JSON REST Web Services Connector - Properties

In this agent, the configuration of the properties attributes is very important due to they define the functionality of the integration:

This agent has five families of properties:

Family	Description
Load	Used to retrieve all the objects in the target system
Select	Used to retrieve an object in the target system
Insert	Used to create an object in the target system
Update	Used to update an object in the target system
Delete	Used to remove an object in the target system

These families are involved in the following processes:

Process	Families
Reconcile automatic task	Load + select
Authoritative automatic task	Load + select
Sync new object	Select + Insert
Sync updated object	Select + Update
Sync deleted object	Select + Delete

These are the properties attributes grouped by family:

Load

Property	Description
loadPath (required)	Denotes the path (relative to webserver root) where the WebService is located. It can contain variable names in the form of **${variableName}**. JSON connector will replace that name for the actual value. Eventually, complex expressions can be written in, but it's discouraged

Property	Description
loadMethod (required)	Denotes the HTTP method to use: PUT, POST, GET and DELETE are allowed
loadEncoding (required)	Type of encoded data that will be used.
loadParams (optional)	Put the character '-' in case you would avoid its value
loadTemplate (optional)	Name of the corresponding template defined on the XML Templates.
loadResults (optional)	But highly recommended) denotes the JSON portion that contains current data for the object. If this element is not present, or empty, the connector will conclude the object does not exist yet. This property will contain a simple JSON attribute name, but complex scripts are also allowed.
loadSuccessCodes (optional)	The HTTP codes to be interpreted as OK.
loadFailureCodes (optional)	The HTTP codes to be interpreted as Error.
loadNext (optional)	Next page to fetch. When the response gives us the URL of the next page to fetch, you must type the tag name of this attribute.
loadPagination (optional)	Complex script to get the next call that has to be done.
loadCondition (optional)	Script to prevent a call. To prevent the call must return false.
loadHeader (optional)	Use this property to send HTTP header(s). More than one header can be sent by adding multiple properties loadHeader1, loadHeader2, and so on. The value of the header is "HEADER:VALUE", for example "Accept:application/json".

Select

Property	Description
selectPath (required)	Denotes the path (relative to webserver root) where the WebService is located. It can contain variable names in the form of **${variableName}**. JSON connector will replace that name for the actual value. Eventually, complex expressions can be written in, but it's discouraged
selectMethod (required)	Denotes the HTTP method to use: PUT, POST, GET and DELETE are allowed

Property	Description
selectEncoding (required)	Denotes the encoding used to send to the target webservice. **application/json** and **application/x-www-form-urlencoded** are supported. The first one is used by default to POST and PUT requests. The second one is used by default for GET and DELETE requests
selectParams (optional)	Put the character '-' in case you would avoid its value
selectTemplate (optional)	Name of the corresponding template defined on the XML Templates.
selectResults (optional)	Denotes the JSON portion that contains current data for the object. If this element is not present, or empty, the connector will conclude the object does not exist yet. This property will contain a simple JSON attribute name, but complex scripts are also allowed
selectSuccessCodes (optional)	The HTTP codes to be interpreted as OK.
selectFailureCodes (optional)	The HTTP codes to be interpreted as Error.
selectNext (optional)	Next page to fetch. When the response gives us the URL of the next page to fetch, you must type the tag name of this attribute.
selectPagination (optional)	Complex script to get the next call that has to be done.
selectCondition (optional)	Script to prevent a call. To prevent the call must return false.
selectHeader (optional)	Use this property to send HTTP header(s). More than one header can be sent by adding multiple properties selectHeader1, selectHeader2, and so on. The value of the header is "HEADER:VALUE", for instance, "Accept:application/json".

Insert

Property	Description
insertPath (required)	Denotes the path (relative to webserver root) where the webservice is located.
insertMethod (required)	Denotes the HTTP method to use: PUT, POST, GET and DELETE are allowed
insertEncoding (required)	Denotes the encoding used to send to the target webservice. **application/json** and **application/x-www-form-urlencoded** are supported. The first one is used by default to POST and PUT requests. The second one is used by default for GET and DELETE requests
insertTemplate (optional)	Name of the corresponding template defined on the XML Templates.

Property	Description
insertParams (optional)	Type in the attributes that will be sent to the rest server. If this property is not set, all attributes will be sent.
insertResults (optional)	Denotes the JSON portion that contains current data for the object. If this element is not present, or empty, the connector will conclude the object does not exist yet. This property will contain a simple JSON attribute name, but complex scripts are also allowed
insertSuccessCodes (optional)	The HTTP codes to be interpreted as OK.
insertFailureCodes (optional)	The HTTP codes to be interpreted as Error.
insertCondition (optional)	Script to prevent a call. To prevent the call must return false.
insertHeader (optional)	Use this property to send HTTP header(s). More than one header can be sent by adding multiple properties insertHeader1, insertHeader2, and so on. The value of the header is "HEADER:VALUE", for example "Accept:application/json".

Update

Property	Description
updatePath (required)	Denotes the path (relative to webserver root) where the webservice is located
updateMethod (required)	Denotes the HTTP method to use: PUT, POST, GET and DELETE are allowed
updateEncoding (required)	Denotes the encoding used to send to the target webservice. **application/json** and **application/x-www-form-urlencoded** are supported. The first one is used by default to POST and PUT requests. The second one is used by default for GET and DELETE requests
updateParams (optional)	Type in the attributes that will be sent to the rest server. If this property is not set, all attributes will be sent.
updateResults (optional)	Denotes the JSON portion that contains current data for the object. If this element is not present, or empty, the connector will conclude the object does not exist yet. This property will contain a simple JSON attribute name, but complex scripts are also allowed

Property	Description
updateSuccessCodes (optional)	The HTTP codes to be interpreted as OK.
updateFailureCodes (optional)	The HTTP codes to be interpreted as Error.
updateCondition (optional)	Script to prevent a call. To prevent the call must return false.
updateHeader (optional)	Use this property to send HTTP header(s). More than one header can be sent by adding multiple properties updateHeader1, updateHeader2, and so on. The value of the header is "HEADER:VALUE", for example "Accept:application/json".

Delete

Property	Description
deletePath (required)	Denotes the path (relative to webserver root) where the webservice is located
deleteMethod (required)	Denotes the HTTP method to use: PUT, POST, GET and DELETE are allowed
deleteEncoding (required)	Denotes the encoding used to send to the target webservice. **application/json** and **application/x-www-form-urlencoded** are supported. The first one is used by default to POST and PUT requests. The second one is used by default for GET and DELETE requests
deleteParams (optional)	Type in the attributes that will be sent to the rest server. If this property is not set, all attributes will be sent.
deleteResults (optional)	Denotes the JSON portion that contains current data for the object. If this element is not present, or empty, the connector will conclude the object does not exist yet. This property will contain a simple JSON attribute name, but complex scripts are also allowed
deleteSuccessCodes (optional)	The HTTP codes to be interpreted as OK.
deleteFailureCodes (optional)	The HTTP codes to be interpreted as Error.
deleteCondition (optional)	Script to prevent a call. To prevent the call must return false.
deleteHeader (optional)	Use this property to send HTTP header(s). More than one header can be sent by adding multiple properties deleteHeader1, deleteHeader2, and so on. The value of the header is "HEADER:VALUE", for example "Accept:application/json".

How to retrieve data from the response with the *Results properties

a) One level

If the JSON has one level you have to avoid the property

```
{
   "userName" : "soffid"
}
```

b) Two level

If the JSON has two levels you have to create the property *Result and put the name of the parent attribute, for example:

```
{
   "user" : {
      "userName" : "soffid"
   }
}
```

And the property must be for example loadResults = user

c) More than two levels

If the JSON has more than two levels you have to create the property *Result and put the atributes in the next pattern

*Results = attribure1{"attribute2"}{"attribute3"}...

For example:

```
{
"data" : {
      "user" : {
         "userName" : {
            "string" : "soffid"
         }
      }
   }
}
```

And the property must be for example:

loadResults = data{"user"}{"userName"}

Revision #3
Created 27 July 2022 09:47:06
Updated 27 July 2022 10:59:33