Getting Started: Azure Injector Quick Start

Prerequisites

- Ignition with Google Cloud Injector Module installed
  - Review the Cirrus Link Module Installation documentation for installation details.
- Ignition Designer installed
  - Review the Inductive Automation documentation for Launching Designer against the Ignition gateway
- An existing Microsoft Azure account with an active IoT Hub and a registered device.
  - Documentation on creating an IoT Hub and registering an new device can be found here

Summary

This tutorial will provide step-by-step instructions for the following:

- Configuring the Azure Injector Module to connect to an existing Azure IoT Hub
- Publishing live Tag data and events to the connected Azure IoT Hub

Upon completion of this module you will have an Ignition Gateway connected and publishing live Tag data to an Azure IoT Hub.

Architecture

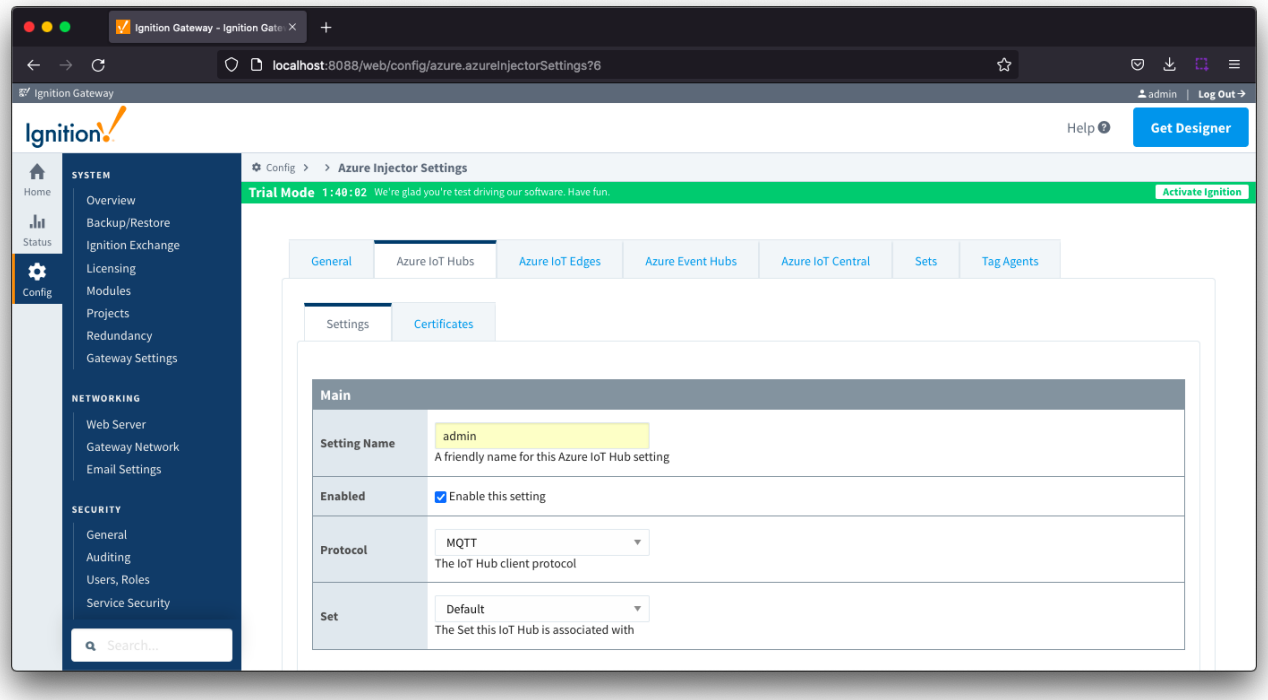
Tutorial

Step 1: Configure the Azure Injector Module

Once you have Ignition and the Azure Injector Module installed and running we can setup the configuration to connect to the active IoT Hub.

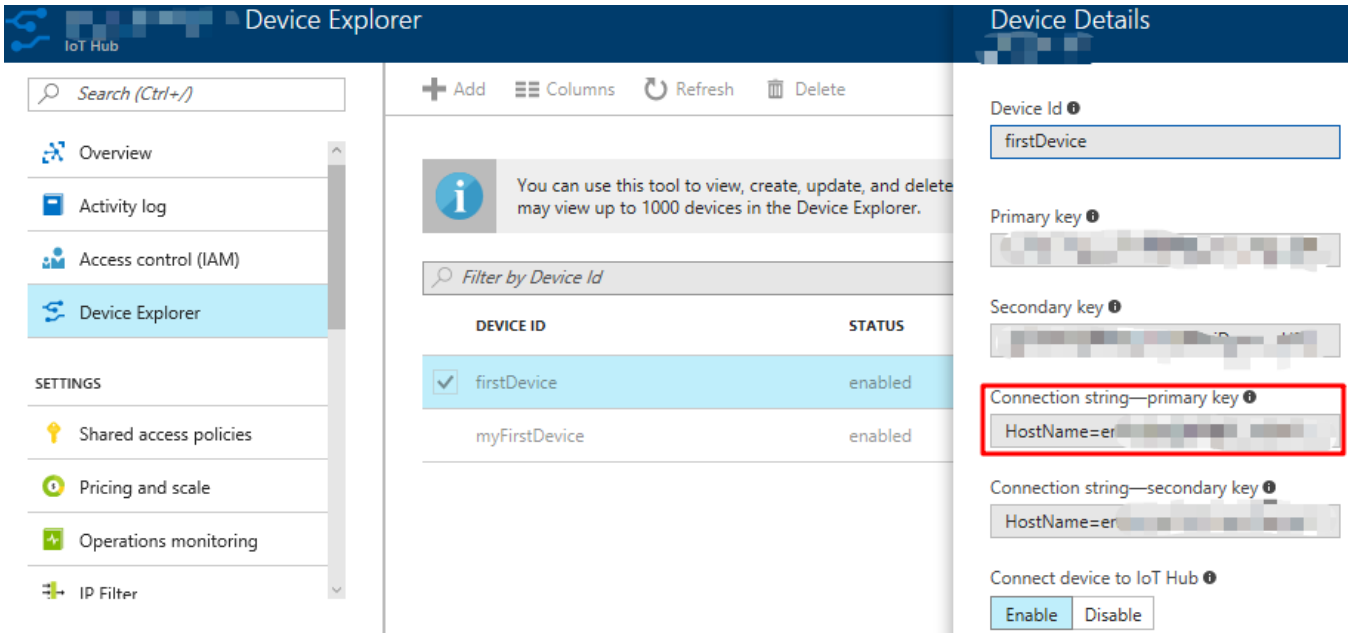
Review the Azure Injector Module configuration guide for more details on each tab

Navigate to the Azure Injector Module configuration section from the left side bar in the Ignition Gateway. From the Azure IoT Hubs tab, click on the "Create new Azure IoT Hub Setting..." link to bring up the following configuration form:

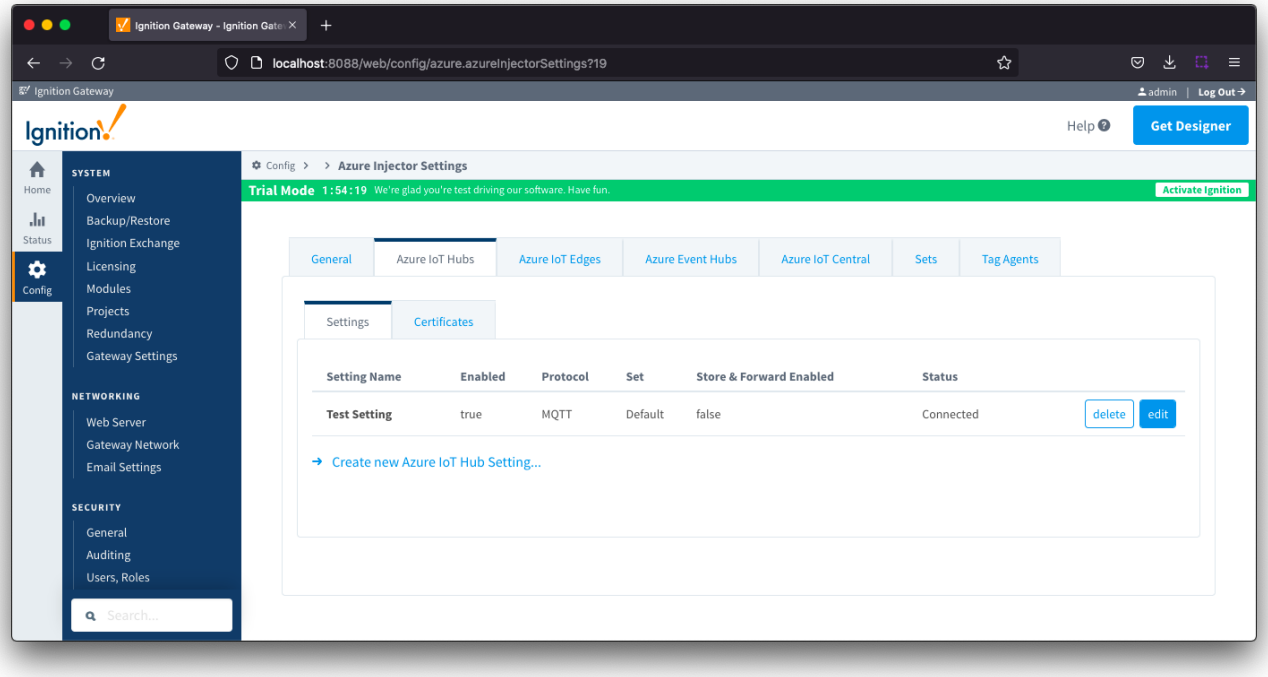


Set the following parameters:

- Setting Name
  - This can be any unique identifier. For this tutorial we will use "Test Setting".
- Password
  - This the IoT Hub Device Connection String from your Azure IoT Hub. This can be found by navigating to a provisioned Device within the Device Explorer of the IoT Hub that you wish to use.



Click on "Create New Azure IoT Hub Setting" to finish creating the new configuration setting.



Now the Azure Injector module is connected to the IoT Hub, we have to determine if there are are changes needed to the Tag Agent tab to be able to push data.

If you already have Ignition tags defined, for example from the Ignition OPC UA Server, then depending on the depth of your tag tree you may need to configure the Sparkplug Settings.

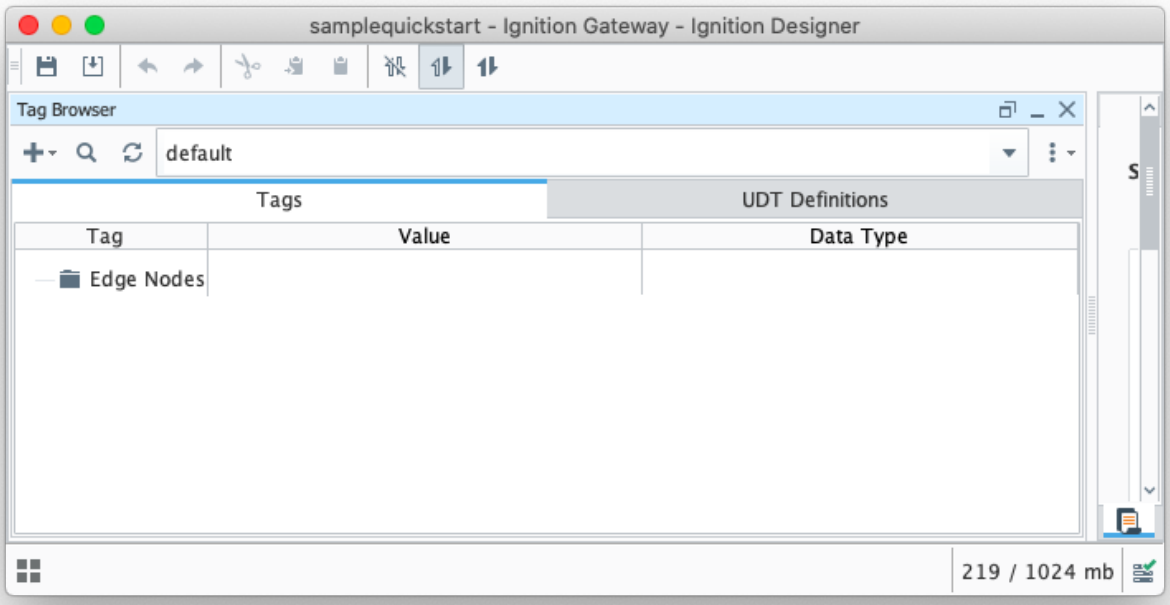
> Review the Cloud Injector Tag Agents and Tag Trees document which describes how Cloud Injector Agent configurations interact with Ignition tag trees to push messages and tag change events to the cloud service. It explains how tags get identified to be pushed as well as what specific 'topics' will be included with the messages. It also goes over some example configurations to show how the system will behave in different scenarios.

Once the Tag Agent is setup as needed, you can jump to Step 3: Publishing data.

If you do not have Ignition tags defined we will do that in the next step with a tag tree depth that requires no additional Sparkplug settings.

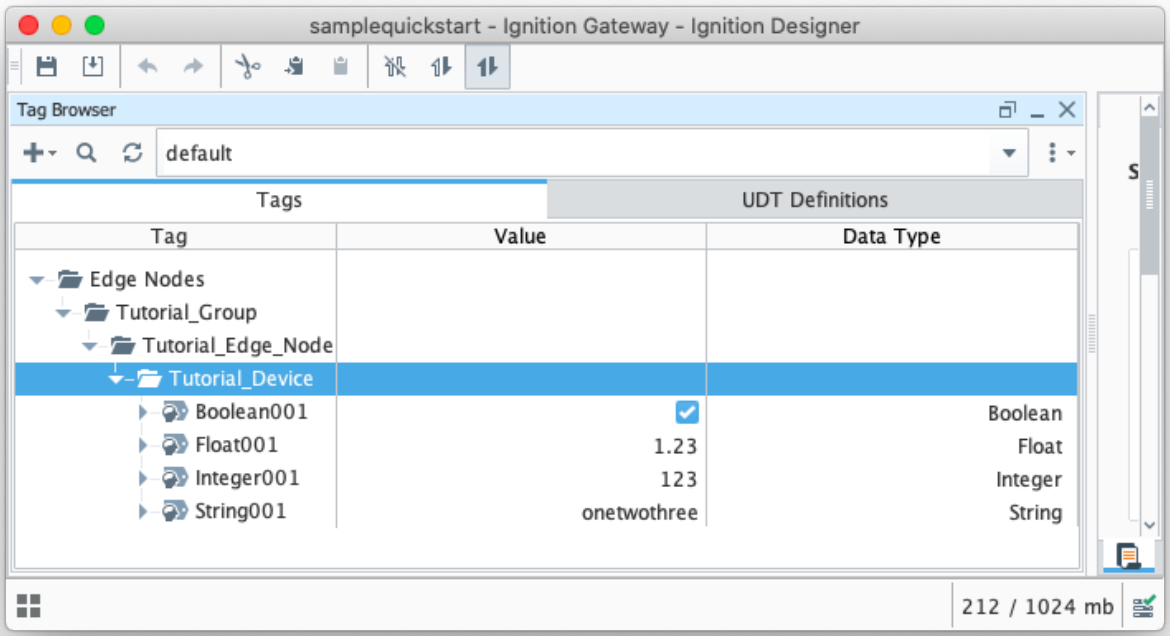
Step 2: Create tags to be published in Designer

When the Azure Injector module is installed in Ignition, an Edge Node folder is automatically created in the 'default' Ignition tag provider.



Create a tree structure under this folder as shown below with some memory tags - this folder structure creates the same hierarchy that is described in the Sparkplug B specification of Group ID, Edge ID, and Device ID.

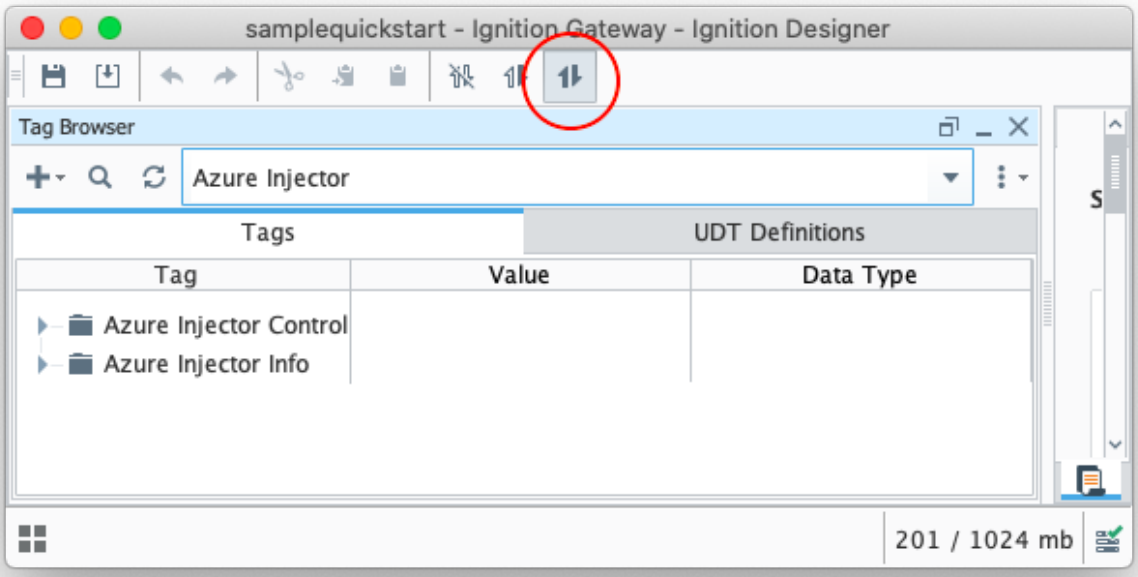
Refer to the Ignition Tag Browser and Creating Tags documentation for assistance in configuring Ignition tags



Step 3: Publishing data

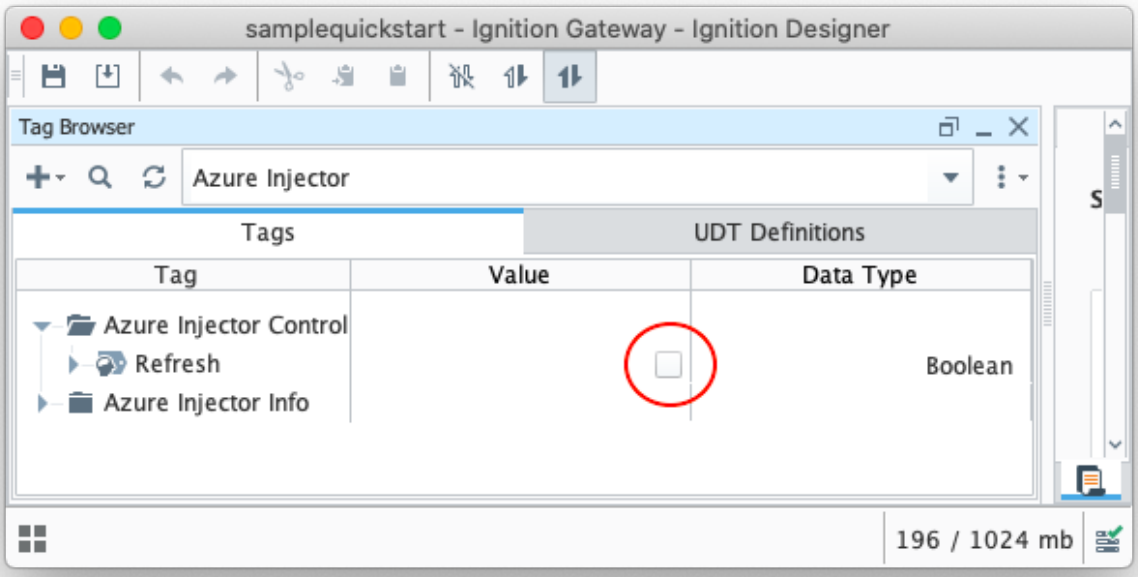
When the Azure Injector module is installed in Ignition, an Azure Injector tag provider is automatically created. This folder will contain both information tags about the module's version and state, as well as control tags for refreshing the module and Tag Agents.

Make sure that the Ignition Designer has read/write communications turned on by selecting the Project/Comm Read/Write button highlighted in the image below.



Review the Inductive Automation Designer documentation for additional assistance on setting the project communication mode

To refresh the default Tag Agent, open the folder "Azure Injector Control" and click on the Refresh Boolean. When this happens, the Tag Agent will scan the "Edge Nodes" folder and find the new Memory Tags that we have created, construct JSON payloads representing those tags with their current values and publish the payload to the Azure IoT Hub that we have configured.



The Boolean tag will not change to true. This is really a one-shot and as a result, the tag will not change to true.

The Azure Injector Tag Agent will publish two JSON payloads to the Azure IoT hub. The format of these messages closely follows the Sparkplug B Specification's payload structure.

The first payload represents the Edge Node and will contain the following:

- The Sparkplug elements: Namespace, Group ID, Edge Node ID. They will be grouped under "topic".
- A "timestamp" for when the payload was constructed.
- A "bdSeq" sequence number to track the "session" of the Tag Agent.
- Any Edge Node tags defined in the "Tutorial Edge Node" folder (in our example we have none).

It will look something like this:

First Payload


```
{
  "topic": {
    "namespace": "spBv1.0",
    "groupId": "Tutorial Group",
    "edgeNodeId": "Tutorial Edge Node"
  },
  "payload": {
    "timestamp": 1504739061495,
    "metrics": [
      {
        "name": "bdSeq",
        "timestamp": 1504739061495,
        "dataType": "Int64",
        "value": 0
      }
    ],
    "seq": 0
  }
}
```


The second payload represents the Device and will contain the following:

- The Sparkplug elements: Namespace, Group ID, Edge Node ID, Device ID. They will be grouped under "topic".
- A "timestamp" for when the payload was constructed.
- Any Device tags defined in the "Tutorial Device" folder.

It will look something like this:

Second Payload


```
{
  "topic": {
    "namespace": "spBv1.0",
    "groupId": "Tutorial Group",
    "edgeNodeId": "Tutorial Edge Node",
    "deviceId": "Tutorial Device"
  },
  "payload": {
    "timestamp": 1504739061501,
    "metrics": [
      {
        "name": "Boolean001",
        "timestamp": 1504739061546,
        "dataType": "Boolean",
        "properties": {
          "Quality": {
            "type": "Int32",
            "value": 192
          }
        },
        "value": true
      },
      {
        "name": "String001",
        "timestamp": 1504739061546,
        "dataType": "String",
        "properties": {
          "Quality": {
            "type": "Int32",
            "value": 192
          }
        },
        "value": "onetwothree"
      },
      {
        "name": "Integer001",
        "timestamp": 1504739061546,
        "dataType": "Int32",
        "properties": {
          "Quality": {
            "type": "Int32",
            "value": 192
          }
        },
        "value": 123
      },
      {
        "name": "Float001",
        "timestamp": 1504739061546,
        "dataType": "Float",
        "properties": {
          "Quality": {
            "type": "Int32",
            "value": 192
          }
        },
        "value": 1.23
      }
    ],
    "seq": 1
  }
}
```

Step 4: Use Ignition Designer to Publish Tag Data (Live Tag Values Changes)

Now we can change the values of the new Memory Tags and generate payloads that contain the Tag change events. Click on the value of the "Boolean001" Memory Tag to change it's value.

This will result in the following payload to be constructed to represent this Tag change event and pushed to the Azure IoT Hub:

Change Event Payload

```
{
  "topic": {
    "namespace": "spBv1.0",
    "groupId": "Tutorial Group",
    "edgeNodeId": "Tutorial Edge Node",
    "deviceId": "Tutorial Device"
  },
  "payload": {
    "timestamp": 1504740884529,
    "metrics": [
      {
        "name": "Boolean001",
        "timestamp": 1504740883526,
        "dataType": "Boolean",
        "value": false
      }
    ],
    "seq": 2
  }
}
```

Step 5: Azure IoT Hub Applications

It is beyond the scope of this tutorial to show how to design an application in Azure to handle the payloads as they are pushed in to the Azure IoT Hub. For additional information on developing applications to consume this data see https://docs.microsoft.com/en-us/azure/.

Additional Resources

- Inductive Automation's Ignition download with free trial
  - Current Ignition Release
- Cirrus Link Solutions Modules for Ignition
  - Ignition Strategic Partner Modules
- Support questions
  - Check out the Cirrus Link Forum: https://forum.cirrus-link.com/
  - Contact support: support@cirrus-link.com
- Sales questions
  - Email: sales@cirrus-link.com
  - Phone: +1 (844) 924-7787
- About Cirrus Link
  - https://www.cirrus-link.com/about-us/