cortex®
Intelligent Automation

Cortex Flow API User Guide

Cortex Limited
Kings Park House,
22 Kings Park Road,
Southampton,
SO15 2AT

T +44 23 8254 8990
E **info@cortex-ia.com**

Status: Release
Release Date: 11/09/2020
Author: Cortex Product Development
Document Version: 0.1
Software Version: 7.1

Preface

About this User Guide

This User Guide explains how to use the Flow API. The guide is organised into the following sections:

- **Section 1 - Authenticating with the Flow API**

  This section describes how to pass authentication credentials when calling the Flow API.

- **Section 2 - Using the Flow API**
  This section describes how to get available flows, trigger a flow execution and stop a flow execution using the Flow API.

- **Section 3 - Passing Complex Arguments**
  This section describes how to pass complex arguments into a flow execution using both structures and lists

Audience

The intended audience is primarily those responsible for integrating 3rd party systems, such as websites and web applications with the Cortex platform. This User Guide assumes a basic working knowledge of API testing tools and uses Postman in its examples.

Abbreviations used in this Document

API – Application Programming Interface

HTTP – Hypertext Transfer Protocol

JSON – JavaScript Object Notation

UUID – Universally Unique Identifier

Revisions to this Document

The following revisions have been made to this document

Date	Author	Revision	Notes
08/09/2020	D. Smith	0.1	First Draft

1 Authenticating with the Flow API

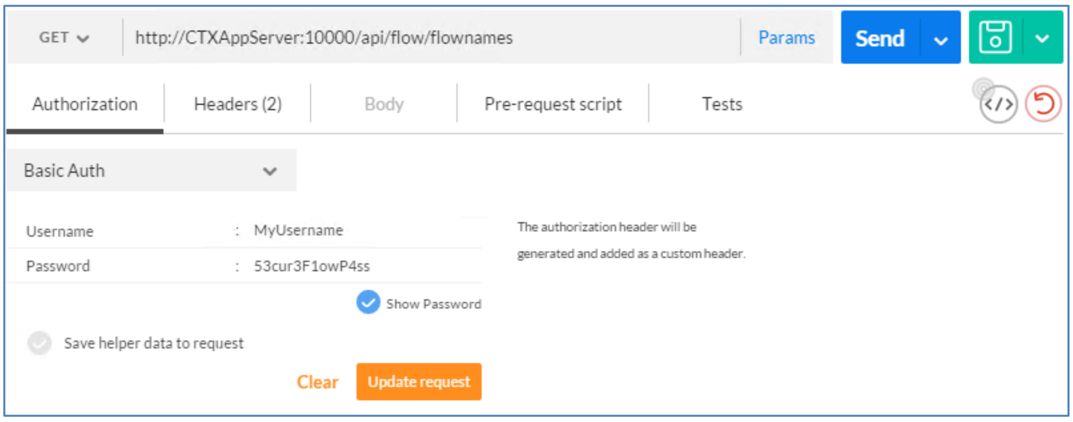
To use the Flow API, authentication credentials that match those specified in the Flow API configuration must be included with every HTTP request.

These credentials include a Username and Password and must be sent using Basic Auth, which is currently the only authentication option supported.

1.1 Configuring Credentials using Basic Auth in Postman

To send an authenticated request:

1. Find out the Username and Password that must be used:
   a. These can be found in the Innovise.Cortex.Web.Owin.dll.config file, located in the Cortex Flow Interface Service installation directory.
   b. Username is the value specified in the AllowedUser setting. It does not support Cortex encrypted strings.
   c. Password is the value specified in the AllowedPassword setting. It does support Cortex encrypted strings.
2. Select the Authorization tab in Postman
3. In the dropdown, change the authorisation type to Basic Auth. A dialog should appear to enter the Username and Password
4. Enter the required Username and Password
5. Click ‘Update Request’
6. The call can then be sent.



GET http://CTXAppServer:10000/api/flow/flownames

Basic Auth

Username : MyUsername
Password : 53cur3F1owP4ss

✓ Show Password

The authorization header will be generated and added as a custom header.

Figure 1 - Using Credentials for a request

2 Using the Flow API

2.1 Getting Flows

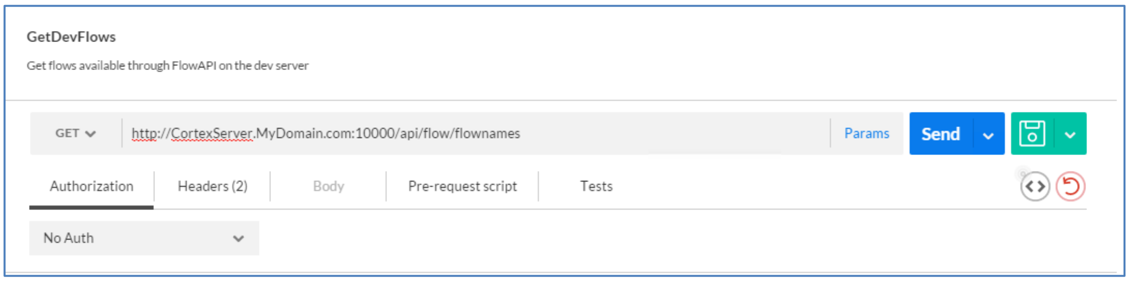
2.1.1 Getting available flows

To list the available flows, enter the following in Postman:

HTTP Action: GET

URL: http://*<server>*:10000/api/flow/flownames

This should return a list of the flows available to call. This uses the Display Name property on a flow – for a flow to be accessible via the Flow API the export property must be set to 'True' in the flow's LivePortal Properties.


GetDevFlows

Get flows available through FlowAPI on the dev server

GET http://CortexServer.MyDomain.com:10000/api/flow/flownames

Authorization: No Auth

Figure 2 - Get available flows

Response:

```
[
        "FLOW1",
        "FLOW2",
        "FLOW3"
]
```

2.2 Triggering a Flow Execution

2.2.1 Headers

When calling a flow, the headers must be set up as follows:

Header Values: Accept: application/json
Content-Type: application/json

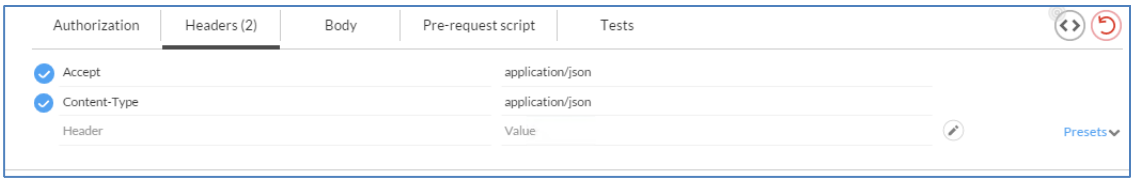

Header	Value
✓ Accept	application/json
✓ Content-Type	application/json

Figure 3 - Headers on flow execution call

2.2.2 Body

The body must be set up for any required parameters using JSON format:

- Name
  This is the Flow Display Name (as shown in the Get Flows call), in the format:
  {"Name": "FlowDisplayName"}
- Arguments
  Any arguments which should be passed to the flow, in the format:
  {"Arguments": {"Var1": "Value1", "Var2": "Value2"} }
- ReturnParameters
  Any Cortex variables that should be returned, in the format:
  {"ReturnParameters":["OutputVar1", "OutputVar2"]

These Body parameters should all be enclosed in one set of braces and separated by commas:

```
{
        "Name": "FlowDisplayName",
        "Arguments": "{"arg1": "val1"},
        "ReturnParameters": ["txt"]
}
```

2.2.3 Calling a Flow (Synchronous)

This method calls the flow then waits for it to finish. This is mainly used when the flow should return some value(s) or if the flow is required to complete before progressing. To call one of the available flows, enter the following in Postman:

HTTP Action: POST

URL: http://*<server>*:10000/api/flow/startflow

Header Values: Accept: application/json
Content-Type: application/json

Body:
{

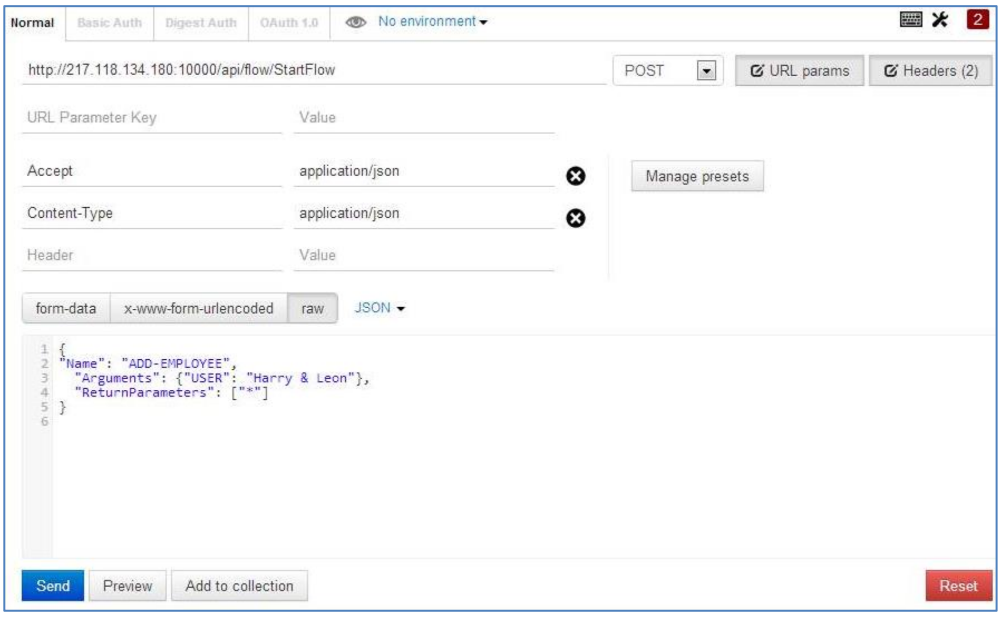
"Name": "Flow Name",

"Arguments": {"TEXT-VAR": "ABC", "INT-VAR": 123, "FLOAT-VAR": 45.67},

"ReturnParameters": ["OUT-VAR-1", "OUT-VAR-2"]

}

Returns: This returns any parameters listed in "ReturnParameters"



http://217.118.134.180:10000/api/flow/StartFlow

Accept	application/json
Content-Type	application/json

```
{
"Name": "ADD-EMPLOYEE",
  "Arguments": {"USER": "Harry & Leon"},
  "ReturnParameters": ["*"]
}
```

Figure 4 - Calling a flow (Synchronous)

2.2.4 Calling a Flow (Asynchronous)

This method calls a flow then leaves it to complete (‘Fire & Forget’).

HTTP Action: POST

URL: http://*<server>*:10000/api/flow/StartFlowAsync

Header Values: Accept: application/json
Content-Type: application/json

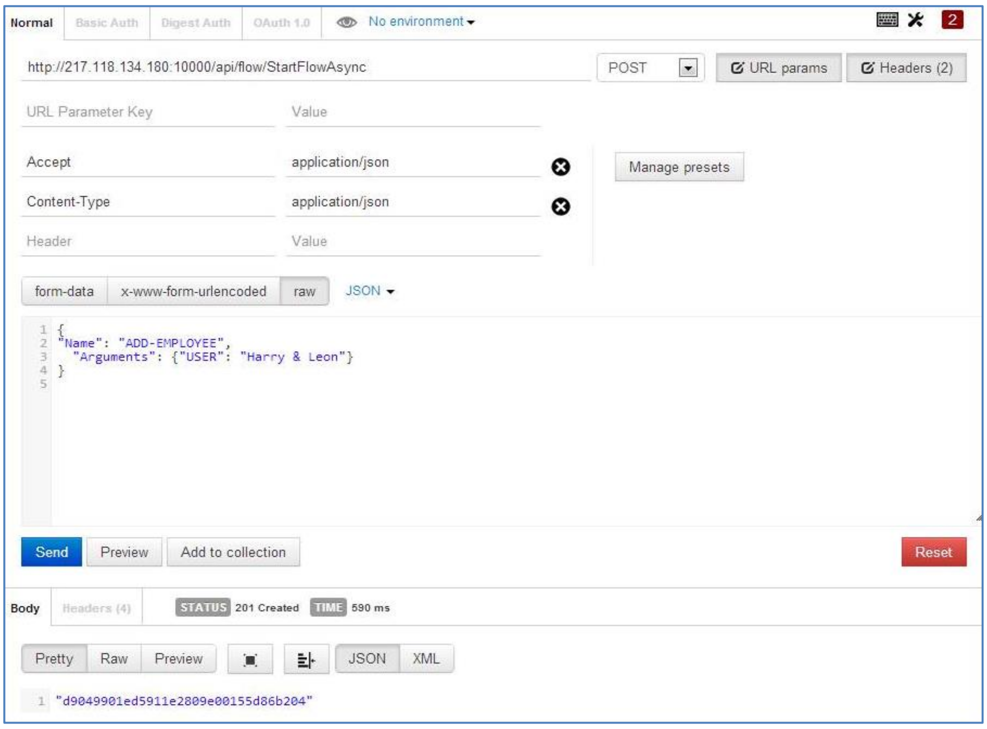
Body:
{

"Name": "Flow Name",

"Arguments": {"TEXT-VAR": "ABC", "INT-VAR": 123, "FLOAT-VAR": 45.67}

}

Returns: This should return the UUID of the flow execution that has been started.



http://217.118.134.180:10000/api/flow/StartFlowAsync

URL Parameter Key	Value
Accept	application/json
Content-Type	application/json
Header	Value

```
{
"Name": "ADD-EMPLOYEE",
  "Arguments": {"USER": "Harry & Leon"}
}
```

STATUS 201 Created TIME 590 ms

```
"d9049901ed5911e2809e00155d86b204"
```

Figure 5 - Calling a flow (Asynchronous)

2.2.5 Stopping a Flow Execution

This method stops a flow execution using the UUID.

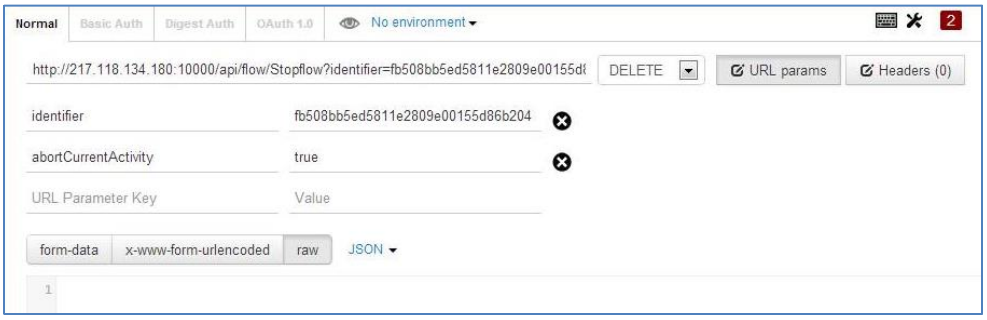
HTTP Action: DELETE

URL:
http://<*server*>:10000/api/flow/Stopflow?identifier=<*executionID*>&abortCurrentActivity=true

Header Values: Identifier: <*Execution UUID*>
abortCurrentActivity: true

Body: Nothing is required in the body to stop a flow.

Returns: 'True' if successful, 'False' if not found or unsuccessful.



http://217.118.134.180:10000/api/flow/Stopflow?identifier=fb508bb5ed5811e2809e00155d

URL Parameter Key	Value
identifier	fb508bb5ed5811e2809e00155d86b204
abortCurrentActivity	true
URL Parameter Key	Value

Figure 6 - Stopping a flow execution

3 Passing Complex Arguments

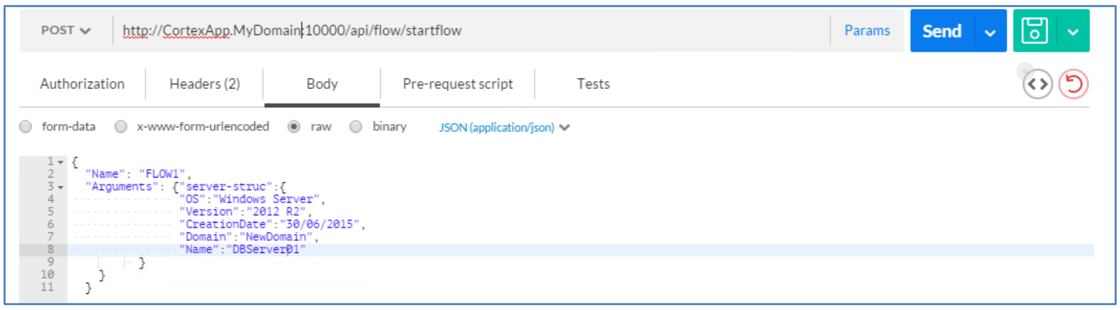
Complex Arguments can be passed in as structures or lists to Cortex via the Flow API. These should be included under the comma-separated 'Arguments'. They can be sent in addition to any other arguments.

3.1 Structures

To pass a structure, the argument should be built up as follows:

```
"<str-var>":{
        "param_a":"Parameter A",
        "param_b":"Parameter B",
        "param_c":"Parameter C",
        "param_d":"Parameter D",
        "param_e":"Parameter E"
 }
```

Where `<str-var>` is the name of the structure variable in Cortex, `"param_a"` is an attribute name of the structure and `"Parameter A"` is the associated value.



POST http://CortexApp.MyDomain:10000/api/flow/startflow

Body: raw [✓] — JSON (application/json)

```
{
  "Name": "FLOW1",
  "Arguments": {"server-struc":{
                "OS":"Windows Server",
                "Version":"2012 R2",
                "CreationDate":"30/06/2015",
                "Domain":"NewDomain",
                "Name":"DBServer01"
        }
    }
  }
```

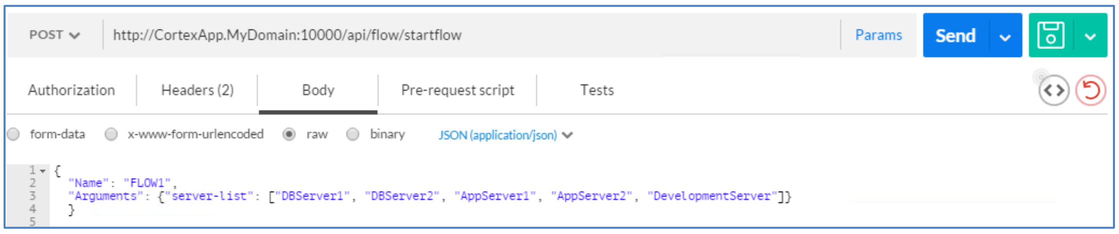
Figure 7 - Passing complex arguments as a structure

3.2 Lists

To pass a list, the argument should be built up as follows:

```
"<list-var>": [1,2,3]
```

Where `<list-var>` is the name of the list variable in Cortex, and the comma separated items in the array are the elements of the list to pass to the flow.



POST http://CortexApp.MyDomain:10000/api/flow/startflow

Body: raw [✓] — JSON (application/json)

```
{
  "Name": "FLOW1",
  "Arguments": {"server-list": ["DBServer1", "DBServer2", "AppServer1", "AppServer2", "DevelopmentServer"]}
  }
```

Figure 8 - Passing complex arguments as a list