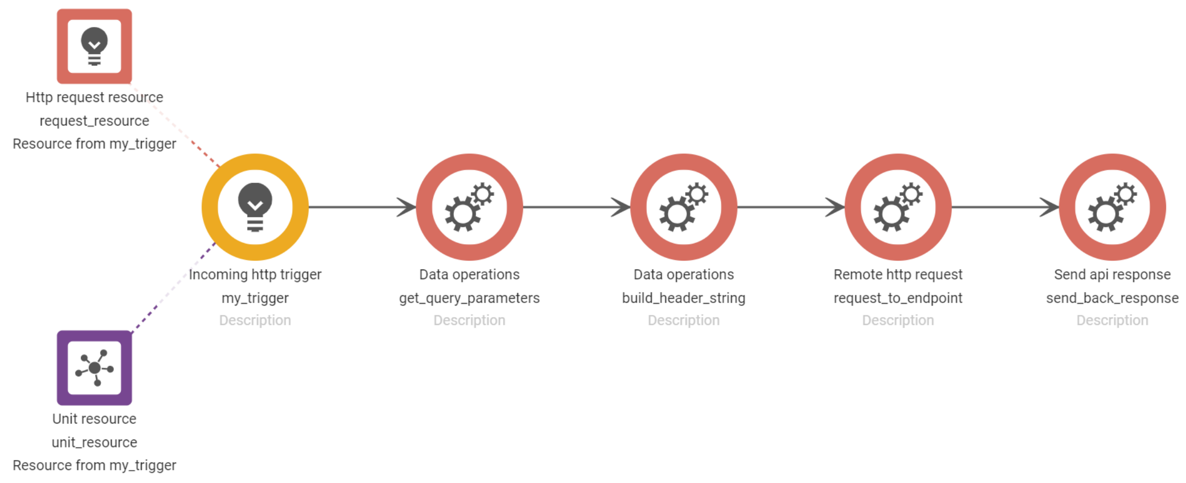
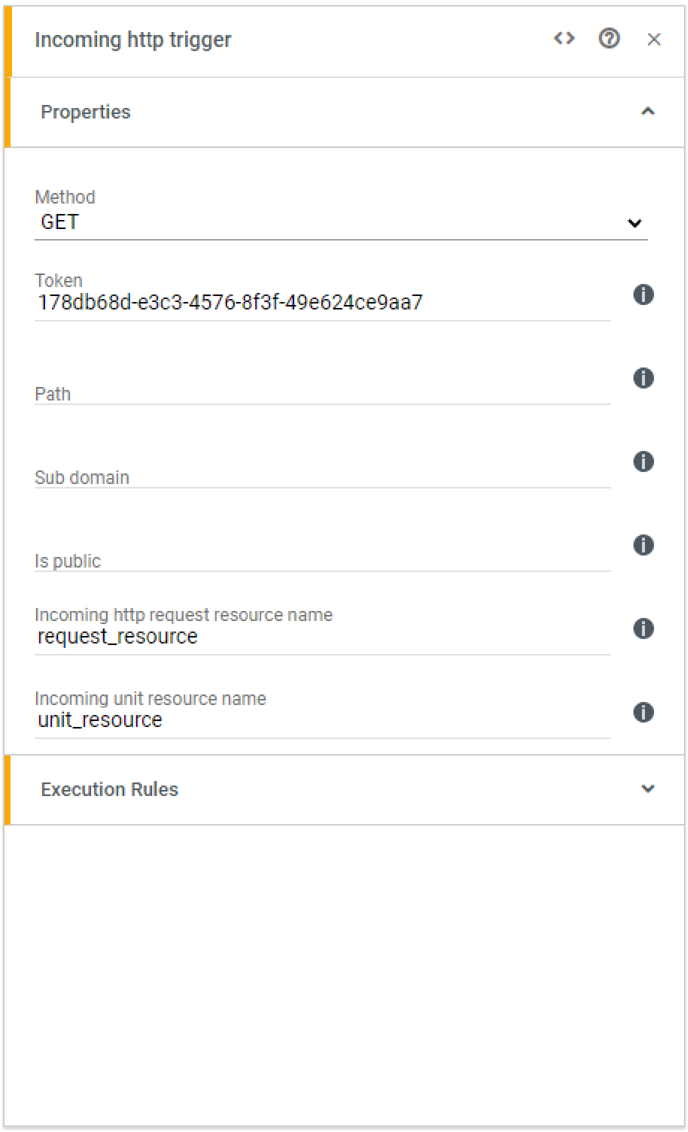


Image 2: API converter workflow.


Incoming http trigger

Properties

Method
GET

Token
178db68d-e3c3-4576-8f3f-49e624ce9aa7

Path

Sub domain

Is public

Incoming http request resource name
request_resource

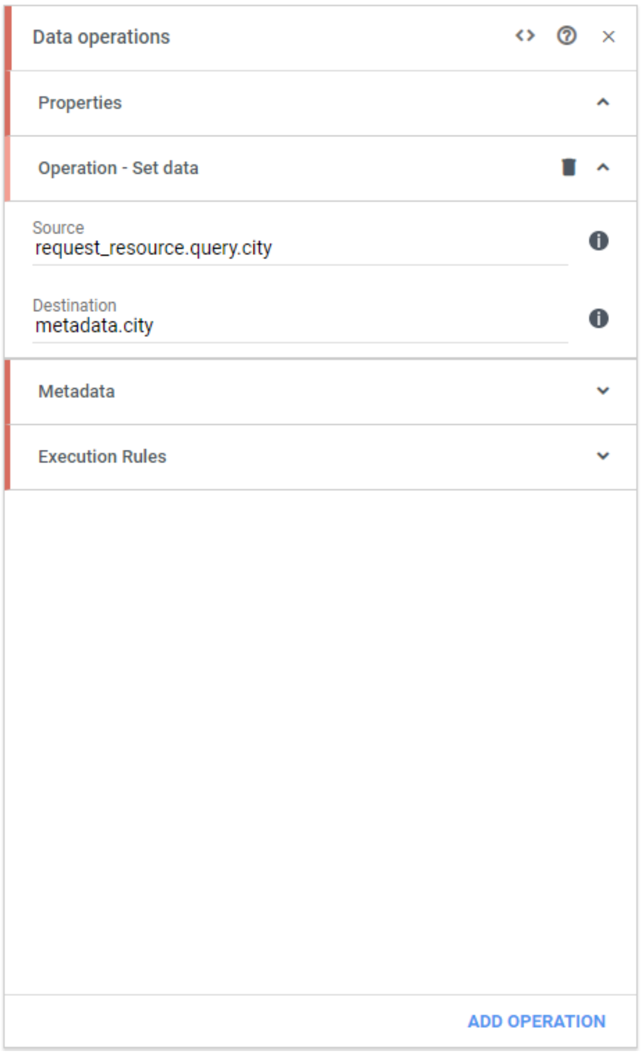
Incoming unit resource name
unit_resource

Execution Rules

Image 3: Set Method to "GET" and use the API token, created in the previous step.

Trigger

Use the trigger "Incoming http trigger". It will react on an incoming HTTP request (the one we created in the previous step). Set the method (GET) and the token you created in the administrative interface and set in your web URL. This is your authentication token and is used to connect the workflow to the request, as it also serves as an API key, so only you who know the token will be able to initialize the workflow.



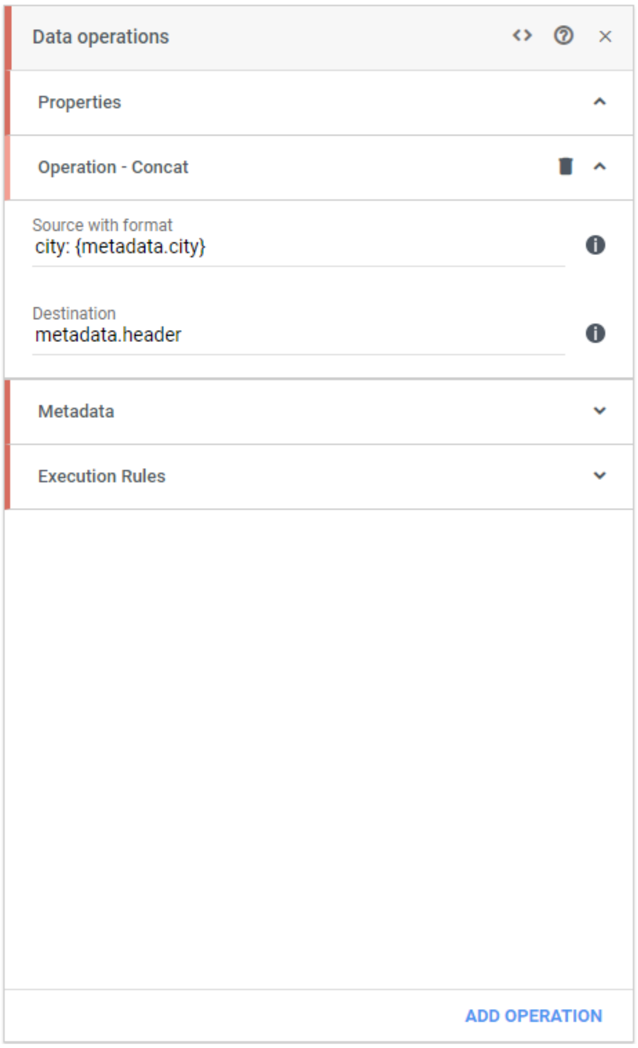
Source
request_resource.query.city

Destination
metadata.city

Image 4: Input parameter "city=Stockholm" extracted from the request resource.

Get query parameter

All the information about the request is stored in the resource "request_resource", which is automatically generated when an incoming HTTP request is received. In the "Data Operations" job, use "Set data" to extract the query from the incoming request and store it in a metadata called city. The metadata, along with any resources, act as a portfolio of all your data and information in the workflow context.



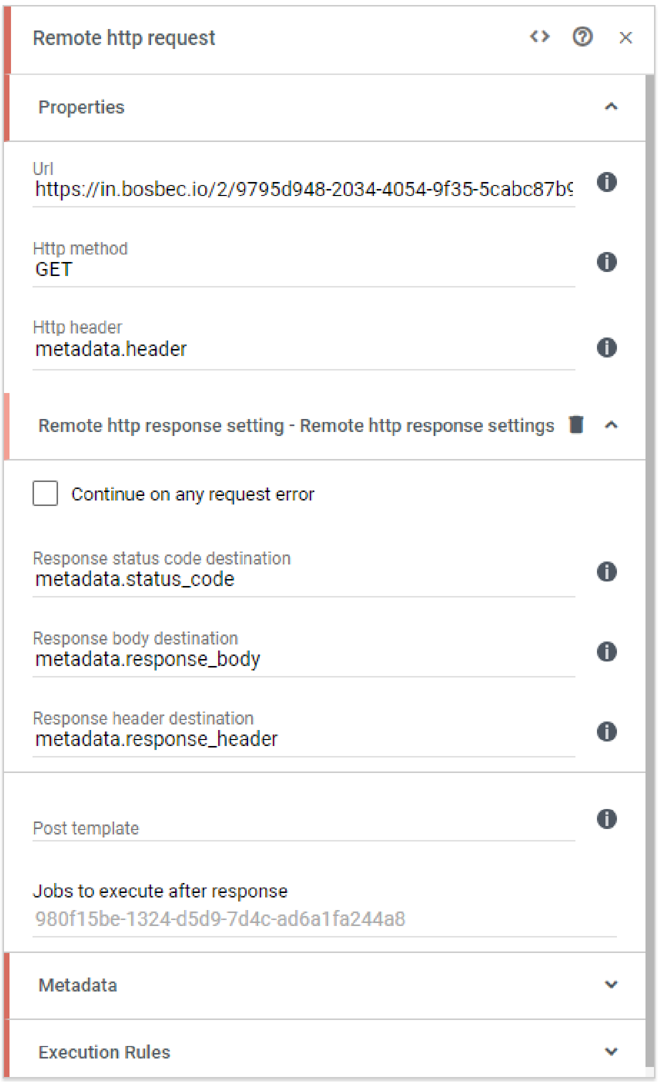
Source with format
city: {metadata.city}

Destination
metadata.header

Image 5: String concatenation with the key "city" and the value in our metadata.

Request setup

qNext, we use a "Data Operations" job and "Concat" to concatenate a key-value string which will be our header in the request we will send from the workflow. We store this result in "metadata.header".



Remote http request

Properties

Url
https://in.bosbec.io/2/9795d948-2034-4054-9f35-5cabc87b9

Http method
GET

Http header
metadata.header

Remote http response setting - Remote http response settings

☐ Continue on any request error

Response status code destination
metadata.status_code

Response body destination
metadata.response_body

Response header destination
metadata.response_header

Post template

Jobs to execute after response
980f15be-1324-d5d9-7d4c-ad6a1fa244a8

Metadata

Execution Rules

Image 6: The request to the endpoint from the workflow.

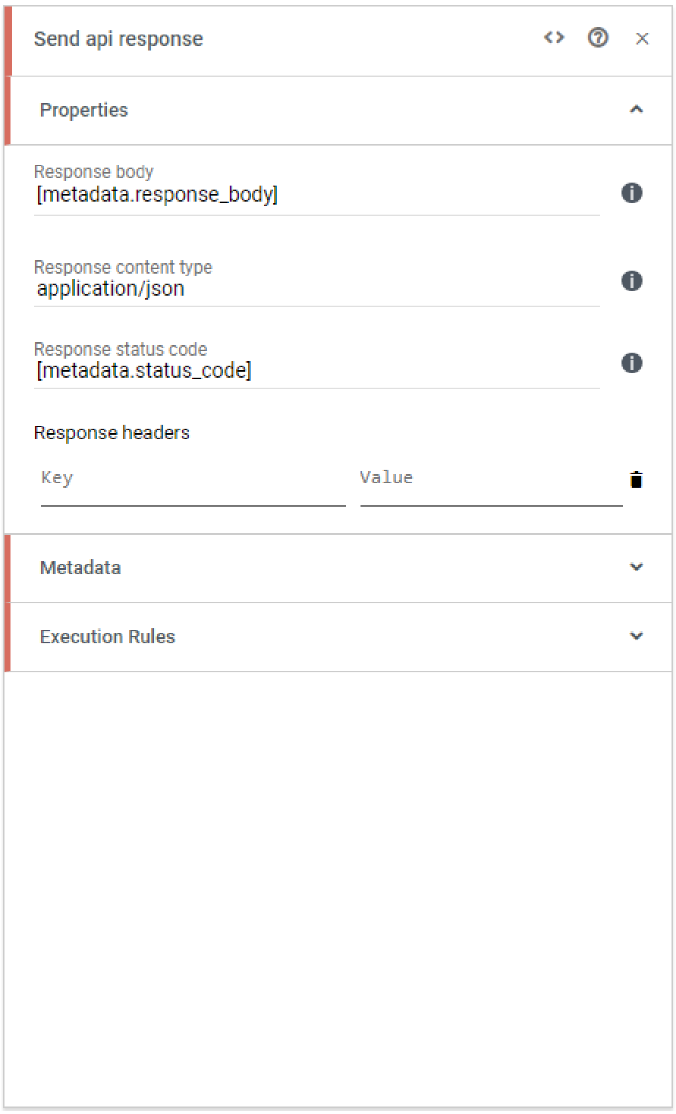
Request to the endpoint

Now that we have extracted the information from our system we can make the request that suits the endpoint. In this case the endpoint only accepts the input parameters from a REST request header.

We have simulated an endpoint with another Bosbec Workflow, but in your case set the external endpoint URL in the URL property.

Set the HTTP method, in this case "GET", and set the HTTP header as "metadata.header", where we stored the input parameter from our request resource.

Since we want to return the response back to our system we want to store it someplace. In the "Remote http response setting", set the destination for status code, response body and response header.



Send api response

Properties

Response body
[metadata.response_body]

Response content type
application/json

Response status code
[metadata.status_code]

Response headers

Key | Value

Metadata

Execution Rules

Image 7: Sending the response back to the system.

Return the response to the system

At this stage we have sent the input parameter to the workflow, extracted the information we need and sent a request to the endpoint.

We have also received the response to the workflow, which we want to send back to the system that made the original request.

Remember the “Remote HTTP Request” job in the previous step? We set some variables which would handle the response. This is the information we now want to use to send the response back to the system.

Use the job “Send api response” and set the response body and status code. Remember to set the content type to the type you want your response. In this example we want it as application/json.

The request is then returned in a JSON format at the start of the chain, this is an example of how a web URL can be sent to a Bosbec workflow which modifies and forwards the request which then returns the following JSON payload:

```
{
    “temperature_c”: 20,
    “rain_mm”:0,
    “windspeed_mps”:5,
    “airhumidity_perc”: 53,
    “airpressure_hpa”: 1011
}
```

Remember to connect all the jobs and activate the workflow, now you have your IT solution up and running!

Further reading

This solution is based is based on the concept of working with incoming and outgoing data in the Bosbec Workflow Builder. If you want to learn more about these concepts, read more at https://help.bosbec.io/.