vRealize Orchestrator Dynamic Types Plug-in Generator v3

1. Goal

This tool is designed to help create third party integrations in vRealize Automation with limiting as much as possible the requirement of writing the code for it.

Via custom resources vRealize Automation can deploy, operate, decommission anything as long as it can be inventoried, created, operated, deleted via a vRealize Orchestrator plug-in.

It can be created within the vRealize Orchestrator designer integrating any third party API providing unique objects. The most common use case is to integrate via REST API and this is what this generator focus on.

2. Use cases

1. Provide in the service catalog the deployment and operations of custom resources that are dependent on a third party API (on premise or cloud based).
2. Include some of these custom resources in cloud templates for operating them as part of the cloud template deployment
3. Use inventory object as workflow inputs to select object in drop downs, tree views

3. What this tool resolves ?

Implementing such extensibility requires the understanding of the third party APIs (most often REST based) as well as experience building plug-ins with vRO.

Building plug-ins for vRO can be done two ways:

- Using the plug-in SDK. This is mainly how VMware technology partners build vRO plug-ins. This requires a Java developer and follow a typical product development cycle.
- Using Dynamic Types. This is mainly how customers / partners / VMware field build plug-ins. This can be done entirely in vRO in JavaScript.

This generator:

- Removes a lot of the complexity involved in completing a plug-in from scratch
- Reduce drastically the time required by generating most of the code
- Determine object hierarchy based on REST object paths

4. How Dynamic Types plug-in Generator is working

At a high level this tool is:

- Gathering information pertinent to the integration from the API documentation
    1. If a Swagger / OpenAPI URL is provided : Automated via a workflow using the swagger as input. This allows to build a basic plug-in without having access to the third party REST host.
    2. if not : with a wizard workflow that ask the user the strict minimum required information to create a plug-in and its objects and gathering the other information interacting with the API. This way requires access to a live third party REST host
- Automate the creation of the plug-in element (Plug-in / Objects / Objects relationships)
- Generate code (actions) integrating the third party API for providing the functionality of the plug-in (Getting objects, CRUD operations).
- Future : Generate workflows for CRUD operations (Create , Delete, Operations workflows that are needed to create vRA custom resources).

Upon workflow completion the end user should get:

- A new Dynamic Types plug-in
- Inventory objects for this plug-in
- A set of CRUD operations actions for each object. These actions can be added to the Create, update, delete workflows used for vRA custom resources.

The completion level this tool reach depends on how standard the API is and how well it is documented.

For example

- It is working well without having to write Javascript code if the API :
    - Uses standard authentication mechanism supported by the vRO REST plug-in
    - Observes the object hierarchy in the URL path
    - Returns an object that is defined by a specific schema and include standard id and name properties

Some manual code updates are necessary if the API:

- Uses specific authentication (I.E call to auth URL to get a token and using this as header)
- Does not observe object hierarchy in the URL path
- If the returned objects are generic (I.E map)
- If the returned object has different nested children objects
- If the URL requires extra parameters after the main path

More consequent work is required such as determining object hierarchy, major update on find object methods if the API:

- Has complex authentication mechanism (I.E AWS)
- Does not use HTTP GET to get objects
- Requires URL parameters that cannot be listed by the API

5. Pre-requisite

- A REST API returning JSON or XML objects
- A supported authentication in the vRO REST plug-in (no coding involved) / or a documented way to use header based authentication (creating an authentication action is required)
- GET method to get objects. ideally :
    - /api/parents being the url to get all parents objects
    - /api/parents/{parentId} being the url to get a single parents object
    - /api/parents/{parentId}/children being the url to get all children objects of the parent that has parentId Id
    - /api/parents returning an array of objects including properties with at least an id and name properties
- A documentation for the REST API or better a swagger / open API

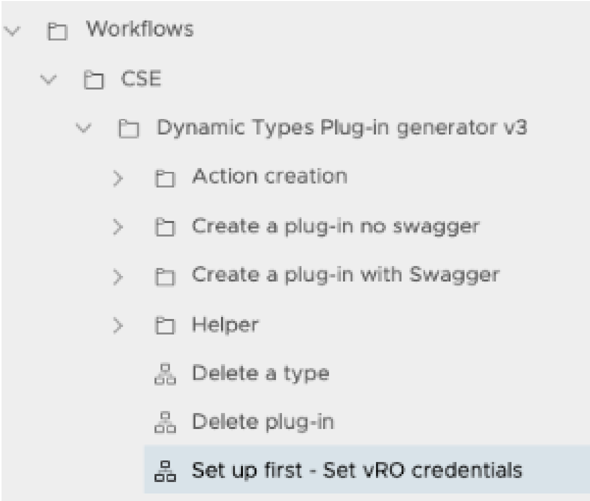
The more the REST API will comply with these rules, the less update work there will be on the plug-in

You need to collect the following information:

- What is the API URL endpoint
- What is the API authentication type (Basic, OAuth1, OAuth2, ...)
- If the authentication is not one of the defined type in vRO "Add a REST Host" the process of making the custom authentication, for example :
    - What URL to call to get a token
    - What to pass to this URL (typically a body with username and password)
    - What part of the response need to be retrieved to be used as an authentication header
- What are the objects you need to provide in the plug-in

6. Installation & first setup

- Import the Dynamic Types Plug-in Generator package
- Run "Set up first - Set vRO credentials"
    - vRO hostname is the same as in your browser URL (without https:// before and /orchestration-ui after)
    - Username is the username you used to log in vRO


Workflows
CSE
Dynamic Types Plug-in generator v3
Action creation
Create a plug-in no swagger
Create a plug-in with Swagger
Helper
Delete a type
Delete plug-in
Set up first - Set vRO credentials

This workflow is saving vRO its credentials information for being able to create via vRO REST API the elements used by the plug-in.

7. Create a plug-in from Swagger / Open API

If the REST API provides a Swagger / Open API then the creation of the plug-in can be automated.

7.1. REST host authentication test

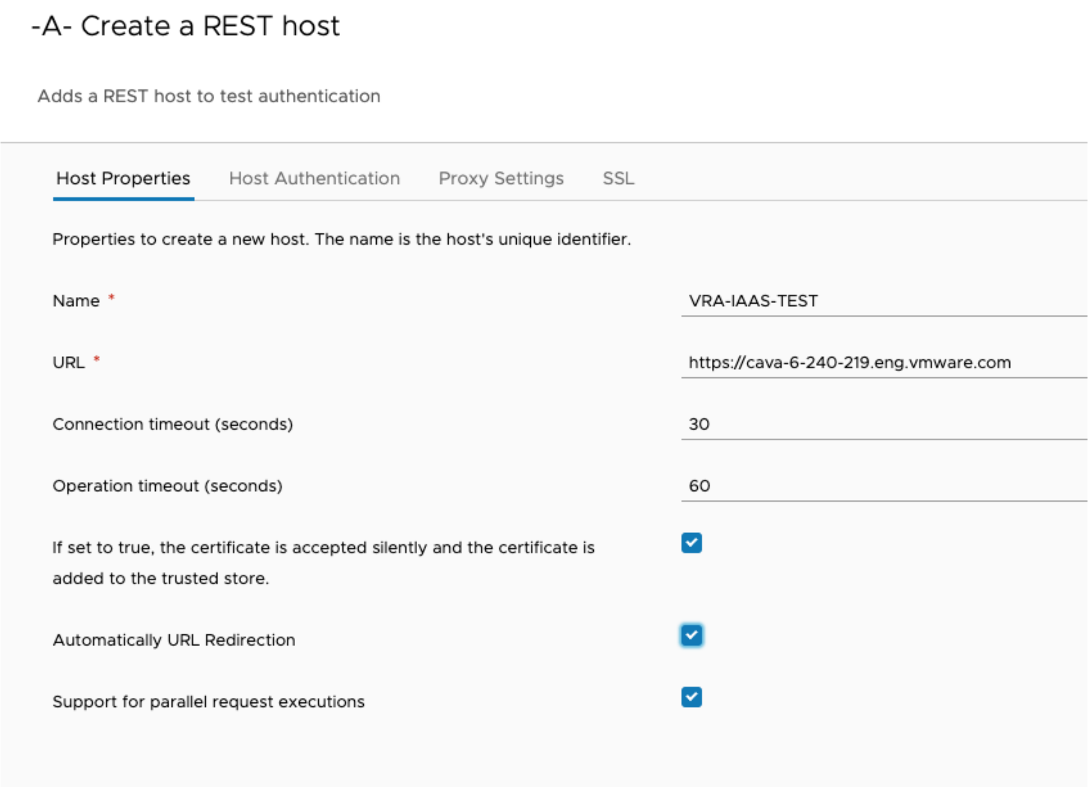
Before creating the plug-in we need to insure requests can be run on the REST host.

Workflows for testing the REST Host authentication are under Create a plug-in with Swagger / ~~1~~ Test REST Host Authentication:

- -A- Create a REST Host - Copy of library "Add a REST host". Alternatively run the library workflow
- -B- Create a REST operation - Copy of library "Add a REST operation"
- -C- Invoke a REST operation with headers from action - Custom workflow that can optionally use a custom build header authentication action
- -D- Remove a REST Host - Copy of library "Remove a REST host"

Run "-A- Create a REST Host" workflow

- Include its URL with https
- Set the certificate to be accepted silently



-A- Create a REST host

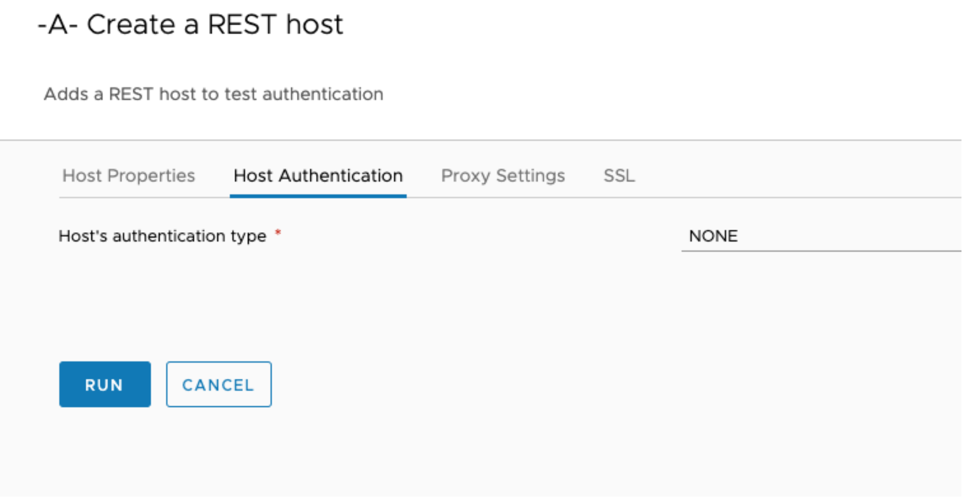
Adds a REST host to test authentication

Properties to create a new host. The name is the host's unique identifier.

Name *	VRA-IAAS-TEST
URL *	https://cava-6-240-219.eng.vmware.com
Connection timeout (seconds)	30
Operation timeout (seconds)	60
If set to true, the certificate is accepted silently and the certificate is added to the trusted store.	☑
Automatically URL Redirection	☑
Support for parallel request executions	☑

On the second tab set the Authentication type. If standard set the Authentication type from the drop down. If header based authentication (I.E vRA 8, vRA Cloud) set it to NONE)

Run the workflow.



-A- Create a REST host

Adds a REST host to test authentication

Host's authentication type * NONE

7.1.1. Custom authentication header authentication

If you selected "NONE" for Authentication type and need to authenticate using an authentication header you need to create an action. If you selected one of the other authentications you can skip this part.

A few actions are provided as examples in the com.vmware.coe.dynamicTypes.pluginGeneratorV3.getCustomHeadersActions scripting module.

This action can have the following inputs in this order:

- restHost of type REST:RESTHost
- method of type string
- urlTemplate of type string
- content of type string

and must return a Properties type object containing the Authorization header

It is not mandatory to use all of them in the action but they are provided in case they could be used.

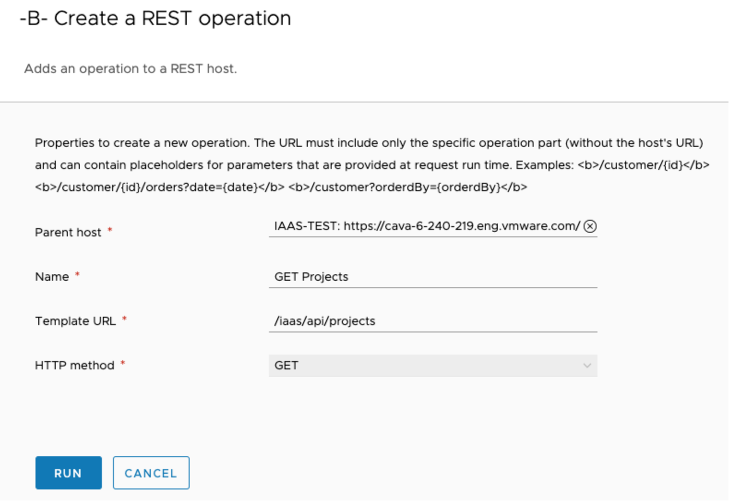
A typical implementation is

- POST a body to an authentication URL (either including username password or authentication token)
- extract the token from the response
- Set and return an header property from

Samples on how to retrieve configuration elements content is in the getCloudServicesCustomHeaders and getvRA82CustomHeaders actions.

Start "-B- Create a REST operation"

Use a simple URL from the REST documentation to GET an object type



-B- Create a REST operation

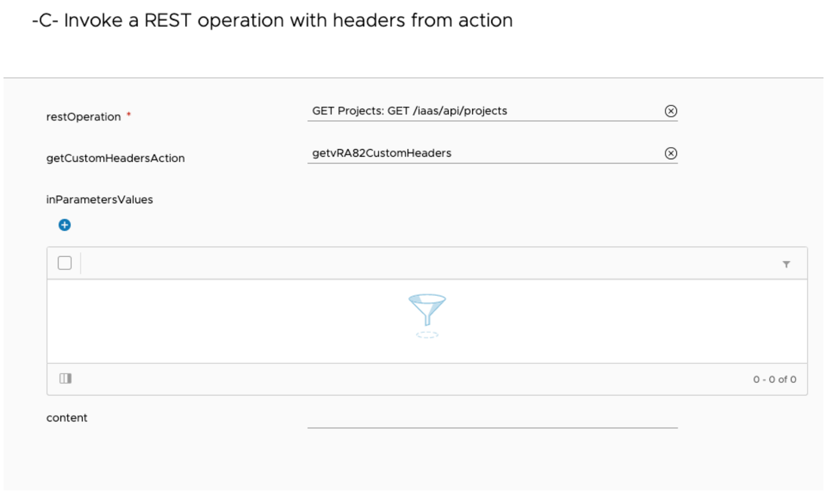
Adds an operation to a REST host.

Properties to create a new operation. The URL must include only the specific operation part (without the host's URL) and can contain placeholders for parameters that are provided at request run time. Examples: <b>/customer/{id}</b> <b>/customer/{id}/orders?date={date}</b> <b>/customer?orderdBy={orderdBy}</b>

Parent host *	IAAS-TEST: https://cava-6-240-219.eng.vmware.com/
Name *	GET Projects
Template URL *	/iaas/api/projects
HTTP method *	GET

Start "-C- Invoke a REST operation with headers from action"

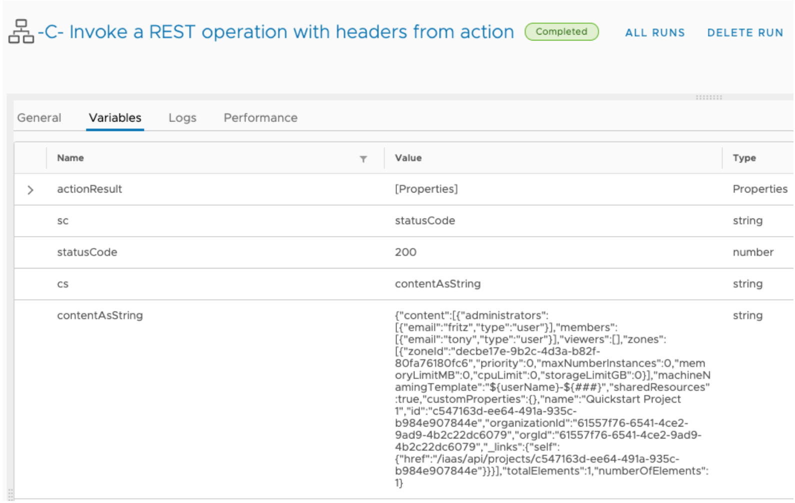
- If you selected "NONE" for Authentication type and need to authenticate using an authentication header you need to select your getCustomHeadersAction action as in the example below.
- If you selected one of the other authentications you can keep the getCustomHeadersAction field empty.



-C- Invoke a REST operation with headers from action

restOperation *	GET Projects: GET /iaas/api/projects
getCustomHeadersAction	getvRA82CustomHeaders
inParametersValues	
content	

Check you get a statusCode of 200 and the expected content in contentAsString



-C- Invoke a REST operation with headers from action

Name	Value	Type
actionResult	[Properties]	Properties
sc	statusCode	string
statusCode	200	number
cs	contentAsString	string
contentAsString	{"content":[{"administrators":[{"email":"fritz","type":"user"}],"members":[{"email":"tony","type":"user"}],"viewers":[],"zones":[{"zoneId":"decbe17e-9b2c-4d3a-b82f-80fa76180fc6","priority":0,"maxNumberInstances":0,"memoryLimitMB":0,"cpuLimit":0,"storageLimitGB":0}],"machineNamingTemplate":"${userName}-${###}","sharedResources":true,"customProperties":{},"name":"Quickstart Project 1","id":"c547163d-ee64-491a-935c-b984e907844e","organizationId":"61557f76-6541-4ce2-9ad9-4b2c22dc6079","orgId":"61557f76-6541-4ce2-9ad9-4b2c22dc6079","_links":{"self":{"href":"/iaas/api/projects/c547163d-ee64-491a-935c-b984e907844e"}}}],"totalElements":1,"numberOfElements":1}	string

In case the custom header action fails update it and restart "-C- Invoke a REST operation with headers from action" until success.

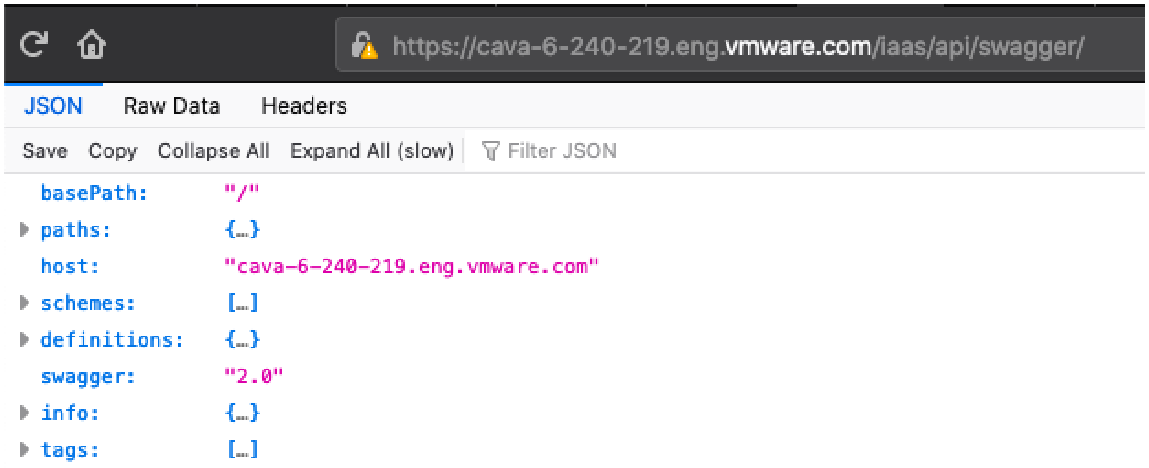
In case you get a different error code check error codes here https://en.wikipedia.org/wiki/List_of_HTTP_status_codes to determine where the issue could be.

This part must be working for the plug-in to work

Once successful you can run "-D- Remove a REST host"

7.2. Creating the plug-in

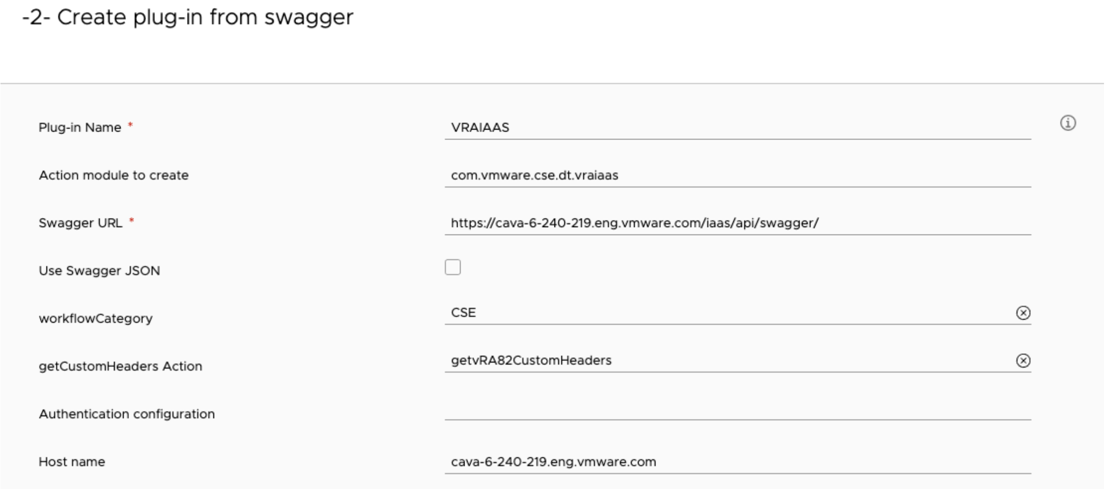
- Enter a plug-in name (Capital letters only)
- You can use the default name for the action module or rename it at your convenience
- Enter the Swagger URL. To check the URL is valid :
  - If you open this URL in Chrome you should get a JSON.
  - if you open this URL in Firefox you should get something like this:



```
  basePath:      "/"
▶ paths:         {…}
  host:          "cava-6-240-219.eng.vmware.com"
▶ schemes:       […]
▶ definitions:   {…}
  swagger:       "2.0"
▶ info:          {…}
▶ tags:          […]
```

- In case you do not have a swagger URL and are provided with the JSON file you can Check :Use Swagger JSON and paste it.
- WorkflowCategory is the location where CRUD operation workflows will be created
- Set your getCustomHeaders Action in case you set the REST Host Authentication type to "NONE" and need to use a Header Authorization
- Authentication configuration is in case your getHeaderAction can get credentials from different configuration elements // Need to check this
- The host name should be automatically set to the one in the Swagger URL. In case you pasted a swagger JSON you need to enter a valid host name for your target REST system
- RUN

Example below for the VRA IaaS API.



-2- Create plug-in from swagger

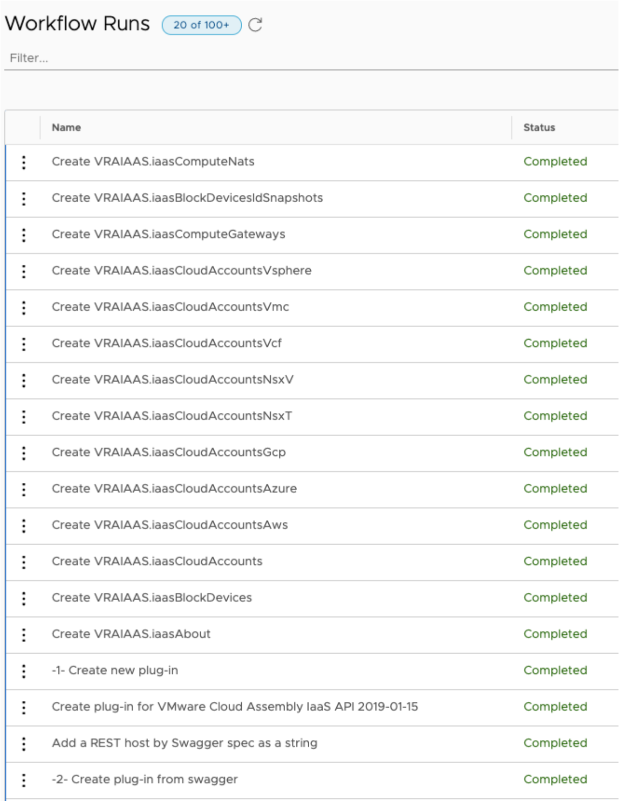
Plug-in Name *	VRAIAAS
Action module to create	com.vmware.cse.dt.vraiaas
Swagger URL *	https://cava-6-240-219.eng.vmware.com/iaas/api/swagger/
Use Swagger JSON	□
workflowCategory	CSE
getCustomHeaders Action	getvRA82CustomHeaders
Authentication configuration	
Host name	cava-6-240-219.eng.vmware.com

The workflow should run a few minutes. It will:

- Run Library workflow "Add a REST Host by Swagger spec as a string"
- Create a plug-in for (API name in swagger)
  - -1- Create new plug-in
  - For each object type:
    - Create plugin-name.objectName
    - Update findById for plugin-name.objectName (if an URL was found to get the object by ID)
  - Create CRUD operations for GET, PUT, POST, PATCH, DELETE operations

For easier troubleshooting each type creation / update is run as a separate workflow.

The "-2- Create plug-in from Swagger" is not supposed to fail but it can provide warning and error messages in its log when it may have issues finding the necessary information to create a type.



Workflow Runs 20 of 100+

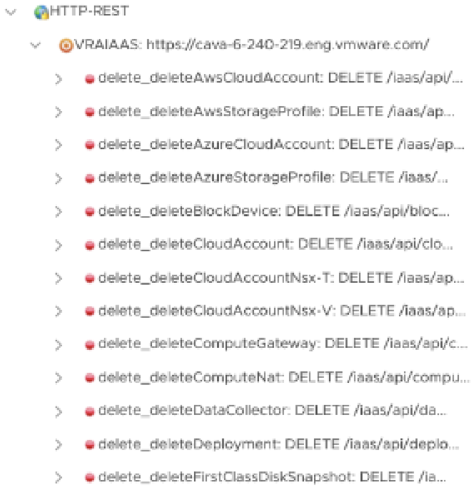
Filter...

Name	Status
Create VRAIAAS.iaasComputeNats	Completed
Create VRAIAAS.iaasBlockDevicesIdSnapshots	Completed
Create VRAIAAS.iaasComputeGateways	Completed
Create VRAIAAS.iaasCloudAccountsVsphere	Completed
Create VRAIAAS.iaasCloudAccountsVmc	Completed
Create VRAIAAS.iaasCloudAccountsVcf	Completed
Create VRAIAAS.iaasCloudAccountsNsxV	Completed
Create VRAIAAS.iaasCloudAccountsNsxT	Completed
Create VRAIAAS.iaasCloudAccountsGcp	Completed
Create VRAIAAS.iaasCloudAccountsAzure	Completed
Create VRAIAAS.iaasCloudAccountsAws	Completed
Create VRAIAAS.iaasCloudAccounts	Completed
Create VRAIAAS.iaasBlockDevices	Completed
Create VRAIAAS.iaasAbout	Completed
-1- Create new plug-in	Completed
Create plug-in for VMware Cloud Assembly IaaS API 2019-01-15	Completed
Add a REST host by Swagger spec as a string	Completed
-2- Create plug-in from swagger	Completed

Once completed there are quite a few changes in the inventory:

"Add a REST Host by Swagger spec as a string" added the REST host with all its operations.

NB that the plug-in will leverage the REST Host so this must be kept in the inventory. However the plug-in will not use the REST operations. These are used only during the generation of the actions for the plug-in.



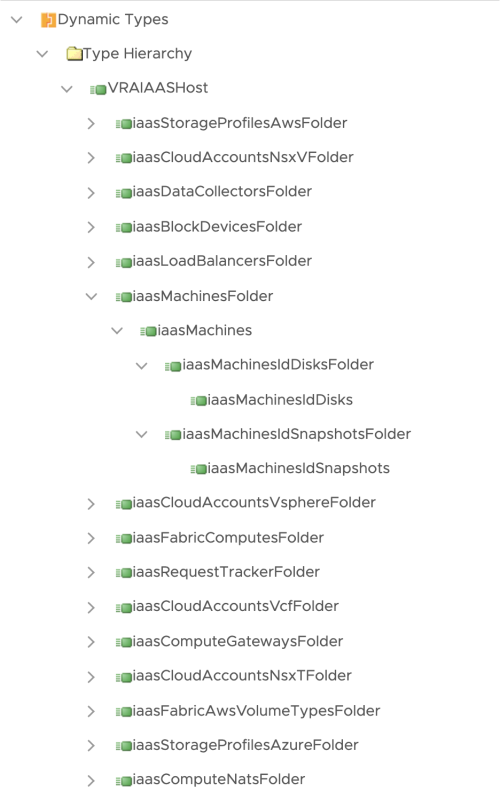
HTTP-REST

VRAIAAS: https://cava-6-240-219.eng.vmware.com/

- delete_deleteAwsCloudAccount: DELETE /iaas/api/...
- delete_deleteAwsStorageProfile: DELETE /iaas/ap...
- delete_deleteAzureCloudAccount: DELETE /iaas/ap...
- delete_deleteAzureStorageProfile: DELETE /iaas/...
- delete_deleteBlockDevice: DELETE /iaas/api/bloc...
- delete_deleteCloudAccount: DELETE /iaas/api/clo...
- delete_deleteCloudAccountNsx-T: DELETE /iaas/ap...
- delete_deleteCloudAccountNsx-V: DELETE /iaas/ap...
- delete_deleteComputeGateway: DELETE /iaas/api/c...
- delete_deleteComputeNat: DELETE /iaas/api/compu...
- delete_deleteDataCollector: DELETE /iaas/api/da...
- delete_deleteDeployment: DELETE /iaas/api/deplo...
- delete_deleteFirstClassDiskSnapshot: DELETE /ia...

Under Dynamic Types / Type Hierarchy there is

- A host type for the plug-in (I.E VRAIAASHost)
    - The folders types for the objects that are under this host (I.E iaasMachinesFolder)
        - The object types under the folders (I.E iaasMachines)
            - And if these objects have children objects folders and objects (I.E iaasMachinesIdDisksFolder & iaasMachinesIdDisks)


Dynamic Types
Type Hierarchy
VRAIAASHost
iaasStorageProfilesAwsFolder
iaasCloudAccountsNsxVFolder
iaasDataCollectorsFolder
iaasBlockDevicesFolder
iaasLoadBalancersFolder
iaasMachinesFolder
iaasMachines
iaasMachinesIdDisksFolder
iaasMachinesIdDisks
iaasMachinesIdSnapshotsFolder
iaasMachinesIdSnapshots
iaasCloudAccountsVsphereFolder
iaasFabricComputesFolder
iaasRequestTrackerFolder
iaasCloudAccountsVcfFolder
iaasComputeGatewaysFolder
iaasCloudAccountsNsxTFolder
iaasFabricAwsVolumeTypesFolder
iaasStorageProfilesAzureFolder
iaasComputeNatsFolder

7.3. Updating the REST Host

The REST host was created by the library workflow "Add a REST Host by Swagger spec as a string" with an authentication set to "NONE".

If you did set your REST host to an authentication different then "NONE" during the REST Host Authentication test run "-3- Update a REST host" and update the Authentication type and credentials.

If you set it to "NONE" you can skip this step.

8. Checking and updating the generated plug-in

With the plug-in generated it is necessary to check the inventory objects. The generator makes some assumption to find the objects returned by the API in different structures as well as to find their properties including the mandatory id and names properties.

The goal is to update anything that is not

8.1. Verifying the generated objects

Open the namespace in the inventory.

- Unfold the host.
- Unfold each folder under the host.
- Unfortunately since recent version of vRO there is no more check on the objects to know if they have children (the hasChildrenInRelation binding in the DT object definition). You can check if the object is expected to have children in the Type hierarchy.
- Click on each object type to check its properties

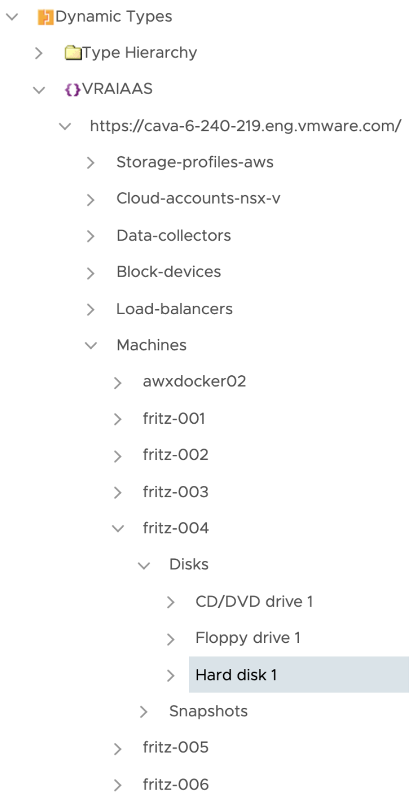
Note that as of this version of vRO (8.2 Patch 1) there is a known issue where right after being generated the inventory may not show the folder and default object icons.

If unfolding is either displaying children objects or no object because there are not expected to have any all is well.

If any error occurs in the code that has been generated it should display a visible error message. The next section will go through possible issues.

If you want to make sure there is no error you can tail the vRO logs and check for exception.

Below is an example of a plug-in generated with the vRA IaaS API. This API works well.



- Dynamic Types
  - Type Hierarchy
  - VRAIAAS
    - https://cava-6-240-219.eng.vmware.com/
      - Storage-profiles-aws
      - Cloud-accounts-nsx-v
      - Data-collectors
      - Block-devices
      - Load-balancers
      - Machines
        - awxdocker02
        - fritz-001
        - fritz-002
        - fritz-003
        - fritz-004
          - Disks
            - CD/DVD drive 1
            - Floppy drive 1
            - Hard disk 1
          - Snapshots
        - fritz-005
        - fritz-006

Bellow is an example of object with the following properties:

The mandatory / minimal ones:

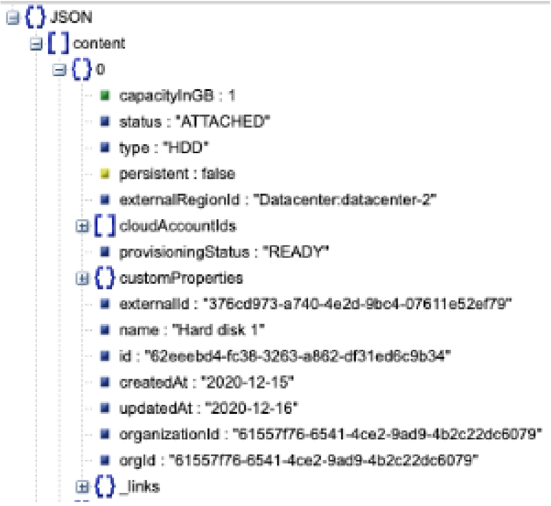
- id : This is a mandatory property allowing the plug-in to identify the object. It is a / separated string including the REST plug-in Host ID then the REST ID of the parent object (Here machine), then the REST ID of the child object (here disk ID).
- name : This is also a mandatory property as this is the main one that will help the end user identify the object

The internal ones :

- @fullType : this defines the vRO types we created. Here DynamicTypes:VRAIAAS.iaasMachinesIdDisks

The debug properties. These properties are added to the inventory object to help troubleshoot possible issues

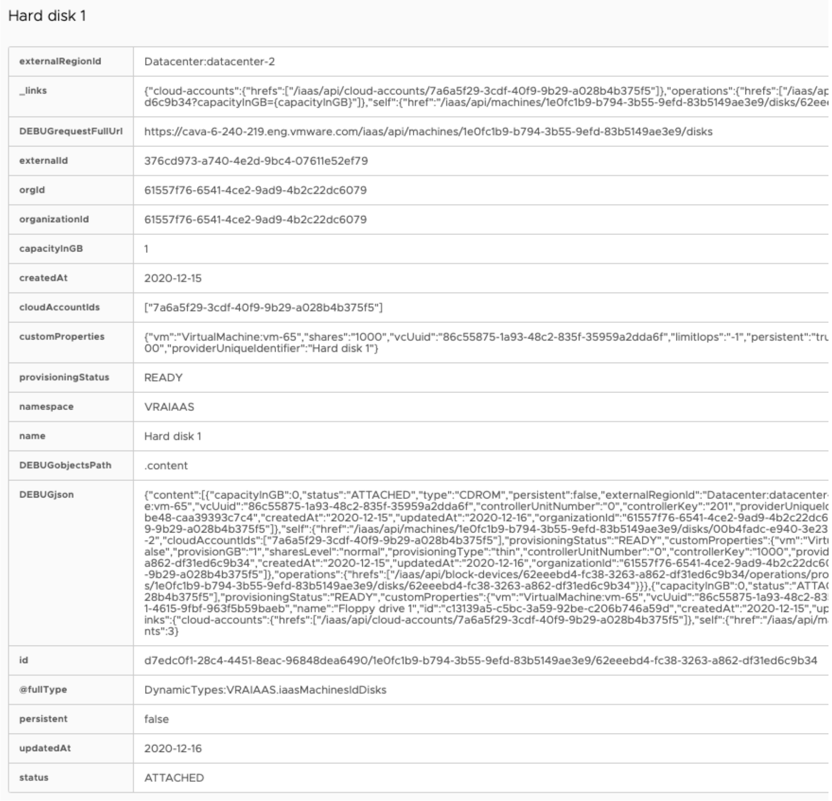
- DEBUGrequestFullUrl : The URL that was used to get the object containing all the children objects (I.E https://cava-6-240-219.eng.vmware.com/iaas/api/machines/1e0fc1b9-b794-3b55-9efd-83b5149ae3e9/disks)
- DEBUGjson : The result of the request made to get the object containing all the children objects



```
{ } JSON
  [ ] content
    { } 0
      capacityInGB : 1
      status : "ATTACHED"
      type : "HDD"
      persistent : false
      externalRegionId : "Datacenter:datacenter-2"
      [ ] cloudAccountIds
      provisioningStatus : "READY"
      { } customProperties
      externalId : "376cd973-a740-4e2d-9bc4-07611e52ef79"
      name : "Hard disk 1"
      id : "62eeebd4-fc38-3263-a862-df31ed6c9b34"
      createdAt : "2020-12-15"
      updatedAt : "2020-12-16"
      organizationId : "61557f76-6541-4ce2-9ad9-4b2c22dc6079"
      orgId : "61557f76-6541-4ce2-9ad9-4b2c22dc6079"
      { } _links
```

- DEBUGobjectsPath : The property in the object returned that return an array of children objects (In this example ".content")

The object properties : All the properties contained in the object (I.E status, capacityInGB, ...)



Hard disk 1

externalRegionId	Datacenter:datacenter-2
_links	{"cloud-accounts":{"hrefs":["/iaas/api/cloud-accounts/7a6a5f29-3cdf-40f9-9b29-a028b4b375f5"]},"operations":{"hrefs":["/iaas/ap d6c9b34?capacityInGB={capacityInGB}"]},"self":{"href":"/iaas/api/machines/1e0fc1b9-b794-3b55-9efd-83b5149ae3e9/disks/62ee
DEBUGrequestFullUrl	https://cava-6-240-219.eng.vmware.com/iaas/api/machines/1e0fc1b9-b794-3b55-9efd-83b5149ae3e9/disks
externalId	376cd973-a740-4e2d-9bc4-07611e52ef79
orgId	61557f76-6541-4ce2-9ad9-4b2c22dc6079
organizationId	61557f76-6541-4ce2-9ad9-4b2c22dc6079
capacityInGB	1
createdAt	2020-12-15
cloudAccountIds	["7a6a5f29-3cdf-40f9-9b29-a028b4b375f5"]
customProperties	{"vm":"VirtualMachine:vm-65","shares":"1000","vcUuid":"86c55875-1a93-48c2-835f-35959a2dda6f","limitIops":"-1","persistent":"tru 00","providerUniqueIdentifier":"Hard disk 1"}
provisioningStatus	READY
namespace	VRAIAAS
name	Hard disk 1
DEBUGobjectsPath	.content
DEBUGjson	{"content":[{"capacityInGB":0,"status":"ATTACHED","type":"CDROM","persistent":false,"externalRegionId":"Datacenter:datacenter- e:vm-65","vcUuid":"86c55875-1a93-48c2-835f-35959a2dda6f","controllerUnitNumber":"0","controllerKey":"201","providerUniquel be48-caa39393c7c4","createdAt":"2020-12-15","updatedAt":"2020-12-16","organizationId":"61557f76-6541-4ce2-9ad9-4b2c22dc6 9-9b29-a028b4b375f5"]},"self":{"href":"/iaas/api/machines/1e0fc1b9-b794-3b55-9efd-83b5149ae3e9/disks/00b4fadc-e940-3e23 -2","cloudAccountIds":["7a6a5f29-3cdf-40f9-9b29-a028b4b375f5"],"provisioningStatus":"READY","customProperties":{"vm":"Virt alse","provisionGB":"1","sharesLevel":"normal","provisioningType":"thin","controllerUnitNumber":"0","controllerKey":"1000","provid a862-df31ed6c9b34","createdAt":"2020-12-15","updatedAt":"2020-12-16","organizationId":"61557f76-6541-4ce2-9ad9-4b2c22dc60 -9b29-a028b4b375f5"]},"operations":{"hrefs":["/iaas/api/block-devices/62eeebd4-fc38-3263-a862-df31ed6c9b34/operations/pro s/1e0fc1b9-b794-3b55-9efd-83b5149ae3e9/disks/62eeebd4-fc38-3263-a862-df31ed6c9b34"}}},{"capacityInGB":0,"status":"ATTA 28b4b375f5"],"provisioningStatus":"READY","customProperties":{"vm":"VirtualMachine:vm-65","vcUuid":"86c55875-1a93-48c2-83 1-4615-9fbf-963f5b59baeb","name":"Floppy drive 1","id":"c13139a5-c5bc-3a59-92be-c206b746a59d","createdAt":"2020-12-15","up inks":{"cloud-accounts":{"hrefs":["/iaas/api/cloud-accounts/7a6a5f29-3cdf-40f9-9b29-a028b4b375f5"]},"self":{"href":"/iaas/api/m nts":3}
id	d7edc0f1-28c4-4451-8eac-96848dea6490/1e0fc1b9-b794-3b55-9efd-83b5149ae3e9/62eeebd4-fc38-3263-a862-df31ed6c9b34
@fullType	DynamicTypes:VRAIAAS.iaasMachinesIdDisks
persistent	false
updatedAt	2020-12-16
status	ATTACHED

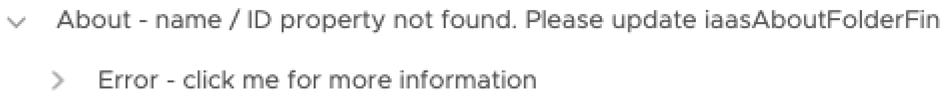
8.2. Fixing issues, updating the generated code

There are cases where the generator does not find a children object type, its id and name properties. There are multiple reasons for that.

It can be that the Swagger is documenting somme generic objects types like maps that do not document the properties. It can also be that the URL is not returning array of objects or that the objects has multiple sub objects.

The plug-in generator tries to generate the object anyway as it is a lot less work to fix these than creating them from scratch. It also let you know when there is obviously a problem and give you information to what action need to be updated.

If a folder name end up with "name / ID property not found. Please update"



About - name / ID property not found. Please update iaasAboutFolderFin

Error - click me for more information

8.2.1. Troubleshooting using the folder object properties

With clicking on the folder you can get a lot of information that was captured from the swagger and the generation of the code finding the children of the parent object.

The properties that determine how the JSON from the HTTP query response will be used. This 3 properties when not determined are causing failures.

- objectPath is the path in the JSON file to the array of objects
- idProperty is the property that will be used for object id
- nameProperty is the property that will be used for the object name

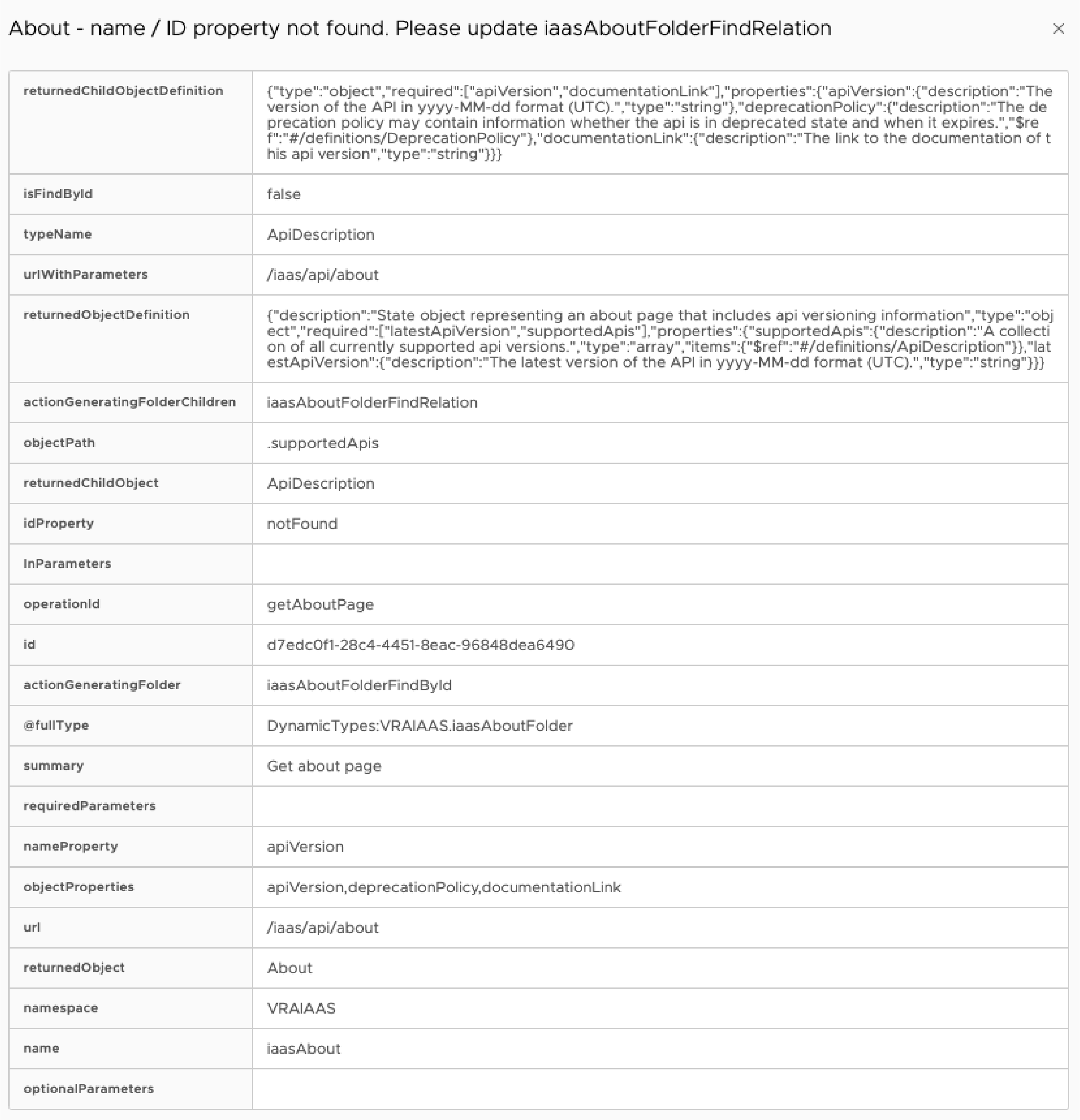
In the example below the objectPath and nameProperty seems to be defined but the idProperty was set to "notFound" as there were no obvious id property.

The returnedObjectDefinition property is the part of the swagger that define what is returned by the URL and is used to set the objectPath. In the example below the .supportedApisPath was chose as this is an array of objects of type ApiDescription.
The returnedChildObjectDefinition property is what swagger documents for the APiDescription object. The generator determined apiVersion may be the name of the object but did not find any property that is named in an obvious way to find the id property. Hence the "notFound" idProperty

Two other important properties are:

- actionGeneratingFolderChildren : This is the action that is responsible to run the GET urlWithParameters (with replacing the parameters if any), getting the objects from the objectPath, creating the object using the idProperty and nameProperty and setting other properties. This is the file you need to edit whenever the objectPath, the ideProperty or nameProperty are not the right ones, when you need to add / remove other properties.
- actionGeneratingFolder : this is the action that generate the parent folder with its name and its properties. You can hardcode the name of this folder if the one generated is not ideal, you can remove a lot of the properties that are useful at development time but may not be for the end users.

Required / optional parameters. Sometimes the swagger API mark some parameters as required. This is important information particularly if these are after the "?". The generator does not set these and they will need to be set within the actionGeneratingFolderChildren action.



About - name / ID property not found. Please update iaasAboutFolderFindRelation

returnedChildObjectDefinition	{"type":"object","required":["apiVersion","documentationLink"],"properties":{"apiVersion":{"description":"The version of the API in yyyy-MM-dd format (UTC).","type":"string"},"deprecationPolicy":{"description":"The deprecation policy may contain information whether the api is in deprecated state and when it expires.","$ref":"#/definitions/DeprecationPolicy"},"documentationLink":{"description":"The link to the documentation of this api version","type":"string"}}}
isFindById	false
typeName	ApiDescription
urlWithParameters	/iaas/api/about
returnedObjectDefinition	{"description":"State object representing an about page that includes api versioning information","type":"object","required":["latestApiVersion","supportedApis"],"properties":{"supportedApis":{"description":"A collection of all currently supported api versions.","type":"array","items":{"$ref":"#/definitions/ApiDescription"}},"latestApiVersion":{"description":"The latest version of the API in yyyy-MM-dd format (UTC).","type":"string"}}}
actionGeneratingFolderChildren	iaasAboutFolderFindRelation
objectPath	.supportedApis
returnedChildObject	ApiDescription
idProperty	notFound
InParameters	
operationId	getAboutPage
id	d7edc0f1-28c4-4451-8eac-96848dea6490
actionGeneratingFolder	iaasAboutFolderFindById
@fullType	DynamicTypes:VRAIAAS.iaasAboutFolder
summary	Get about page
requiredParameters	
nameProperty	apiVersion
objectProperties	apiVersion,deprecationPolicy,documentationLink
url	/iaas/api/about
returnedObject	About
namespace	VRAIAAS
name	iaasAbout
optionalParameters	

If you unfold the folder, if the request managed to go far enough there will be a "Error - click me for more information" object

This object is generated to provide further information on what was returned by the query and may suggest fixes.

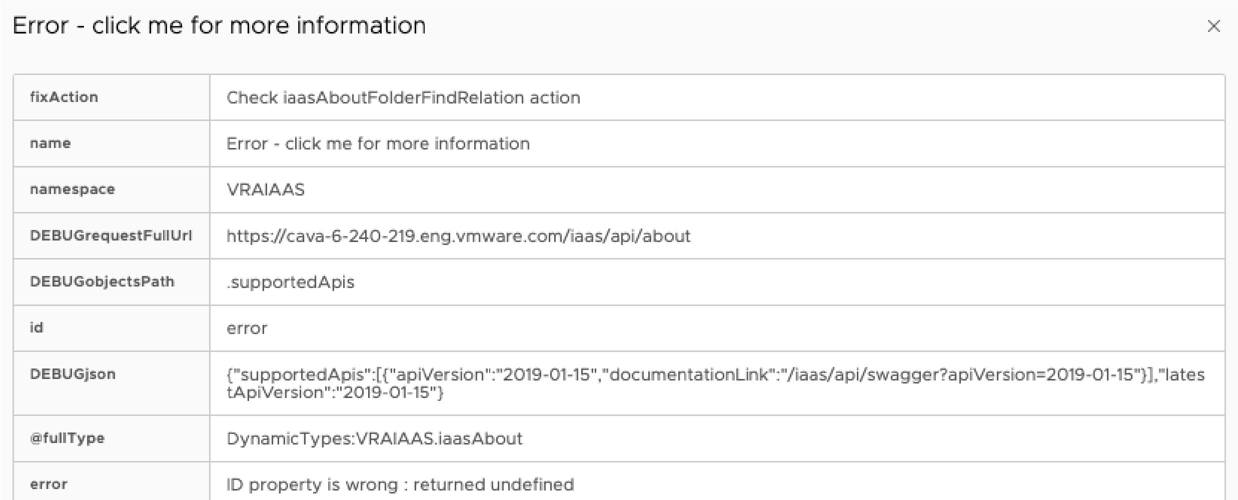
8.2.2. Troubleshooting using the "Error - click me for more information" object

If you click on the "Error - click me for more information" object it should display a number of properties helping to troubleshoot:

- The error property attempts to provide more information. Here "ID property is wrong : returned undefined"
- The fix action directs you to the action you need to edit for the fix. In the case ite objectPath property was no good it may suggest other possible properties returning array of objects.
- The DEBUGrequestFullUrl indicates the URL that was used (with parameters replaced with parent object IDs in case of children objects). This way the url can be tested separately to see if it leads to a problem. It may happen that the credentials do not allow to get from a particular URL. Another possible issue are some required parameters that are not included in the query. In this case the url used to make the query can be edited in the action to provide these.

- Another important property is the DEBUGjson which gives the result of the query. In the case below we can see that the object returned has few information. We may choose to delete completely this object if it is not useful or to provide some of this information as leaf object.

For example here he can still use the supportedApis objectPath, keep "apiVersion" for the name property and also use it for the ID property as this will always be a unique string. We could as well used the documentationLink one but shorter strings are preferred as the value of this property will be used as the last part of the whole object ID that include the RESTHost ID and the parent object IDs.



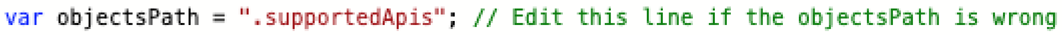
Error - click me for more information

fixAction	Check iaasAboutFolderFindRelation action
name	Error - click me for more information
namespace	VRAIAAS
DEBUGrequestFullUrl	https://cava-6-240-219.eng.vmware.com/iaas/api/about
DEBUGobjectsPath	.supportedApis
id	error
DEBUGjson	{"supportedApis":[{"apiVersion":"2019-01-15","documentationLink":"/iaas/api/swagger?apiVersion=2019-01-15"}],"latestApiVersion":"2019-01-15"}
@fullType	DynamicTypes:VRAIAAS.iaasAbout
error	ID property is wrong : returned undefined

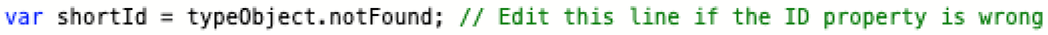
8.2.3. Updating / fixing the children objects with editing the [objectType]FolderFindRelation action

The [objectType]FolderFindRelation action is a crucial part of the code written for the plug-in. While most of its code can in most case stay the same the objectsPatch, the id and name property are likely to be different.

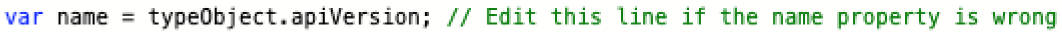
The parts that are the most likely to be edited are commented on the same line. This is the case for these properties which can be edited on these 3 lines:



```
var objectsPath = ".supportedApis"; // Edit this line if the objectsPath is wrong
```



```
var shortId = typeObject.notFound; // Edit this line if the ID property is wrong
```



```
var name = typeObject.apiVersion; // Edit this line if the name property is wrong
```

In the case above the objectsPath looks to be OK when we check on the DEBUGjson property of the Error object.

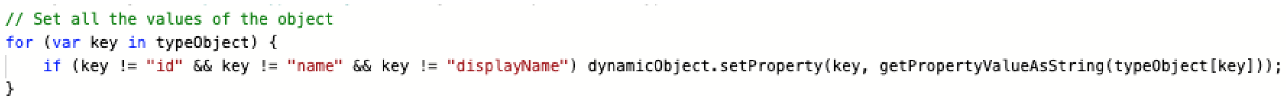
The shortId set as typeObject.notFound is definitely wrong and can be replaced by typeObject.apiVersion.

The name set as typeObject.apiVersion is fine.

Other lines of code that are in the generated action also are commented to suggest they could be removed if wanted as these:

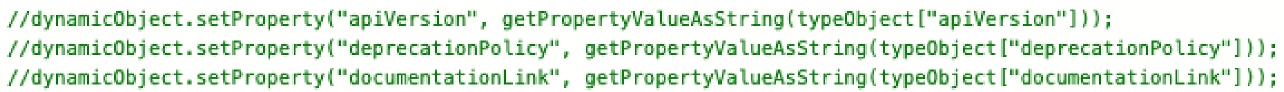
- Provide troubleshooting information (I.E the addDebugProperties() function)
- Avoid the action to throw exceptions (I.E the many try catch to allow troubleshooting from the UI instead of the vRO logs)
- Handle specific use cases (I.E out of order parameters between parent and child URLs, children objects as string links instead of JSON, children object being a single object)

Also by default the action is setting object properties dynamically using these lines.



```
// Set all the values of the object
for (var key in typeObject) {
    if (key != "id" && key != "name" && key != "displayName") dynamicObject.setProperty(key, getPropertyValueAsString(typeObject[key]));
}
```

But do provide with an example of setting the properties individually so you could remove the above lines and choose which one of the properties you want to have on the object, rename these properties as needed etc

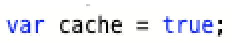


```
//dynamicObject.setProperty("apiVersion", getPropertyValueAsString(typeObject["apiVersion"]));
//dynamicObject.setProperty("deprecationPolicy", getPropertyValueAsString(typeObject["deprecationPolicy"]));
//dynamicObject.setProperty("documentationLink", getPropertyValueAsString(typeObject["documentationLink"]));
```

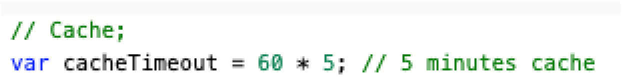
These properties are the one that were found in the swagger file. In some case the generator does not pick up the right ones so you may have to edit these based on the DEBUGjson content.

Another thing to look at is the cache settings. Cache was built in as there are situations were the plug-in will call to this action many times to get the very same results.

Cache can be controlled per object type editing the next two lines:



```
var cache = true;
```



```
// Cache;
var cacheTimeout = 60 * 5; // 5 minutes cache
```

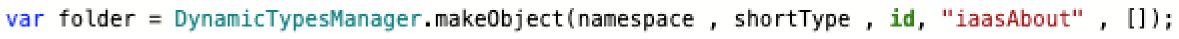
There may be cases where other changes need to be applied such as executing additional REST queries to get some properties needed in the object. The action can be customize as much as needed.

8.2.4. Updating / fixing the children objects with editing the [objectType]FolderFindId action

This action defines the folder for the object. In some cases you may want to rename the folder, add / remove folder properties.

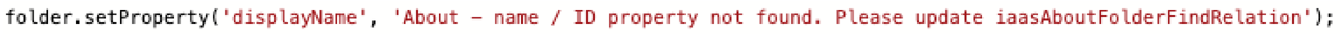
For example when the plug-in generated the folder it added "name / ID property not found. Please update iaasFolderFindRelation"

The name of the folder can be set via the name property in the makeObject call. In the case above it is the name of the object.



```
var folder = DynamicTypesManager.makeObject(namespace , shortType , id, "iaasAbout" , []);
```

But it can be overwritten bu the displayName property.

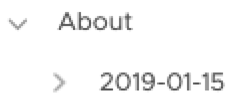


```
folder.setProperty('displayName', 'About - name / ID property not found. Please update iaasAboutFolderFindRelation');
```

So basically if we want to remove the "name / ID property not found. Please update iaasFolderFindRelation" we can either edit the displayName property or delete the line completely to keep the name set in the make object call. In this case I set the displayName property to "About"

There are also a lot of properties set on folders that are for informational . debugging purposes during the development of the plug-in. All these properties can be removed.

Here is the result after the changes to fix iaasAboutFolderFindRelation and IaasAboutFolderFindById



In some cases the API does not return much information or information that is not worth exposing to the plug-in. In this case the "Delete a type" workflow can be used. However be careful not to delete a parent type that is needed by a children type that is needed,

9. Create a plug-in from non swagger REST documentation

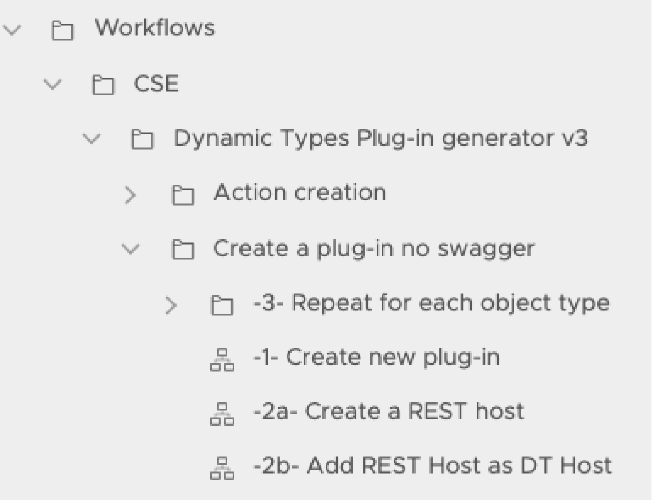
In the case the REST API is not providing a swagger / openAPI (I.E PDF based or HTML based REST API documentation) or in the case you need to create only specific inventory objects from this API it is possible by using a wizard style workflow that interacts with the API.

9.1. Pre requisites

To make sure you have proper REST Host and its credentials, proper authentication method follow section 8.1.

9.2. Creating the plug-in and REST Host

The workflows are located in "Create a plug-in no swagger"

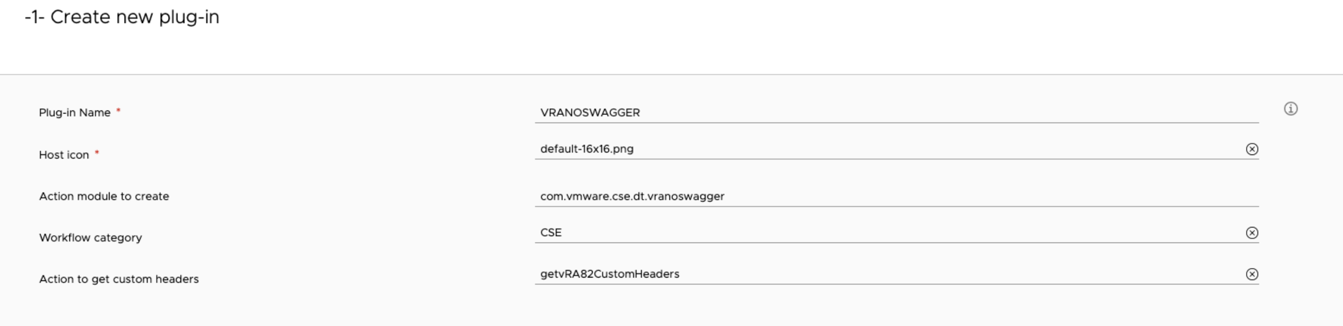


- Workflows
  - CSE
    - Dynamic Types Plug-in generator v3
      - Action creation
      - Create a plug-in no swagger
        - -3- Repeat for each object type
        - -1- Create new plug-in
        - -2a- Create a REST host
        - -2b- Add REST Host as DT Host

Run "-1- Create new plug-in "

- Enter a plug-in name
- Set a host icon (you can upload icons in vRO resources)
- Set the action module to be created (default from the plug-in name)
- Set a workflow category (Not used ATM, may be used for creating CRUD workflows)
- If the REST Host authentication requires passing a custom authorization header set the action you created in section 8.1

RUN

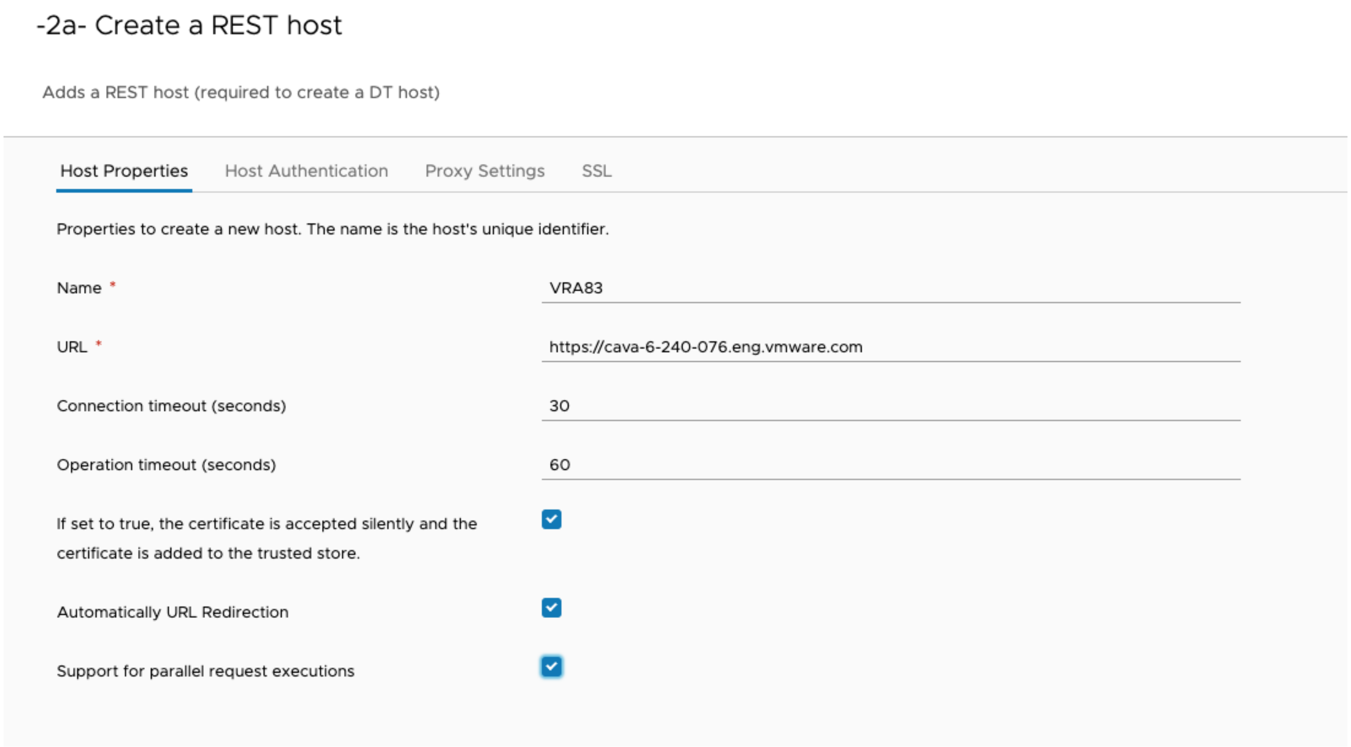


-1- Create new plug-in

Plug-in Name *	VRANOSWAGGER
Host icon *	default-16x16.png
Action module to create	com.vmware.cse.dt.vranoswagger
Workflow category	CSE
Action to get custom headers	getvRA82CustomHeaders

Run "-2a- Create a REST host" - This is the same workflow as the library "Add a REST host" workflow

- Set a REST Host name
- Enter the host URL
- Set the accept certificate, Automatic URL redirection and support for parallel operations to true



-2a- Create a REST host

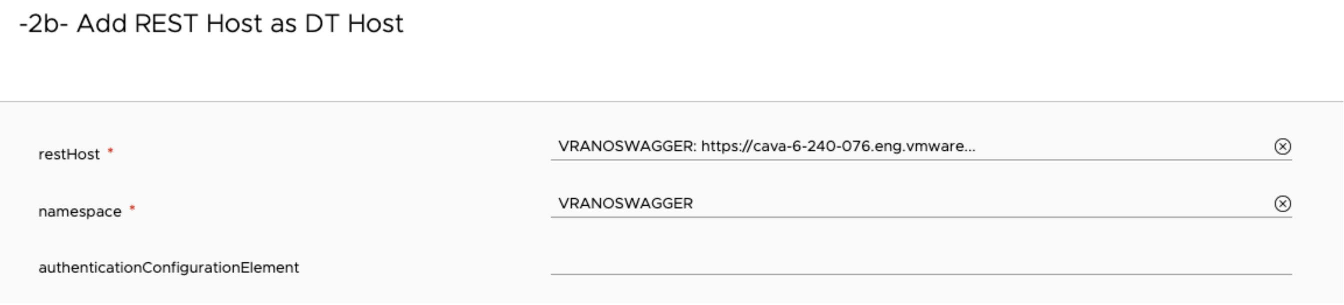
Adds a REST host (required to create a DT host)

Properties to create a new host. The name is the host's unique identifier.

Name *	VRA83
URL *	https://cava-6-240-076.eng.vmware.com
Connection timeout (seconds)	30
Operation timeout (seconds)	60
If set to true, the certificate is accepted silently and the certificate is added to the trusted store.	☑
Automatically URL Redirection	☑
Support for parallel request executions	☑

Run "-2b- Add REST Host as DT Host"

- Select the REST Host
- Select the plug-in namespace
- The third input is optional. It allows to bind a configuration element to the host so it can be used from the getCustomHeaderAction action

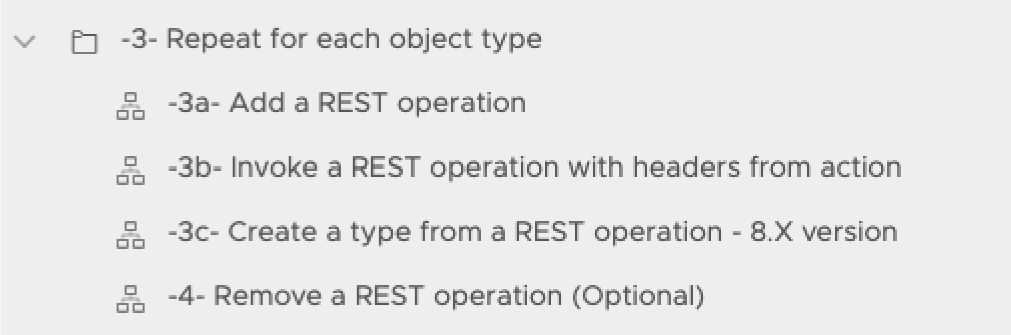


-2b- Add REST Host as DT Host

restHost *	VRANOSWAGGER: https://cava-6-240-076.eng.vmware...
namespace *	VRANOSWAGGER
authenticationConfigurationElement	

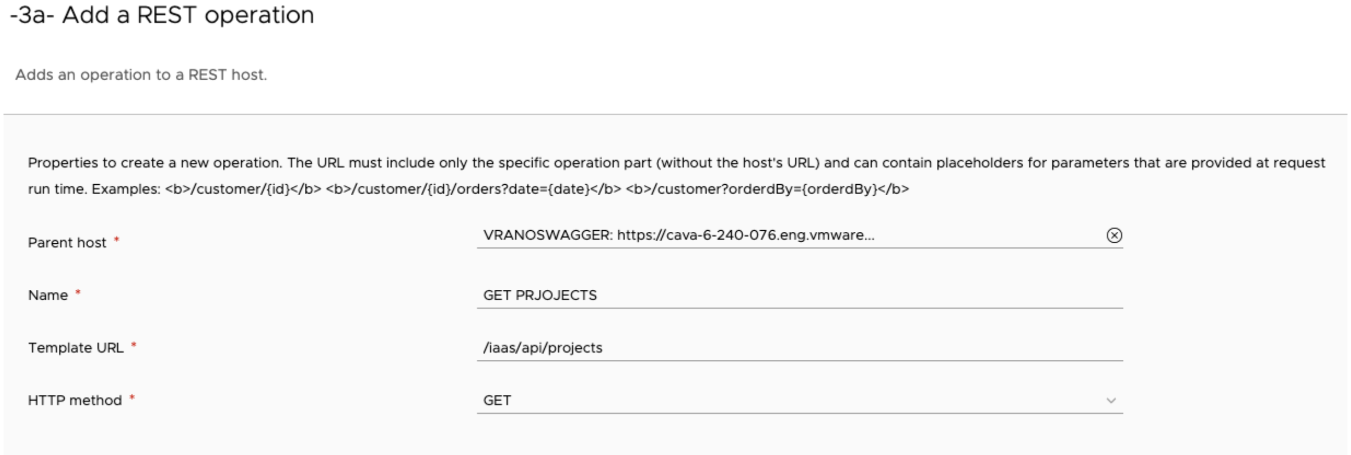
9.3. Creating a type under the host type

The workflows for creating the plug-in types are under the "-3- Repeat for each object type" folder.



- -3- Repeat for each object type
  - -3a- Add a REST operation
  - -3b- Invoke a REST operation with headers from action
  - -3c- Create a type from a REST operation - 8.X version
  - -4- Remove a REST operation (Optional)

Run the "-3a- Add a REST operation" workflow - this is the same workflow as the "Add a REST operation" workflow from the library



-3a- Add a REST operation

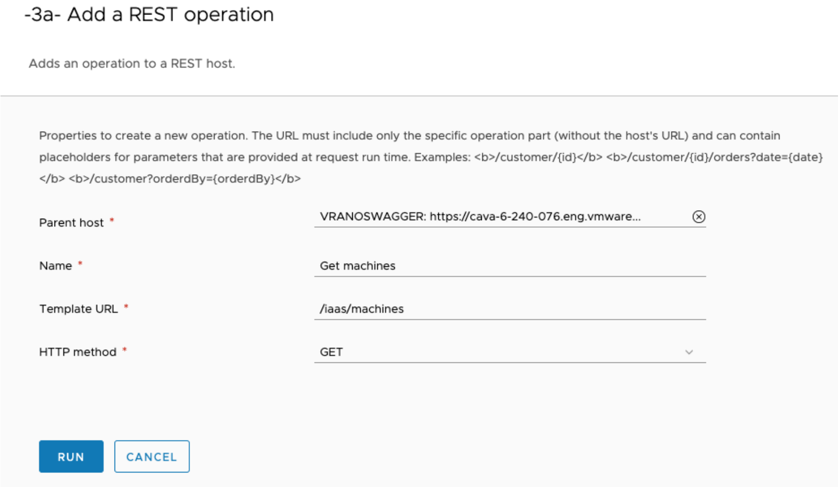
Adds an operation to a REST host.

Properties to create a new operation. The URL must include only the specific operation part (without the host's URL) and can contain placeholders for parameters that are provided at request run time. Examples: <b>/customer/{id}</b> <b>/customer/{id}/orders?date={date}</b> <b>/customer?orderBy={orderBy}</b>

Parent host *	VRANOSWAGGER: https://cava-6-240-076.eng.vmware...
Name *	GET PRJOJECTS
Template URL *	/iaas/api/projects
HTTP method *	GET

Run "-3b- Invoke a REST operation with headers from action"

- Select the operation
- Select the action used to create the Authorization header (if needed)



-3a- Add a REST operation

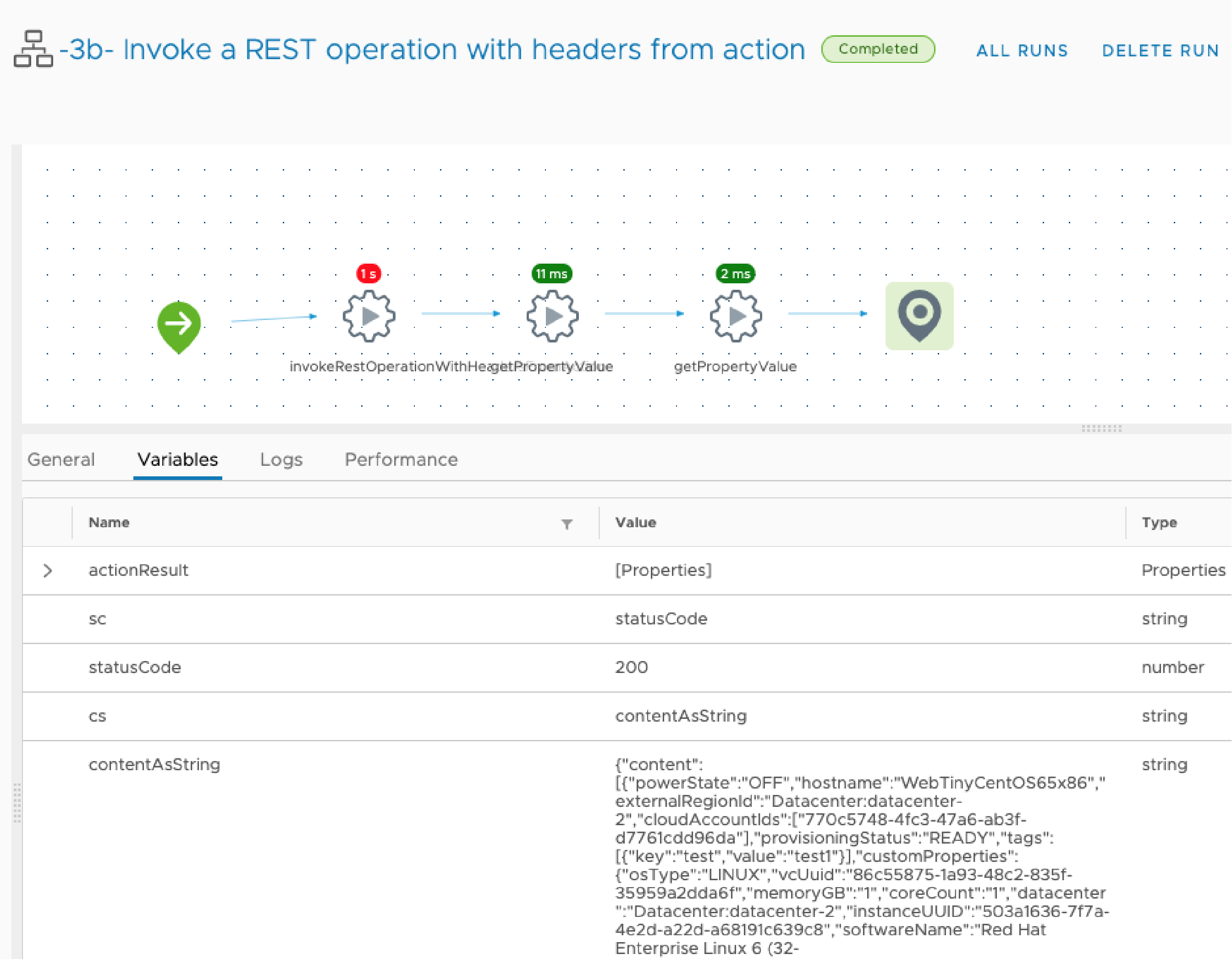
Adds an operation to a REST host.

Properties to create a new operation. The URL must include only the specific operation part (without the host's URL) and can contain placeholders for parameters that are provided at request run time. Examples: <b>/customer/{id}</b> <b>/customer/{id}/orders?date={date}</b> <b>/customer?orderBy={orderBy}</b>

Parent host *	VRANOSWAGGER: https://cava-6-240-076.eng.vmware...
Name *	Get machines
Template URL *	/iaas/machines
HTTP method *	GET

Check the request status is fine (200 for a GET operation)

Check contentAsString variable is set with returned objects.

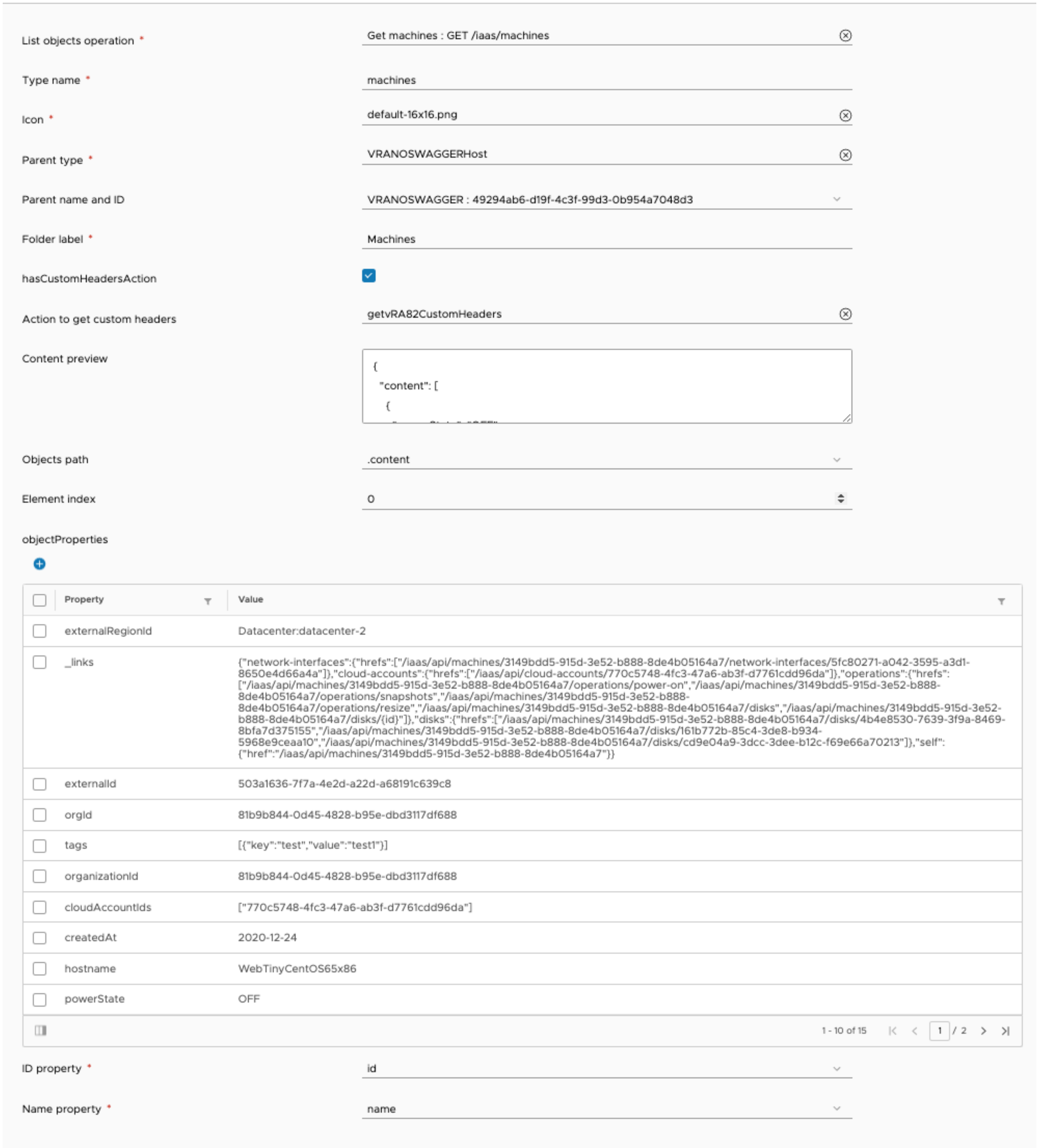

-3b- Invoke a REST operation with headers from action
Completed
ALL RUNS
DELETE RUN
1 s
11 ms
2 ms
invokeRestOperationWithHeagetPropertyValue
getPropertyValue
General
Variables
Logs
Performance

Name	Value	Type
actionResult	[Properties]	Properties
sc	statusCode	string
statusCode	200	number
cs	contentAsString	string
contentAsString	{"content":[{"powerState":"OFF","hostname":"WebTinyCentOS65x86","externalRegionId":"Datacenter:datacenter-2","cloudAccountIds":["770c5748-4fc3-47a6-ab3f-d7761cdd96da"],"provisioningStatus":"READY","tags":[{"key":"test","value":"test1"}],"customProperties":{"osType":"LINUX","vcUuid":"86c55875-1a93-48c2-835f-35959a2dda6f","memoryGB":"1","coreCount":"1","datacenter":"Datacenter:datacenter-2","instanceUUID":"503a1636-7f7a-4e2d-a22d-a68191c639c8","softwareName":"Red Hat Enterprise Linux 6 (32-	string

Run "-3c- Create a type from a REST operation"

- Select the REST operation you just created
- Enter a type name
- Set an icon for this type (icons can be uploaded as resource elements separately)
- Select a the parent type for this object. If the URL has no parameter the parent type should be the namespace name + "Host"
- Enter a folder label
- If you are using an action to return an authorization header check "hasCustomHeaderAction" and set your action
- If you got the previous steps right you should see a JSON based text in "Content Preview". You may copy and paste this JSON in a JSON viewer to check in which property the array of objects is stored
- Objects path should list all the possible paths to an array of object. Select the right one.
- Element index is used to select which of the object you want to see preview in the next objectProperties input
- The objectProperties input is used to see what is the data retrieved and to choose which properties should be kept in the Dynamic Types object. You can select and remove the ones that are not needed
- Set the ID property. This must be a unique ID, this must be the ID used in /objectType/{objectId} URL to get a specific object by ID
- Set the Name property. This should be a unique name allowing to identify the object.

RUN



List objects operation *	Get machines : GET /iaas/machines
Type name *	machines
Icon *	default-16x16.png
Parent type *	VRANOSWAGGERHost
Parent name and ID	VRANOSWAGGER : 49294ab6-d19f-4c3f-99d3-0b954a7048d3
Folder label *	Machines
hasCustomHeadersAction	☑
Action to get custom headers	getvRA82CustomHeaders

Content preview

```
{
  "content": [
   {
```

Objects path	.content
Element index	0

objectProperties

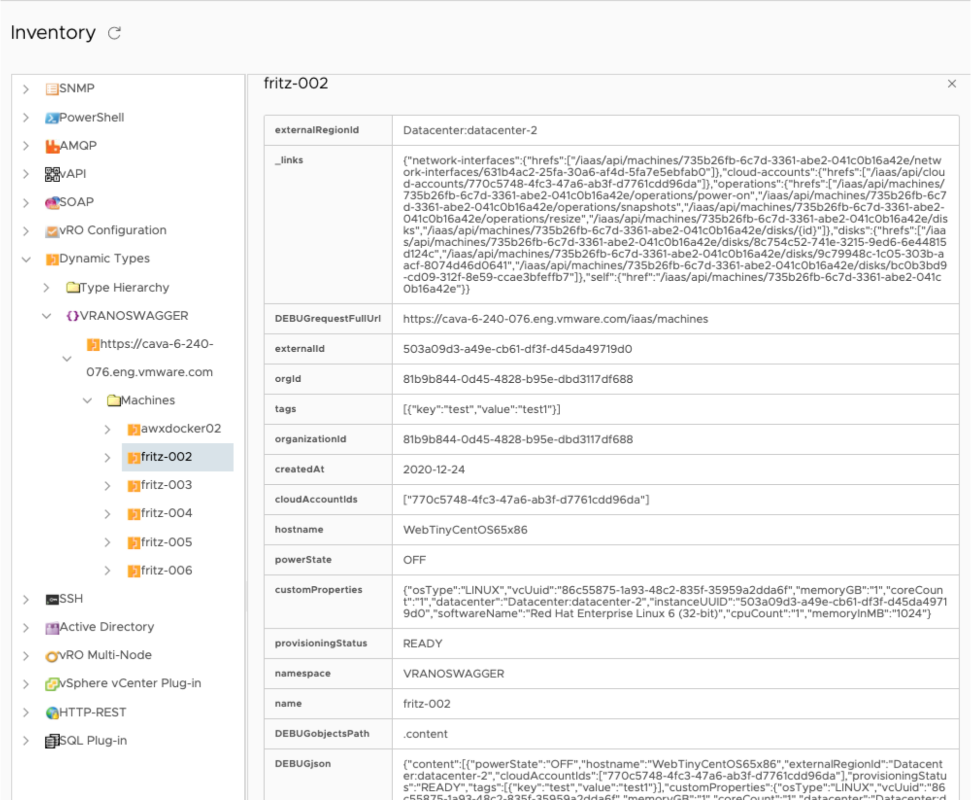
□	Property	Value
□	externalRegionId	Datacenter:datacenter-2
□	_links	{"network-interfaces":{"hrefs":["/iaas/api/machines/3149bdd5-915d-3e52-b888-8de4b05164a7/network-interfaces/5fc80271-a042-3595-a3d1-8650e4d66a4a"]},"cloud-accounts":{"hrefs":["/iaas/api/cloud-accounts/770c5748-4fc3-47a6-ab3f-d7761cdd96da"]},"operations":{"hrefs":["/iaas/api/machines/3149bdd5-915d-3e52-b888-8de4b05164a7/operations/power-on","/iaas/api/machines/3149bdd5-915d-3e52-b888-8de4b05164a7/operations/snapshots","/iaas/api/machines/3149bdd5-915d-3e52-b888-8de4b05164a7/operations/resize","/iaas/api/machines/3149bdd5-915d-3e52-b888-8de4b05164a7/disks","/iaas/api/machines/3149bdd5-915d-3e52-b888-8de4b05164a7/disks/{id}"]},"disks":{"hrefs":["/iaas/api/machines/3149bdd5-915d-3e52-b888-8de4b05164a7/disks/4b4e8530-7639-3f9a-8469-8bfa7d375155","/iaas/api/machines/3149bdd5-915d-3e52-b888-8de4b05164a7/disks/161b772b-85c4-3de8-b934-5968e9ceaa10","/iaas/api/machines/3149bdd5-915d-3e52-b888-8de4b05164a7/disks/cd9e04a9-3dcc-3dee-b12c-f69e66a70213"]},"self":{"href":"/iaas/api/machines/3149bdd5-915d-3e52-b888-8de4b05164a7"}}
□	externalId	503a1636-7f7a-4e2d-a22d-a68191c639c8
□	orgId	81b9b844-0d45-4828-b95e-dbd3117df688
□	tags	[{"key":"test","value":"test1"}]
□	organizationId	81b9b844-0d45-4828-b95e-dbd3117df688
□	cloudAccountIds	["770c5748-4fc3-47a6-ab3f-d7761cdd96da"]
□	createdAt	2020-12-24
□	hostname	WebTinyCentOS65x86
□	powerState	OFF

1 - 10 of 15 1 / 2

ID property *	id
Name property *	name

Once the workflow complete check the vRO Inventory.

You should see the folder object then the child objects. You can click and check the properties. There are a number of properties that are added to the object automatically starting with "DEBUG". You can click on the parent folder to get the name of the action returning the child objects and edit it to modify the properties returned if needed.



fritz-002

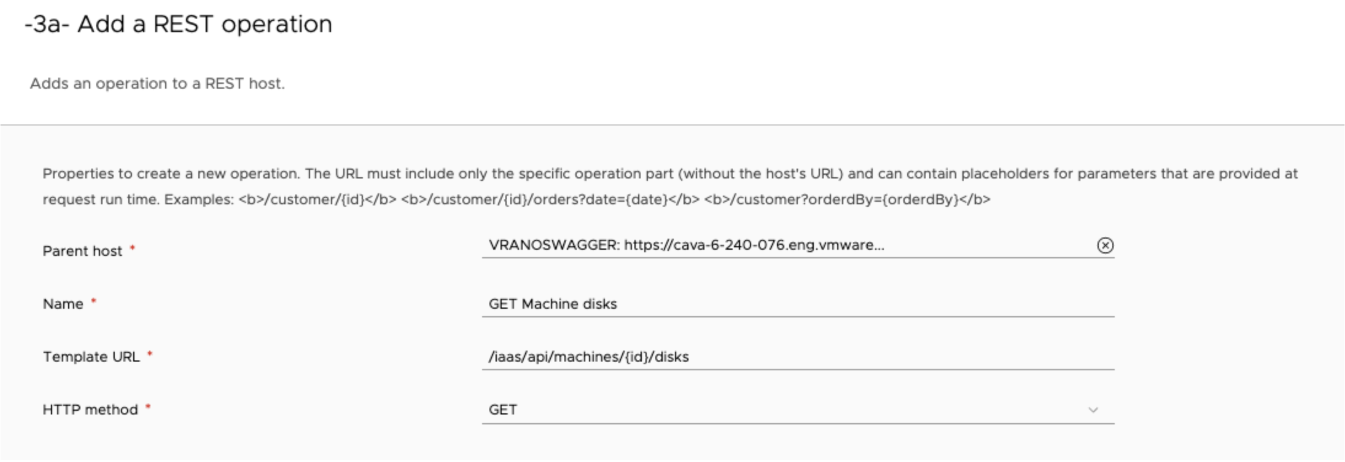
externalRegionId	Datacenter:datacenter-2
_links	{"network-interfaces":{"hrefs":["/iaas/api/machines/735b26fb-6c7d-3361-abe2-041c0b16a42e/network-interfaces/631b4ac2-25fa-30a6-af4d-5fa7e5ebfab0"]},"cloud-accounts":{"hrefs":["/iaas/api/cloud-accounts/770c5748-4fc3-47a6-ab3f-d7761cdd96da"]},"operations":{"hrefs":["/iaas/api/machines/735b26fb-6c7d-3361-abe2-041c0b16a42e/operations/power-on","/iaas/api/machines/735b26fb-6c7d-3361-abe2-041c0b16a42e/operations/snapshots","/iaas/api/machines/735b26fb-6c7d-3361-abe2-041c0b16a42e/operations/resize","/iaas/api/machines/735b26fb-6c7d-3361-abe2-041c0b16a42e/disks","/iaas/api/machines/735b26fb-6c7d-3361-abe2-041c0b16a42e/disks/{id}"]},"disks":{"hrefs":["/iaas/api/machines/735b26fb-6c7d-3361-abe2-041c0b16a42e/disks/8c754c52-741e-3215-9ed6-6e44815d124c","/iaas/api/machines/735b26fb-6c7d-3361-abe2-041c0b16a42e/disks/9c79948c-1c05-303b-aacf-8074d46d0641","/iaas/api/machines/735b26fb-6c7d-3361-abe2-041c0b16a42e/disks/bc0b3bd9-cd09-312f-8e59-ccae3bfeffb7"]},"self":{"href":"/iaas/api/machines/735b26fb-6c7d-3361-abe2-041c0b16a42e"}}
DEBUGrequestFullUrl	https://cava-6-240-076.eng.vmware.com/iaas/machines
externalId	503a09d3-a49e-cb61-df3f-d45da49719d0
orgId	81b9b844-0d45-4828-b95e-dbd3117df688
tags	[{"key":"test","value":"test1"}]
organizationId	81b9b844-0d45-4828-b95e-dbd3117df688
createdAt	2020-12-24
cloudAccountIds	["770c5748-4fc3-47a6-ab3f-d7761cdd96da"]
hostname	WebTinyCentOS65x86
powerState	OFF
customProperties	{"osType":"LINUX","vcUuid":"86c55875-1a93-48c2-835f-35959a2dda6f","memoryGB":"1","coreCount":"1","datacenter":"Datacenter:datacenter-2","instanceUUID":"503a09d3-a49e-cb61-df3f-d45da49719d0","softwareName":"Red Hat Enterprise Linux 6 (32-bit)","cpuCount":"1","memoryInMB":"1024"}
provisioningStatus	READY
namespace	VRANOSWAGGER
name	fritz-002
DEBUGobjectsPath	.content
DEBUGjson	{"content":[{"powerState":"OFF","hostname":"WebTinyCentOS65x86","externalRegionId":"Datacenter:datacenter-2","cloudAccountIds":["770c5748-4fc3-47a6-ab3f-d7761cdd96da"],"provisioningStatus":"READY","tags":[{"key":"test","value":"test1"}],"customProperties":{"osType":"LINUX","vcUuid":"86c55875-1a93-48c2-835f-35959a2dda6f","memoryGB":"1","coreCount":"1","datacenter":"Datacenter:d

9.4. Creating a type under another type

You may have object types requiring to be under another type. In the example below the disk type under the machine type.

Typically the URL for such type does have the parent type in the URL followed by a parameter that is the ID of the the parent object.

First add the REST Operation. The template URL must provide the parent id between {} as in the example below. You can find these URLs in the API documentation



-3a- Add a REST operation

Adds an operation to a REST host.

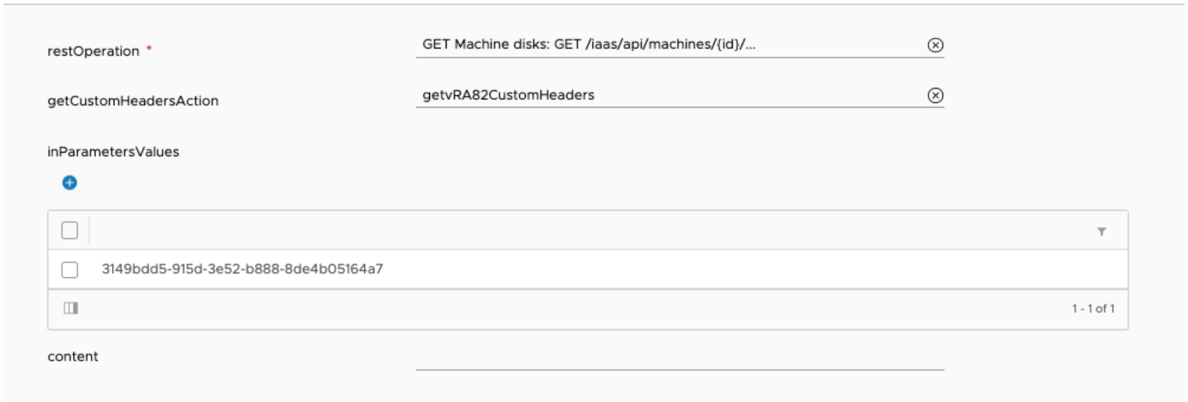
Properties to create a new operation. The URL must include only the specific operation part (without the host's URL) and can contain placeholders for parameters that are provided at request run time. Examples: <b>/customer/{id}</b> <b>/customer/{id}/orders?date={date}</b> <b>/customer?orderdBy={orderdBy}</b>

Parent host *	VRANOSWAGGER: https://cava-6-240-076.eng.vmware...
Name *	GET Machine disks
Template URL *	/iaas/api/machines/{id}/disks
HTTP method *	GET

Test this operation with running "-3b- Invoke a REST operation with headers from action"

This time you will have to provide a parameter value. If you look at the contentAsString variable from the last run of this workflow you should have in the objects their ids. You can copy and paste the id of one of them when adding a parameter.

-3b- Invoke a REST operation with headers from action



restOperation *	GET Machine disks: GET /iaas/api/machines/{id}/...
getCustomHeadersAction	getvRA82CustomHeaders

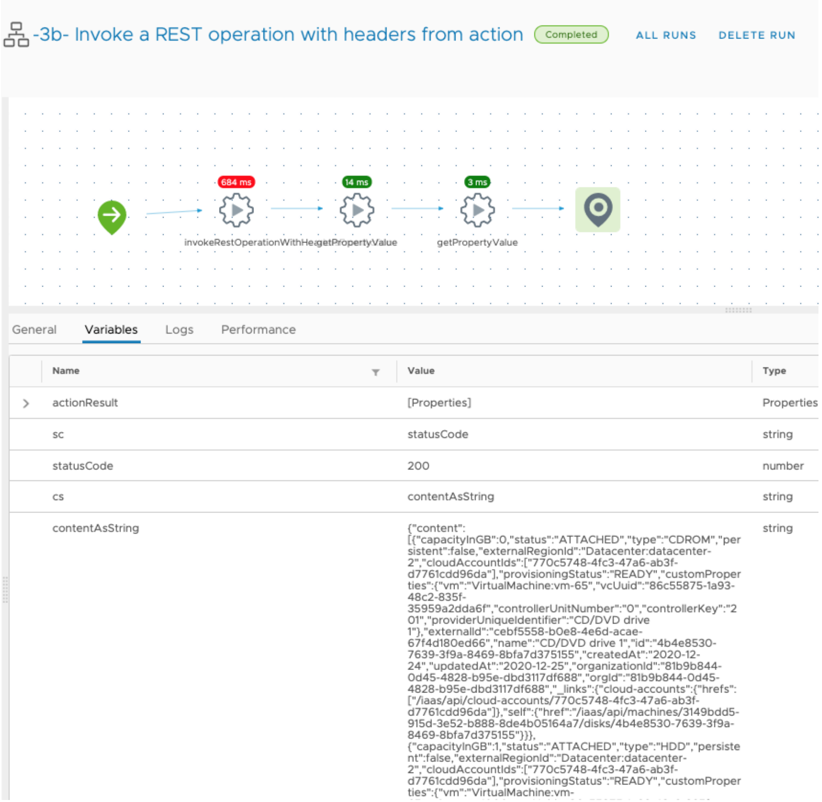
inParametersValues

□	
□	3149bdd5-915d-3e52-b888-8de4b05164a7

1 - 1 of 1

content

You should get a 200 status code with a new contentAsString set with the JSON representing the child objects.



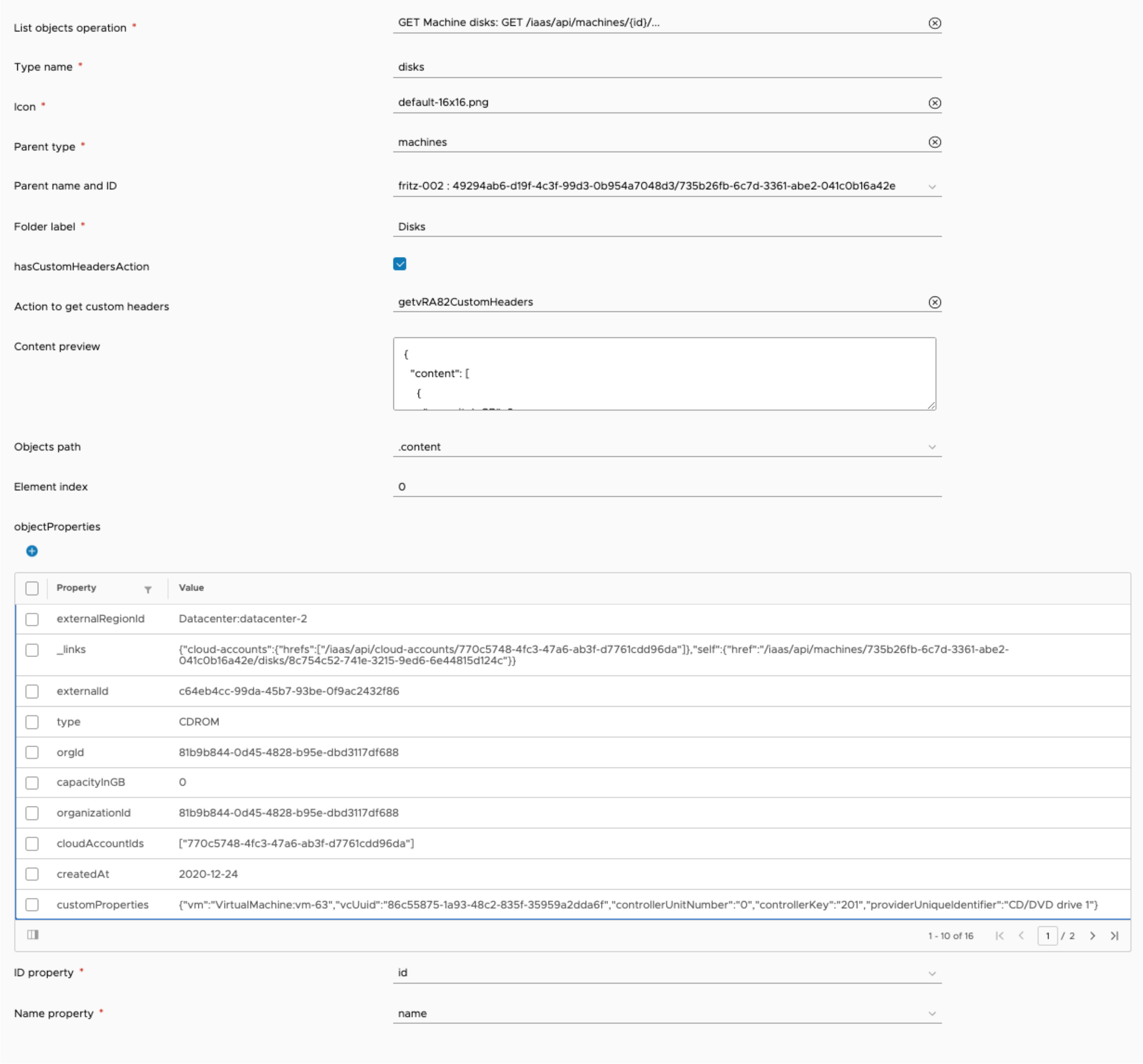
Name	Value	Type
actionResult	[Properties]	Properties
sc	statusCode	string
statusCode	200	number
cs	contentAsString	string
contentAsString	{"content":[{"capacityInGB":0,"status":"ATTACHED","type":"CDROM","persistent":false,"externalRegionId":"Datacenter:datacenter-2","cloudAccountIds":["770c5748-4fc3-47a6-ab3f-d7761cdd96da"],"provisioningStatus":"READY","customProperties":{"vm":"VirtualMachine:vm-65","vcUuid":"86c55875-1a93-48c2-835f-35959a2dda6f","controllerUnitNumber":"0","controllerKey":"201","providerUniqueIdentifier":"CD/DVD drive 1"},"externalId":"cebf5558-b0e8-4e6d-acae-67f4d180ed66","name":"CD/DVD drive 1","id":"4b4e8530-7639-3f9a-8469-8bfa7d375155","createdAt":"2020-12-24","updatedAt":"2020-12-25","organizationId":"81b9b844-0d45-4828-b95e-dbd3117df688","orgId":"81b9b844-0d45-4828-b95e-dbd3117df688","_links":{"cloud-accounts":{"hrefs":["/iaas/api/cloud-accounts/770c5748-4fc3-47a6-ab3f-d7761cdd96da"]},"self":{"href":"/iaas/api/machines/3149bdd5-915d-3e52-b888-8de4b05164a7/disks/4b4e8530-7639-3f9a-8469-8bfa7d375155"}}},{"capacityInGB":1,"status":"ATTACHED","type":"HDD","persistent":false,"externalRegionId":"Datacenter:datacenter-2","cloudAccountIds":["770c5748-4fc3-47a6-ab3f-d7761cdd96da"],"provisioningStatus":"READY","customProperties":{"vm":"VirtualMachine:vm-	string

Run "-3c- Create a type from a REST operation"

This is similar to the previous run you did except:

- Make sure to select the operation to get the child objects you just tested
- Parent type should be the parent type you previously created
- In parent name and ID select one item. This is the equivalent of passing and ID as you did in the previous step.

You can follow the previous create instructions for the other inputs.



List objects operation *	GET Machine disks: GET /iaas/api/machines/{id}/...
Type name *	disks
Icon *	default-16x16.png
Parent type *	machines
Parent name and ID	fritz-002 : 49294ab6-d19f-4c3f-99d3-0b954a7048d3/735b26fb-6c7d-3361-abe2-041c0b16a42e
Folder label *	Disks
hasCustomHeadersAction	☑
Action to get custom headers	getvRA82CustomHeaders
Content preview	{ "content": [{
Objects path	.content
Element index	0

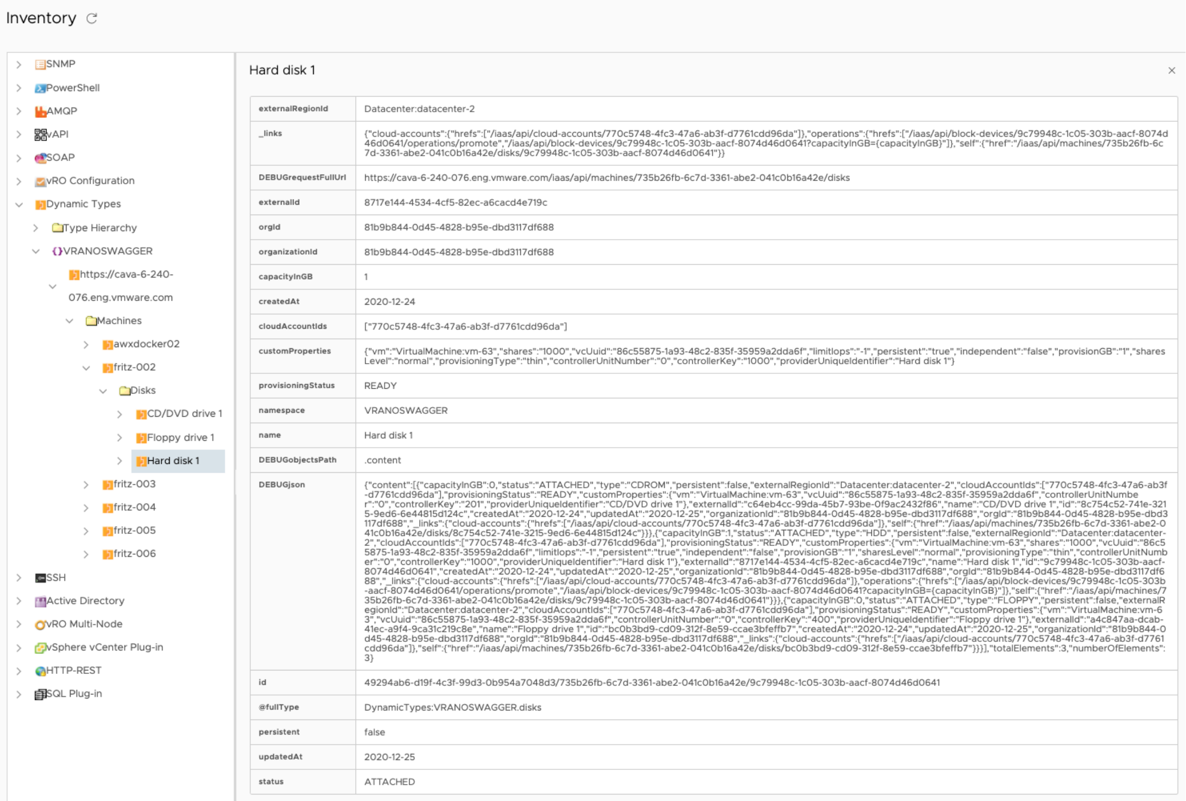
objectProperties

□	Property	Value
□	externalRegionId	Datacenter:datacenter-2
□	_links	{"cloud-accounts":{"hrefs":["/iaas/api/cloud-accounts/770c5748-4fc3-47a6-ab3f-d7761cdd96da"]},"self":{"href":"/iaas/api/machines/735b26fb-6c7d-3361-abe2-041c0b16a42e/disks/8c754c52-741e-3215-9ed6-6e44815d124c"}}
□	externalId	c64eb4cc-99da-45b7-93be-0f9ac2432f86
□	type	CDROM
□	orgId	81b9b844-0d45-4828-b95e-dbd3117df688
□	capacityInGB	0
□	organizationId	81b9b844-0d45-4828-b95e-dbd3117df688
□	cloudAccountIds	["770c5748-4fc3-47a6-ab3f-d7761cdd96da"]
□	createdAt	2020-12-24
□	customProperties	{"vm":"VirtualMachine:vm-63","vcUuid":"86c55875-1a93-48c2-835f-35959a2dda6f","controllerUnitNumber":"0","controllerKey":"201","providerUniqueIdentifier":"CD/DVD drive 1"}

ID property *	id
Name property *	name

Once the workflow complete check the vRO Inventory.

You should see under the parent object the child object folder then the child objects. You can click and check the properties.



Inventory

- SNMP
- PowerShell
- AMQP
- vAPI
- SOAP
- vRO Configuration
- Dynamic Types
  - Type Hierarchy
  - VRANOSWAGGER
    - https://cava-6-240-076.eng.vmware.com
      - Machines
        - awxdocker02
        - fritz-002
          - Disks
            - CD/DVD drive 1
            - Floppy drive 1
            - Hard disk 1
        - fritz-003
        - fritz-004
        - fritz-005
        - fritz-006
- SSH
- Active Directory
- vRO Multi-Node
- vSphere vCenter Plug-in
- HTTP-REST
- SQL Plug-in

Hard disk 1

externalRegionId	Datacenter:datacenter-2
_links	{"cloud-accounts":{"hrefs":["/iaas/api/cloud-accounts/770c5748-4fc3-47a6-ab3f-d7761cdd96da"]},"operations":{"hrefs":["/iaas/api/block-devices/9c79948c-1c05-303b-aacf-807446d0641/operations/promote","/iaas/api/block-devices/9c79948c-1c05-303b-aacf-8074d46d0641?capacityInGB={capacityInGB}"]},"self":{"href":"/iaas/api/machines/735b26fb-6c7d-3361-abe2-041c0b16a42e/disks/9c79948c-1c05-303b-aacf-8074d46d0641"}}
DEBUGrequestFullUrl	https://cava-6-240-076.eng.vmware.com/iaas/api/machines/735b26fb-6c7d-3361-abe2-041c0b16a42e/disks
externalId	8717e144-4534-4cf5-82ec-a6cacd4e719c
orgId	81b9b844-0d45-4828-b95e-dbd3117df688
organizationId	81b9b844-0d45-4828-b95e-dbd3117df688
capacityInGB	1
createdAt	2020-12-24
cloudAccountIds	["770c5748-4fc3-47a6-ab3f-d7761cdd96da"]
customProperties	{"vm":"VirtualMachine:vm-63","shares":"1000","vcUuid":"86c55875-1a93-48c2-835f-35959a2dda6f","limitIops":"-1","persistent":"true","independent":"false","provisionGB":"1","sharesLevel":"normal","provisioningType":"thin","controllerUnitNumber":"0","controllerKey":"1000","providerUniqueIdentifier":"Hard disk 1"}
provisioningStatus	READY
namespace	VRANOSWAGGER
name	Hard disk 1
DEBUGobjectsPath	.content
DEBUGjson	{"content":[{"capacityInGB":0,"status":"ATTACHED","type":"CDROM","persistent":false,"externalRegionId":"Datacenter:datacenter-2","cloudAccountIds":["770c5748-4fc3-47a6-ab3f-d7761cdd96da"],"provisioningStatus":"READY","customProperties":{"vm":"VirtualMachine:vm-63","vcUuid":"86c55875-1a93-48c2-835f-35959a2dda6f","controllerUnitNumber":"0","controllerKey":"201","providerUniqueIdentifier":"CD/DVD drive 1"},"externalId":"c64eb4cc-99da-45b7-93be-0f9ac2432f86","name":"CD/DVD drive 1","id":"8c754c52-741e-3215-9ed6-6e44815d124c","createdAt":"2020-12-24","updatedAt":"2020-12-25","organizationId":"81b9b844-0d45-4828-b95e-dbd3117df688","orgId":"81b9b844-0d45-4828-b95e-dbd3117df688","_links":{"cloud-accounts":{"hrefs":["/iaas/api/cloud-accounts/770c5748-4fc3-47a6-ab3f-d7761cdd96da"]},"self":{"href":"/iaas/api/machines/735b26fb-6c7d-3361-abe2-041c0b16a42e/disks/8c754c52-741e-3215-9ed6-6e44815d124c"}}},{"capacityInGB":1,"status":"ATTACHED","type":"HDD","persistent":false,"externalRegionId":"Datacenter:datacenter-2","cloudAccountIds":["770c5748-4fc3-47a6-ab3f-d7761cdd96da"],"provisioningStatus":"READY","customProperties":{"vm":"VirtualMachine:vm-63","shares":"1000","vcUuid":"86c55875-1a93-48c2-835f-35959a2dda6f","limitIops":"-1","persistent":"true","independent":"false","provisionGB":"1","sharesLevel":"normal","provisioningType":"thin","controllerUnitNumber":"0","controllerKey":"1000","providerUniqueIdentifier":"Hard disk 1"},"externalId":"8717e144-4534-4cf5-82ec-a6cacd4e719c","name":"Hard disk 1","id":"9c79948c-1c05-303b-aacf-8074d46d0641","createdAt":"2020-12-24","updatedAt":"2020-12-25","organizationId":"81b9b844-0d45-4828-b95e-dbd3117df688","orgId":"81b9b844-0d45-4828-b95e-dbd3117df688","_links":{"cloud-accounts":{"hrefs":["/iaas/api/cloud-accounts/770c5748-4fc3-47a6-ab3f-d7761cdd96da"]},"operations":{"hrefs":["/iaas/api/block-devices/9c79948c-1c05-303b-aacf-8074d46d0641/operations/promote","/iaas/api/block-devices/9c79948c-1c05-303b-aacf-8074d46d0641?capacityInGB={capacityInGB}"]},"self":{"href":"/iaas/api/machines/735b26fb-6c7d-3361-abe2-041c0b16a42e/disks/9c79948c-1c05-303b-aacf-8074d46d0641"}}},{"capacityInGB":0,"status":"ATTACHED","type":"FLOPPY","persistent":false,"externalRegionId":"Datacenter:datacenter-2","cloudAccountIds":["770c5748-4fc3-47a6-ab3f-d7761cdd96da"],"provisioningStatus":"READY","customProperties":{"vm":"VirtualMachine:vm-63","vcUuid":"86c55875-1a93-48c2-835f-35959a2dda6f","controllerUnitNumber":"0","controllerKey":"400","providerUniqueIdentifier":"Floppy drive 1"},"externalId":"a4c847aa-dcab-41ec-a9f4-9ca31c219c8e","name":"Floppy drive 1","id":"bc0b3bd9-cd09-312f-8e59-ccae3bfeffb7","createdAt":"2020-12-24","updatedAt":"2020-12-25","organizationId":"81b9b844-0d45-4828-b95e-dbd3117df688","orgId":"81b9b844-0d45-4828-b95e-dbd3117df688","_links":{"cloud-accounts":{"hrefs":["/iaas/api/cloud-accounts/770c5748-4fc3-47a6-ab3f-d7761cdd96da"]},"self":{"href":"/iaas/api/machines/735b26fb-6c7d-3361-abe2-041c0b16a42e/disks/bc0b3bd9-cd09-312f-8e59-ccae3bfeffb7"}}}],"totalElements":3,"numberOfElements":3}
id	49294ab6-d19f-4c3f-99d3-0b954a7048d3/735b26fb-6c7d-3361-abe2-041c0b16a42e/9c79948c-1c05-303b-aacf-8074d46d0641
@fullType	DynamicTypes:VRANOSWAGGER.disks
persistent	false
updatedAt	2020-12-25
status	ATTACHED