Flow Orchestration

> ⚠ The examples and configuration information in this topic use a model named simpleuser with the default endpoints generated.

The API Orchestration user interface is accessed from the APIs List page on the API Builder Console by selecting a **Flow** icon or a **Create Flow** i con for a generated or imported endpoint depending upon the current endpoint status. The API Orchestration user interface is divided into the following panels:

- Flow-node list (left side of the API Orchestration user interface) - Provides a graphical listing of the Model and Core flow-nodes. The default Core flow-node types are: Custom, Codeblock, Compose, Condition, Delay, HTTP, JSON, and Set Context. A model flow-node is displayed for each configured model and an endpoint flow-node is displayed for each imported endpoint.
- Flow editor (center of the API Orchestration user interface) - Provides a graphical space to view, edit, and create flows.
- Flow-node configuration (right side of the API Orchestration user interface) - Provides the functionality to configure the Name, Method, Parameters, and Outputs of flow-nodes.

This topic is divided into the following sections:

Model flow-node

The Model flow-node methods, parameters, and outputs are described in the following sections.

Methods

The method selections for a Model flow-node are:

- `count` - Gets a count of records.
- `create` - Creates a new model object.
- `delete` - Deletes the model object.
- `deleteAll` - Deletes all the model objects.
- `distinct` - Finds unique values using the provided field.
- `findAll` - Finds all model instances.
- `findAndModify` - Finds one model instance and modifies it.
- `findByID` - Finds model instance by ID.
- `query` - Queries for particular model records.
- `update` - Updates a model instance.
- `upsert` - Creates a model record if not found, or updates the model record if found.

Parameters

The Model flow-node parameters are described in the following sections.

count parameters

The `count` method parameters are:

Parameter	Type	Default	Description	Configuration selection
`limit`	integer	`10`	The number of records to fetch.	Selector, Number
`order`	object	`-`	A dictionary of key-value pairs describing the field(s) for sorting. The field name is the key and the value is set to either -1 for ascending order or 1 for descending order.	Selector, Object
`page`	integer	`1`	Starting page number.	Selector, Number
`per_page`	integer	`10`	Results per page.	Selector, Number
`sel`	object	`-`	A dictionary of key-value pairs describing which fields to include in the query results. The field name is the key and the value is set to 1.	Selector, Object
`skip`	integer	`-`	The number of records to skip.	Selector, Number
`unsel`	object	`-`	A dictionary of key-value pairs describing which fields to exclude from the query results. The field name is the key and the value is set to 1.	Selector, Object
`where`	string	`-`	The JSON-encoded object specifying field constraints. The field name is the key and the value is the constraint statement or value.	Selector, String

All parameters can be enabled or disabled.

create parameters

The `create` method parameter is:

Parameter	Type	Configuration selection
`data`	object	Selector, Object

delete parameters

The `delete` method parameter is:

Parameter	Type	Configuration selection
`id`	any	Selector, String, Number, Boolean, Object, Array, Null

deleteAll parameters

There are no configurable `deleteAll` method parameters.

distinct parameters

The `distinct` method parameters are:

Parameter	Type	Default	Description	Configuration selection
`field`	string	`-`	The field must be distinct.	Selector, String
`limit`	integer	`10`	The number of records to fetch.	Selector, Number
`order`	object	`-`	A dictionary of key-value pairs describing the field(s) for sorting. The field name is the key and the value is set to either -1 for ascending order or 1 for descending order.	Selector, Object

page	number	1	Starting page number.	Selector, Number
per_page	integer	10	Results per page.	Selector, Number
sel	object	-	A dictionary of key-value pairs describing which fields to include in the query results. The field name is the key and the value is set to 1.	Selector, Object
skip	integer	-	The number of records to skip.	Selector, Number
unsel	object	-	A dictionary of key-value pairs describing which fields to exclude from the query results. The field name is the key and the value is set to 1.	Selector, Object
where	string	-	The JSON-encoded object specifying field constraints. The field name is the key and the value is the constraint statement or value.	Selector, String

The `limit`, `order`, `page`, `per_page`, `sel`, `skip`, `unsel`, and `where` parameters can be enabled or disabled.

findAll parameters

There are no configurable `findAll` method parameters.

findAndModify parameters

The `findAndModify` method parameters are:

Parameter	Type	Default	Description	Configuration selection
data	object	-	-	Selector, Object
args	object	-	Optional parameters.	Selector, Object
limit	integer	10	The number of records to fetch.	Selector, Number
order	object	-	A dictionary of key-value pairs describing the field(s) for sorting. The field name is the key and the value is set to either -1 for ascending order or 1 for descending order.	Selector, Object
page	integer	1	Starting page number.	Selector, Number
per_page	integer	10	Results per page.	Selector, Number
sel	object	-	A dictionary of key-value pairs describing which fields to include in the query results. The field name is the key and the value is set to 1.	Selector, Object
skip	integer	-	The number of records to skip.	Selector, Number
unsel	object	-	A dictionary of key-value pairs describing which fields to exclude from the query results. The field name is the key and the value is set to 1.	Selector, Object
where	string	-	The JSON-encoded object specifying field constraints. The field name is the key and the value is the constraint statement or value.	Selector, String

The `args`, `limit`, `order`, `page`, `per_page`, `sel`, `skip`, `unsel`, and `where` parameters can be enabled or disabled.

findByID parameters

The `findByID` method parameter is:

Parameter	Type	Default	Description	JSONPath	Configuration selections
id	any	-	-	$.params.id	Selector, String, Number, Boolean, Object, Array, Null

query parameters

The `query` method parameters are:

Parameter	Type	Default	Description	Configuration selection

`limit`	integer	`10`	The number of records to fetch.	Selector, Number
`order`	object	-	A dictionary of key-value pairs describing the field(s) for sorting. The field name is the key and the value is set to either -1 for ascending order or 1 for descending order.	Selector, Object
`page`	integer	-	Starting page number.	Selector, Number
`per_page`	integer	`10`	Results per page.	Selector, Number
`sel`	object	-	A dictionary of key-value pairs describing which fields to include in the query results. The field name is the key and the value is set to 1.	Selector, Object
`skip`	integer	-	The number of records to skip.	Selector, Number
`unsel`	object	-	A dictionary of key-value pairs describing which fields to exclude from the query results. The field name is the key and the value is set to 1.	Selector, Object
`where`	string	-	The JSON-encoded object specifying field constraints. The field name is the key and the value is the constraint statement or value.	Selector, String

All parameters can be enabled or disabled.

update parameters

The `update` method parameters are:

Parameter	Type	Description	Configuration selection
`data`	object	Dependent on configured model fields.	Selector, Object
`id`	any	-	Selector, String, Number, Boolean, Object, Array, Null

upsert parameters

The `upsert` method parameter is:

Parameter	Type	Configuration selections
`data`	object	Selector, Object

Outputs

The Model flow-node outputs are described in the following sections.

count outputs

The `count` method output is:

Output	Type	Description	Save output value as:
`next`	number	Successfully counted records of simpleuser.	`$.count`

create outputs

The `create` method output is:

Output	Type	Description	Save output value as:
`next`	object	Successfully created a simpleuser.	`$.model`

delete outputs

The `delete` method outputs are:

Output	Type	Description	Save output value as:

`next`	object	Successfully deleted the simpleuser.	`$.delete`
`notfound`	any	No model instance found.	-

deleteAll outputs

The `deleteAll` method output is:

Output	Type	Description	Save output value as:
`next`	array	Successfully deleted all the simpleusers.	-

distinct outputs

The `distinct` method output is:

Output	Type	Description	Save output value as:
`next`	array	Successfully found all unique values of simpleuser.	`$.models`

findAll outputs

The `findAll` method output is:

Outputs	Type	Description	Save output value as:
next	array	Successfully found all simpleusers.	`$.models`

findAndModify outputs

The `findAndModity` method outputs are:

Output	Type	Description	Save output value as:
`next`	object	Successfully found and modified simpleuser.	`$.model`
`notfound`	any	No matching model found.	-

findByID outputs

The `findByID` method outputs are:

Output	Type	Description	Save output value as:
`next`	object	Successfully found instance of simpleuser by ID.	`$.model`
`notfound`	any	No model instance found.	-

query outputs

The `query` method output is:

Output	Type	Description	Save output value as:
`next`	array	Successfully queried the simpleuser.	`$.models`

update outputs

The `update` method outputs are:

Output	Type	Description	Save output value as:
`next`	object	Successfully updated the simpleuser.	`$.model`
`notfound`	any	No model instance found.	-

upsert outputs

The `upsert` method outputs are:

Output	Type	Description	Save output value as:
`update`	object	Successfully updated the model.	`$.model`
`insert`	object	Successfully inserted the model.	`$.model`

Codeblock flow-node

The Codeblock flow-node methods, parameters, and outputs for the Greet Codeblock flow-node are described in the following sections. Each method in a Codeblock flow-node corresponds to a single codeblock. The Greet Codeblock flow-node is created when a new application is created, but it can be deleted.

Methods

The default method for the Greet Codeblock flow-node is:

- `Greet` - Some codeblock to run with the greet flow.

Parameter

The `Greet` method parameter is:

Parameter	Type	Configuration selection
`username`	string	Selector, String

Outputs

The Greet method outputs are:

Output	Type	Description	Save output value as:
`next`	string	The codeblock completed.	`$.greeting`
`error`	object	The codeblock failed to complete.	`$.error`

Condition flow-node

The Condition flow-node methods, parameters, and outputs are described in the following sections.

Methods

The default methods for a Condition flow-node are:

- `equals` - Tests is a value is equal.
- `exists` - Tests does a value exists, true or false.
- `greater-than` - Tests is a value is greater than another value.
- `greater-than-equal` - Tests is a value is greater than or equal to another value.
- `less-than` - Tests is a value is less than another value.
- `less-than-equal` - Tests is a value is less than or equal to another value.

Parameters

The Condition parameters are described in the following sections.

equals parameters

The `equals` method parameters are:

Parameter	Type	Minimum length	Description	Configuration selection

`source`	any	`1`	The input to test.	Selector, String, Number, Boolean, Object, Array, Null
`value`	any	`1`	The value to test input against.	Selector, String, Number, Boolean, Object, Array, Null

exists parameters

The `exists` method parameter is:

Parameter	Type	Minimum length	Description	Configuration selection
`source`	any	-	The input to test.	Selector, String, Number, Boolean, Object, Array, Null

greater-than parameters

The `greater-than` method parameters are:

Parameter	Type	Minimum length	Description	Configuration selection
`source`	any	`1`	The input to test.	Selector, String, Number, Boolean, Object, Array, Null
`value`	any	`1`	The value to test input against.	Selector, String, Number, Boolean, Object, Array, Null

greater-than-equal parameters

The `greater-than-equal` method parameters are:

Parameter	Type	Minimum length	Description	Configuration selection
`source`	any	`1`	The input to test.	Selector, String, Number, Boolean, Object, Array, Null
`value`	any	`1`	The value to test input against.	Selector, String, Number, Boolean, Object, Array, Null

less-than parameters

The `less-than` method parameters are:

Parameter	Type	Minimum length	Description	Configuration selection
`source`	any	`1`	The input to test.	Selector, String, Number, Boolean, Object, Array, Null
`value`	any	`1`	The value to test input against.	Selector, String, Number, Boolean, Object, Array, Null

less-than-equal parameters

The `less-than-equal` method parameters are:

Parameter	Type	Minimum length	Description	Configuration selection
`source`	any	`1`	The input to test.	Selector, String, Number, Boolean, Object, Array, Null
`value`	any	`1`	The value to test input against.	Selector, String, Number, Boolean, Object, Array, Null

Outputs

The Condition flow-node outputs are described in the following sections.

equals outputs

The `equals` method outputs are:

Output	Type	Description	Save output value as:
`true`	boolean	The condition tested true.	`$.equals`
`false`	boolean	The condition tested false.	`$.equals`

exists outputs

The `exists` method outputs are:

Output	Type	Description	Save output value as:
`true`	boolean	The condition tested true.	`$.exists`
`false`	boolean	The condition tested false.	`$.exists`

greater-than outputs

The `greater-than` method outputs are:

Output	Type	Description	Save output value as:
`true`	boolean	The condition tested true.	`$.greaterThan`
`false`	boolean	The condition tested false.	`$.greaterThan`

greater-than-equal outputs

The `greater-than-equal` method outputs are:

Output	Type	Description	Save output value as:
`true`	boolean	The condition tested true.	`$.greaterThanEqual`
`false`	boolean	The condition tested false.	`$.greaterThanEqual`

less-than outputs

The `greater-than-equal` method outputs are:

Output	Type	Description	Save output value as:
`true`	boolean	The condition tested true.	`$.lessThan`
`false`	boolean	The condition tested false.	`$.lessThan`

less-than-equal outputs

The `less-than-equal` method outputs are:

Output	Type	Description	Save output value as:
`true`	boolean	The condition tested true.	`$.lessThanEqual`
`false`	boolean	The condition tested false.	`$.lessThanEqual`

Delay flow-node

The Delay flow-node method, parameter, and output are described in the following sections.

Method

The method for a Delay flow-node is:

- `delay` - Wait for a certain amount of time before continuing to the next flow-node.

Parameter

The Delay flow-node parameter is described in the following section.

delay parameters

The `delay` method parameter is:

Parameter	Type	Default	Description	Configuration selection
`delay`	integer	-	The length of delay, in milliseconds.	Selector, Number

Output

The Delay flow-node output is described in the following section.

delay outputs

The `delay` method output is:

Output	Type	Description	Save output value as:
`next`	integer	The delay is completed.	`$.delayed`

HTTP flow-node

Method

The method for a HTTP flow-node is:

- `setHTTPResponse`

Parameter

The HTTP flow-node parameter is described in the following section.

setHTTPResponse parameter

The `setHTTPResponse` method parameters are:

Parameter	Type	Minimum	Maximum	Description	Configuration selection
`status`	integer	`100`	`599`	-	Selector, Number
`body`	any	-	-	-	Selector, String, Number, Boolean, Object, Array, Null
`headers`	object	-	-	-	Selector, Object

The `body` and `headers` parameters can be enabled or disabled.

Output

The HTTP flow-node output is described in the following section.

setHTTPResponse output

The `setHTTPResponse` method does not have any configurable outputs.

Set Context flow-node

The Set Context flow-node method, parameter, and output are described in the following sections.

Method

The method for a Set Context flow-node is:

- `setContext`

Parameter

The Set Context flow-node parameter is described in the following section.

setContext parameter

The `setContext` method parameter is:

Parameter	Type	Default	Description	Configuration selection
`value`	any	-	-	Selector, String, Number, Boolean, Object, Array, Null

Output

The Set Context flow-node output is described in the following section.

setContext output

The `setContext` method output is:

Output	Type	Description	Save output value as:
`next`	any	-	-

Custom flow-nodes

Custom flow-nodes which are not built into the API Builder Console can be created and installed. Axway provides three custom flow-nodes with new projects and more planned for the future. The custom flow-node handlers are created and their methods, parameters, and outputs are defined using the Axway Flow SDK. For additional information, see Axway Flow SDK.

The custom flow-nodes currently provided with a new project are:

- Base64
- Compose
- JSON

Base64 flow-node

The flow-node methods, parameters, output for the Base64 flow-node are described in the following sections. The Base64 flow-node is created when NPM installs the `nodehandler-base64` code. The Base64 flow-node is included in the default application, but it can be removed.

Methods

The Base64 flow-node default methods are:

- `Decode` - Decodes base64 encoded data.
- `Encode` - Encodes base64 data.

Parameters

The Base64 flow-node parameters are described in the following sections.

Decode parameters

The `Decode` method parameters are:

Parameter	Type	Description	Configuration selection
`data`	string	The date to decode.	Selector, String
`as`	string	Expected decoded data format.	Selector, String

The `as` parameter can be enabled or disabled.

Encode parameters

The `Encode` method parameter is:

Parameter	Type	Description	Configuration selection

`data`	any	The data to encode.	Selector, String, Number, Boolean, Object, Array, Null

Outputs

The Base64 flow-node outputs are described in the following sections.

Decode outputs

The `Decode` method outputs are:

Output	Type	Description	Save output value as:
`Decoded`	string	The base64 decoded data.	`$.decoded`
`Error`	any	-	`$.b64error`

Encode outputs

The `Encode` method output is:

Output	Type	Description	Save output value as:
`Encoded`	string	The base64 encoded data.	`$.b64data`

Compose flow-node

The Compose flow-node methods, parameters, and outputs are described in the following sections. The Compose flow-node is created when NPM installs the `nodehandler-dot` code. The Compose flow-node is included in the default application, but it can be removed.

Methods

The default methods for a Compose flow-node are:

- `Format object` - Compose an object by evaluating a template. The evaluated template is JSON parsed and so must be a valid JSON encoded string.
- `Format string` - Compose a string by evaluating a template.

Parameters

The Compose flow-node parameters are described in the following sections.

Format object parameters

The `Format object` method parameters are:

Parameter	Type	Description	Configuration selection
`data`	object	The data to evaluate the template with. Use $ to access the entire context.	Selector, Object
`template`	string	The doT template.	Selector, String

Format string parameters

The `Format string` method parameters are:

Parameter	Type	Description	Configuration selection
`data`	object	The data to evaluate the template with. Use $ to access the entire context.	Selector, Object
`template`	string	The doT template.	Selector, String

Outputs

The Compose flow-node outputs are described in the following sections.

Format object outputs

The `Format object` method outputs are:

Output	Type	Description	Save output value as:
`Next`	any	-	`$.value`
`Error`	any	This output is triggered if the evaluated template is not a valid JSON string. The output value is the error object.	`$.error`

Format string outputs

The `Format string` method outputs are:

Output	Type	Description	Save output value as:
`Next`	any	-	`$.value`
`Error`	any	This output is triggered if the evaluated template is not valid. The output value is error object.	`$.error`

JSON flow-node

The JSON flow-node methods, parameters, and output are described in the following sections. The JSON flow-node is created when NPM installs the `nodehandler-json` code. The JSON flow-node is included in the default application, but it can be removed.

Methods

The default methods for a JSON flow-node are:

- `Parse` - The parse method parses a JSON string, constructing the JavaScript value or object described by the string.
- `Stringify` - The stringify method converts a JavaScript value to a JSON string.

Parameters

The JSON flow-node parameters are described in the following sections.

Parse parameters

The `Parse` method parameter is:

Parameter	Type	Description	Configuration selection
`json`	string	The JSON string to parse.	Selector, String

Stringify parameters

The `Stringify` method parameters are:

Parameter	Type	Description	Configuration selection
`value`	any	The value to convert to a JSON string.	Selector, String, Number, Boolean, Object, Array, Null
`space`	any	A string or number object that's used to insert white space into the output JSON string for readability purposes. If this is a number, it indicates the number of space characters to use as white space; this number is capped at 10. If this is a string, its maximum length is 10; the string is used as white space. If this parameter is not provided, no white space is used.	Selector, String, Number, Boolean, Object, Array, Null

The `space` parameter can be enabled or disabled.

Outputs

The JSON flow-node outputs are described in the following sections.

Parse outputs

The `Parse` method outputs are:

Output	Type	Description	Save output value as:
`Next`	any	-	`$.value`
`Error`	any	This output is triggered if the input is not a valid JSON string. The output value is the error object.	`$.error`

Stringify outputs

The `Stringify` method output is:

Output	Type	Description	Save output value as:
`Next`	string	-	`$.json`