|  |  |
| --- | --- |
| **Example Title:** | Influx 2.x extension with trender mashup |
| **Description:** | This example application uses the Influx 2 Extension to query via the flux language the InfluxDB. |
| **Key Concepts:** | Extension, Flux QL |
| **Latest TW Version Used:** | ThingWorx 9.3.1 |
| **Extensions Required:** | Flux2Connector extension is required |
| **Original Tech Sales Engineers:** | Tobias Wobben (v1.0)  Costin Badici  Antonia Trippner |
| **Example Revision:** | 1. First Release – 5/2/2022 |

**Overview:**

This example application uses the flux2connector to query the InfluxDB 2 OSS database with the ThingWorx Platform, including most of the features of the Influx DB function. The used query is Flux language.

Link to Flux language description:  
<https://docs.influxdata.com/influxdb/cloud/query-data/flux/>

**Entities Provided:**

1. An entities export has been provided in the Entities\_TrenderDemo.xml file
2. An extension flux2connector.zip

**Java libs used for the extension:**

<https://github.com/influxdata/influxdb-client-java/releases/tag/v5.0.0>

**Shared Demo Instance Provided:**

The following PTC Cloud Portal instance has been shared out:

<https://pp-2204260714pq.portal.ptc.io/Thingworx>

Web App Login: Administrator / ch@ngMe@F1rstL0gon

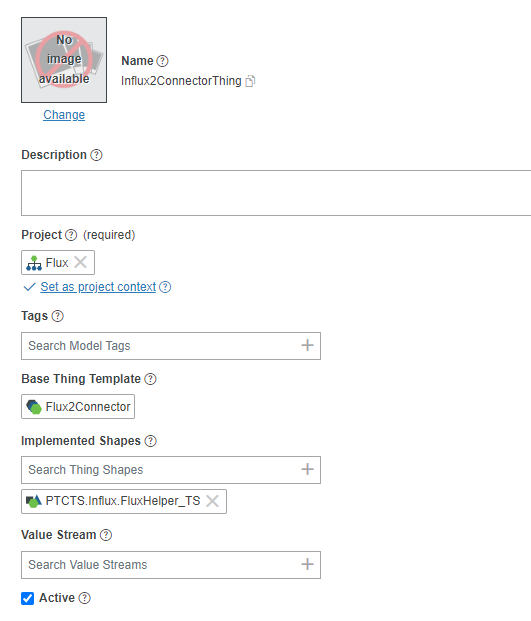
The environment notes provide the link to the runtime application and all necessary login details.

**Example Thing / Services description:**

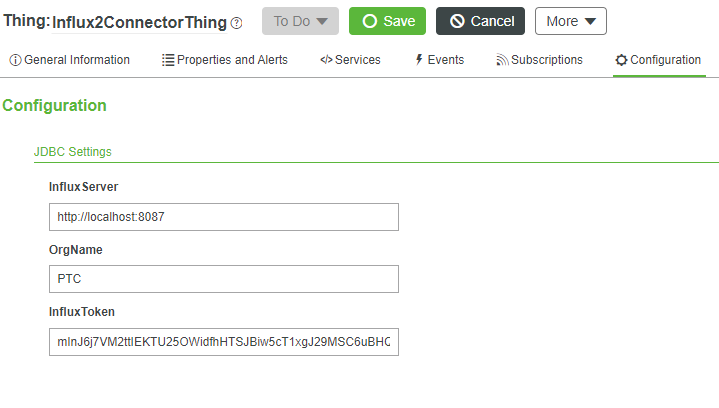
Influx2ConnectorThing

BaseThing: Flux2Connector (extension)

Implemented Shape: PTCTS.Influx.FluxHelper\_TS (Javascript add-on services)



**Configuration:**



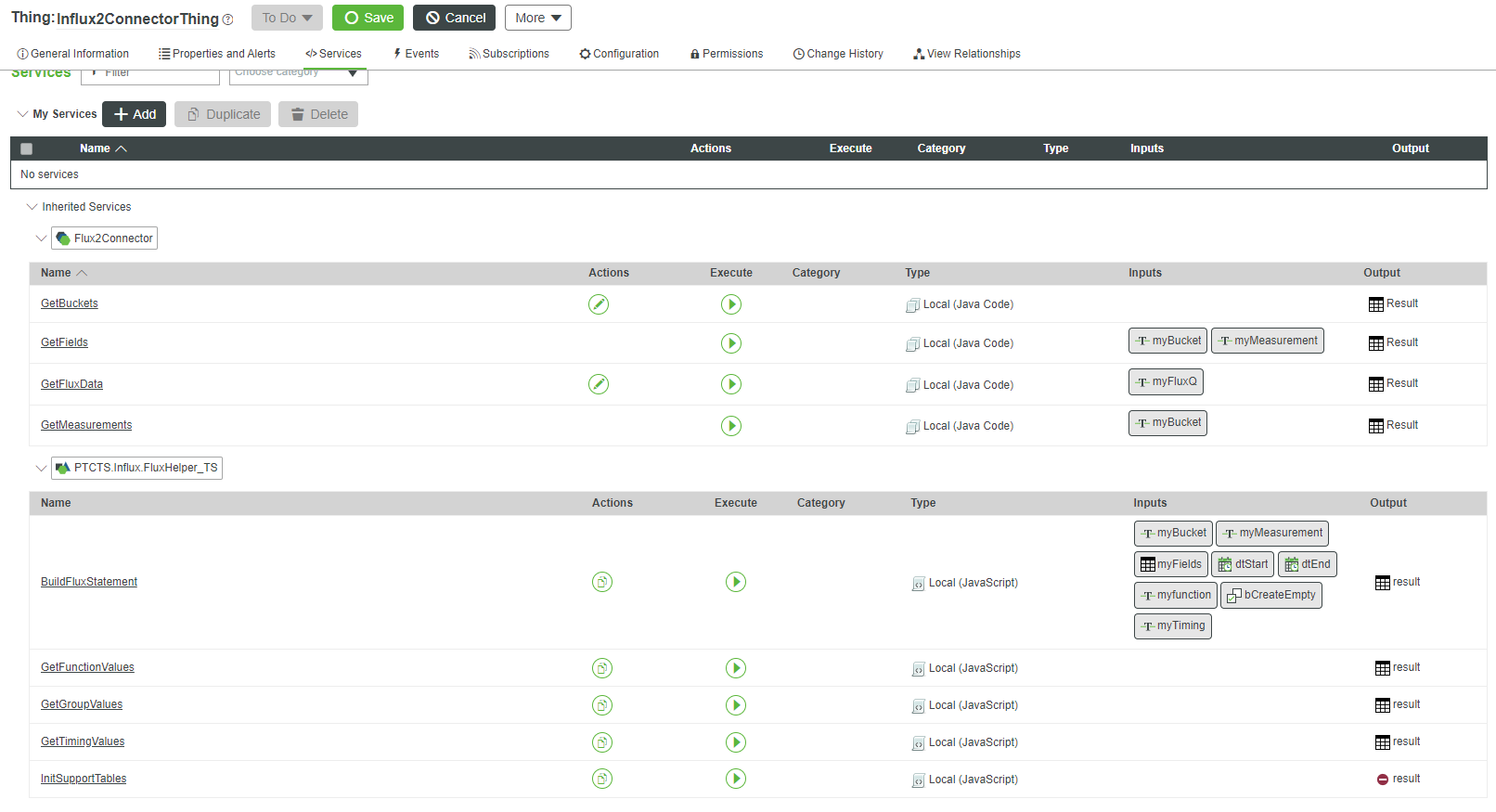
To configure the thing, you need to change following items to your needs.

InfluxServer: Address of the Influx instance

OrgName: Organization name of the Influx instance

InfluxToken: the token you have crated belonging to your organization

**Services:**



**Services:**

**Helper functions:**

1. GetBuckets: Returns all buckets (databases) from the Influx instance
2. GetMeasurements (input: bucket) Returns all measurements (tables) form the Influx bucket
3. GetFields (input: bucket & measurement) Returns all field from the measurement

**Data function**

1. GetFluxData(Input: Flux statement)  
   Example statement (query the last 1h of a temperature form the SigmaTile valuestream grouped by 10 second values)

from(bucket: "ThingWorx")  
 |> range(start: now()-1h, stop: now())  
 |> filter(fn: (r) => r["\_measurement"] == "SigmaTile")  
 |> filter(fn: (r) => r["\_field"] == "Temperature")  
 |> filter(fn: (r) => r["valuestreamname"] == "SigmaTile\_VS")  
 |> aggregateWindow(every: 10s, fn: mean, createEmpty: false)  
 |> yield(name: "mean")

**Thingshape (PTCTS.Influx.FluxHelper\_TS) services:**

1. BuildFluxStatement: Helper function for the trender. Needs the input parameter to create the flux statement.

**Example for the SigmaTile query above:**

**Imput Params:**

myBucket: **ThingWorx**

myMeasurement: **SigmaTile**

myFields (InfoTable / DS StringList): **Temperature**

dtStart: “**2022-05-01 08:00:00**”

dtEnd: “**2022-05-01 09:00:00**”

myfunction: **mean**

bCreateEmpty: **false**

myTiming: **10s**

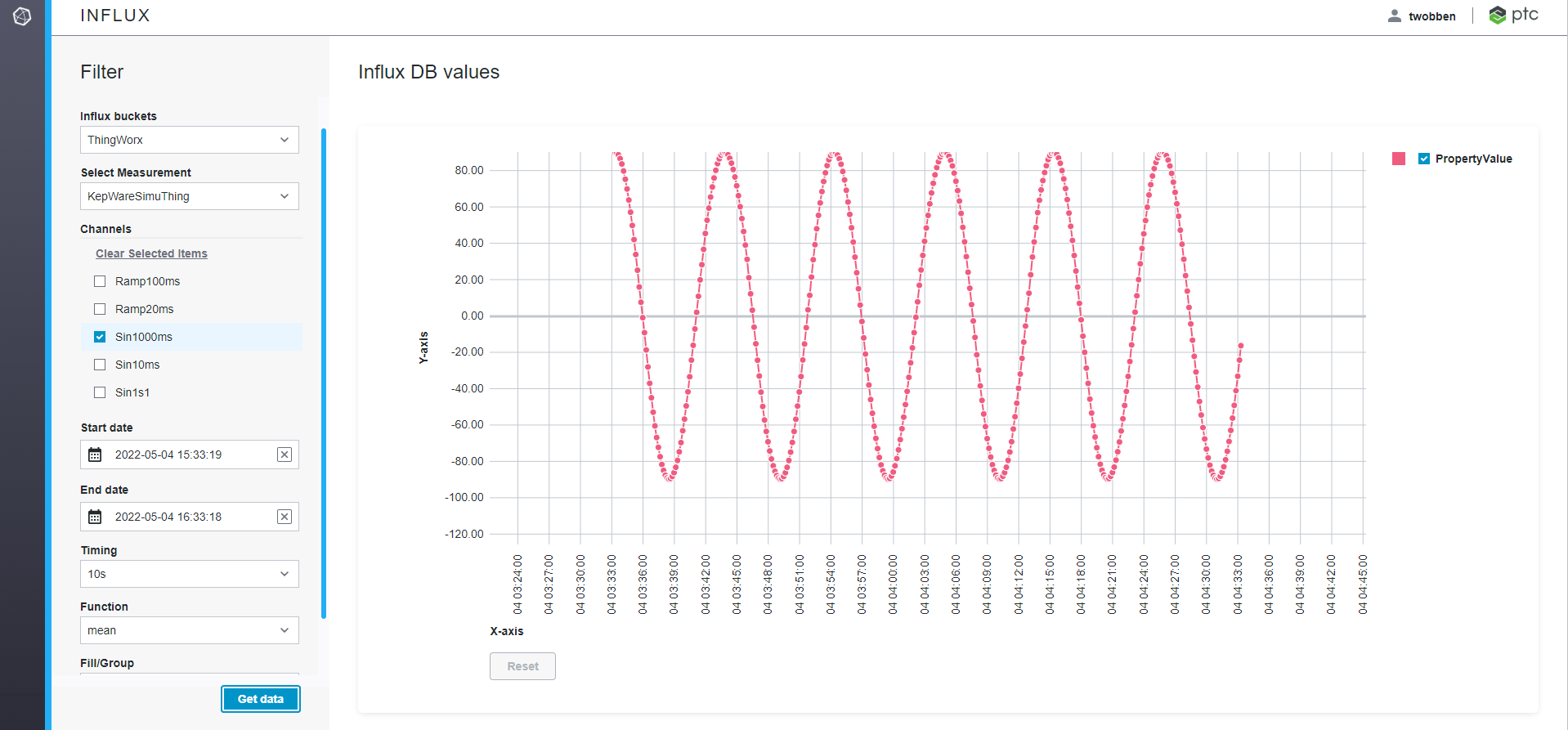
1. GetFunctionValues/GetGroupValues/GetTimingValues: Gets from the data tables the values for the drop down menues for the trender mashup
2. InitSupportTables: Filles the FunctionValues/Group/Timing datatables with standard values

**Walkthrough:**

1. Connect your ThingWorx persistence provider to your Indflux2 oss instance.
   1. User
   2. Org
   3. Token

Is from your Influx instance is needed.

1. Connect one of our things to a value stream (Influx persistence provider)
2. Import the Flux2Connector.zip extension
3. Import the Entities\_TrenderDemo.xml
4. Run the service InitSupportTables from the Influx2ConnectorThing
5. Open the Mashup PTCTS.InfluxUI.Main\_MU



1. Choose bucket, measurement, channels, date range, timing etc
2. Click on Get data