Event Ingestion in Data Platform

Estimated time to read: 5 minutes

Event ingestion is the process of publishing events for immediate consumption and/or storage in the **Data Platform** backend database. Events can be streamed in real time, where each event is processed as the source emits it, or ingested in batches.

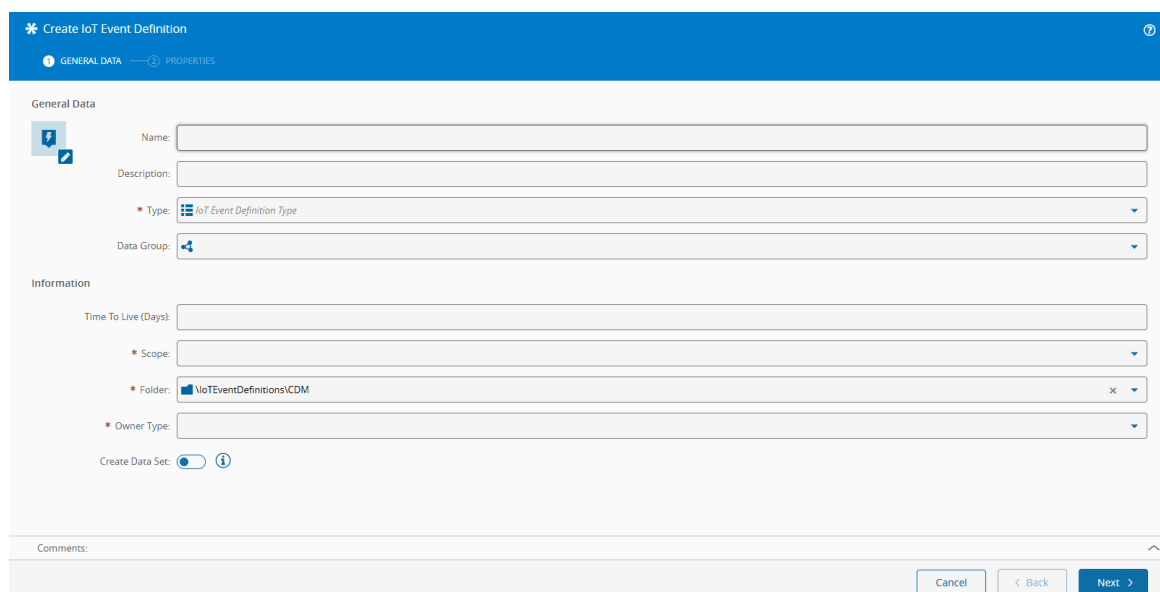This tutorial will demonstrate how to use a basic **Data Platform** event to ingest data.

Overview

In this example we will create an event that receives values from different sensors on a shop floor, and show how that data is made available as it is being published.
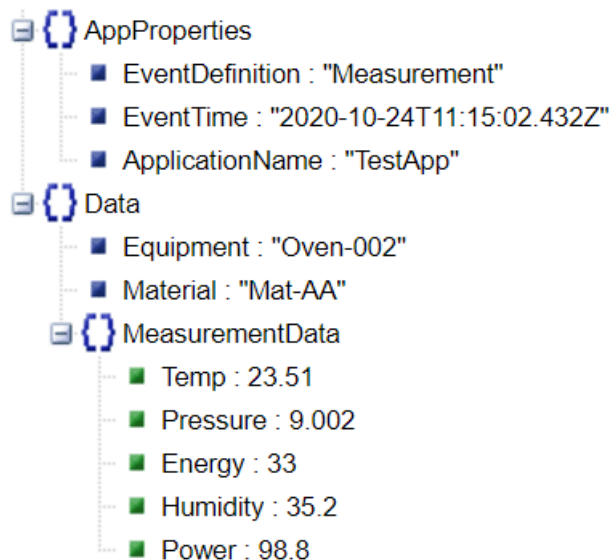
Creating an IoT Event Definition

In the **Data Platform** page group of the main menu, navigate to the **IoT Event Definitions** tile and create a new **IoT Event Definition**:

- Enter a name - bear in mind that the name will identify the **IoT Event** throughout the **Data Platform** infrastructure, so it should be a meaningful name for later reference.

- Set the Type to **Data Platform**.

- Set the Scope to **Data Platform**.

- Select the desired Owner Type.

- Set the **Create Data Set** option to `true` - if set to `true`, a dataset will be created for the data ingested with this event; always set this option to `true` if direct access to the data will be required in the future (example: data collections).
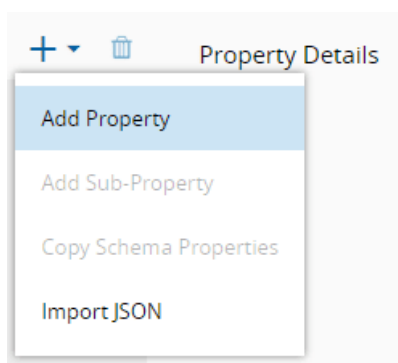

We will create an event with the following structure:

```
□ { } AppProperties
    ■ EventDefinition : "Measurement"
    ■ EventTime : "2020-10-24T11:15:02.432Z"
    ■ ApplicationName : "TestApp"
□ { } Data
    ■ Equipment : "Oven-002"
    ■ Material : "Mat-AA"
    □ { } MeasurementData
        ■ Temp : 23.51
        ■ Pressure : 9.002
        ■ Energy : 33
        ■ Humidity : 35.2
        ■ Power : 98.8
```

AppProperties is a set of properties that the publisher of the events (example: the client that will produce the events) should add to identify itself, while **Data** is the actual event payload.

Adding properties to the event definition can be done manually or by importing a JSON structure such as the one shown above. The system will infer the data types from the actual values and generate a property for each of the fields.

```
+ ▾    🗑        Property Details
┌─────────────────────────────┐
│ Add Property                │
│ Add Sub-Property            │
│ Copy Schema Properties      │
│ Import JSON                 │
└─────────────────────────────┘
```
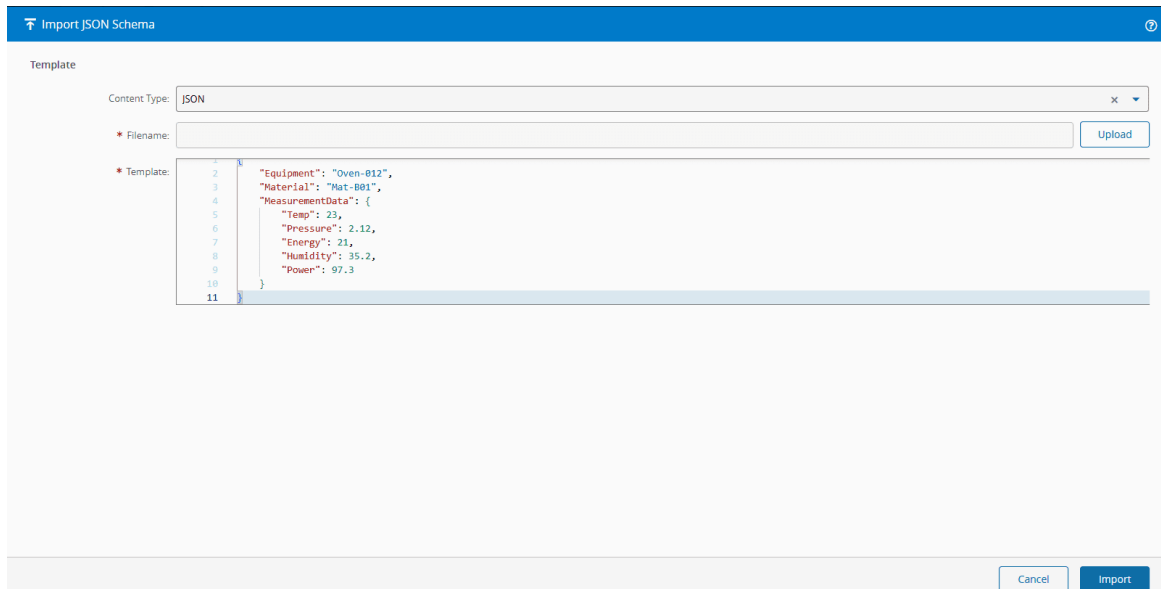
Several properties are essential for the proper functioning of the event ingestion, so specify for each property:

- **Name** - name of the property, that must matches the property in the JSON document that represents the event.

- **Array** - indicates if the property is an array of objects.

- **Mandatory** - indicates if the property is mandatory. If `true`, this information will be used to validate all the arriving events.

- **Indexed** - indicates if the property should be indexed. If `true`, the property will become part of the key that uniquely identifies an event; events with the same key will be automatically deduplicated; if the client sends many events with the same key, then only the event that was sent last will be saved in the data set.

- **Data Type** - the data type of the property. Type **IoTSchema** allows you to create sub-groups of properties.

- **Default Value** - specifies the default value for the property. If the property is not present in the json of the event that was posted, the default value will be set automatically.
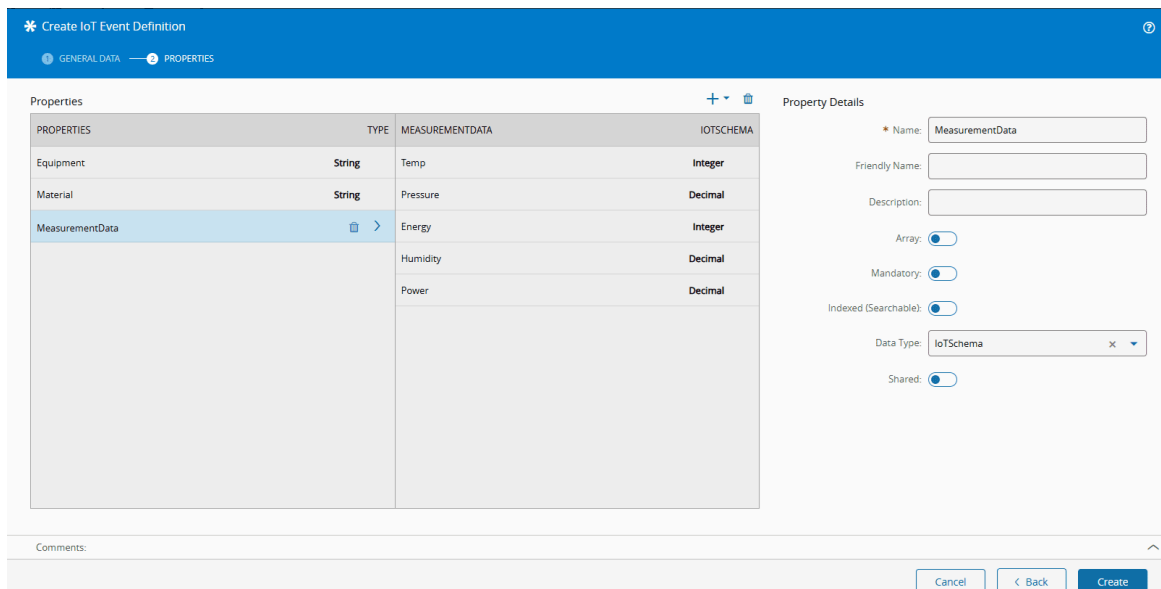
> **ℹ Info**
>
> For more information, see Create IoT Event Definition.

As previously mentioned, rather than adding the properties one by one, it is possible to import the structure by selecting **Import JSON** and providing a sample JSON:
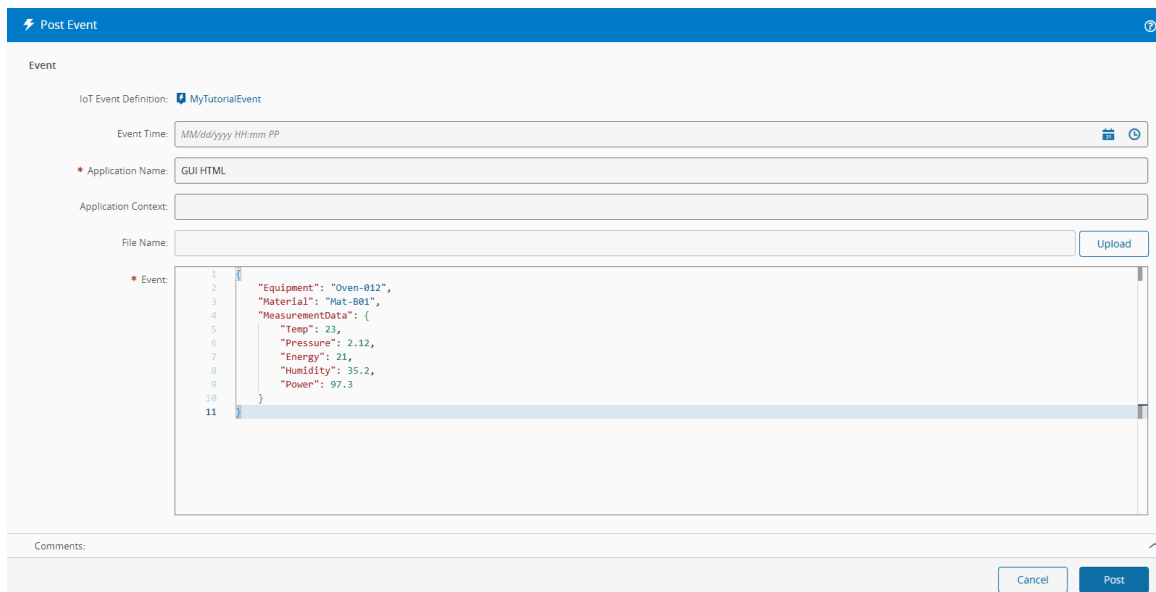

After creating the properties manually or importing from a JSON file, the **IoT Event Definition** can be created.

When the event is created, it is possible to start posting events using the `Post Event` host api. We can simulate this by using the **Post** button in the **IoT Events** page. Simply navigate to that page by going to the **IoT Event Definitions** page (which you can access from the **Data Platform** page group), and select the **IoT Event** you created from the list. Then, select the **Post** button on the top ribbon, and type the JSON document for the event you want to post:
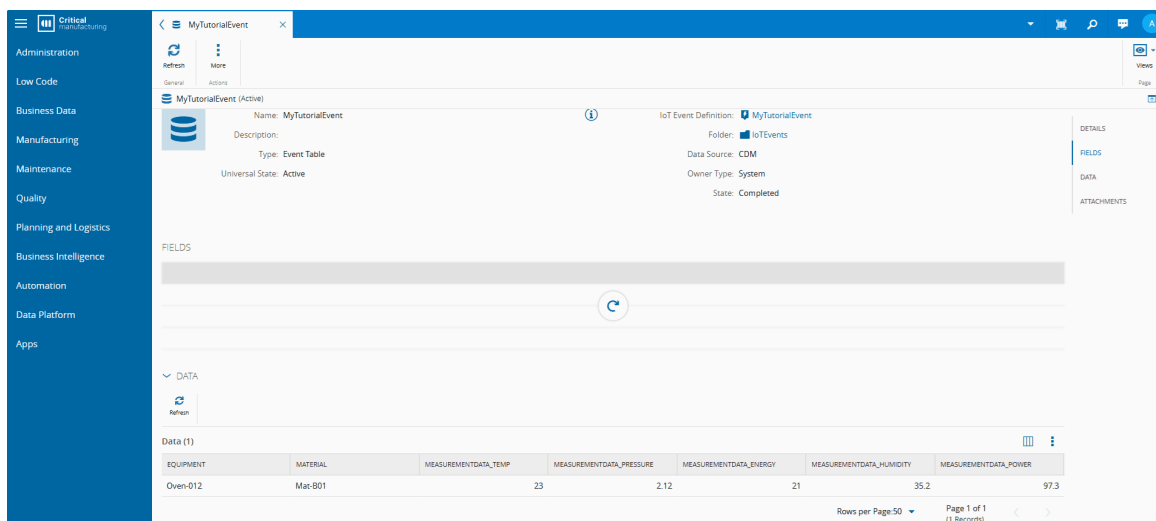
Since the **Create Data Set** option was set to `true` when we created the **IoT Event Definition**, once this event is posted we can immediately see the data in the corresponding **Data Set**. Simply select the **Data Sets** tile in the **Data Platform** page group, then navigate the **IoTEvents** folder to find the data set for the event. The data set name and folder will match the name and folder of your iot event.

When you find the data set, simply select the **Data** section and you'll be able to see the data for the event that you just posted:

This data is now available to be accessed in multiple ways (OData, Grafana Dashboards, etc.). Thus, if you have a client continuously posting events using the `Post Event` host api, **Data Platform** will make this data immediately available for consumption.