UiPath – Orchestrator API

Author: Siddhesh Sawant

Introduction

UiPath Orchestrator is a well-known component now in the world of Robotic Process Automation. It is the heart of automation management. It gives the power to provision, deploy, trigger, monitor, measure, track, and ensure the security of every robot in organization. You can manage everything from your browser—or your mobile device. The UiPath Orchestrator API is meant to help easily access and manage all resources available in the Orchestrator web interface.

This article is mainly about Orchestrator API and how we can leverage this API as a part of automation to generate the execution report for the business. The article assumes having some knowledge of APIs as some little details may not be possible to explain.

Use Case

Every business using automation for their day to day work, requires an execution report stating what all records that bot has processed. This report is kind of visibility for businesses where human interaction is required due to any failure. In short, business would like to know if the record is processed successfully or if there was any failure.

Report for every day execution of bot

A call center agent gets help from bot which is running every day at scheduled time, processing some number of records fetching from queue. The bot is developed to maintain the statuses of every record in memory/data table, and at the end of the execution an email can be shared with agents, attaching Excel/csv file with statuses for each record.

This will work well for few records, but what if the bot is processing up to or more than million records. Storing in huge amount of data in memory is not advisable. There is a risk of issue during the execution and bot may not be able to send the email and will end up losing all the details maintained in memory as bot stops.

Report for weekly or monthly or quarterly details of execution

Administration of financial service company need report on all the records processed by bot. They need supervision and maintenance of a separate track of all records for internal audit purposes processed within specific time period. A bot running every day or weekly or monthly doesn't matter here. The administration is concerned with report of particular duration.

How to use Orchestrator API

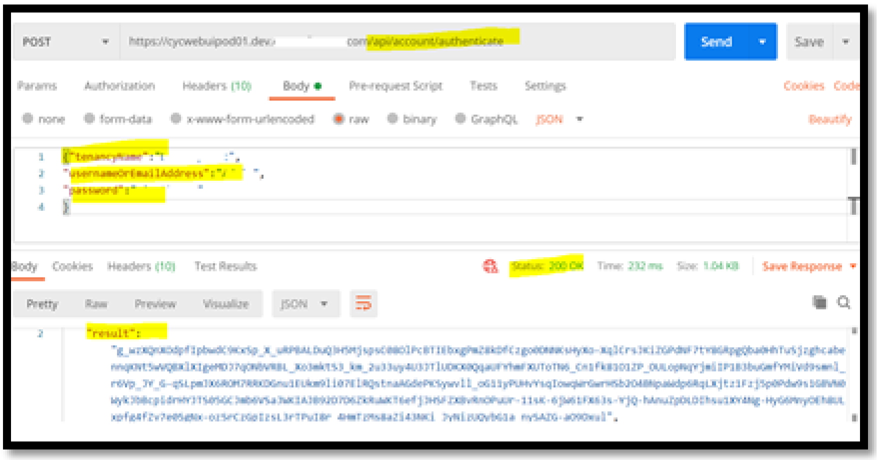
We will check how the API can be used to generate the execution report and can be very helpful for a projects where bots are required to process large number of records or bot is needed to provide execution details for specific time period.

Authentication:

As per any API calls, Orchestrator API would also require an authentication first. This authentication will generate a bearer token. Once bearer token is generated, it can be used with subsequent request to the Orchestrator.

Authenticate API accepts tenancyName, username & password as their input parameters. The token is the output which is nothing but the string value.

UiPath's HttpClient activity can be used to authenticate the API request.

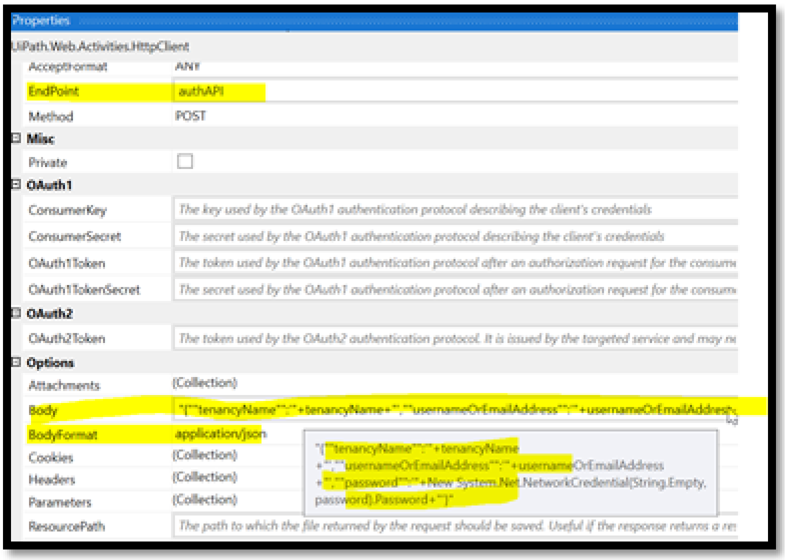


https://cycwebuipod01.dev. com/api/account/authenticate

```
{"tenancyName":"    ",
"usernameOrEmailAddress":"    ",
"password":"    "
}
```

Status: 200 OK Time: 232 ms Size: 1.04 KB

```
"result":
    "[illegible]",
```



UiPath.Web.Activities.HttpClient

AcceptFormat	ANY
EndPoint	authAPI
Method	POST
⊟ **Misc**	
Private	□
⊟ **OAuth1**	
ConsumerKey	*The key used by the OAuth1 authentication protocol describing the client's credentials*
ConsumerSecret	*The secret used by the OAuth1 authentication protocol describing the client's credentials*
OAuth1Token	*The token used by the OAuth1 authentication protocol after an authorization request for the consum*
OAuth1TokenSecret	*The secret used by the OAuth1 authentication protocol after an authorization request for the consum*
⊟ **OAuth2**	
OAuth2Token	*The token used by the OAuth2 authentication protocol. It is issued by the targeted service and may n*
⊟ **Options**	
Attachments	(Collection)
Body	"{""tenancyName"":""+tenancyName+"",""usernameOrEmailAddress"":""+usernameOrEmailAddress+
BodyFormat	application/json
Cookies	(Collection)
Headers	(Collection)
Parameters	(Collection)
ResourcePath	*The path to which the file returned by the request should be saved. Useful if the response returns a re*

```
"{""tenancyName"":""+tenancyName
+"",""usernameOrEmailAddress"":""+usernameOrEmailAddress
+"",""password"":""+New System.Net.NetworkCredential(String.Empty,
password).Password+""}"
```

Subsequent Request:

Now we have the token generated. We can make the next API request which can bring the details of records processed from queue for specific date time.

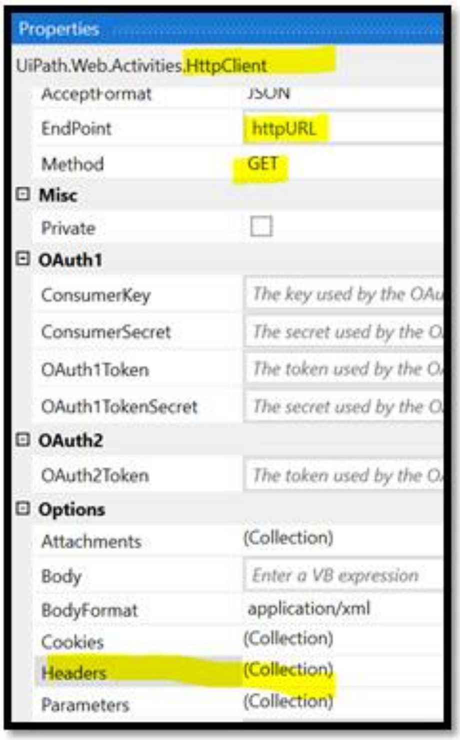
Here is the API call that fetches all the queue item information processed by all the bots having status success and failed. The *$expand* and the *$filter* are the keywords of the API calls. This API call accepts date time of the specific day, queue id and the status of the records that are supposed to be fetched.

https://cycwebuipop01.xxxx.com/odata/QueueItems?$expand=Robot&$filter=(EndProcessing gt 2020-09-23T10:33:09.00Z and QueueDefinitionId eq 7 and (Status eq 'Failed' or Status eq 'Successful'))

Point to remember is that the date time use for filtering the records should be in Universal Time Format (UTC). Because Orchestrator UiPath updates the date details for records in Universal Time Format only.

There are certain header parameters that should be included when making these API calls. Token generated from authenticate request is also supposed to be part of the header parameters.

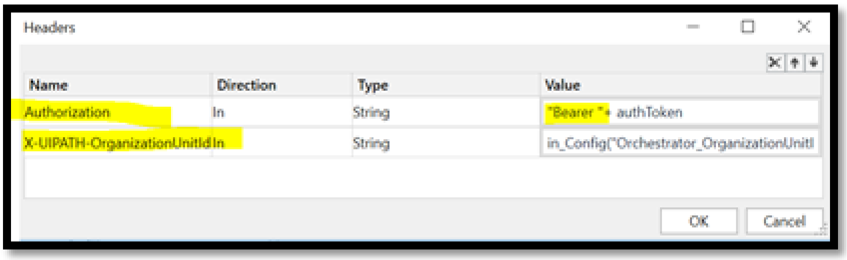
UiPath's HttpClient activity can be used to make the subsequent API request.



Properties

UiPath.Web.Activities.HttpClient

AcceptFormat	JSON
EndPoint	httpURL
Method	GET
⊟ **Misc**	
Private	□
⊟ **OAuth1**	
ConsumerKey	*The key used by the OAu*
ConsumerSecret	*The secret used by the O*
OAuth1Token	*The token used by the O*
OAuth1TokenSecret	*The secret used by the O*
⊟ **OAuth2**	
OAuth2Token	*The token used by the O*
⊟ **Options**	
Attachments	(Collection)
Body	*Enter a VB expression*
BodyFormat	application/xml
Cookies	(Collection)
Headers	(Collection)
Parameters	(Collection)



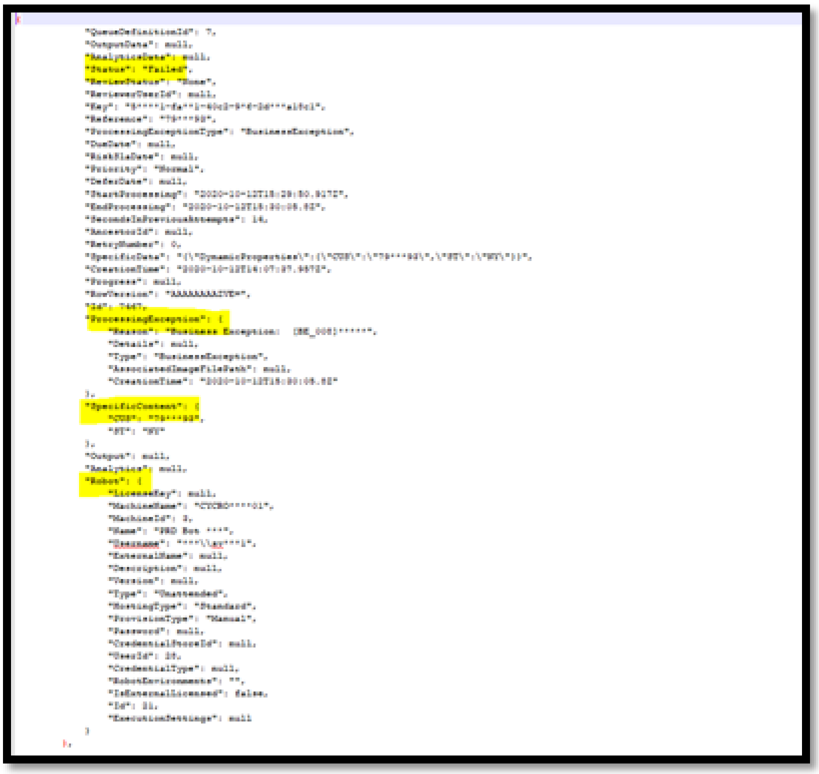
Headers

Name	Direction	Type	Value
Authorization	In	String	"Bearer "+ authToken
X-UIPATH-OrganizationUnitId	In	String	in_Config("Orchestrator_OrganizationUnitI

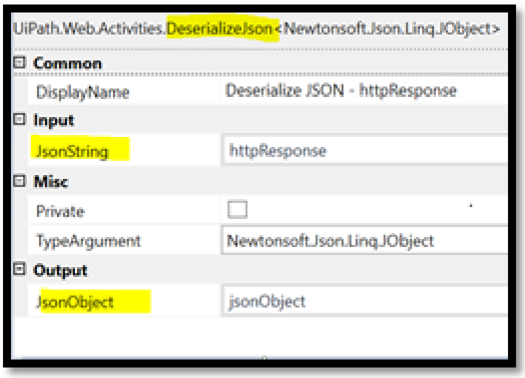
Authorization and *X-UIPATH-OrganizationUnitId* are the 2 required parameters. The token generated from authenticate request should be added in authorization parameter followed by word "Bearer" and *X-UIPATH-OrganizationUnitId* is nothing but the Orchestrator folder id in which the queue is created. Most of the time it is Default folder only.

The output of the API call is in json string format which can be converted into jsonObject using "DeserializeJson" activity of UiPath

Sample Json string:



```
        "QueueDefinitionId": 7,
        "OutputData": null,
        "AnalyticsData": null,
        "Status": "Failed",
        "ReviewStatus": "None",
        "ReviewerUserId": null,
        "Key": "5****1-fa**1-40c2-9*6-2d***a1fc1",
        "Reference": "79***93",
        "ProcessingExceptionType": "BusinessException",
        "DueDate": null,
        "RiskSlaDate": null,
        "Priority": "Normal",
        "DeferDate": null,
        "StartProcessing": "2020-10-12T15:29:50.917Z",
        "EndProcessing": "2020-10-12T15:30:05.82",
        "SecondsInPreviousAttempts": 14,
        "AncestorId": null,
        "RetryNumber": 0,
        "SpecificData": "{\"DynamicProperties\":{\"CUS\":\"79***93\",\"ST\":\"NY\"}}",
        "CreationTime": "2020-10-12T14:07:37.497Z",
        "Progress": null,
        "RowVersion": "AAAAAAAAIVE=",
        "Id": 7447,
        "ProcessingException": {
            "Reason": "Business Exception:  [BE_008]*****",
            "Details": null,
            "Type": "BusinessException",
            "AssociatedImageFilePath": null,
            "CreationTime": "2020-10-12T15:30:05.82"
        },
        "SpecificContent": {
            "CUS": "79***93",
            "ST": "NY"
        },
        "Output": null,
        "Analytics": null,
        "Robot": {
            "LicenseKey": null,
            "MachineName": "CTCRO****01",
            "MachineId": 3,
            "Name": "PRD Bot ***",
            "Username": "****\\ar***1",
            "ExternalName": null,
            "Description": null,
            "Version": null,
            "Type": "Unattended",
            "HostingType": "Standard",
            "ProvisionType": "Manual",
            "Password": null,
            "CredentialStoreId": null,
            "UserId": 19,
            "CredentialType": null,
            "RobotEnvironments": "",
            "IsExternalLicensed": false,
            "Id": 11,
            "ExecutionSettings": null
        }
    },
```



UiPath.Web.Activities.DeserializeJson<Newtonsoft.Json.Linq.JObject>

⊟ **Common**

DisplayName: Deserialize JSON - httpResponse

⊟ **Input**

JsonString: httpResponse

⊟ **Misc**

Private: □

TypeArgument: Newtonsoft.Json.Linq.JObject

⊟ **Output**

JsonObject: jsonObject

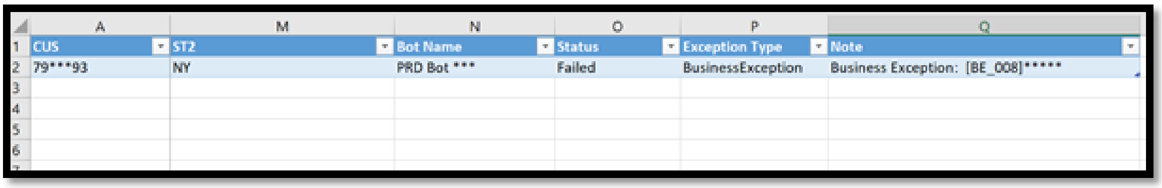
Status property will tell us whether the records was processed successfully or not.

Processing Exception property will give the exception details.

Specific Content property will give all the input data from queue item

Robot property will tell us the robot that process the record.

The Json Object then can be read to create a data table which can be then exported in spreadsheet as report for the business.



CUS	ST2	Bot Name	Status	Exception Type	Note
79***93	NY	PRD Bot ***	Failed	BusinessException	Business Exception: [BE_008]*****

Wrap-up

Hopefully this article gave you an idea about using Orchestrator API to fetch the queue item details from orchestrator.

Reference: https://docs.uipath.com/orchestrator/reference/jobs-requests

About Sogeti

Sogeti is a leading provider of technology and engineering services. Sogeti delivers solutions that enable digital transformation and offers cutting-edge expertise in Cloud, Cybersecurity, Digital Manufacturing, Digital Assurance & Testing, and emerging technologies. Sogeti combines agility and speed of implementation with strong technology supplier partnerships, world class methodologies and its global delivery model, Rightshore®. Sogeti brings together more than 25,000 professionals in 15 countries, based in over 100 locations in Europe, USA and India. Sogeti is a wholly-owned subsidiary of Capgemini SE, listed on the Paris Stock Exchange.

Learn more about us at

www.sogeti.com