

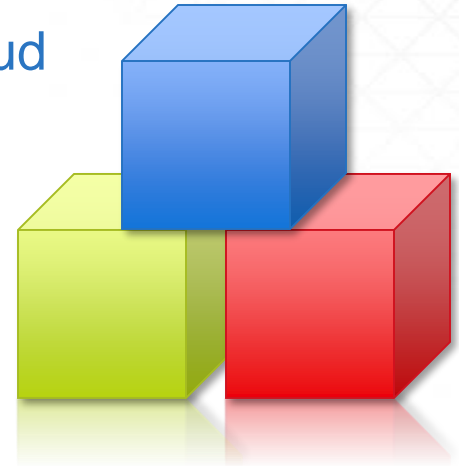


Building Integration Solutions with the Niagara JSON Toolkit

The Niagara JSON Toolkit is an Enabler

“JSON is the de-facto data encoding standard for the IoT.”

With the new Niagara JSON Toolkit, you have the building blocks to more simply use the Niagara platform to connect to IoT enterprise software applications, cloud services, databases, and third-party applications.



Accessing Valuable Niagara Data

The JSON Toolkit extends Niagara's rich object model and connectivity options by encoding point, alarm and history data into JSON format. Coupled with the MQTT driver this opens new possibilities for cloud connectivity and extracting value from the data Niagara can expose.



Unlocking new potential with JSON



REST
HTTP



```
{  
  "format": "JSON",  
  "uses": [  
    "UI / Charting / JavaScript",  
    "Web Service API's",  
    "Storage - NoSQL",  
    "Configuration files",  
    "IoT - Sensor Data / Control"  
  ]  
}
```




Couchbase

 mongoDB.



Example Use Cases

JSON messages have many uses, some typical use cases could include:

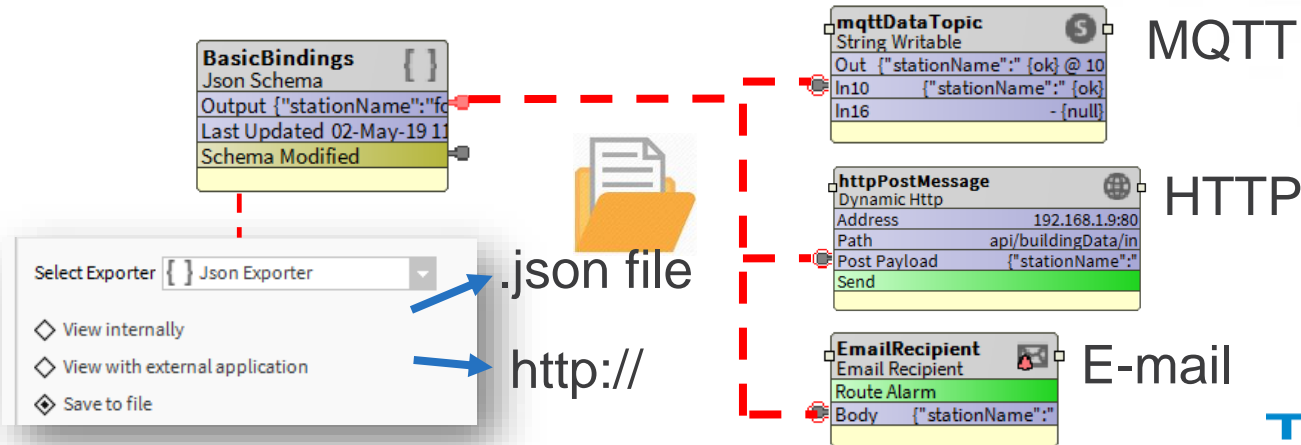
- Sending/Receiving JSON messages via an MQTT broker (for cloud connectivity)
- Sending/Receiving JSON messages via HTTP (for web services communication)
- Creating a JSON message to populate a chart in a web page
- Sending/Receiving JSON messages to control/monitor a local device
- Exporting station data in JSON format to save in a file

Does the toolkit only work with MQTT...?

No, the toolkit does not depend on mqtt.

The toolkit is intentionally decoupled from any single transport.

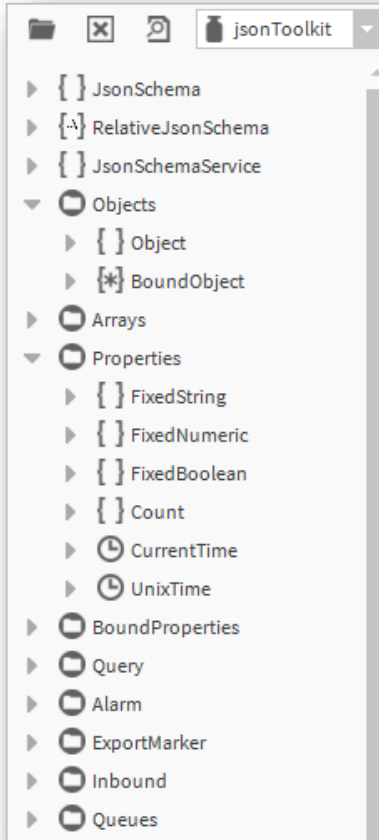
The JSON schema objects and message handlers may simply be linked to any transport mechanism you desire including MQTT String points.



What is the JSON Toolkit?

- ❑ The JSON Toolkit is a module which adds support for JSON messages within a Niagara station. This lets you create bespoke JSON string messages that can contain live point values, point tags and facet values, history data, alarm data and the results of queries such as bql or neql etc.
- ❑ Rather than mandating a fixed 'Niagara' JSON format, we allow the end user to build up JSON messages to suit their requirements.
- ❑ It also gives you tools for some handling of incoming JSON messages.

What is the JSON Toolkit?



There is a palette of tools that can be dragged into a station to construct a JSON 'schema' for data export

- ❑ Start with a JSONSchemaService
- ❑ Drag on a JsonSchema or RelativeJsonSchema
- ❑ Construct Objects, BoundObjects and Arrays with the different elements available


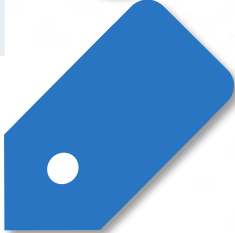
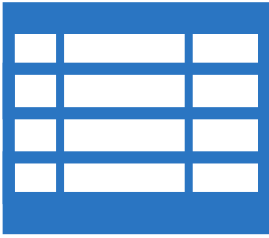
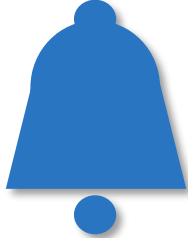

What is a JSON schema?

A JSON schema is a component within the JSON toolkit which allows you to build bespoke JSON strings.

You may build up a nested structure of objects, arrays and properties and include within them:

N	live point vales	alarm data	facet values
	history data	query results	tag values

B E



TRIDIUM

What does it look like in workbench?

The screenshot displays the 'BasicBindings (Json Schema)' interface. The central editor shows a JSON object with the following structure:

```
{
  "stationName": "jsonDemo",
  "myApiVersion": 3.1415,
  "messageId": 21480,
  "timestamp": "2019-02-25 12:15:32.084+0000",
  "whatIsJson": "json is a lightweight data-interchange format. It is easy for humans to read and",
  "numberWithHistory": {
    "out": 25.94,
    "in10": 25.94,
    "in16": 0,
    "nestedString": "Properties, Arrays and Objects may be nested within other Objects or Arrays!",
  },
  "selectedSlots": [

```

The left sidebar lists the following elements:

- Enabled: true
- License Status: {ok}
- Last Updated: 25-Feb-2019 12:15 PM GMT
- Config: Json Schema Config Folder
- Queries: Json Schema Query Folder
- root: Json Schema Object
 - stationName: Json Schema String Property
 - myApiVersion: Json Schema Numeric Property
 - messageld: Json Schema Count Property
 - timestamp: Json Schema Current Time Property
 - whatIsJson: Json Schema Bound Property
 - multipleSlots: Json Schema Bound Object

The right sidebar contains the following actions:

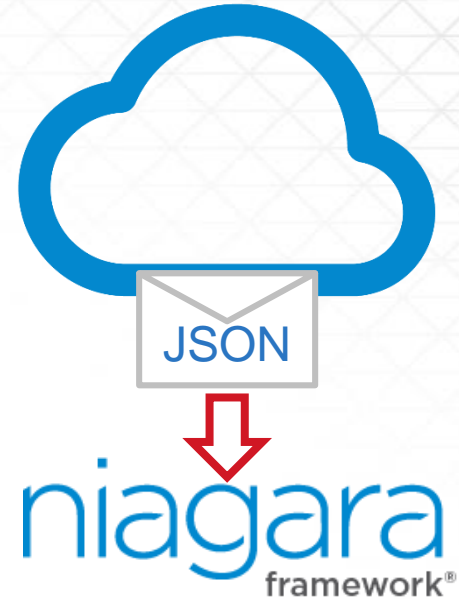
- Generate
- Copy
- Clear Output
- Output History
- Stats
- Indented Display

Drag the toolkit elements onto a Schema and preview the results

Does it handle incoming messages?

There are components to help users with incoming messages, allowing them to:

- route incoming JSON messages
- dissect them to select values from them
- acknowledge alarms
- set or change point values



Summary

Available now:

- Already added the **jsonToolkit** feature to all Demo licenses
- Refresh licenses to take advantage of this
- Download from Niagara Central Licensing: <http://bit.ly/jsont>
- Module contains the user guide with full details and examples

Production use:

- Purchase **DR-JSON** software option for JACE/portability controllers
- Or **DR-S-JSON** for Supervisors
- Flat price – not capacity based, but **requires active SMA**
- Contact your Tridium representative with pricing enquiries

