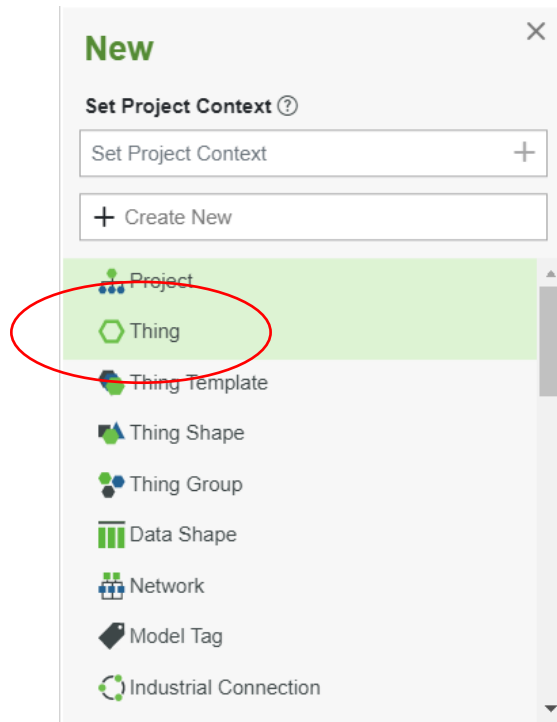


Displaying Time Series Data from Edge Device in a Mashup

1. Create new Thing




2. Fill out General Information

- a. Name = Initials_Temperature_Sensor
- b. Project = ESRAP_Meetings_Exercises
- c. Base Thing Template = RemoteThing

Thing:New Thing - 4 * ? To Do ▼ Save Cancel

[General Information](#) [Properties and Alerts](#) [Services](#) [Events](#)

General Information

 **Name** ? (required)
JA_Temperature_Sensor

[Change](#)

Description ?

Project ? (required)
ESRAP_Meetings_Exercises ✕

[Set as project context](#) ?

Tags ?

+

Base Thing Template ? (required)
RemoteThing ✕

3. Go to Properties and Alerts -> Add Property
 - a. Fill in the information as in the following image:

New Property 7 ✓ ✓ ✕

Name ? (required)
Temperature

Description ?

Base Type ?
NUMBER ▼

Units ?

Min Value ?

Max Value ?

☐ **Has Default Value** ?

☒ **Persistent** ?

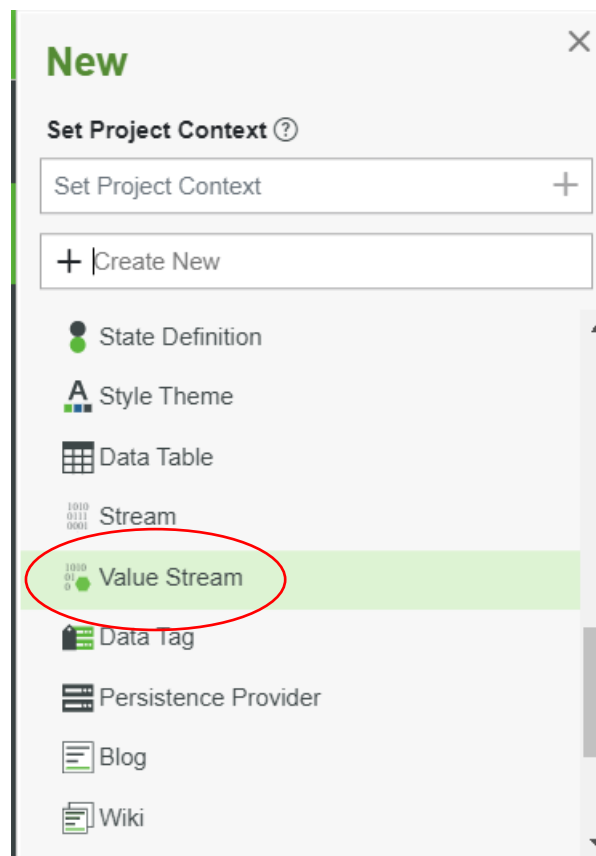
☐ **Read Only** ?

☒ **Logged** ?

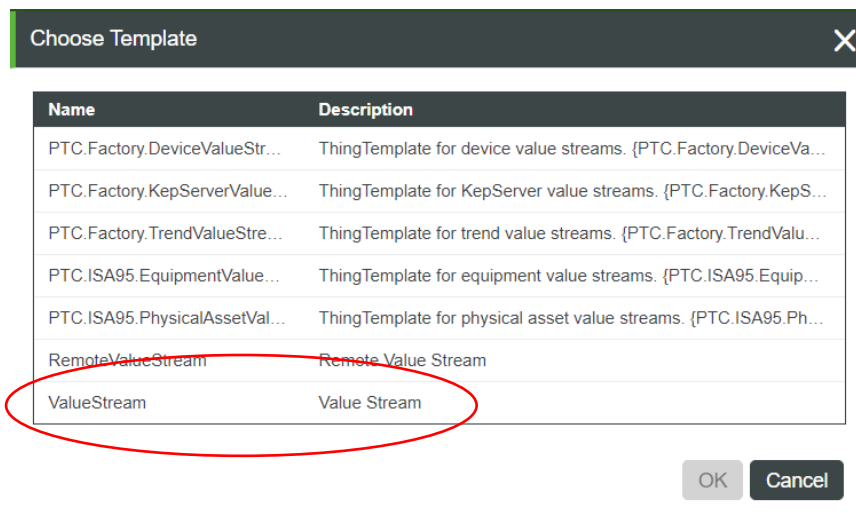
Binding ?
None ▼

- b. Save

4. Create new Value Stream




a. Select ValueStream -> OK



5. Fill in Information:

- Name = Initials_Temperature_Sensor_ValueStream
- Project = ESRAP_Meetings_Exercises

General Information



Name ? (required)
JA_Temperature_Sensor_ValueStream

[Change](#)

Description ?

Project ? (required)
ESRAP_Meetings_Exercises [Set as project context](#) ?

Tags ?
Search Model Tags +

Base Thing Template ? (required)
ValueStream

- c. Save
- 6. Navigate to Initials_Temperature_Sensor Thing
 - a. Add a Value Stream on General Information tab

Thing:JA_Temperature_Sensor ? [To Do](#)

[General Information](#) [Properties and Alerts](#) [Service](#)

ESRAP_Meetings_Exercises [Set as project context](#) ?

Tags ?
Search Model Tags +

Base Thing Template ?
RemoteThing

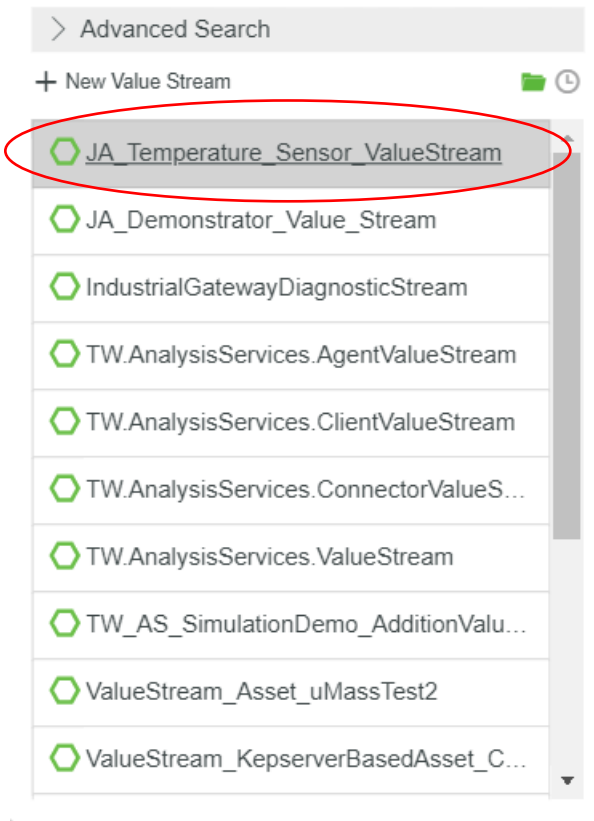
Implemented Shapes ?
Search Thing Shapes +

Value Stream ?
Search Value Streams +

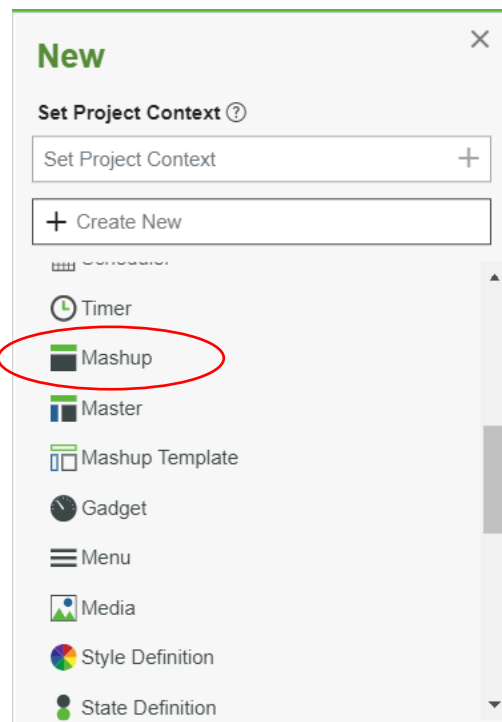
☒ **Active** ?

☐ **Published** ?

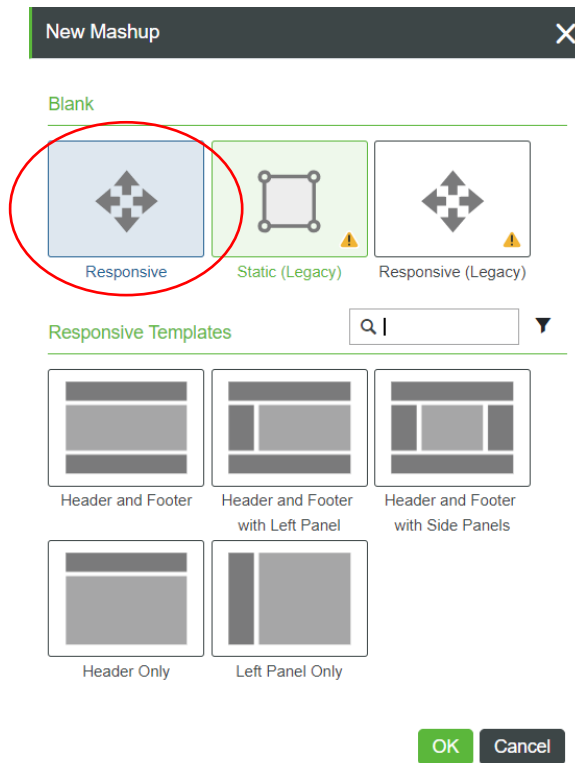
- b. Select the previous created Value Stream



- c. Save
7. Create a Mashup

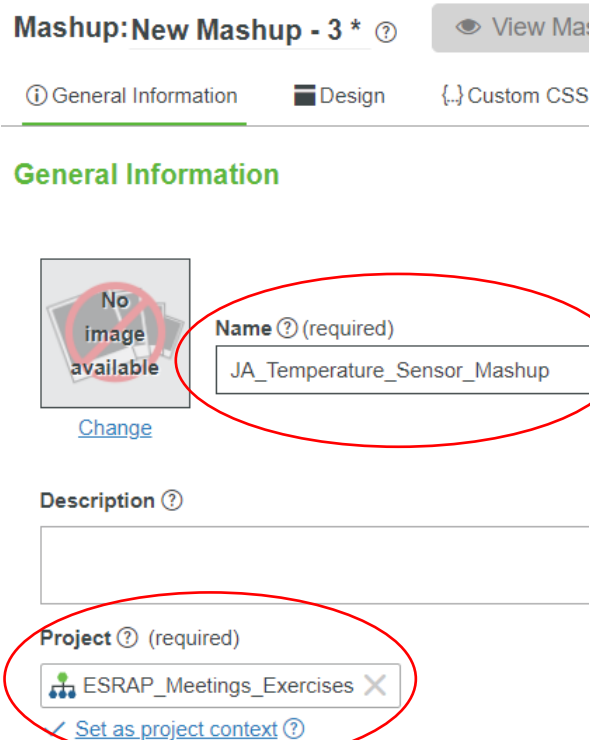


- a. Select Responsive



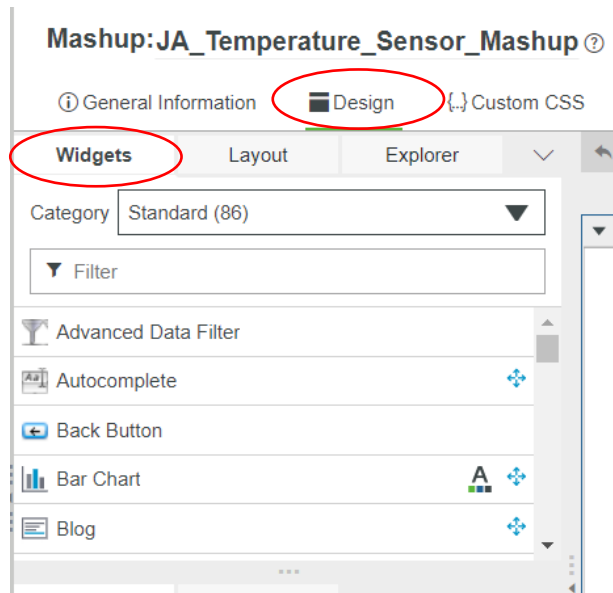
8. Fill in General Information

- Name = Initials_Temperature_Sensor_Mashup
- Project = ESRAP_Meetings_Exercises

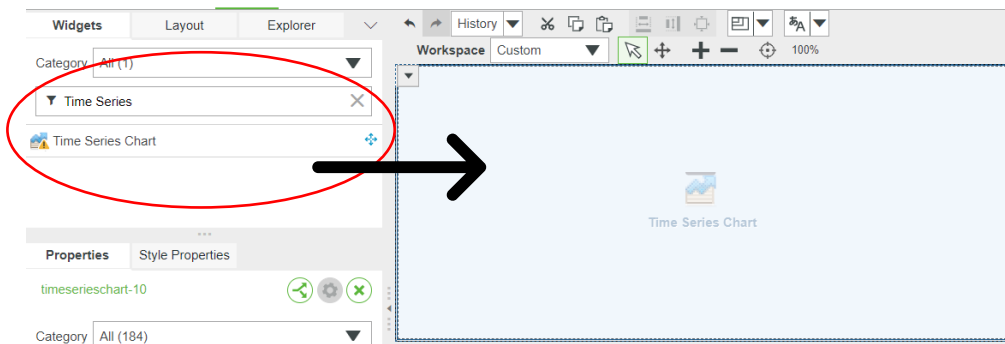


9. Switch to the Design tab

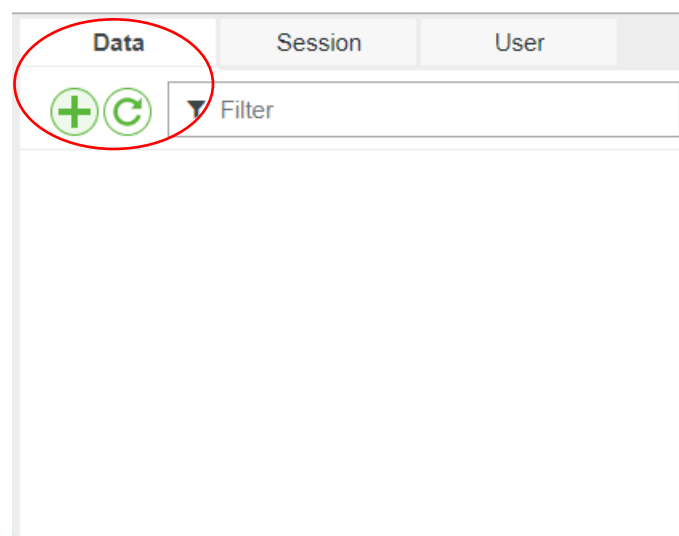
- Select Widgets



- b. Use the Filter to select Time Series Chart
- c. Drag and drop the Time Series Chart to the container



10. On the right side of the screen navigate to Data and click on add



11. In the Add Data window select the Temperature_Sensor Thing
 - a. Use the Entity filter to find the Thing

Select Entity

Entity Type

Search entity types ▼

Entity

JA_Tem

JA_Temperature_Sensor

JA_Temperature_Sensor_Mashup

JA_Temperature_Sensor_ValueStream

- b. Use the Services Filter to find the QueryPropertyHistory Service and click on the arrow to select it

Select Entity / Select Service(s)

Selected Entity

JA_Temperature_Sensor X

☐ Show dynamic services ?

Select Service Category

Choose category ▼

Services

QueryPr

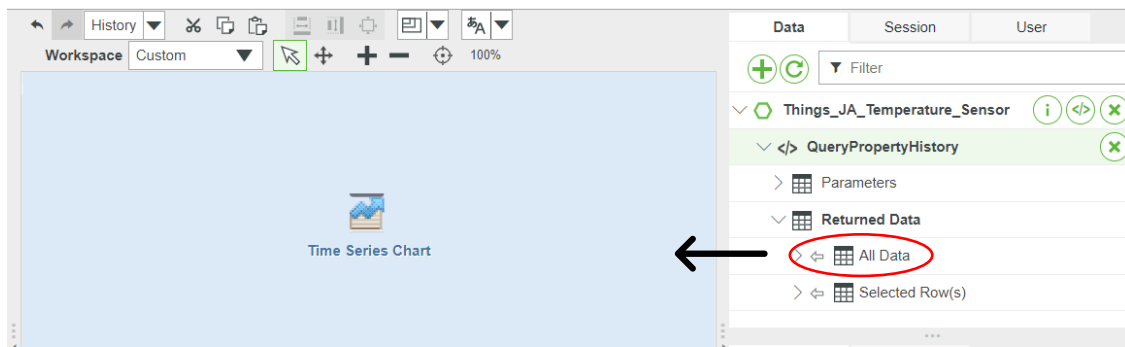
+ QueryPropertyHistory →

- c. Check the Execute on Load checkbox

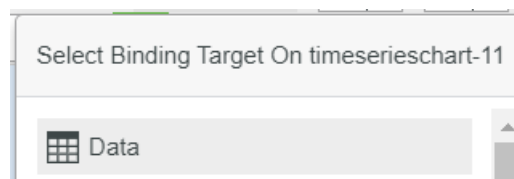
Selected Services

Delete	Entity	Service	Execute on Load
X	JA_Temperature_Sensor	QueryPropertyHistory	<input checked="" type="checkbox"/>

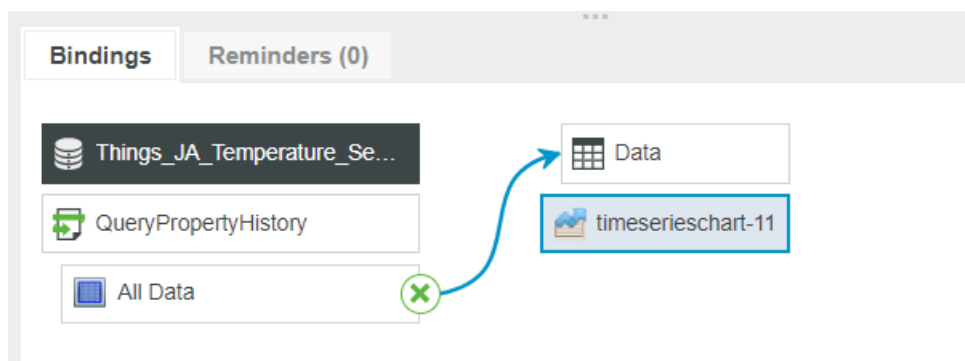
12. From the Data tab drag and drop All Data on the Time Series Chart



a. Select Data



13. Your Bindings in the bottom should look like this



14. Click on the Time Series Chart

- On the left side you can see the Properties tab
- Select following values (use the Filter to find it quicker):

The screenshot shows the 'Properties' tab for a 'timeserieschart-11' widget. The 'Category' is set to 'All (152)'. The 'Filter' dropdown is circled in red. Below the properties panel, there are two data field configurations: 'DataField1' with a 'Temperature' dropdown and 'XAxisField' with a 'timestamp' dropdown.

15. Now we need some data

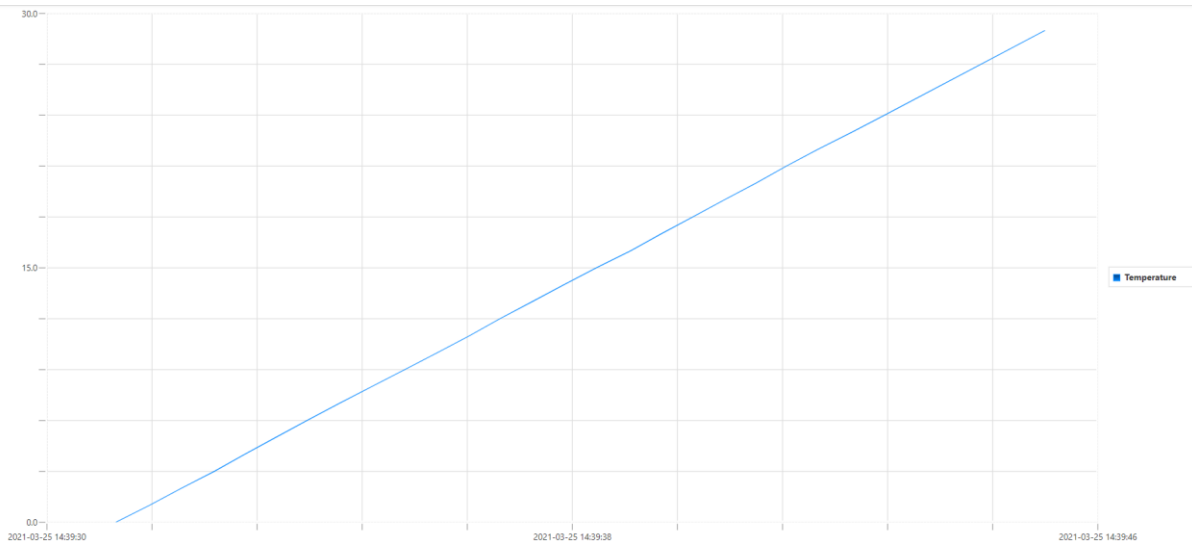
- You can go to the Initials_Temperature_Sensor Thing and manually change the value of the Temperature property a few times
- Or you can use the provided python script that uses the Thingworx REST API to push property data
 - Create New -> Application Key
 - Name = Initials_Temperature_Sensor_Appkey
 - Project = ESRAP_Meetings_Exercises
 - User Name Reference = Select your own user (TWX will tell you that no admin user should be used)
 - Save
 - In the python script copy your generated Key ID and fill in your initials (example: '/Things/JA_Temperature_Sensor/Properties/*')

```
def callToTwx(propertyValue):
    url = 'https://pp-2101111403aw.portal.ptc.io/Thingworx'
    headers = { 'Content-Type': 'application/json', 'appKey': 'Insert Key ID from your Application Key', 'accept': 'application/json' }
    payload = { 'Temperature': propertyValue }
    response = requests.put(url + '/Things/Initials_Temperature_Sensor/Properties/*', headers=headers, json=payload, verify=False)
    return response
```

16. The script needs the “requests” package (pip install requests)

- Run the script from command line: python updateTwxProperties.py
- Tested with Python 3.7 and 3.9

17. In Thingworx Composer head back to your Mashup and click “View Mashup”, if you used the python script it will look like this:



18. If you want to delete the stored data, use the PurgeAllPropertyHistory Service of the Temperature_Sensor Thing