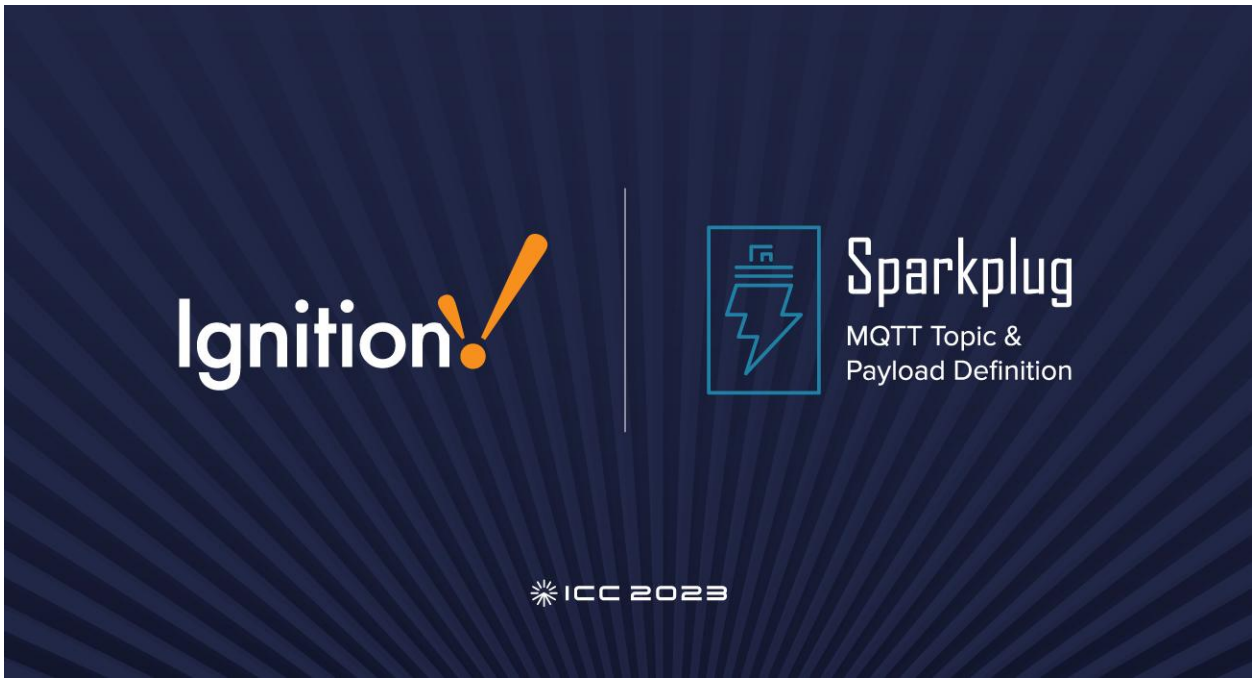
ICC 2023 Community-Powered Sparkplug Data Dash Instructions



At the Ignition Community Conference (ICC), we're always exploring ways to elevate both the technology and the community behind Ignition to new heights. That's why at ICC 2023, we're doing something completely new — our first-ever Sparkplug Data Dash!

Overview

We want to set up an Ignition Cloud Edition Perspective dashboard that displays data being published to a cloud MQTT server by community-built data models at the edge from around the world. The dashboard will present a map and listing of all contributors' data models. Plus, it will feature community-built Perspective templates that provide visualization of the published data models. And, to make it even cooler, the community-built UDTs and Perspective templates will be publicly available on the Ignition Exchange, so the community can benefit from these free resources to kickstart their next project.

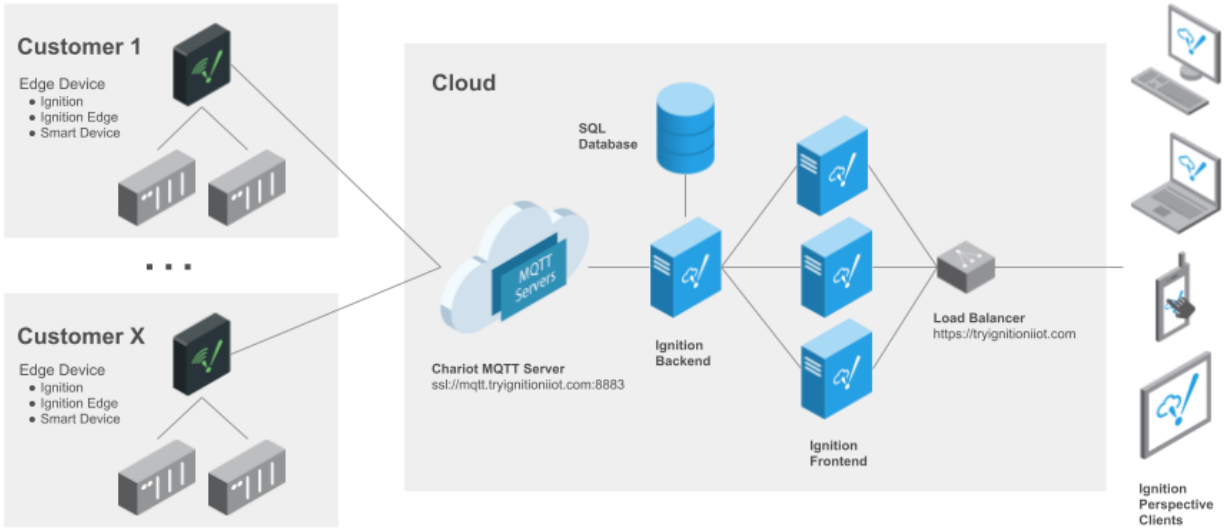
That is where you come in! We invite anyone in the Ignition community to participate, whether you're an end-user, integrator, vendor, or even a home-automation enthusiast using Ignition Maker Edition. All we ask is that you build one data model (UDT) and one corresponding Perspective template that can represent anything from a machine to a sensor to an oil well or even a thermostat. You can leverage Ignition or Ignition Edge to build and publish the UDT or a smart sensor that supports MQTT Sparkplug templates natively. If you are feeling adventurous, feel free to build multiple data models and templates.

Not to toot our own horn, but we think this is an amazing (and fun) way to show off the power of Ignition and MQTT Sparkplug through real data (with context) and practical examples of

edge-to-cloud solutions — and to engage as members of the Ignition community by building something really cool together. With your help, we can showcase data and a UNS (unified namespace) coming in from hundreds or thousands of edge devices around the world, all automatically discovered on a stunning cloud dashboard.

Sparkplug Data Dash Components & Architecture

There are three components to the data dash: data models built at the Edge, the cloud MQTT server data is published to, and an Ignition Cloud Edition application/dashboard showing off the data.



Data Models at the Edge

This part of the architecture is 100% provided by the Ignition Community. Here a community member will develop at least one data model (UDT) representing a machine or asset in the field and publish that to the MQTT server using Sparkplug. This will take full advantage of Sparkplug templating so the data models will be automatically discovered by client applications. In this case, Ignition Cloud Edition will discover the models and turn them into UDTs automatically. This allows for the data to be defined with context at the edge, maintaining a single source of truth for that data.

The data can be published by one of the following:

- Ignition Edge IIoT
- Standard Ignition using the MQTT Transmission Module
- Smart device/sensor that supports MQTT Sparkplug and templates (data models)

The data can come from a real PLC or simulator. We highly recommend using a sandbox environment or a test environment. We don't want production PLCs published to the cloud. The purpose is to show off the data model itself, not necessarily where the data came from. Data

can definitely be bi-directional, especially if you want to show off setpoints or control. We encourage you to set up a Docker container, VM or test device that will publish the data.

These data models will be automatically discovered in the Ignition Cloud Edition application. We would love to know what other configuration you would like to apply to the UDT in the cloud. For example, you may want history or alarms configured on some tags. If you are using Ignition at the edge, we will end up with 2 sets of UDTs. In this case, we may want to upload both sets of UDTs since the UDT at the edge will show how to map data from a PLC where the UDT in the cloud will show history or alarms.

Cloud MQTT Server

This part of the architecture will be provided by Inductive Automation and Cirrus Link Solutions. The MQTT Server is Cirrus Link's Chariot MQTT Server. The URL and connection information will be provided upon submission of the application. Each participant will have their own credentials and ACLs to avoid conflicts and to keep data secure.

Ignition Cloud Edition Dashboard

This part of the architecture will be provided by Inductive Automation. There will be a backend and frontend in the cloud. The backend will receive the data from the MQTT server and apply any configuration, such as history and alarming. The frontend will be highly available behind a load balancer to handle a high amount of Perspective client sessions.

Our Sales Engineering team is building the dashboard application. It will mainly consist of screens that show off the entire MQTT infrastructure and a map that shows where data is being published. When you click on a pin on the map, you will get more details on the device that is being published and see the discovered UDTs and their respective Perspective templates. The community member will also develop a Perspective template to show a visual representation of the data model. This template will be uploaded to the Ignition Cloud Edition dashboard and matched with the UDT that is published. That way the dashboard can show real HMIs or visualizations versus just raw values.

The Ignition application will be available at https://tryignitioniiot.com (coming in late August). We will contact you once the application is up and running.

Knowledge Required

You will need a general understanding of the following:

- Building UDTs in Ignition (or with a device that supports Sparkplug templates)
- Building Perspective templates
- Binding & Tag Indirection
- Basic Understanding of MQTT & Sparkplug
- MQTT Transmission Module (for publishing data if using Ignition)
- MQTT Engine Module (for testing)

How To Participate

Just complete these five easy steps:

1. Please fill out and submit this form. We just need some basic information about you and how you'd like to participate.
2. After you submit the form, we at IA will send you an email containing the credentials for the MQTT server so you can connect and publish data, as well as the instructions on how to get started and how to submit your Perspective template and data model configuration instructions (such as history or alarms required for your template) to us.
3. Build the UDT and the template according to the requirements below.
4. Publish the data to the broker, test it locally (using Ignition and the MQTT Engine Module), then send us, via email, the template along with any instructions for configuring the data model on the cloud Ignition server. That way we can test out your template on the real system prior to the event.
5. Upload your template to the Ignition Exchange prior to the event and send us the link to your resource. That way we can link your template in the cloud dashboard to your Ignition Exchange resource for recognition of your amazing work!

Disclaimer

Please don't upload any sensitive, confidential, or private data to the Sparkplug Data Dash project. Make sure your data model and Perspective template are scrubbed of any customer data.

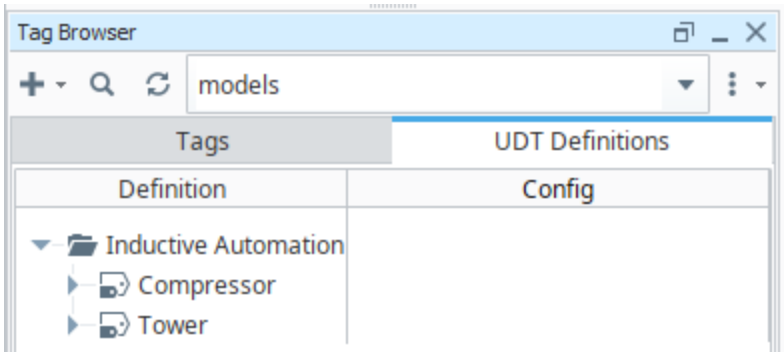
Requirements

Please read this section carefully. There will be hundreds of data models being published to the cloud and we want to set up proper naming structures and standards to avoid conflicts. Please follow the instructions for the UDT naming convention, MQTT Sparkplug topic namespace, and Perspective template requirements.

UDT Naming

All of the UDTs are automatically discovered in Ignition but will exist in a single tag provider called MQTT Engine. We want to avoid two different people publishing on the exact same UDT path. This can easily be avoided by having universally unique template names or paths. One easy way is to simply define your UDT definition or template under a path that includes your company name.

If you are using Ignition or Ignition Edge simply put the UDT inside of a folder representing your company. For example, here are 2 data models we created inside of a "Inductive Automation" folder:



If you are using another device that supports MQTT Sparkplug, please ensure the name of the data model will be universally unique. You can use a company name or serial number of the device.

MQTT Sparkplug Topic Namespace

MQTT Sparkplug defines a topic namespace to uniquely identify data coming from different vendors and edge devices. Again, we don't want two different people publishing data on the same topic namespace. This can easily be avoided by defining a namespace for this endeavor.

The MQTT Sparkplug namespace consists of three elements:

- **Group ID** - Provides for a logical grouping of Sparkplug Edge Nodes
- **Edge Node ID** - Uniquely identifies the Sparkplug Edge Node within the infrastructure
- **Device ID** - identifies a device attached (physically or logically) to the Sparkplug Edge Node

We require you to use specific group ids and edge node ids. However, you can publish on one or more devices (device ids) under your edge node. Please use the following Group ID/Edge Node ID:

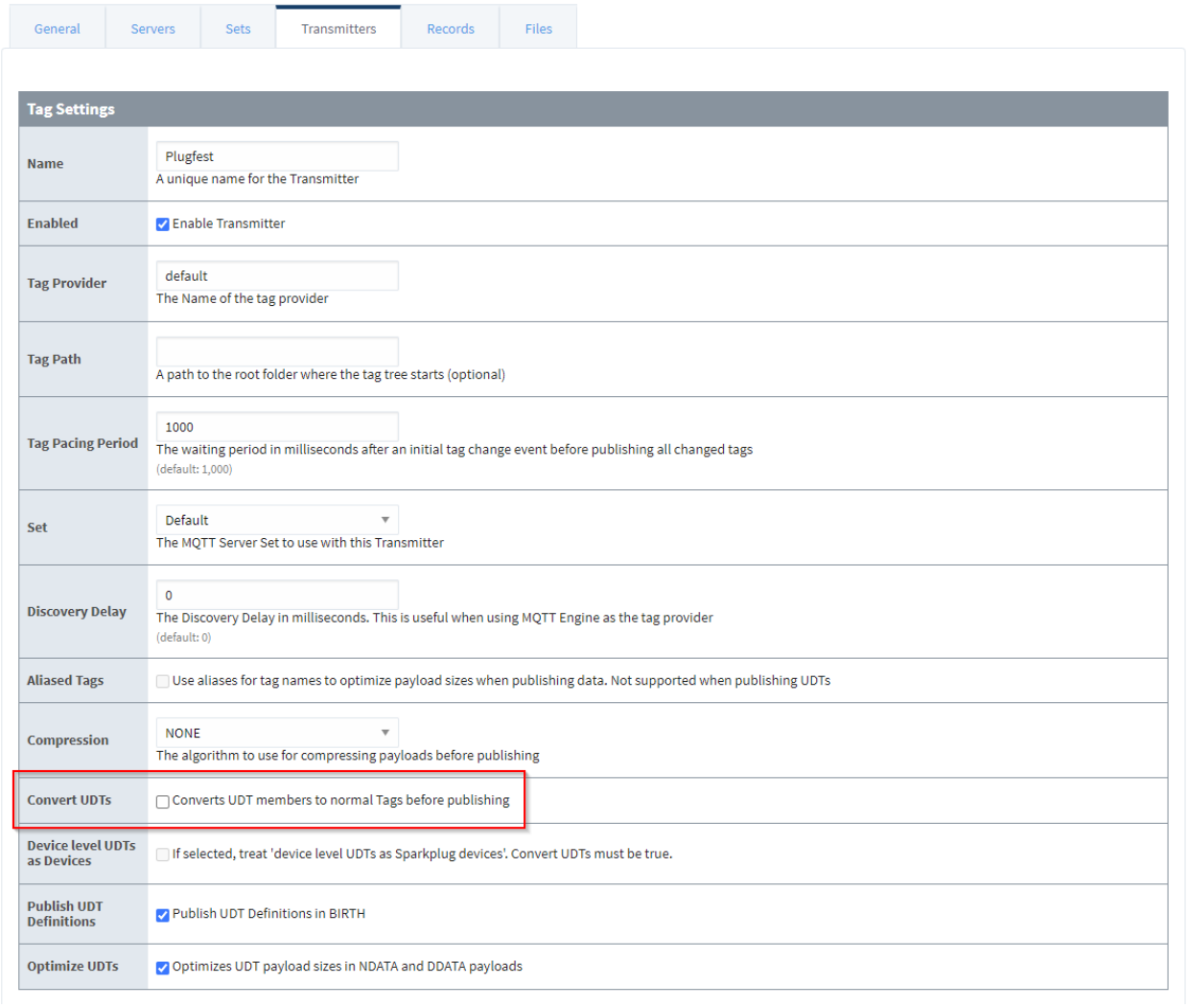
Group ID = Plugfest * *Same for everyone*
Edge Node ID = Your Company Name

Please use the same company name as you did with your UDT or template path. For example, if we wanted to publish data from one device the namespace would be:

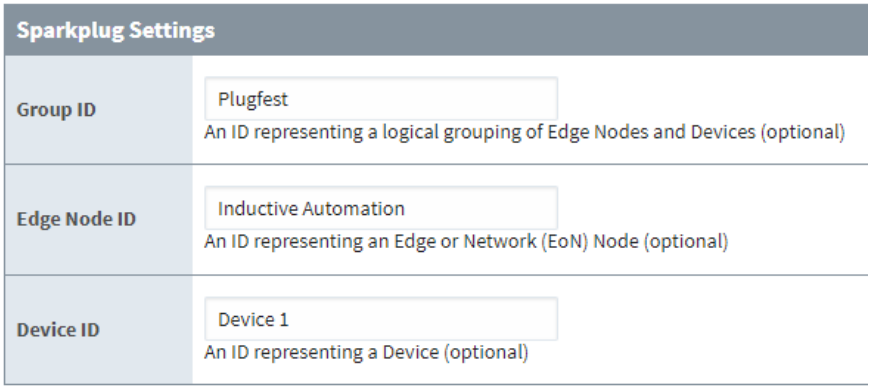
```
Plugfest/Inductive Automation/Device1
```

MQTT Transmission Settings

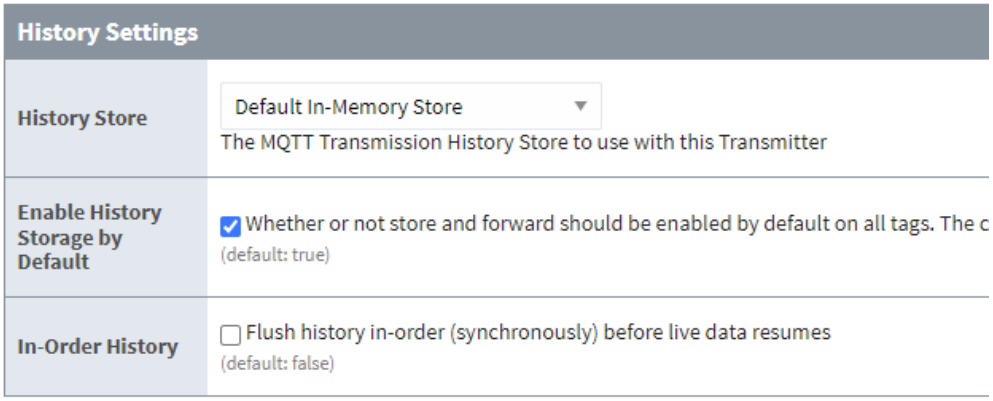
If you are using MQTT Transmission on Ignition Edge or standard Ignition, please ensure you uncheck the setting to convert UDTs into normal tags on your transmitter:



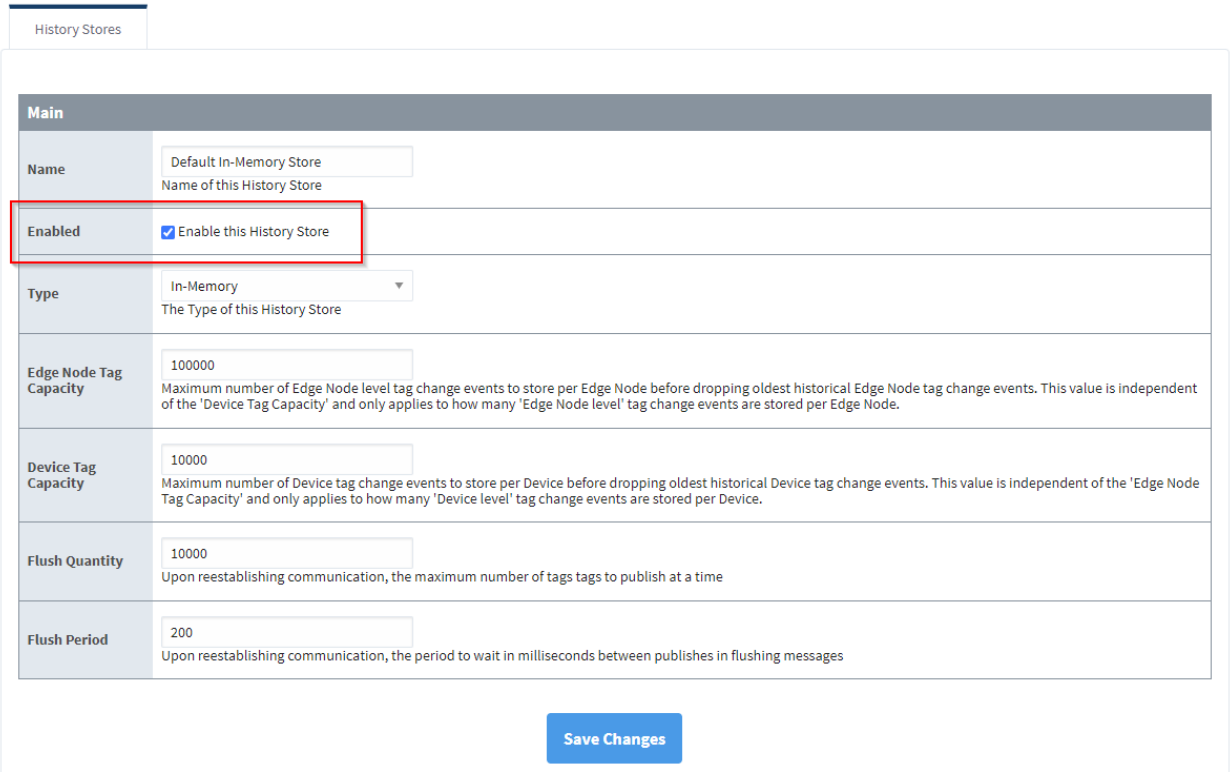
Make sure the transmitter uses the proper topic namespace under the advanced settings:



You can either specify the Device ID inside of the transmitter or use folders in the tag provider to represent multiple devices.
Lastly, it is recommended that you set up the history settings to cache data on the edge if you lose connection to the broker. You can do that on the transmitter under the history settings:



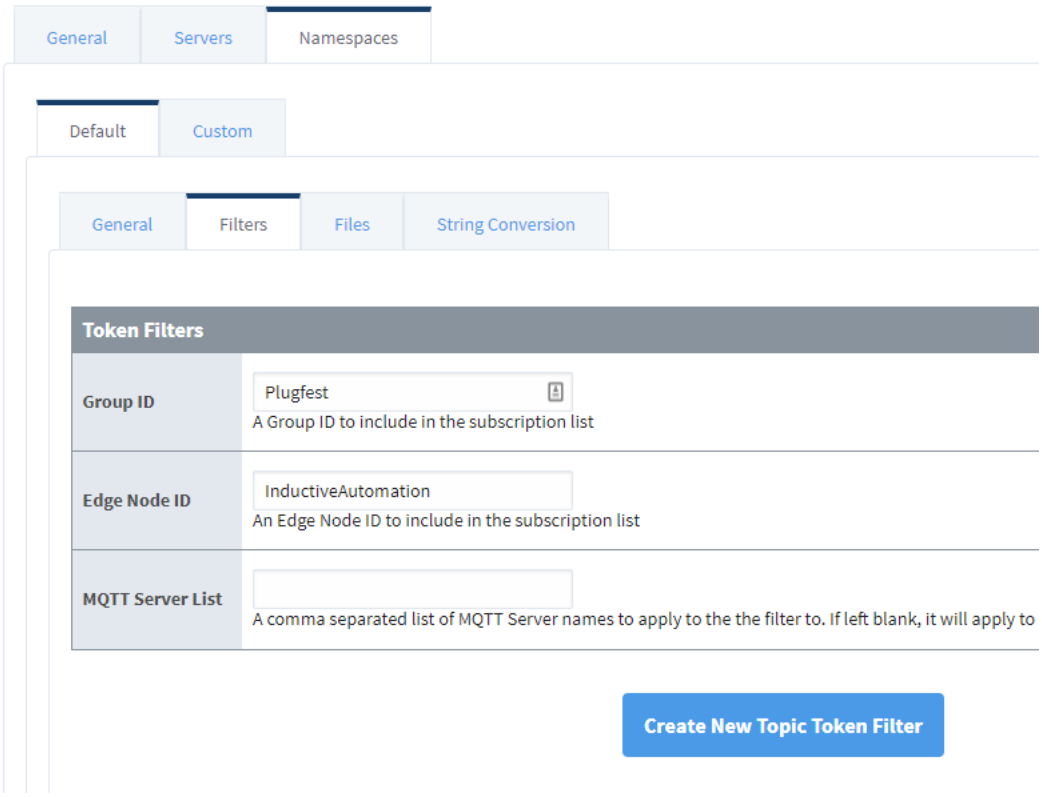
You also have to make sure that the “Default In-Memory Store” is enabled. You can do this by editing the store in the MQTT Transmission > History page:



MQTT Engine Settings

If you are using MQTT Engine to test our data model and Perspective template, you will need to filter Engine on only your Edge Node Id or Engine will not connect to the broker. Follow these instructions:

1. Go into the Ignition configuration section
2. Select MQTT Engine > Settings in the menu
3. Click on the Namespaces tab
4. Edit the Sparkplug B namespace
5. Click on the Filters tab
6. Click the “Create new Topic Token Filter…” link
7. Enter in Plugfest for the Group Id, your Edge Node Id (provided in the email), and leave MQTT Server List empty.



8. Click “Create New Topic Token Filter” and ensure your MQTT server connection is connected and working properly.
9. Please contact Travis if you run into any issues.

Perspective Template

One of the two assets you are providing is a visualization template in Perspective to show off your UDT or data model. The application will only be referencing a single view that you develop. However, that view can have embedded views. You just can’t provide a full Perspective project.

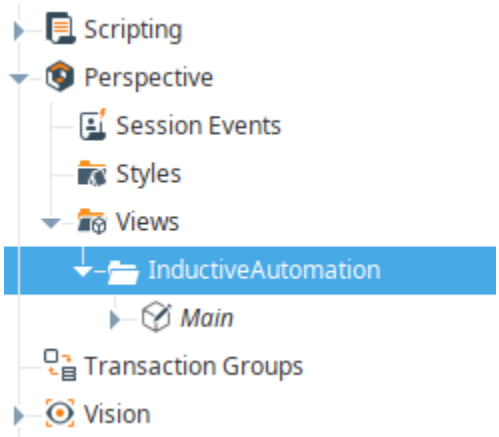
There are a few approaches to building a template in Perspective but we want to ensure the template is built a certain way so it can be easily integrated into the cloud dashboard application and to make it super useful for the community.

Naming Convention

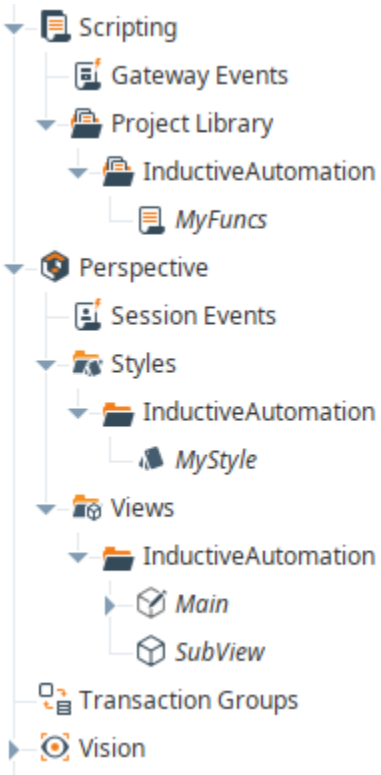
Your template could just consist of a single view or it could consist of multiple views, styles, and script libraries. We need to be able to import your resource into our cloud dashboard project and avoid conflicts with other contributors.

Follow these instructions:

1. Create a folder to hold your view that will be unique. It is recommended to use your Edge Node Id or company name. **Please ensure there are no spaces or special characters (underscores are fine).** The reason for no spaces is that Perspective styles (if you use them) cannot have spaces.



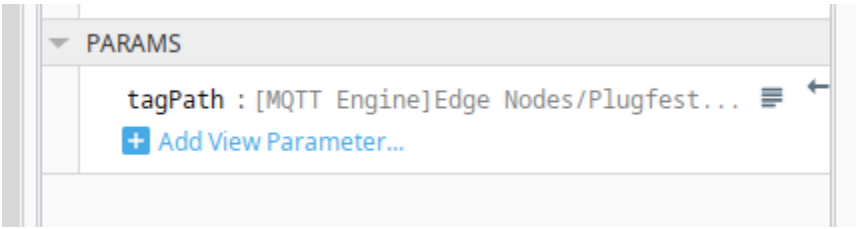
2. Make sure to name your primary view **Main**. See pic above. That way we know exactly which view you expect to be shown in the dashboard.
3. If your view requires embedded views, styles, or script libraries, please make sure to follow the same naming convention. Add a folder for each section with the same name. See the picture below.



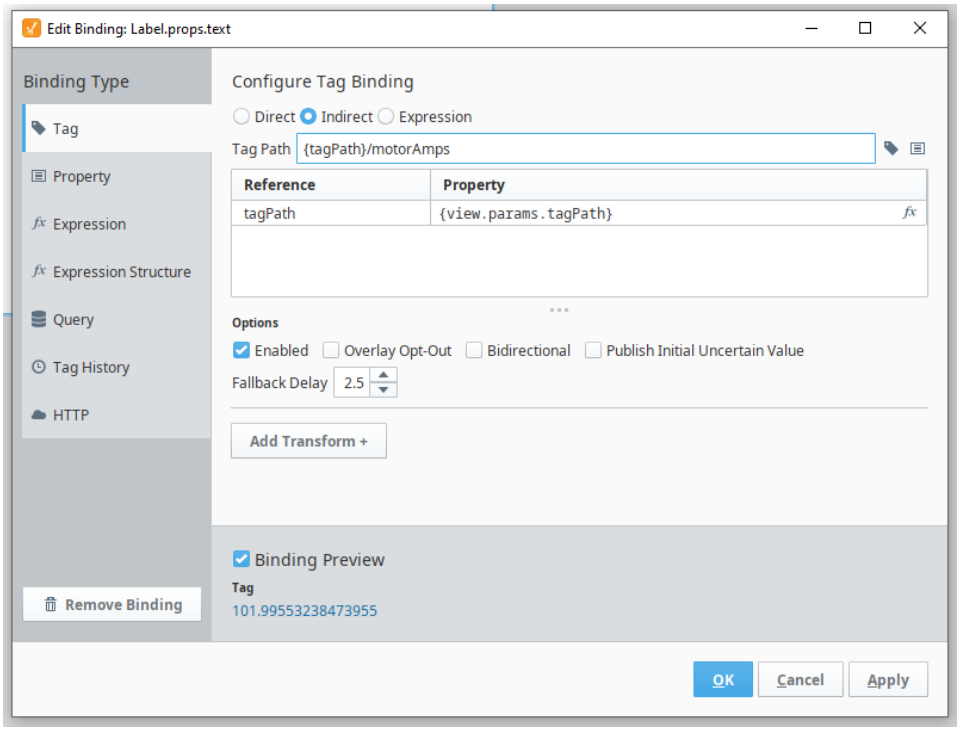
Template Parameter & Indirect Tag Bindings
Your template will have tag bindings to show data. We require that you build the template using tag indirection with a single parameter called tagPath. That way the template will work for you locally and for the cloud application, since the tag paths will be different. For the cloud application, the tags will be located in the MQTT Engine tag provider.

Follow these instructions:

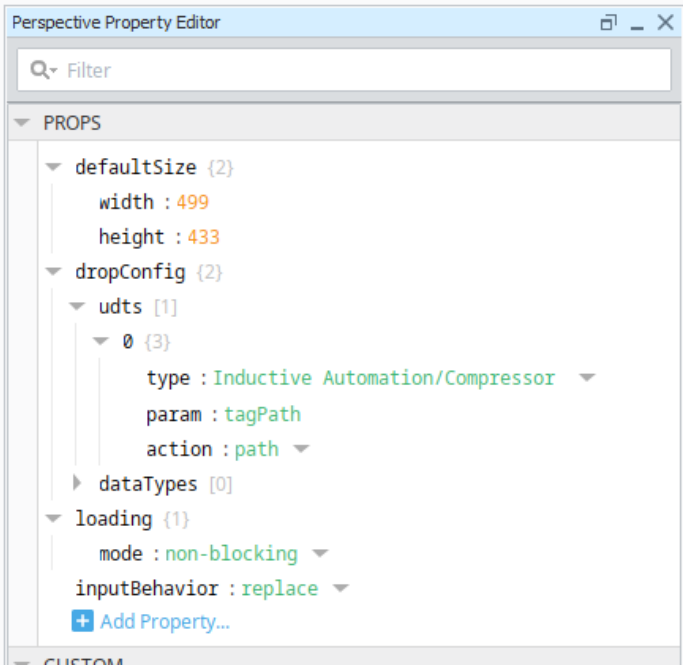
1. Add a template parameter called tagPath. This will hold the fully qualified tag path to the top level UDT. The reason is that we can use this template inside of any container, especially when building dynamic screens.



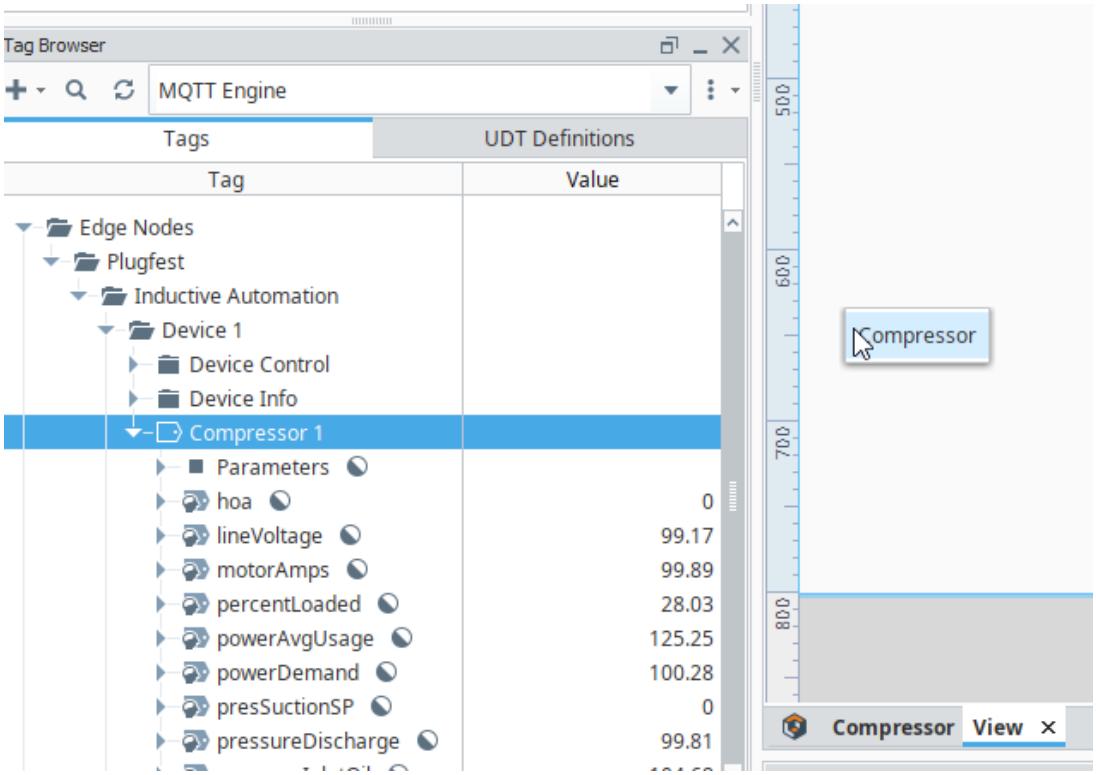
2. Make sure the tag bindings are using indirect bindings that leverage the tagPath parameter with the tag you are interested in. For example, if I want the motorAmps inside of the compressor UDT:



3. The last requirement is to configure the UDT drop target on the template. That way a user can easily drag and drop a UDT instance on a screen and the template will popup and set the parameter accordingly. Simply add a UDT object to the dropConfig > udts property on the view properties. Click on the view and add a new array element to the dropConfig > udts. Set the type to your UDT path. Set the param to “tagPath”. Set the action to “path”. It will look like this:



This will set the tagPath property to the fully qualified path of the tag when dropped on a screen.



Querying History in Your Template

If you would like to query history in your template, please make sure to use the proper fully qualified real time tag path using the tagPath parameter that is passed in. You don't want the template to be tied to a specific datasource or tag path. This will require you to use indirection on the tag history binding in Perspective. Follow these steps:

1. Create a custom **Array** property on the component called **tags**
2. Add one object to the array with path, aggregate, and alias values:

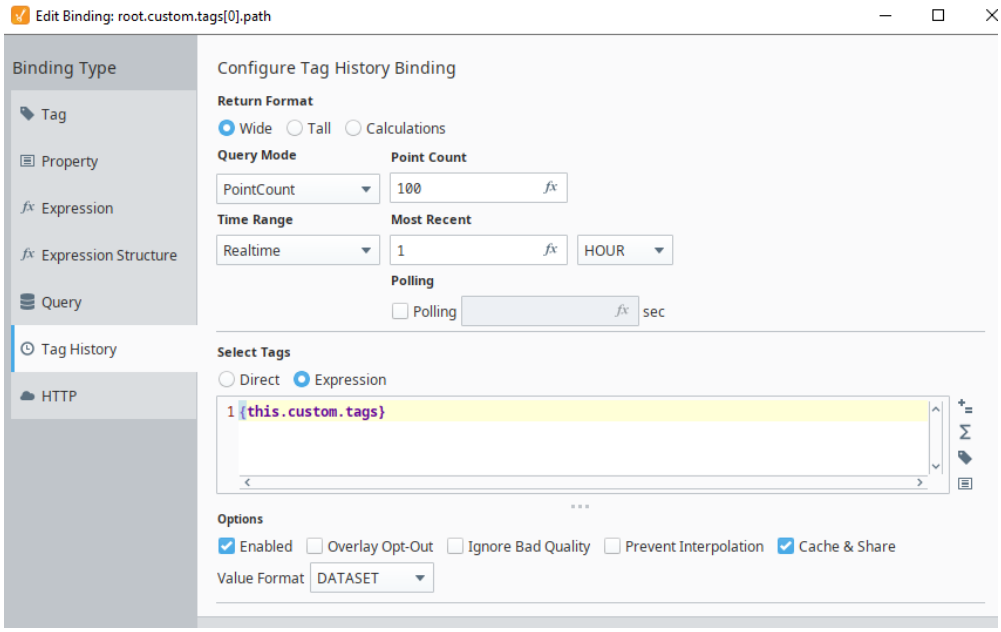
```
{
  "path": "[MQTT Engine]Edge Nodes/Plugfest/Inductive Automation/Device 1/Compressor 1/motorAmps",
  "aggregate": "MinMax",
  "alias": "motorAmps"
}
```

   You can set the aggregate and alias values to whatever you want.

3. Bind the path value to an expression that creates a dynamic path based on the tagPath parameter (realtime tag path):

```
{view.params.tagPath} + "/motorAmps"
```

4. Use the tags property in the tag history binding under expression:



This will allow the template to be used locally and centrally where the tag paths could be different.

Visit this page for more details on using dynamic paths: https://docs.inductiveautomation.com/display/DOC81/Tag+History+Bindings+in+Perspective#TagHistoryBindingsinPerspective-UsingDynamicTagPaths

Showing Alarm Status in Your Template

If you would like to show alarm status information in your template simply bind to the alarm tags meta information or use the alarm status component filtered to your tags or use the system.alarm.queryAlarmStatus function filtered to your alarms. Again, we want to use the tagPath parameter passed in to build a source path dynamically. For example, on the alarm status component you can bind the filters > active > conditions > source to the following expression:

```
"prov:MQTT Engine:/tag:" +
substring({view.params.tagPath}, max(0,
indexOf({view.params.tagPath}, "]") + 1))
+"*"
```

Ignition Licenses

If you plan on using Ignition Edge or standard Ignition to publish data, Inductive Automation will provide you with a trial license that will be good through the end of 2023. Please contact Travis Cox at travis@inductiveautomation.com to get a license.

Questions

Please contact Travis Cox at travis@inductiveautomation.com for any questions or concerns. Inductive Automation's Sales Engineering team is available for any assistance on building UDTs or Perspective templates. Please reach out.