Technical Document

Niagara Analytics Reference

niagara

This PDF is generated from *docs.niagara-community.com* on: June 18, 2025

About this reference

This reference documents the pre-defined algorithms, components, blocks, web charts, views, and ORD schemes used by the Niagara Analytics Framework.

Audience

The information in this document is for Systems Integrators and Facility Managers who are responsible for configuring the tools used to manage complex building systems.

Document Content

Chapters document the Analytics Tag Dictionary, algorithm library, components, views, windows, logic blocks, Px views, Ux charts and ORD schemes. An index is provided to help you find the specific information you are looking for.

Product Documentation

This document is part of the Niagara technical documentation library. Released versions of Niagara software include a complete collection of technical information that is provided in both online help and PDF formats.

Document change log

This topic summarizes the history of this document.

May 30, 2023

Corrected technical errors throughout the document.

Added missing algorithms to Chapter 2.

Consolidated Logic block examples from the *Niagara Analytics Guide* with Logic block Property Sheets to provide more complete help topics.

October 19, 2022

Added NaN (Not a Number) to the **Invalid Value Filter** (BInvalidValueFilter) block.

Added NaN (Not a Number) to the Outlier property description (**BAnalyticDataDefinition** and **Analytic Data Manager** view description).

Removed the bullet identifying an infinite number as a condition for making a value invalid (**Invalid Value Filter** block)

Removed infinity as an invalid value condition for **Invalid Value Switch**.

Rewrote the introductory text for the **Deadband Filter** (BDeadbandFilter) block.

Removed this sentence from the Deadband Mode property description: “For this option, the Deadband should be in the range 0–1.”

Related documentation

These documents are available for learning how to use Niagara Analytics Framework

- *Niagara Analytics Guide*

  documents the code you can use to extend this product.

- *Niagara Hierarchies Guide* provides information about adding metadata to objects.
- *Niagara Relations Guide* provides general information about how to create Px graphics.

Analytics tag dictionary

The Analytics Tag Dictionary provides a set of tags and tag groups specifically designed for the framework.

To create your own framework-related tag dictionary, drag this Analytics dictionary from the analytics-lib palette to the **TagDictionaryService** in the Nav tree, then customize it by copying and pasting tags from other dictionaries.

There are no unique relations definitions for the framework.

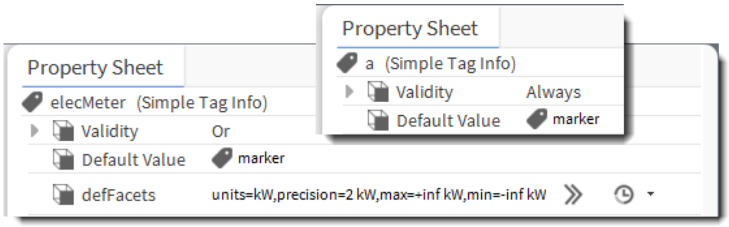
- **Tag Definitions**
  Tags and the hierarchies to which they belong provide powerful tools for configuring data analysis. This topic documents the tag definitions that come with the Analytics tag dictionary.
- **Tag Group Definitions**
  These groups of tags are unique to the framework. This topic documents the group definitions that come with the Analytics tag dictionary.

Tag Definitions

Tags and the hierarchies to which they belong provide powerful tools for configuring data analysis. This topic documents the tag definitions that come with the Analytics tag dictionary.

The properties on each Simple Tag Info **Property Sheet** may or may not include default facets.

Figure 1. Simple Tag Info properties



Property Sheet

elecMeter (Simple Tag Info)

Validity	Or
Default Value	marker
defFacets	units=kW,precision=2 kW,max=+inf kW,min=-inf kW

Property Sheet

a (Simple Tag Info)

Validity	Always
Default Value	marker

To view these properties, double-click the tag in the tag dictionary.

Property	Value	Description
Validity	Always or Or	Provides a way to identify a data source.
Default Value	marker or a value	A tag that is identified as a marker tag does not require a value. For value tags, use this property to configure a default value.
defFacets	units, precision, min, max, etc.	Clicking the chevron to the right of this property opens a standard **Config Facets** window. The facets you configure here serve as defaults for the point to which this tag is assigned. Other facet configurations can override these settings.

Pre-defined tag definitions

The framework comes with several pre-defined tag definitions in addition to the Haystack tags. You precede each of these tags with a:.

Tag definition	Identifies an entity as a source of...
a	values to be used by Analytics. All points used for Analytics must be tagged with this tag as they count towards the total allowed by your framework license.
ahuMode	an air handling unit mode.
chilledWaterUsage	chilled water usage indicator.
differential	differences between amounts of things.
economizer	information about a part of a building's cooling system that uses cool outdoor air to cool rather than operating the air conditioning compressor.
elecMeter	input from an electric meter.
enthalpy	input from a thermodynamic quantity equivalent to the total heat content of a system.
roomHumidity	room humidity.
secondary	information from a secondary controller.
space	information from a building space.
static	static pressure.
status	information related to performance.
steamUsage	information from a steam boiler.
steamWaterUsage	information regarding steam usage.
supply	supply air control.
vahhlmt	Variable Air Humidity High Limit
vahllmt	Variable Air Humidity Low Limit
vathlmt	Variable Air Temperature High Limit
vathlmt	Variable Air Temperature Low Limit
waterMeter	Identifies a water meter.

Parent topic: Analytics tag dictionary

Tag Group Definitions

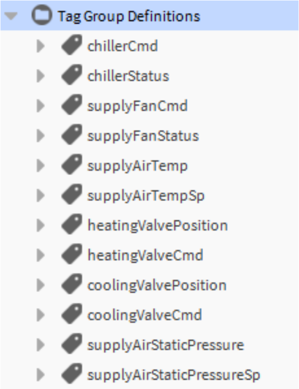
These groups of tags are unique to the framework. This topic documents the group definitions that come with the Analytics tag dictionary.

Pre-defined tags

The number of tags in each group definition varies. Assigning a group definition to a point assigns all tags that the group contains to the point.

The tag group definitions that come with the framework contain all marker tags. You precede these tags with a:.

Figure 1. Pre-defined Tag Group Definitions



- Tag Group Definitions
  - chillerCmd
  - chillerStatus
  - supplyFanCmd
  - supplyFanStatus
  - supplyAirTemp
  - supplyAirTempSp
  - heatingValvePosition
  - heatingValveCmd
  - coolingValvePosition
  - coolingValveCmd
  - supplyAirStaticPressure
  - supplyAirStaticPressureSp

Table 1. Analytics tag dictionary, tag group definitions and tags

Tag group definition	Tags	Suggested use
chilledWaterReturnTemp	chilled, water, return, temp	
chilledWaterSupplyTemp	chilled, water, supply, temp	
chilledWaterSupplyTempSp	chilled, water, supply, temp, sp (setpoint)	
chillerCmd	chiller, cmd	
chillerStatus	chiller, status	
coolingValveCmd	cooling, valve, cmd	
coolingValvePosition	cooling, valve, position	
economizerSp	economizer, sp (setpoint)	
economizerStatus	curStatus, economizer	
energyTargetMonthly	energy, target, monthly	
energyTargetYearly	energy, target, yearly	
heatingHotWaterReturnTempSp	heating, hot, water, temp, sp (setpoint), return	
heatingHotWaterSupplyTemp	heating, hot, water, temp, supply	
heatingHotWaterSupplyTempSp	heating, hot, water, temp, sp (setpoint), supply	
heatingHotWaterSystem	heating, hot water	
heatingValveCmd	heating, valve, cmd	
heatingValvePosition	heating, valve, position	
outsideAirDamperCmd	air, damper, outside, cmd	
outsideAirDamperPos	air, damper outside pos	
outsideAirEnthalpy	air, outside, enthalpy	

Tag group definition	Tags	Suggested use
outsideAirTemp	air, outside, temp	
returnAirEnthalpy	air, outside, enthalpy	
secondaryChilledWaterPump	chilled, water, pump, secondary	
secondaryChilledWaterPumpSpeed	chilled, water, pump, secondary, speed	
secondaryChilledWaterPumpStatus	chilled, water, pump, secondary, curStatus	
secondaryLoopDP	differential, pressure, secondaryLoop	
secondaryLoopDPsp	differential, pressure, secondaryLoop, sp (setpoint)	
spaceDP	differential, pressure, space	
spaceDPSp	differential, pressure, sp (setpoint), space	
spaceExhaustAirFlow	space, exhaust, air, flow	
spaceExhaustAirFlowSp	space, exhaust, air, flow, sp (setpoint)	
spaceSupplyAirFlow	space air flow, supply	
spaceSupplyAirFlowSp	space air flow, sp (setpoint), supply	
spaceAirStaticPressure	supply air, static, pressure	
spaceAirStaticPressureSp	supply air, static, pressure, sp (setpoint)	
supplyAirTemp	supply, air, temp	
supplyAirTempSp	supply, air, temp, sp (setpoint)	
supplyFanCmd	supply, fan, cmd	
supplyFanStatus	supply, fan, status	
terminalUnitDamperPosition	terminal, unit, damper, position	

Parent topic: Analytics tag dictionary

Algorithm library reference

The pre-defined algorithms documented in the following topics come with the software. You may be able to use one or more of them without modification, or as a reference when creating your own custom algorithms.

Each algorithm description contains:

- a brief description of what the algorithm does
- a flowchart view of the algorithm logic
- an image of the **Wire Sheet** for each algorithm
- a table that explains what each logic block is doing

To use an algorithm, add tag and tag groups from the analytics-lib palette's Analytics dictionary to the station, apply the tags as a data model to the necessary components in the station, then drag an algorithm to a logic **Wire Sheet** and configure its object properties for your specific application.

You could use an algorithm with an analytic alert that has either a value or trend request. To process a value request or trend request, you use an algorithm with a control point on an analytic proxy extension. The control point might have an alarm extension for generating alarm records.

To help visualize when a fault condition occurred in the past without generating an alarm record, you can use an algorithm in an analytic web chart or table.

Some of the explanations in this document may assume that the analytic request is going to be a trend request, but it might also be a value request.

- **Chilled_Water_Mixing**
  This algorithm evaluates trend data and returns a true result, indicating that an exception existed, if a chiller received the command to run, it was on line, and the chilled water supply temperature minus the chilled water supply temperature was less than three degrees Fahrenheit for 30 minutes. A false result indicates that nothing was wrong.
- **Chilled_Water_Usage_Minus_10_PC_of_last_Month**
  This algorithm reports an exception if chilled water usage this month is less than last month's usage by more than 10% (the negative delta between this month's and last month's chilled water usage is more than 10%).
- **Chilled_Water_Usage_Minus_5_PC_of_last_Month**
  This algorithm reports an exception if chilled water usage this month is less than last month's usage by more than 5% (the negative delta between this month's and last month's chilled water usage is more than 5%).
- **Chilled_Water_Usage_Plus_10_PC_of_last_Month**
  This algorithm reports an exception if chilled water usage this month is greater than last month's usage by more than 10% (the positive delta between this month's and last month's chilled water usage is more than 10%).
- **Chilled_Water_Usage_Plus_5_PC_of_last_Month**
  This algorithm reports an exception if chilled water usage this month is greater than last month's usage by more than 5% (the positive delta between this month's and last month's chilled water usage is more than 5%).
- **Cooling_Valve_Open_100_Percent**
  This algorithm evaluates trend data. It produces a true or false result for historical data, if a building was occupied, the supply fan was on, the supply fan status indicated that the fan was on, and the cooling valve was wide open (100% open) for 30 minutes. A true result indicates that an exception existed; a false result that nothing was wrong.
- **Damper_Open_During_Warmup**
  When an AHU is warming up in the morning, the outside air damper should not be on or open more

than 5%.

- **EconomizerNotRunning-EnthalpyBased**
  This algorithm returns true if, when a room or building is occupied, the economizer is off and the outside air enthalpy is less than the sum of the economizer setpoint and the return air enthalpy for more than 30 minutes.
- **EconomizerNotRunning-TempBased**
  This algorithm returns true if, when a room or building is occupied, the economizer is off and the outside air temperature is lower than the economizer setpoint for more than 30 minutes.
- **Electric_EUI_Exceeds_Month_Target**
  This algorithm evaluates electricity usage over the period of this month. It returns true if the monthly electric EUI for the site exceeds the target EUI (Energy Use Intensity, or energy per square foot per year). EUI is based on the last available Area. A false result indicates that EUI did not exceed the monthly target.
- **Electric_EUI_Exceeds_Yearly_Target**
  This algorithm evaluates electricity usage over the period of this year. It returns true if the yearly electric EUI for the site exceeds the target EUI (Energy Use Intensity, or energy per square foot per year). EUI is based on last available Area. A false result indicates that EUI did not exceed the yearly target.
- **Electricity_Usage_5_Percent_Less_Than_Last_Year**
  This algorithm compares electricity usage this month with usage from the same month last year. It produces a true result, indicating that an exception exists, if the total monthly usage for this month is less than 95% of the total monthly usage for the same month last year. A false result indicates that nothing was wrong.
- **Electricity_Usage_5_Percent_More_Than_Last_Year**
  This algorithm evaluates trend data for electricity usage. It returns a true result, indicating that an exception existed, if electricity usage for this month is more than electricity usage for the same month last year by more than 5%. A false result indicates that electricity usage was normal.
- **Heating_Valve_Open_100_Percent**
  This algorithm evaluates trend data. It produces a true or false result for historical data, if a building was occupied, the supply fan was on, the supply fan status indicated that the fan was on, and the heating valve was wide open (100% open) for 30 minutes. A true result indicates that an exception existed; a false result that nothing was wrong.
- **HighHotWaterTemperature**
  This algorithm returns true if, when the hot water system is running, the hot water temperature is greater than the hot water supply temperature setpoint by plus five degrees for more than 30 minutes.
- **High_Chilled_Water_Temp**
  This algorithm analyzes trend data and returns a true result, indicating that an exception existed, if a chiller was commanded on, the chiller status (feedback) indicates it was running, and the chilled water supply temperature was greater than 1 degree Fahrenheit above the chilled water supply water temperature setpoint for 30 minutes. A false result indicates that nothing was wrong.
- **High_Outside_Air_Intake**
  This algorithm evaluates trend data collected from an AHU unit. It returns a true result if four conditions are met for an hour. Otherwise it returns false. The four conditions are: system was occupied, the command had been given to turn the supply fan on, the supply fan was on, and the outside air flow coming in was more than 100% of the outside airflow setpoint. A false result indicates that air coming in from the outside was normal.
- **High_Supply_Static_Pressure**
  This algorithm evaluates static pressure trend data. It produces a true result, indicating that an exception existed, if a building was occupied, an AHU unit's supply fan was on, the supply fan status indicated that the fan was on, the supply air static pressure was greater than 110% of the supply air static pressure stepoint, and all these conditions existed for 30 minutes. A false result indicates that nothing was wrong.
- **High_Supply_Temperature**
  This algorithm evaluates supply temperature trend data. It produces a true result if the difference between the supply air temperature and supply air temperature setpoint is greater than three degrees Fahrenheit. A false result indicates that nothing was wrong.
- **HotWaterMixing**
  This algorithm returns true if, when the hot water system is running, the hot water supply temperature minus the hot water return temperature setpoint is less than five degrees for more than 30 minutes.
- **Low_Chilled_Water_Temp**

This algorithm evaluates trend data and returns a true result, indicating that an exception existed, if a chiller was commanded on, the chiller status (feedback) indicates it was running, and the chilled water supply temperature was greater than 1 degree Fahrenheit below the chilled water supply water temperature setpoint for 30 minutes. A false result indicates that nothing was wrong.

- **LowHotWaterTemperature**
  This algorithm returns true if, when the hot water system is running, the hot water temperature is less than the hot water supply temperature setpoint by more than five degrees for more than 30 minutes.
- **Low_Outside_Air_Intake**
  This algorithm evaluates trend data collected from an AHU unit. It returns a true result if four conditions are met for an hour. Otherwise, it returns false. The four conditions are: was occupied, the command had been given to turn the supply fan on, the supply fan was on, and the outside air flow coming in was less than 90% of the outside airflow setpoint. A false result indicates that air coming in from the outside was normal.
- **Low_Supply_Static_Pressure**
  This algorithm evaluates static pressure trend data. It produces a true result, indicating that an exception existed, if a building was occupied, an AHU unit's supply fan was on, the supply fan status indicated that the fan was on, the supply air static pressure was less than 90% of the supply air static pressure stepoint, and all these conditions existed for 30 minutes. A false result indicates that nothing was wrong.
- **Low_Supply_Temperature**
  This algorithm evaluates supply temperature trend data. It produces a true result if the supply air temperature is more than three degrees Fahrenheit below the supply air temperature setpoint. A false result indicates that nothing was wrong.
- **NonConformanceSpaceHumidity**
  This algorithm returns true, when a room is occupied, the command to turn the supply fan on was issued, the supply fan is on, and the humidity is greater than the VAHHLMT (Variable Air Humidity High Limit) or less than the VAHLLMT (Variable Air Humidity Low Limit).
- **NonConformanceSpaceTemperature**
  This algorithm returns true, when a room is occupied, the command to turn the supply fan on was issued, the supply fan is on, and the room temperature is greater than the VATHLMT (Variable Air Temperature High Limit) or less than the VATLLMT (Variable Air Temperature Low Limit).
- **Prolonged_100_Percent_VAV_Damper**
  This algorithm evaluates trend data collected from an AHU unit. It returns true if the facility was occupied, the command to turn the supply fan on was given, the supply fan was on, and the position of the damper was more than 95% open for 30 minutes. A false result indicates that nothing was wrong.
- **PumpExceedingSetpoint**
  This algorithm uses three inputs to determine if the pressure in a secondary chilled water pump exceeds the expected setpoint.
- **PumpNotMeetingSetpoint**
  This algorithm uses four inputs to determine if the pressure in a secondary chilled water pump is below the expected setpoint.
- **Steam_Usage_Minus_10_Percent_of_last_Month**
  This algorithm compares steam usage this month with usage from the previous month. It produces a true result, indicating that an exception exists, if the total steam usage for this month is less than 90% of the total monthly usage for the previous month.
- **Steam_Usage_Minus_5_Percent_of_last_Month**
  This algorithm compares steam usage this month with usage from the previous month. It produces a true result, indicating that an exception exists, if the total steam usage for this month is less than 95% of the total monthly usage for the previous month.
- **Steam_Usage_Plus_10_Percent_of_last_Month**
  This algorithm compares steam usage this month with usage from the previous month. It produces a true result, indicating that an exception exists, if the total steam usage for this month is greater than the total monthly usage for the previous month by more than 10%.
- **Steam_Usage_Plus_5_Percent_of_last_Month**
  This algorithm compares steam usage this month with usage from the previous month. It produces a true result, indicating that an exception exists, if the total steam usage for this month is greater than the total monthly usage for the previous month by more than 5%.
- **Steam_Water_Usage_Minus_10_PC_of_last_Month**
  This algorithm compares steam water usage this month with usage from the previous month. It produces a true result, indicating that an exception exists, if the total steam water usage for this month

is less than 90% of the total monthly usage for the previous month.

- **Steam_Water_Usage_Minus_5_PC_of_last_Month**
  This algorithm compares steam usage this month with usage from the previous month. It produces a true result, indicating that an exception exists, if the total steam usage for this month is less than 95% of the total monthly usage for the previous month.
- **Steam_Water_Usage_Plus_10_PC_of_last_Month**
  This algorithm compares steam water usage this month with usage from the previous month. It produces a true result, indicating that an exception exists, if the total steam water usage for this month is greater than the total monthly usage for the previous month by more than 10%.
- **Steam_Water_Usage_Plus_5_PC_of_last_Month**
  This algorithm compares steam water usage this month with usage from the previous month. It produces a true result, indicating that an exception exists, if the total steam water usage for this month is greater than the total monthly usage for the previous month by more than 5%.
- **Simultaneous_Heat_Cool**
  This algorithm analyzes trend data and returns a true result, indicating that an exception existed, if the building was occupied, the supply fan command was on, the supply fan status was on, the heating valve position or the heating valve command was greater than 0%, and the cooling valve position or cooling valve command was greater than 0% for 15 minutes. A false result indicates that nothing was wrong.
- **SpaceDifferentialPressureHighLow**
  This algorithm returns true if the differential pressure in the space (room or building) is greater than or less than the setpoint for more than 30 minutes.
- **SpaceExhaustFlowIsHigh**
  This algorithm returns true if the space exhaust air flow is greater than the space exhaust airflow setpoint.
- **SpaceSupplyFlowIsHigh**
  This algorithm returns true if the space supply air flow is greater than the space supply air flow setpoint.
- **Unscheduled_Operation**
  This algorithm evaluates trend data collected from an AHU unit. It determines if an AHU unit was running at an unscheduled time. It returns true if the facility is unoccupied, the command was given to turn the supply fan on, or the supply fan is on. It does not matter how long the supply fan was running. A false result indicates that the AHU ran on schedule.
- **Unscheduled_Operation_EFan**
  This algorithm returns true if a supply fan (EFAN) is on or its status is ok and a space (room or building) is unoccupied.
- **Water_Usage_5_Percent_Less_Than_Last_Year**
  This algorithm evaluates trend data for water usage. It returns a true result, indicating that an exception existed, if water usage for this month was less than water usage the same month last year by more than 5%. A false result indicates that water usage was normal.
- **Water_Usage_5_Percent_More_Than_Last_Year**
  This algorithm evaluates trend data for water usage. It returns a true result, indicating that an exception existed, if water usage for this month was greater than water usage the same month last year by more than 5%. A false result indicates that water usage was normal.
- **WeatherNormalizedConsumption**
  This algorithm uses three inputs to calculate normal energy consumption based on the weather.
- **WeatherNormalizedDemand**
  This algorithm uses three inputs to calculate normal energy demand based on the weather.

Chilled_Water_Mixing

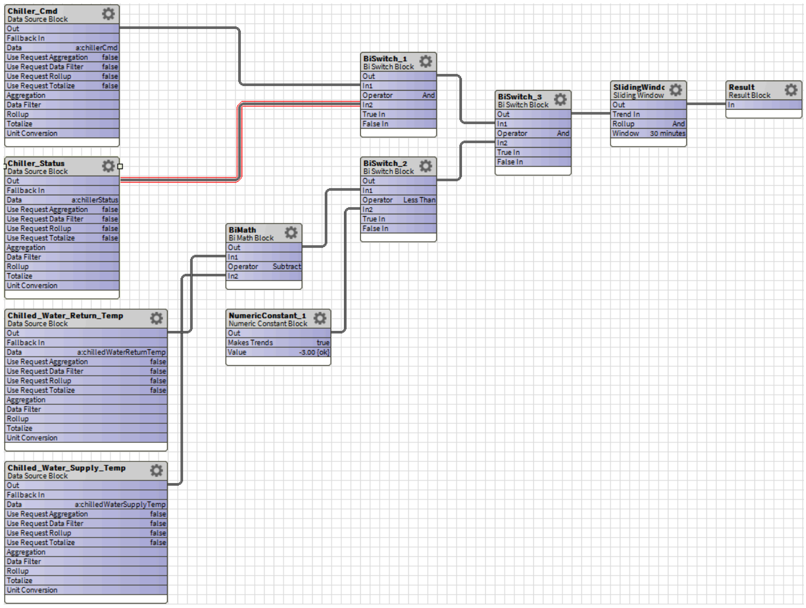
This algorithm evaluates trend data and returns a true result, indicating that an exception existed, if a chiller received the command to run, it was on line, and the chilled water supply temperature minus the chilled water supply temperature was less than three degrees Fahrenheit for 30 minutes. A false result indicates that nothing was wrong.

ORDs

slot:/Algorithm/English/Chilled_Water_Mixing slot:/Algorithm/Metric/Chilled_Water_Mixing

Wire sheet view

Figure 1. Chilled Water Mixing algorithm



Chiller_Cmd
Data Source Block
Out
Fallback In
Data a:chillerCmd
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Use Request Totalize false
Aggregation
Data Filter
Rollup
Totalize
Unit Conversion

Chiller_Status
Data Source Block
Out
Fallback In
Data a:chillerStatus
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Use Request Totalize false
Aggregation
Data Filter
Rollup
Totalize
Unit Conversion

Chilled_Water_Return_Temp
Data Source Block
Out
Fallback In
Data a:chilledWaterReturnTemp
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Use Request Totalize false
Aggregation
Data Filter
Rollup
Totalize
Unit Conversion

Chilled_Water_Supply_Temp
Data Source Block
Out
Fallback In
Data a:chilledWaterSupplyTemp
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Use Request Totalize false
Aggregation
Data Filter
Rollup
Totalize
Unit Conversion

BiMath
Bi Math Block
Out
In1
Operator Subtract
In2

NumericConstant_1
Numeric Constant Block
Out
Makes Trends true
Value -3.00 {ok}

BiSwitch_1
Bi Switch Block
Out
In1
Operator And
In2
True In
False In

BiSwitch_2
Bi Switch Block
Out
In1
Operator Less Than
In2
True In
False In

BiSwitch_3
Bi Switch Block
Out
In1
Operator And
In2
True In
False In

SlidingWindc
Sliding Window
Out
Trend In
Rollup And
Window 30 minutes

Result
Result Block
In

Parent topic: Algorithm library reference

Chilled_Water_Usage_Minus_10_PC_of_last_Month

This algorithm reports an exception if chilled water usage this month is less than last month's usage by more than 10% (the negative delta between this month's and last month's chilled water usage is more than 10%).

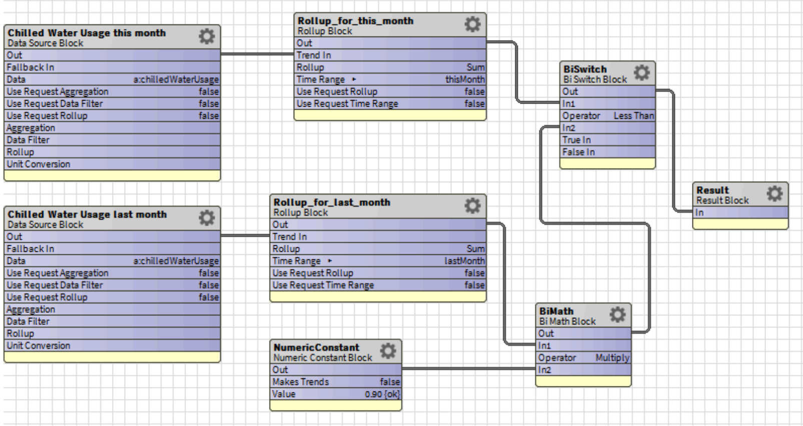
Input ORD

slot:/Algorithm/English/Chilled_Water_Usage_minus_10_pc_of_last_Month

slot:/Algorithm/Metric/Chilled_Water_Usage_minus_10_pc_of_last_Month

Wire sheet view

Figure 1. Chilled water usage wire sheet



Parent topic: Algorithm library reference

Chilled_Water_Usage_Minus_5_PC_of_last_Month

This algorithm reports an exception if chilled water usage this month is less than last month's usage by more than 5% (the negative delta between this month's and last month's chilled water usage is more than 5%).

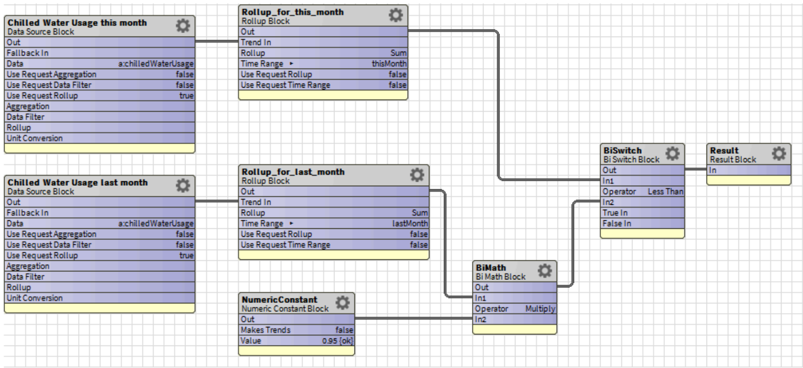
Input ORD

slot:/Algorithm/English/Chilled_Water_Usage_Minus_5_pc_of_last_Month

slot:/Algorithm/Metric/Chilled_Water_Usage_Minus_5_pc_of_last_Month

Wire sheet view

Figure 1. Chilled water usage wire sheet



Parent topic: Algorithm library reference

Chilled_Water_Usage_Plus_10_PC_of_last_Month

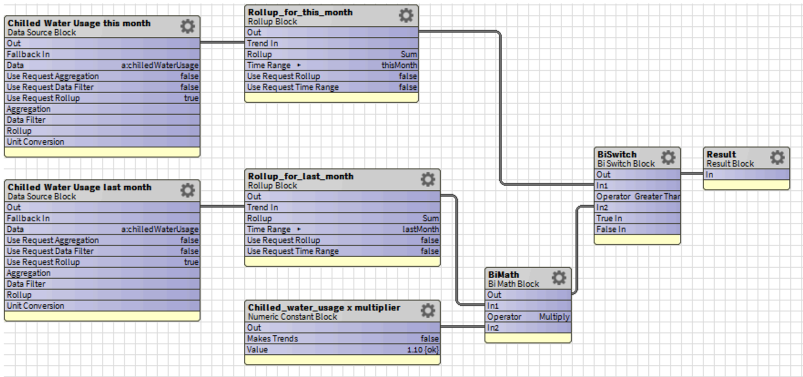
This algorithm reports an exception if chilled water usage this month is greater than last month's usage by more than 10% (the positive delta between this month's and last month's chilled water usage is more than 10%).

Input ORD

slot:/Algorithm/English/Chilled_Water_Usage_Plus_10_PC_of_last_Month slot:/Algorithm/Metric/Chilled_Water_Usage_Plus_10_PC_of_last_Month

Wire sheet view

Figure 1. Chilled water usage wire sheet



Chilled Water Usage this month
Data Source Block
Out
Fallback In
Data a:chilledWaterUsage
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup true
Aggregation
Data Filter
Rollup
Unit Conversion

Rollup_for_this_month
Rollup Block
Out
Trend In
Rollup Sum
Time Range ▸ thisMonth
Use Request Rollup false
Use Request Time Range false

Chilled Water Usage last month
Data Source Block
Out
Fallback In
Data a:chilledWaterUsage
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup true
Aggregation
Data Filter
Rollup
Unit Conversion

Rollup_for_last_month
Rollup Block
Out
Trend In
Rollup Sum
Time Range ▸ lastMonth
Use Request Rollup false
Use Request Time Range false

Chilled_water_usage x multiplier
Numeric Constant Block
Out
Makes Trends false
Value 1.10 {ok}

BiMath
Bi Math Block
Out
In1
Operator Multiply
In2

BiSwitch
Bi Switch Block
Out
In1
Operator Greater Thar
In2
True In
False In

Result
Result Block
In

Parent topic: Algorithm library reference

Chilled_Water_Usage_Plus_5_PC_of_last_Month

This algorithm reports an exception if chilled water usage this month is greater than last month’s usage by more than 5% (the positive delta between this month’s and last month’s chilled water usage is more than 5%).

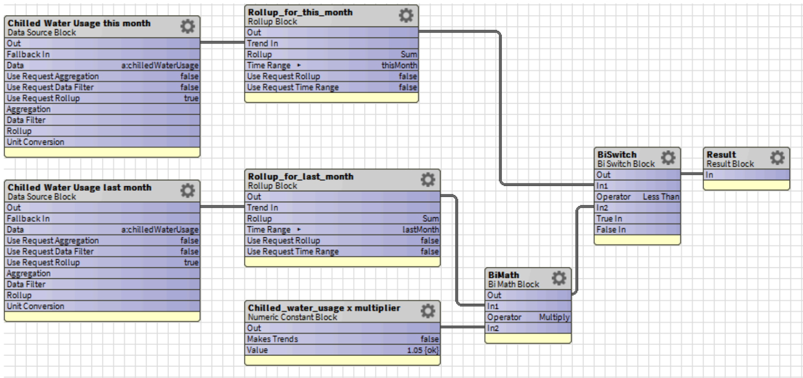
Input ORD

slot:/Algorithm/English/Chilled_Water_Usage_P_5_pc_of_last_Month

slot:/Algorithm/Metric/Chilled_Water_Usage_P_5_pc_of_last_Month

Wire sheet view

Figure 1. Chilled water usage wire sheet



Chilled Water Usage this month
Data Source Block
Out
Fallback In
Data a:chilledWaterUsage
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup true
Aggregation
Data Filter
Rollup
Unit Conversion

Rollup_for_this_month
Rollup Block
Out
Trend In
Rollup Sum
Time Range ▸ thisMonth
Use Request Rollup false
Use Request Time Range false

Chilled Water Usage last month
Data Source Block
Out
Fallback In
Data a:chilledWaterUsage
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup true
Aggregation
Data Filter
Rollup
Unit Conversion

Rollup_for_last_month
Rollup Block
Out
Trend In
Rollup Sum
Time Range ▸ lastMonth
Use Request Rollup false
Use Request Time Range false

Chilled_water_usage x multiplier
Numeric Constant Block
Out
Makes Trends false
Value 1.05 {ok}

BiMath
Bi Math Block
Out
In1
Operator Multiply
In2

BiSwitch
Bi Switch Block
Out
In1
Operator Less Than
In2
True In
False In

Result
Result Block
In

Parent topic: Algorithm library reference

Cooling_Valve_Open_100_Percent

This algorithm evaluates trend data. It produces a true or false result for historical data, if a building was occupied, the supply fan was on, the supply fan status indicated that the fan was on, and the cooling valve was wide open (100% open) for 30 minutes. A true result indicates that an exception existed; a false result that nothing was wrong.

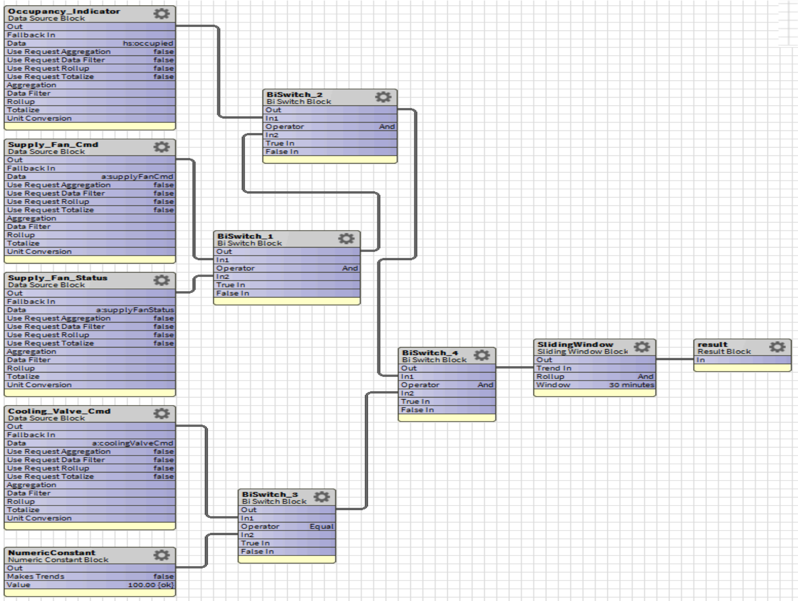
ORDs

slot:/Algorithm/English/Cooling_Valve_Open_100_Percent

slot:/Algorithm/Metric/Cooling_Valve_Open_100_Percent

Wire sheet

Figure 1. Cooling Valve Open 100% algorithm



Parent topic: Algorithm library reference

Damper_Open_During_Warmup

When an AHU is warming up in the morning, the outside air damper should not be on or open more than 5%.

This algorithm returns an exception if two conditions are met:

- An AHU is in warm-up mode and
- The outside air damper command is on or the outside damper position is more than 5% open.

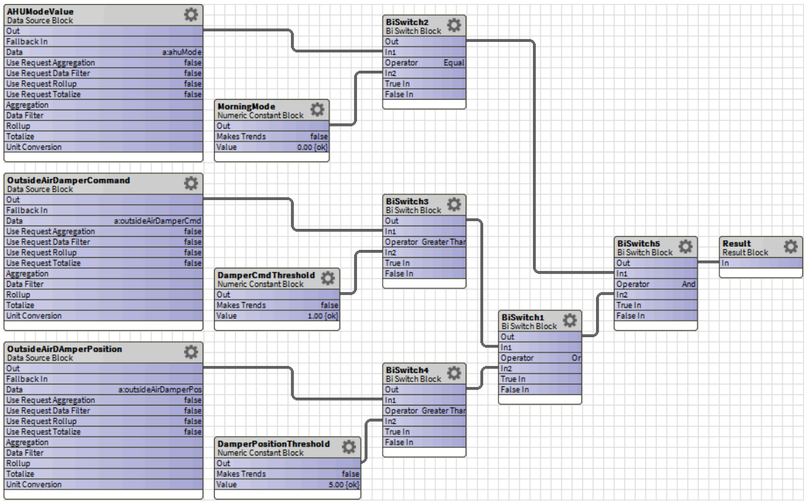
Input ORD

slot:/Algorithm/English/Damper_Open_During_Warmup

slot:/Algorithm/Metric/Damper_Open_During_Warmup

Wire sheet view

Figure 1. Damper open during warm-up wire sheet



Parent topic: Algorithm library reference

EconomizerNotRunning-EnthalpyBased

This algorithm returns true if, when a room or building is occupied, the economizer is off and the outside air enthalpy is less than the sum of the economizer setpoint and the return air enthalpy for more than 30 minutes.

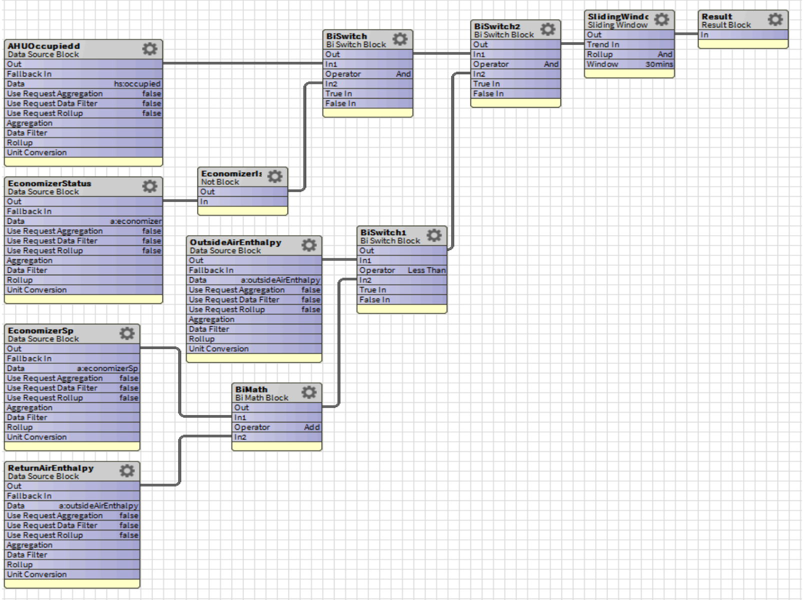
Input ORD

slot:/Algorithm/English/EconomizerNotRunning-EnthalpyBased

slot:/Algorithm/Metric/EconomizerNotRunning-EnthalpyBased

Wire sheet view

Figure 1. Economizer not running, enthalpy-based wire sheet



Parent topic: Algorithm library reference

EconomizerNotRunning-TempBased

This algorithm returns true if, when a room or building is occupied, the economizer is off and the outside air temperature is lower than the economizer setpoint for more than 30 minutes.

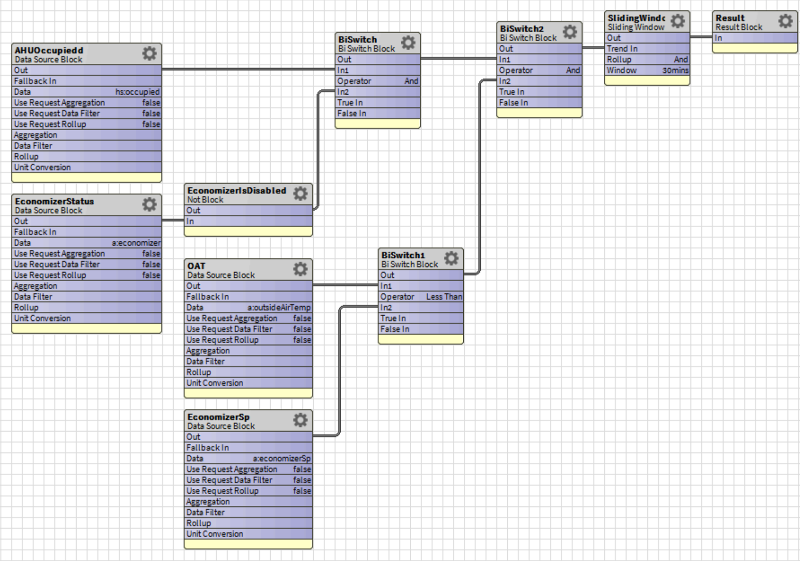
Input ORD

slot:/Algorithm/English/EconomizerNotRunning-TempBased

slot:/Algorithm/Metric/EconomizerNotRunning-TempBased

Wire sheet view

Figure 1. Economizer not running, temperature-based wire sheet



Parent topic: Algorithm library reference

Electric_EUI_Exceeds_Month_Target

This algorithm evaluates electricity usage over the period of this month. It returns true if the monthly electric EUI for the site exceeds the target EUI (Energy Use Intensity, or energy per square foot per year). EUI is based on the last available Area. A false result indicates that EUI did not exceed the monthly target.

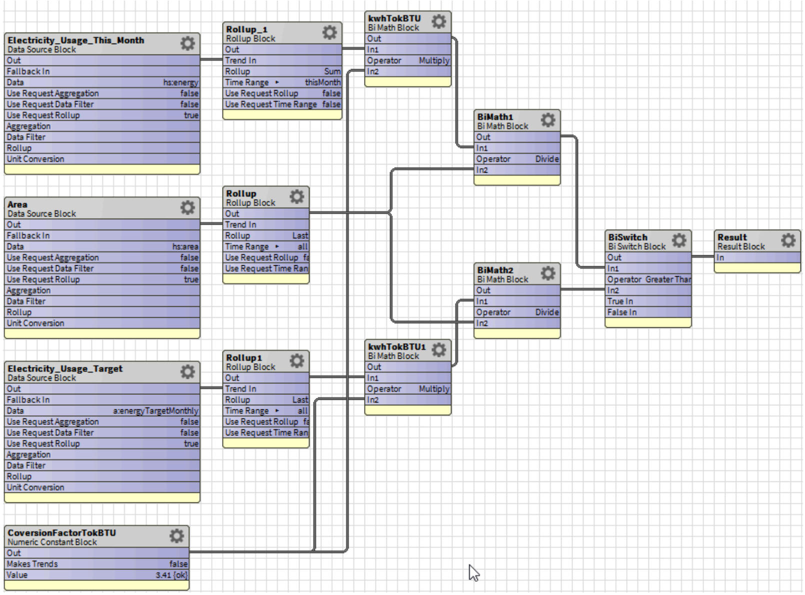
ORDs

slot:/Algorithm/English/Electric_EUI_Exceeds_Month_Target

slot:/Algorithm/Metric/Electric_EUI_Exceeds_Month_Target

Algorithm

Figure 1. Electric EUI Exceeds Month Target algorithm



Electricity_Usage_This_Month
Data Source Block
Out
Fallback In
Data hs:energy
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup true
Aggregation
Data Filter
Rollup
Unit Conversion

Area
Data Source Block
Out
Fallback In
Data hs:area
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup true
Aggregation
Data Filter
Rollup
Unit Conversion

Electricity_Usage_Target
Data Source Block
Out
Fallback In
Data a:energyTargetMonthly
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup true
Aggregation
Data Filter
Rollup
Unit Conversion

CoversionFactorTokBTU
Numeric Constant Block
Out
Makes Trends false
Value 3.41 {ok}

Rollup_1
Rollup Block
Out
Trend In
Rollup Sum
Time Range ▸ thisMonth
Use Request Rollup false
Use Request Time Range false

Rollup
Rollup Block
Out
Trend In
Rollup Last
Time Range ▸ all
Use Request Rollup fa
Use Request Time Ran

Rollup1
Rollup Block
Out
Trend In
Rollup Last
Time Range ▸ all
Use Request Rollup fa
Use Request Time Ran

kwhTokBTU
Bi Math Block
Out
In1
Operator Multiply
In2

kwhTokBTU1
Bi Math Block
Out
In1
Operator Multiply
In2

BiMath1
Bi Math Block
Out
In1
Operator Divide
In2

BiMath2
Bi Math Block
Out
In1
Operator Divide
In2

BiSwitch
Bi Switch Block
Out
In1
Operator Greater Thar
In2
True In
False In

Result
Result Block
In

Parent topic: Algorithm library reference

Electric_EUI_Exceeds_Yearly_Target

This algorithm evaluates electricity usage over the period of this year. It returns true if the yearly electric EUI for the site exceeds the target EUI (Energy Use Intensity, or energy per square foot per year). EUI is based on last available Area. A false result indicates that EUI did not exceed the yearly target.

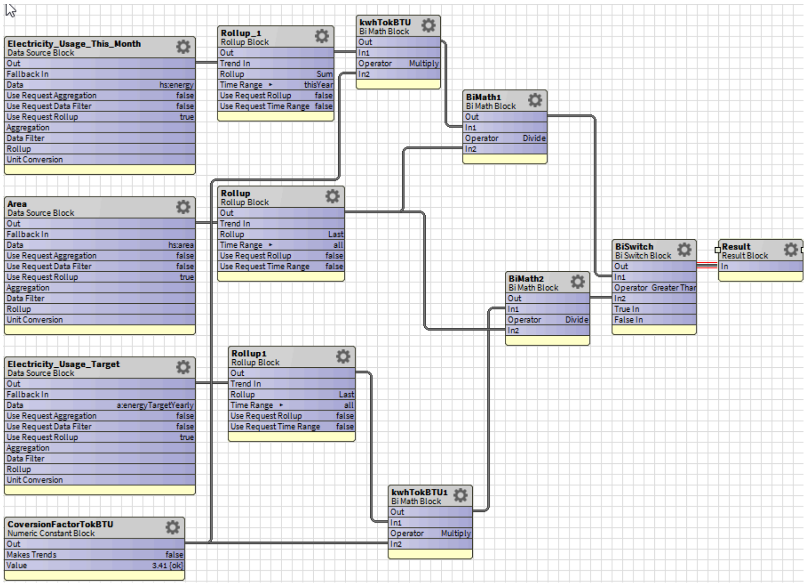
ORDs

slot:/Algorithm/English/Electric_EUI_Exceeds_Yearly_Target

slot:/Algorithm/Metric/Electric_EUI_Exceeds_Yearly_Target

Algorithm

Figure 1. Electric EUI Exceeds Yearly Target algorithm



Parent topic: Algorithm library reference

Electricity_Usage_5_Percent_Less_Than_Last_Year

This algorithm compares electricity usage this month with usage from the same month last year. It produces a true result, indicating that an exception exists, if the total monthly usage for this month is less than 95% of the total monthly usage for the same month last year. A false result indicates that nothing was wrong.

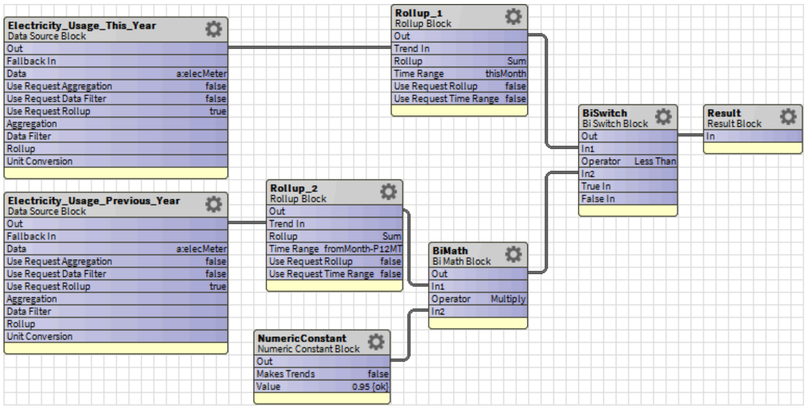
ORDs

slot:/Algorithm/English/Electricity_Usage_5_Percent_Less_Than_Last_Year

slot:/Algorithm/Metric/Electricity_Usage_5_Percent_Less_Than_Last_Year

Wire sheet

Figure 1. Electricity Usage Less Than 95% of Last Year



Parent topic: Algorithm library reference

Electricity_Usage_5_Percent_More_Than_Last_Year

This algorithm evaluates trend data for electricity usage. It returns a true result, indicating that an exception existed, if electricity usage for this month is more than electricity usage for the same month last year by more than 5%. A false result indicates that electricity usage was normal.

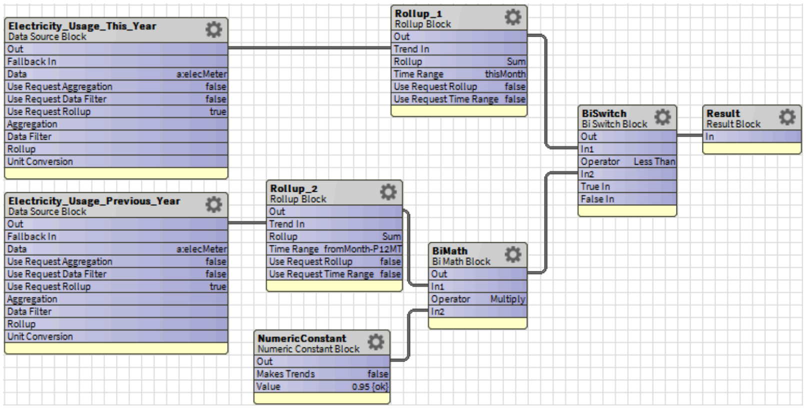
ORDs

slot:/Algorithm/English/Electricity_Usage_5_Percent_More_Than_Last_Year

slot:/Algorithm/Metric/Electricity_Usage_5_Percent_More_Than_Last_Year

Wire sheet

Figure 1. Electricity usage algorithm



Electricity_Usage_This_Year
Data Source Block
Out
Fallback In
Data a:elecMeter
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup true
Aggregation
Data Filter
Rollup
Unit Conversion

Electricity_Usage_Previous_Year
Data Source Block
Out
Fallback In
Data a:elecMeter
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup true
Aggregation
Data Filter
Rollup
Unit Conversion

Rollup_1
Rollup Block
Out
Trend In
Rollup Sum
Time Range thisMonth
Use Request Rollup false
Use Request Time Range false

Rollup_2
Rollup Block
Out
Trend In
Rollup Sum
Time Range fromMonth-P12MT
Use Request Rollup false
Use Request Time Range false

NumericConstant
Numeric Constant Block
Out
Makes Trends false
Value 0.95 {ok}

BiMath
Bi Math Block
Out
In1
Operator Multiply
In2

BiSwitch
Bi Switch Block
Out
In1
Operator Less Than
In2
True In
False In

Result
Result Block
In

Parent topic: Algorithm library reference

Heating_Valve_Open_100_Percent

This algorithm evaluates trend data. It produces a true or false result for historical data, if a building was occupied, the supply fan was on, the supply fan status indicated that the fan was on, and the heating valve was wide open (100% open) for 30 minutes. A true result indicates that an exception existed; a false result that nothing was wrong.

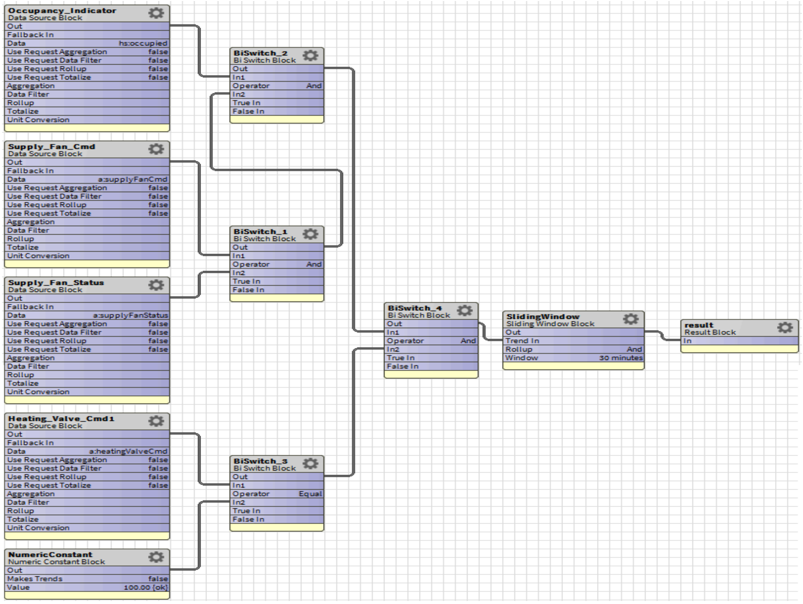
ORDs

slot:/Algorithm/English/Heating_Valve_Open_100_Percent

slot:/Algorithm/Metric/Heating_Valve_Open_100_Percent

Wire Chart

Figure 1. Heating Valve Open 100%



Parent topic: Algorithm library reference

HighHotWaterTemperature

This algorithm returns true if, when the hot water system is running, the hot water temperature is greater than the hot water supply temperature setpoint by plus five degrees for more than 30 minutes.

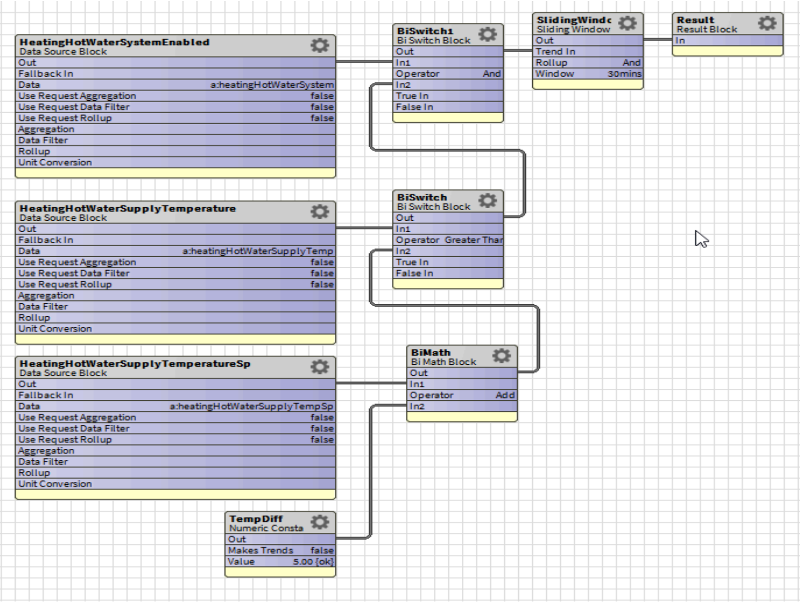
Input ORD

slot:/Algorithm/English/HighHotWaterTemperature

slot:/Algorithm/Metric/HighHotWaterTemperature

Wire sheet view

Figure 1. High hot water temperature wire sheet



Parent topic: Algorithm library reference

High_Chilled_Water_Temp

This algorithm analyzes trend data and returns a true result, indicating that an exception existed, if a chiller was commanded on, the chiller status (feedback) indicates it was running, and the chilled water supply temperature was greater than 1 degree Fahrenheit above the chilled water supply water temperature setpoint for 30 minutes. A false result indicates that nothing was wrong.

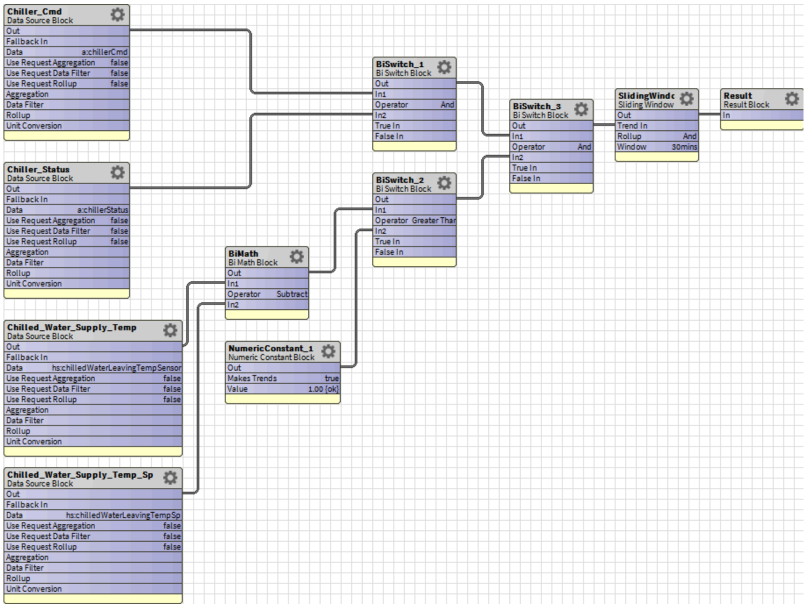
Input ORD

slot:/Algorithm/English/High_Chilled_Water_Temp

slot:/Algorithm/Metric/High_Chilled_Water_Temp

Wire sheet view

Figure 1. High Chilled Water Temp algorithm



Parent topic: Algorithm library reference

High_Outside_Air_Intake

This algorithm evaluates trend data collected from an AHU unit. It returns a true result if four conditions are met for an hour. Otherwise it returns false. The four conditions are: system was occupied, the command had been given to turn the supply fan on, the supply fan was on, and the outside air flow coming in was more than 100% of the outside airflow setpoint. A false result indicates that air coming in from the outside was normal.

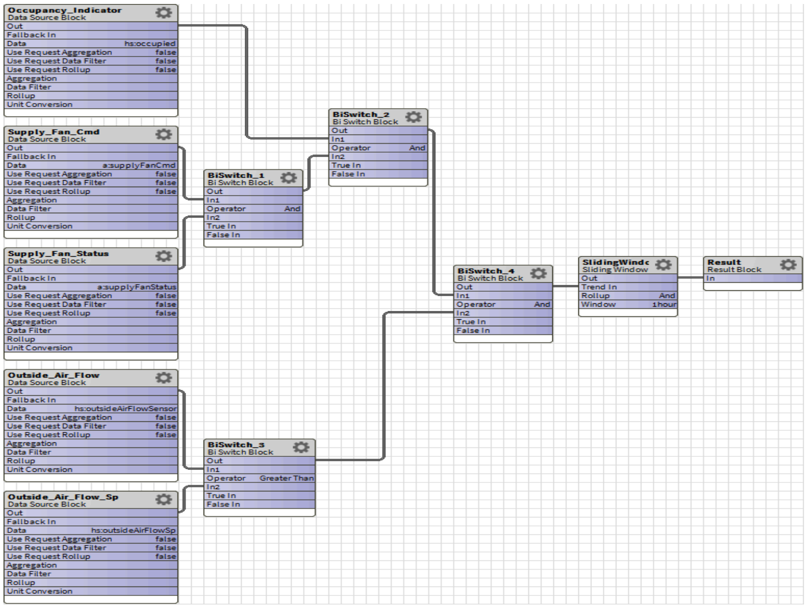
Ords

slot:/Algorithm/English/High_Outside_Air_Intake

slot:/Algorithm/Metric/High_Outside_Air_Intake

Algorithm

Figure 1. High Outside Air Intake algorithm



Parent topic: Algorithm library reference

High_Supply_Static_Pressure

This algorithm evaluates static pressure trend data. It produces a true result, indicating that an exception existed, if a building was occupied, an AHU unit's supply fan was on, the supply fan status indicated that the fan was on, the supply air static pressure was greater than 110% of the supply air static pressure stepoint, and all these conditions existed for 30 minutes. A false result indicates that nothing was wrong.

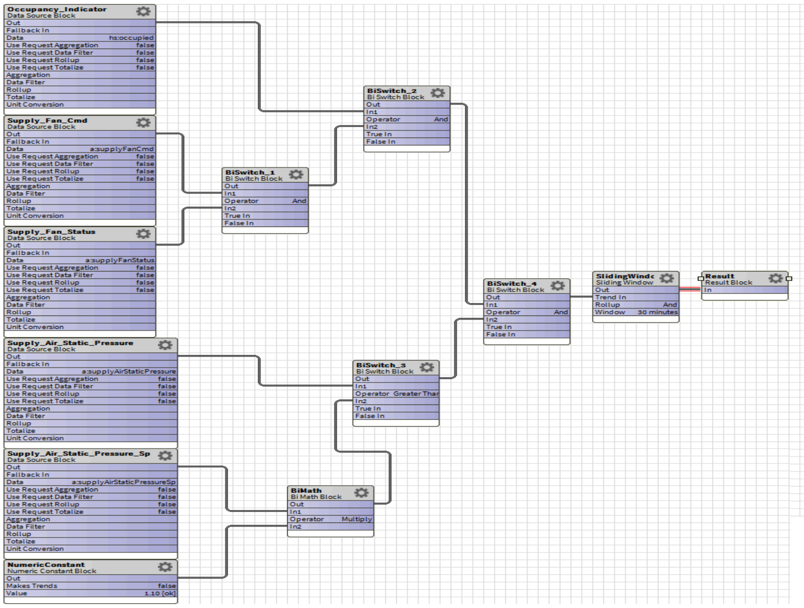
ORDs

slot:/Algorithm/English/High_Supply_Static_Pressure

slot:/Algorithm/Metric/High_Supply_Static_Pressure

Wire sheet

Figure 1. High Supply Static Pressure



Parent topic: Algorithm library reference

High_Supply_Temperature

This algorithm evaluates supply temperature trend data. It produces a true result if the difference between the supply air temperature and supply air temperature setpoint is greater than three degrees Fahrenheit. A false result indicates that nothing was wrong.

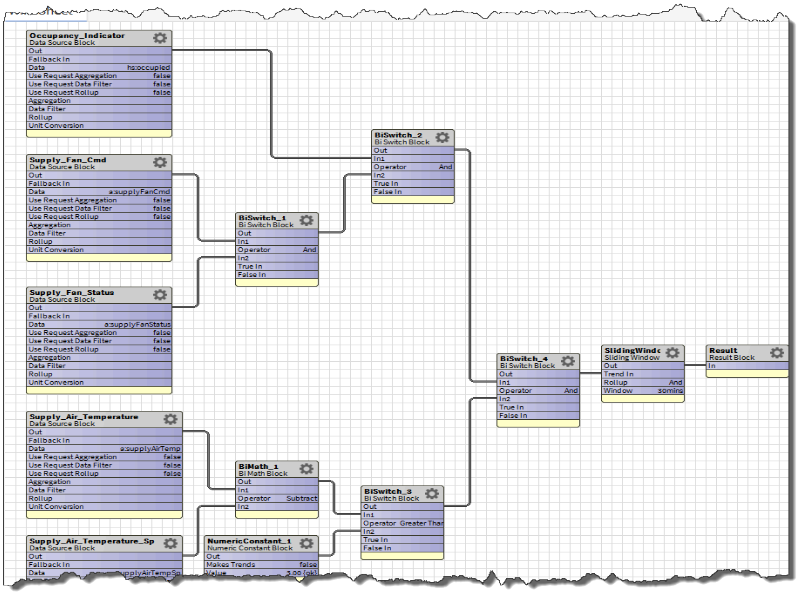
Input ORD

slot:/Algorithm/English/High_Supply_Temperature

slot:/Algorithm/Metric/High_Supply_Temperature

Wire sheet

Figure 1. Fragment of the High Supply Temperature algorithm



Parent topic: Algorithm library reference

HotWaterMixing

This algorithm returns true if, when the hot water system is running, the hot water supply temperature minus the hot water return temperature setpoint is less than five degrees for more than 30 minutes.

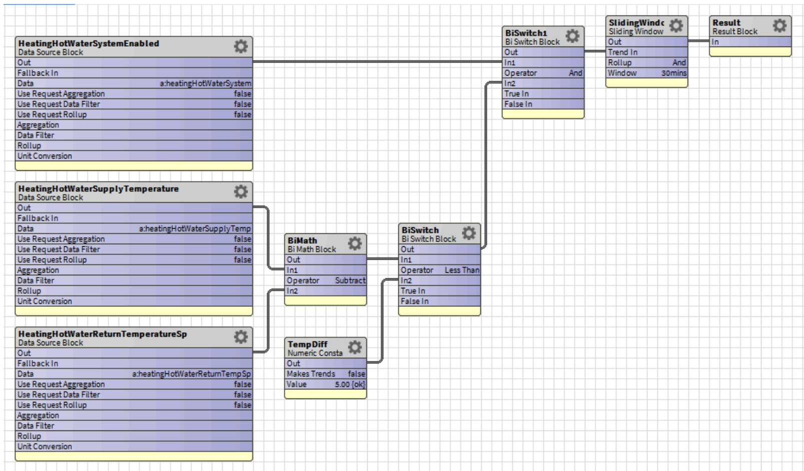
Input ORD

slot:/Algorithm/English/HotWaterMixing

slot:/Algorithm/Metric/HotWaterMixing

Wire sheet view

Figure 1. Hot water mixing wire sheet



Parent topic: Algorithm library reference

Low_Chilled_Water_Temp

This algorithm evaluates trend data and returns a true result, indicating that an exception existed, if a chiller was commanded on, the chiller status (feedback) indicates it was running, and the chilled water supply temperature was greater than 1 degree Fahrenheit below the chilled water supply water temperature setpoint for 30 minutes. A false result indicates that nothing was wrong.

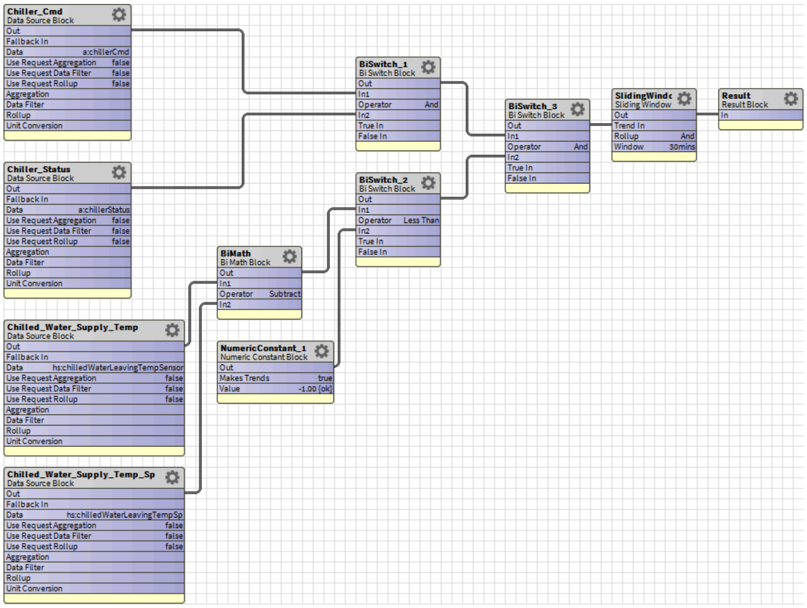
Input ORD

slot:/Algorithm/English/Low_Chilled_Water_Temp

slot:/Algorithm/English/Low_Chilled_Water_Temp

Wire sheet view

Figure 1. Low Chilled Water Temp algorithm



Chiller_Cmd
Data Source Block
Out
Fallback In
Data a:chillerCmd
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Aggregation
Data Filter
Rollup
Unit Conversion

Chiller_Status
Data Source Block
Out
Fallback In
Data a:chillerStatus
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Aggregation
Data Filter
Rollup
Unit Conversion

Chilled_Water_Supply_Temp
Data Source Block
Out
Fallback In
Data hs:chilledWaterLeavingTempSensor
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Aggregation
Data Filter
Rollup
Unit Conversion

Chilled_Water_Supply_Temp_Sp
Data Source Block
Out
Fallback In
Data hs:chilledWaterLeavingTempSp
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Aggregation
Data Filter
Rollup
Unit Conversion

BiMath
Bi Math Block
Out
In1
Operator Subtract
In2

NumericConstant_1
Numeric Constant Block
Out
Makes Trends true
Value -1.00 {ok}

BiSwitch_1
Bi Switch Block
Out
In1
Operator And
In2
True In
False In

BiSwitch_2
Bi Switch Block
Out
In1
Operator Less Than
In2
True In
False In

BiSwitch_3
Bi Switch Block
Out
In1
Operator And
In2
True In
False In

SlidingWindc
Sliding Window
Out
Trend In
Rollup And
Window 30mins

Result
Result Block
In

Parent topic: Algorithm library reference

LowHotWaterTemperature

This algorithm returns true if, when the hot water system is running, the hot water temperature is less than the hot water supply temperature setpoint by more than five degrees for more than 30 minutes.

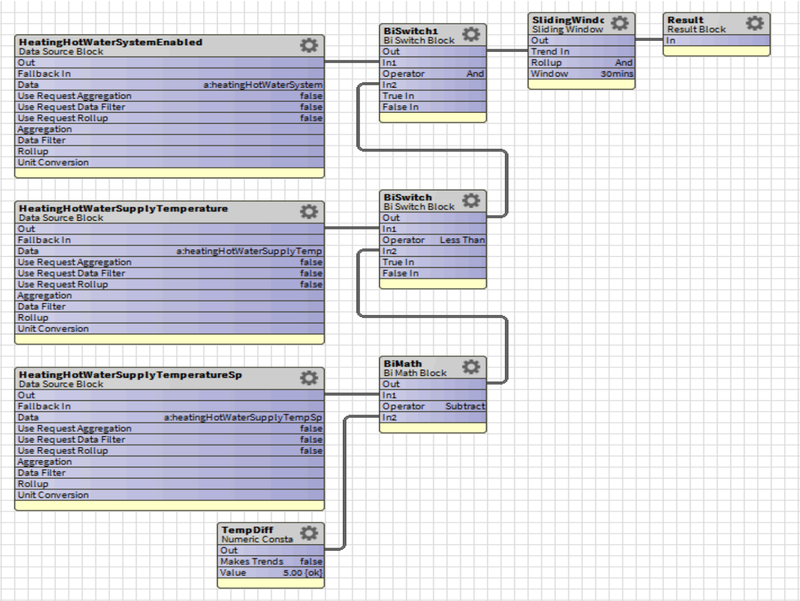
Input ORD

slot:/Algorithm/English/LowHotWaterTemperature

slot:/Algorithm/Metric/LowHotWaterTemperature

Wire sheet view

Figure 1. Low hot water temperature wire sheet



HeatingHotWaterSystemEnabled
Data Source Block
Out
Fallback In
Data a:heatingHotWaterSystem
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Aggregation
Data Filter
Rollup
Unit Conversion

HeatingHotWaterSupplyTemperature
Data Source Block
Out
Fallback In
Data a:heatingHotWaterSupplyTemp
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Aggregation
Data Filter
Rollup
Unit Conversion

HeatingHotWaterSupplyTemperatureSp
Data Source Block
Out
Fallback In
Data a:heatingHotWaterSupplyTempSp
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Aggregation
Data Filter
Rollup
Unit Conversion

TempDiff
Numeric Consta
Out
Makes Trends false
Value 5.00 {ok}

BiSwitch1
Bi Switch Block
Out
In1
Operator And
In2
True In
False In

BiSwitch
Bi Switch Block
Out
In1
Operator Less Than
In2
True In
False In

BiMath
Bi Math Block
Out
In1
Operator Subtract
In2

SlidingWindc
Sliding Window
Out
Trend In
Rollup And
Window 30mins

Result
Result Block
In

Parent topic: Algorithm library reference

Low_Outside_Air_Intake

This algorithm evaluates trend data collected from an AHU unit. It returns a true result if four conditions are met for an hour. Otherwise, it returns false. The four conditions are: was occupied, the command had been given to turn the supply fan on, the supply fan was on, and the outside air flow coming in was less than 90% of the outside airflow setpoint. A false result indicates that air coming in from the outside was normal.

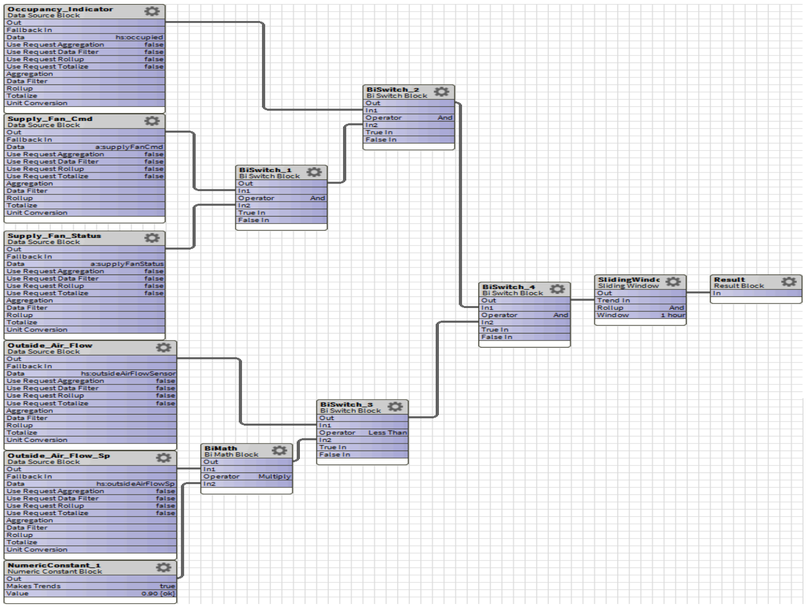
ORDs

slot:/Algorithm/English/Low_Outside_Air_Intake

slot:/Algorithm/Metric/Low_Outside_Air_Intake

Wire sheet

Figure 1. Low Outside Air Intake algorithm



Parent topic: Algorithm library reference

Low_Supply_Static_Pressure

This algorithm evaluates static pressure trend data. It produces a true result, indicating that an exception existed, if a building was occupied, an AHU unit's supply fan was on, the supply fan status indicated that the fan was on, the supply air static pressure was less than 90% of the supply air static pressure stepoint, and all these conditions existed for 30 minutes. A false result indicates that nothing was wrong.

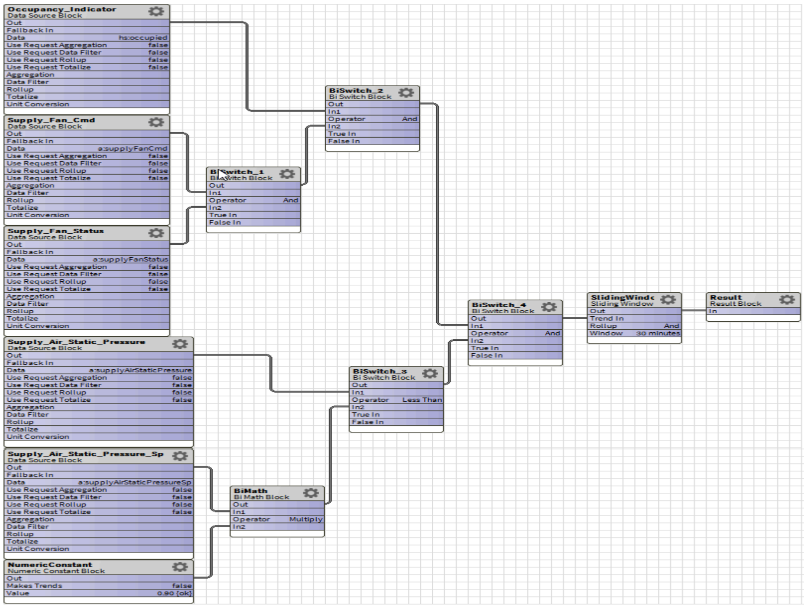
ORDs

slot:/Algorithm/English/Low_Supply_Static_Pressure

slot:/Algorithm/Metric/Low_Supply_Static_Pressure

Wire sheet

Figure 1. Low Supply Static Pressure algorithm



Occupancy_Indicator
Data Source Block
Out
Fallback In
Data hs:occupied
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Use Request Totalize false
Aggregation
Data Filter
Rollup
Totalize
Unit Conversion

Supply_Fan_Cmd
Data Source Block
Out
Fallback In
Data a:supplyFanCmd
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Use Request Totalize false
Aggregation
Data Filter
Rollup
Totalize
Unit Conversion

Supply_Fan_Status
Data Source Block
Out
Fallback In
Data a:supplyFanStatus
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Use Request Totalize false
Aggregation
Data Filter
Rollup
Totalize
Unit Conversion

Supply_Air_Static_Pressure
Data Source Block
Out
Fallback In
Data a:supplyAirStaticPressure
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Use Request Totalize false
Aggregation
Data Filter
Rollup
Totalize
Unit Conversion

Supply_Air_Static_Pressure_Sp
Data Source Block
Out
Fallback In
Data a:supplyAirStaticPressureSp
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Use Request Totalize false
Aggregation
Data Filter
Rollup
Totalize
Unit Conversion

NumericConstant
Numeric Constant Block
Out
Makes Trends false
Value 0.90 {ok}

BiSwitch_1
BiSwitch Block
Out
In1
Operator And
In2
True In
False In

BiSwitch_2
Bi Switch Block
Out
In1
Operator And
In2
True In
False In

BiMath
Bi Math Block
Out
In1
Operator Multiply
In2

BiSwitch_3
Bi Switch Block
Out
In1
Operator Less Than
In2
True In
False In

BiSwitch_4
Bi Switch Block
Out
In1
Operator And
In2
True In
False In

SlidingWindc
Sliding Window
Out
Trend In
Rollup And
Window 30 minutes

Result
Result Block
In

Parent topic: Algorithm library reference

Low_Supply_Temperature

This algorithm evaluates supply temperature trend data. It produces a true result if the supply air temperature is more than three degrees Fahrenheit below the supply air temperature setpoint. A false result indicates that nothing was wrong.

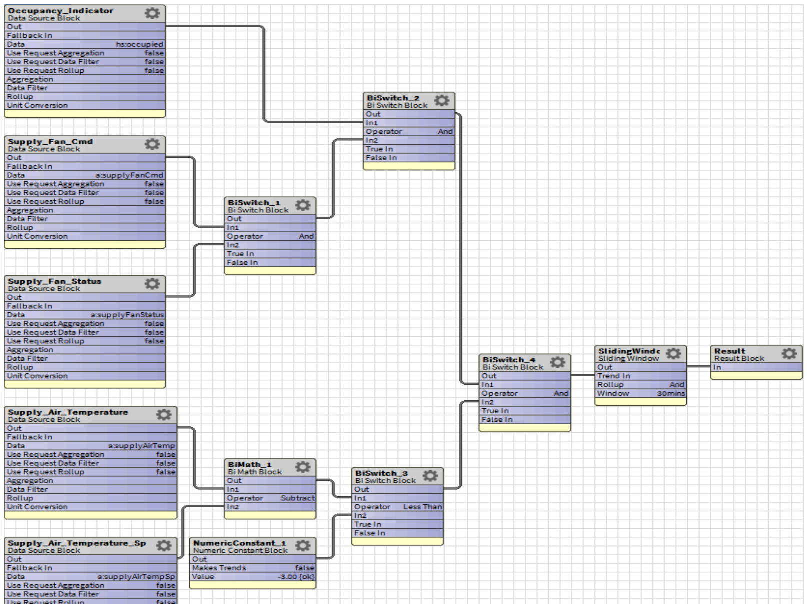
ORDs

slot:/Algorithm/English/Low_Supply_Temperature

slot:/Algorithm/Metric/Low_Supply_Temperature

Wire sheet

Figure 1. Low Supply Temperature algorithm



Parent topic: Algorithm library reference

NonConformanceSpaceHumidity

This algorithm returns true, when a room is occupied, the command to turn the supply fan on was issued, the supply fan is on, and the humidity is greater than the VAHHLMT (Variable Air Humidity High Limit) or less than the VAHLLMT (Variable Air Humidity Low Limit).

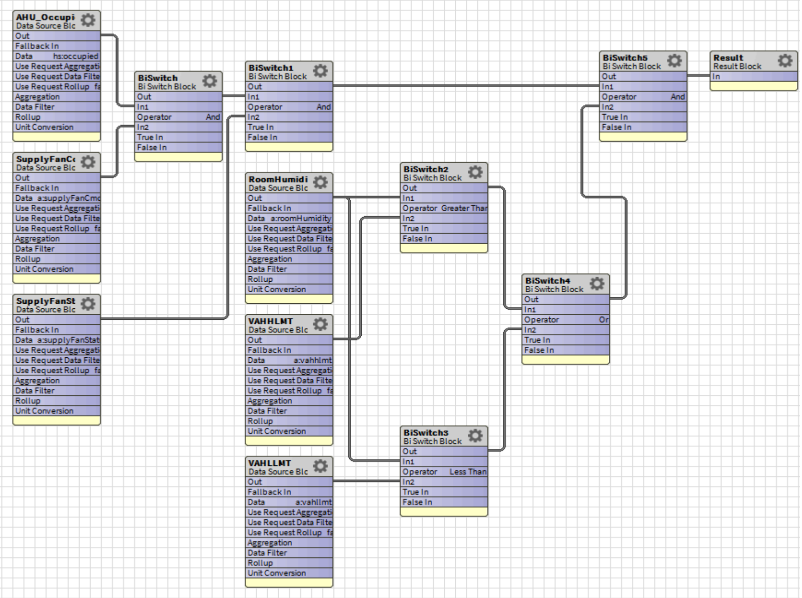
Input ORD

slot:/Algorithm/English/NonConformanceSpaceHumid

slot:/Algorithm/Metric/NonConformanceSpaceHumid

Wire sheet view

Figure 1. Non-conformance space humidity wire sheet



Parent topic: Algorithm library reference

NonConformanceSpaceTemperature

This algorithm returns true, when a room is occupied, the command to turn the supply fan on was issued, the supply fan is on, and the room temperature is greater than the VATHLMT (Variable Air Temperature High Limit) or less than the VATLLMT (Variable Air Temperature Low Limit).

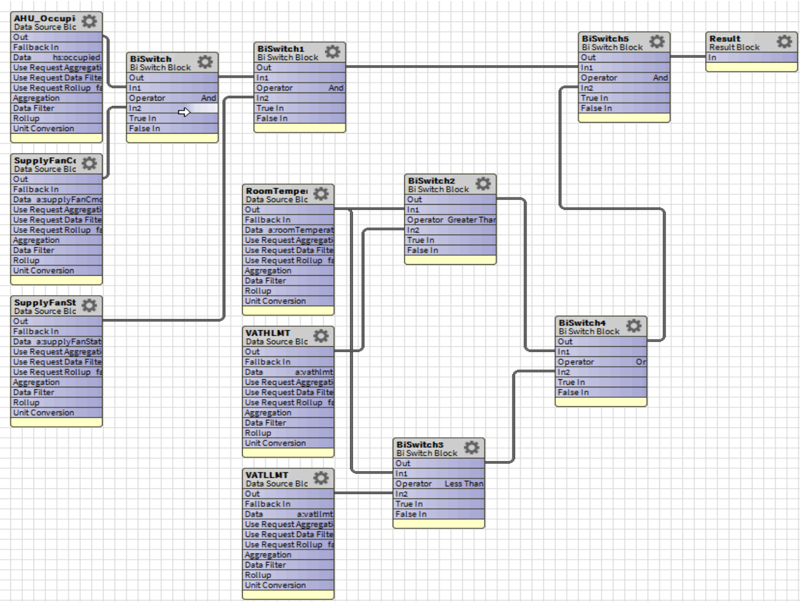
Input ORD

slot:/Algorithm/English/NonConformanceSpaceTemp

slot:/Algorithm/Metric/NonConformanceSpaceTemp

Wire sheet view

Figure 1. Non-conformance space temperature wire sheet



Parent topic: Algorithm library reference

Prolonged_100_Percent_VAV_Damper

This algorithm evaluates trend data collected from an AHU unit. It returns true if the facility was occupied, the command to turn the supply fan on was given, the supply fan was on, and the position of the damper was more than 95% open for 30 minutes. A false result indicates that nothing was wrong.

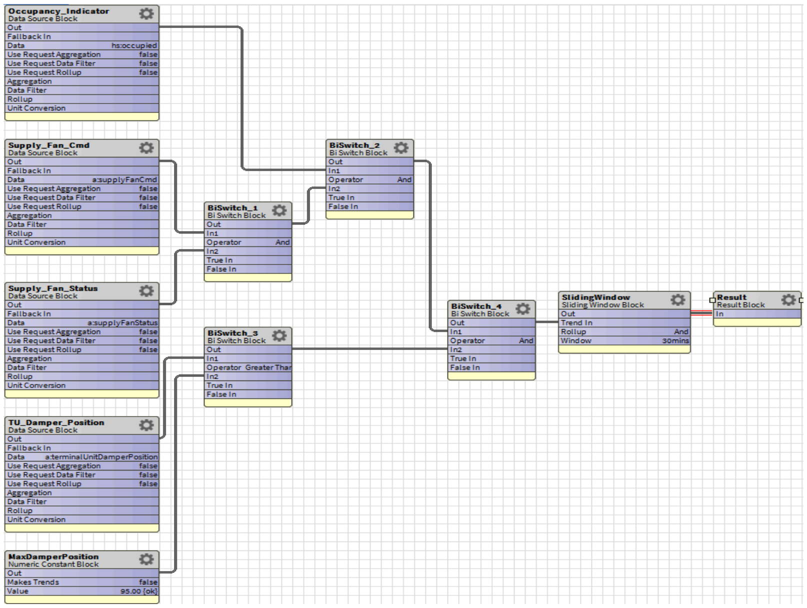
ORDs

slot:/Algorithm/English/Prolonged_100_Percent_VAV_Damper

slot:/Algorithm/Metric/Prolonged_100_Percent_VAV_Damper

Algorithm

Figure 1. Prolonged 100% VAV Damper algorithm



Occupancy_Indicator
Data Source Block
Out
Fallback In
Data hs:occupied
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Aggregation
Data Filter
Rollup
Unit Conversion

Supply_Fan_Cmd
Data Source Block
Out
Fallback In
Data a:supplyFanCmd
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Aggregation
Data Filter
Rollup
Unit Conversion

Supply_Fan_Status
Data Source Block
Out
Fallback In
Data a:supplyFanStatus
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Aggregation
Data Filter
Rollup
Unit Conversion

TU_Damper_Position
Data Source Block
Out
Fallback In
Data a:terminalUnitDamperPosition
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Aggregation
Data Filter
Rollup
Unit Conversion

MaxDamperPosition
Numeric Constant Block
Out
Makes Trends false
Value 95.00 {ok}

BiSwitch_1
Bi Switch Block
Out
In1
Operator And
In2
True In
False In

BiSwitch_3
Bi Switch Block
Out
In1
Operator Greater Thar
In2
True In
False In

BiSwitch_2
Bi Switch Block
Out
In1
Operator And
In2
True In
False In

BiSwitch_4
Bi Switch Block
Out
In1
Operator And
In2
True In
False In

SlidingWindow
Sliding Window Block
Out
Trend In
Rollup And
Window 30mins

Result
Result Block
In

Parent topic: Algorithm library reference

PumpExceedingSetpoint

This algorithm uses three inputs to determine if the pressure in a secondary chilled water pump exceeds the expected setpoint.

The conditions that must be met for this algorithm to return a value of true are:

- Secondary chilled water pump command is on.
- Secondary chilled water pump is on.
- Secondary loop differential pressure is above 110% the setpoint for more than 30 minutes.

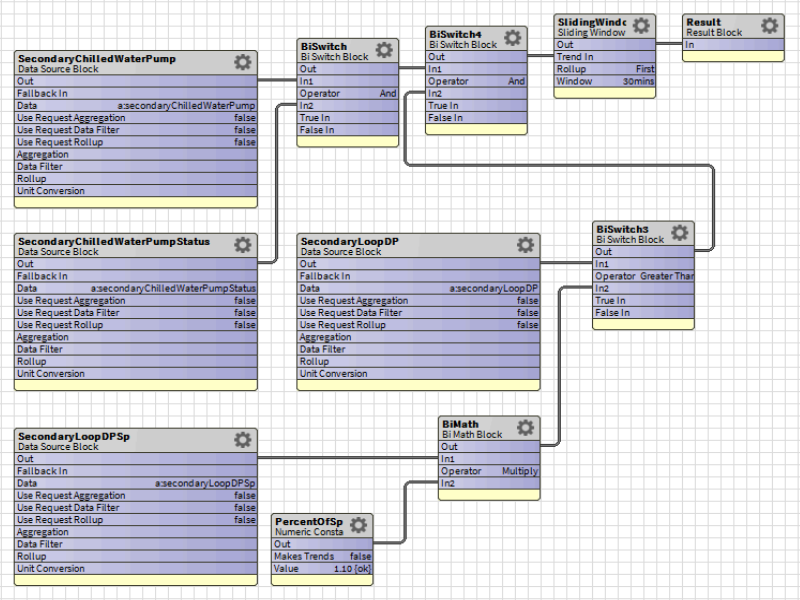
Input ORD

slot:/Algorithm/English/PumpExceedingSetpoint

slot:/Algorithm/Metric/PumpExceedingSetpoint

Wire sheet view

Figure 1. Pump pressure above setpoint wire sheet



SecondaryChilledWaterPump
Data Source Block
Out
Fallback In
Data a:secondaryChilledWaterPump
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Aggregation
Data Filter
Rollup
Unit Conversion

SecondaryChilledWaterPumpStatus
Data Source Block
Out
Fallback In
Data a:secondaryChilledWaterPumpStatus
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Aggregation
Data Filter
Rollup
Unit Conversion

SecondaryLoopDPSp
Data Source Block
Out
Fallback In
Data a:secondaryLoopDPSp
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Aggregation
Data Filter
Rollup
Unit Conversion

BiSwitch
Bi Switch Block
Out
In1
Operator And
In2
True In
False In

BiSwitch4
Bi Switch Block
Out
In1
Operator And
In2
True In
False In

SlidingWindc
Sliding Window
Out
Trend In
Rollup First
Window 30mins

Result
Result Block
In

SecondaryLoopDP
Data Source Block
Out
Fallback In
Data a:secondaryLoopDP
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Aggregation
Data Filter
Rollup
Unit Conversion

BiSwitch3
Bi Switch Block
Out
In1
Operator Greater Thar
In2
True In
False In

BiMath
Bi Math Block
Out
In1
Operator Multiply
In2

PercentOfSp
Numeric Consta
Out
Makes Trends false
Value 1.10 {ok}

Parent topic: Algorithm library reference

PumpNotMeetingSetpoint

This algorithm uses four inputs to determine if the pressure in a secondary chilled water pump is below the expected setpoint.

The conditions that must be met for this algorithm to return a value of true are:

- Secondary chilled water pump command is on.
- Secondary chilled water pump is on (feedback indicates the pump is running).
- Secondary chilled water pump speed is 60 Hz (100%).
- Secondary loop differential pressure is below 90% of the setpoint for more than 30 minutes.

Input ORD

There are two ORDs for both English and Metric folders for an algorithm based on speed in Hz and speed in percentage.

slot:/Algorithm/English/PumpNotMeetingSetpoint

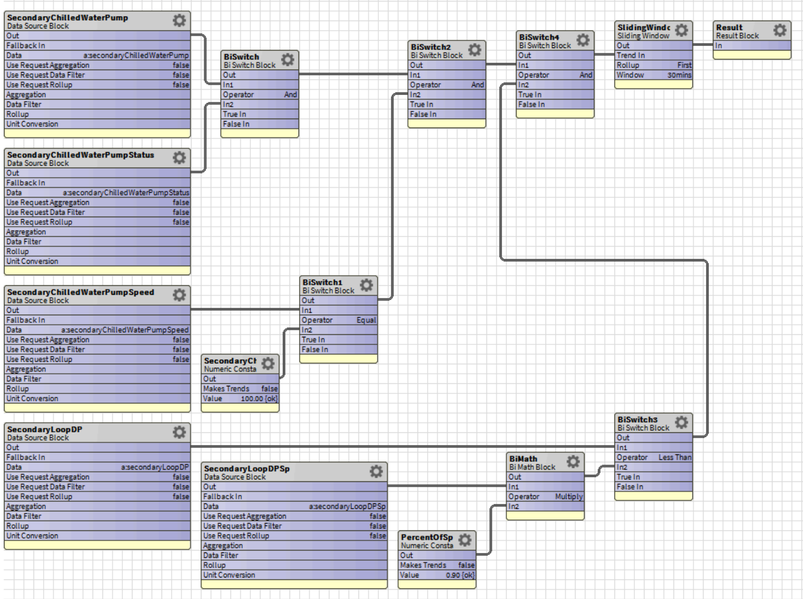
slot:/Algorithm/English/PumpNotMeetingSetpoint_SpeedInPercentage

slot:/Algorithm/Metric/PumpNotMeetingSetpoint

slot:/Algorithm/Metric/PumpNotMeetingSetpoint_SpeedInPercentage

Wire sheet view

Figure 1. Pump pressure below setpoint wire sheet



SecondaryChilledWaterPump
Data Source Block
Out
Fallback In
Data a:secondaryChilledWaterPump
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Aggregation
Data Filter
Rollup
Unit Conversion

SecondaryChilledWaterPumpStatus
Data Source Block
Out
Fallback In
Data a:secondaryChilledWaterPumpStatus
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Aggregation
Data Filter
Rollup
Unit Conversion

SecondaryChilledWaterPumpSpeed
Data Source Block
Out
Fallback In
Data a:secondaryChilledWaterPumpSpeed
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Aggregation
Data Filter
Rollup
Unit Conversion

SecondaryLoopDP
Data Source Block
Out
Fallback In
Data a:secondaryLoopDP
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Aggregation
Data Filter
Rollup
Unit Conversion

BiSwitch
Bi Switch Block
Out
In1
Operator And
In2
True In
False In

SecondaryCh
Numeric Consta
Out
Makes Trends false
Value 100.00 {ok}

BiSwitch1
Bi Switch Block
Out
In1
Operator Equal
In2
True In
False In

SecondaryLoopDPSp
Data Source Block
Out
Fallback In
Data a:secondaryLoopDPSp
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Aggregation
Data Filter
Rollup
Unit Conversion

PercentOfSp
Numeric Consta
Out
Makes Trends false
Value 0.90 {ok}

BiSwitch2
Bi Switch Block
Out
In1
Operator And
In2
True In
False In

BiMath
Bi Math Block
Out
In1
Operator Multiply
In2

BiSwitch4
Bi Switch Block
Out
In1
Operator And
In2
True In
False In

BiSwitch3
Bi Switch Block
Out
In1
Operator Less Than
In2
True In
False In

SlidingWindo
Sliding Window
Out
Trend In
Rollup First
Window 30mins

Result
Result Block
In

Parent topic: Algorithm library reference

Steam_Usage_Minus_10_Percent_of_last_Month

This algorithm compares steam usage this month with usage from the previous month. It produces a true result, indicating that an exception exists, if the total steam usage for this month is less than 90% of the total monthly usage for the previous month.

Input ORD

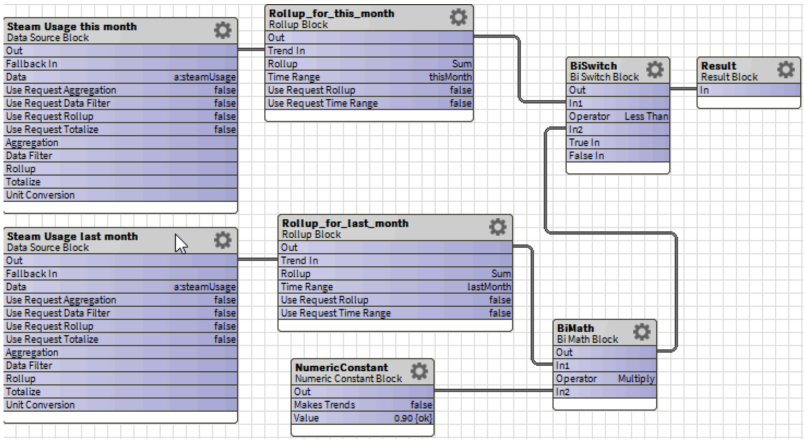
slot:/Algorithm/English/Steam_Usage_minus_10_Percent_of_last_Month

slot:/Algorithm/Metric/Steam_Usage_minus_10_Percent_of_last_Month

The algorithm uses two rollup blocks, numeric constant, calculation and switch block to calculate the result.

Wire Sheet

Figure 1. Steam usage minus 10 percent of last month wire sheet



Steam Usage this month
Data Source Block
Out
Fallback In
Data a:steamUsage
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Use Request Totalize false
Aggregation
Data Filter
Rollup
Totalize
Unit Conversion

Rollup_for_this_month
Rollup Block
Out
Trend In
Rollup Sum
Time Range thisMonth
Use Request Rollup false
Use Request Time Range false

BiSwitch
Bi Switch Block
Out
In1
Operator Less Than
In2
True In
False In

Result
Result Block
In

Steam Usage last month
Data Source Block
Out
Fallback In
Data a:steamUsage
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Use Request Totalize false
Aggregation
Data Filter
Rollup
Totalize
Unit Conversion

Rollup_for_last_month
Rollup Block
Out
Trend In
Rollup Sum
Time Range lastMonth
Use Request Rollup false
Use Request Time Range false

NumericConstant
Numeric Constant Block
Out
Makes Trends false
Value 0.90 {ok}

BiMath
Bi Math Block
Out
In1
Operator Multiply
In2

Parent topic: Algorithm library reference

Steam_Usage_Minus_5_Percent_of_last_Month

This algorithm compares steam usage this month with usage from the previous month. It produces a true result, indicating that an exception exists, if the total steam usage for this month is less than 95% of the total monthly usage for the previous month.

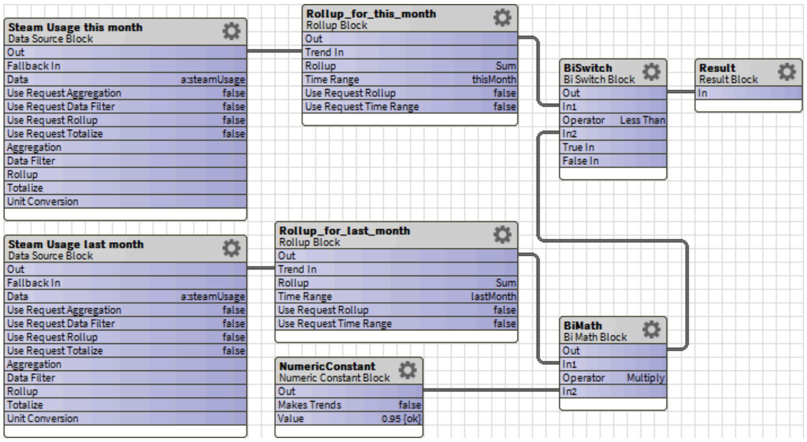
Input ORD

slot:/Algorithm/English/Steam_Usage_minus_5_Percent_of_last_Month

slot:/Algorithm/Metric/Steam_Usage_minus_5_Percent_of_last_Month

The algorithm uses two rollup blocks, numeric constant, calculation and switch block to calculate the result.

Figure 1. Steam usage minus 5 percent of last month wire sheet



Parent topic: Algorithm library reference

Steam_Usage_Plus_10_Percent_of_last_Month

This algorithm compares steam usage this month with usage from the previous month. It produces a true result, indicating that an exception exists, if the total steam usage for this month is greater than the total monthly usage for the previous month by more than 10%.

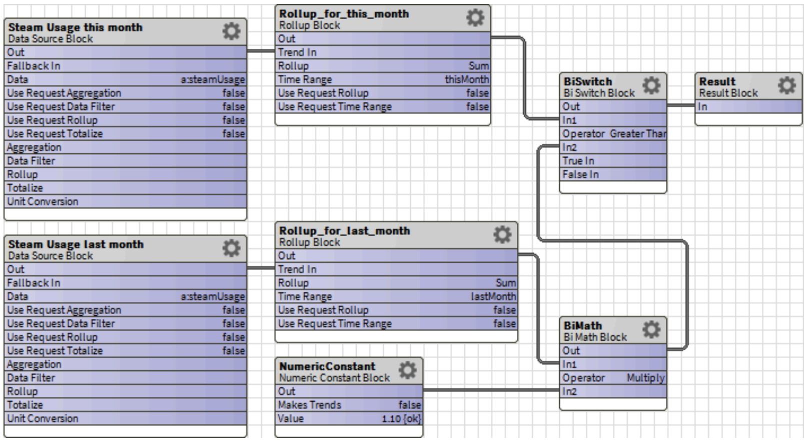
Input ORD

slot:/Algorithm/English/Steam_Usage_plus_10_Percent_of_last_Month

slot:/Algorithm/Metric/Steam_Usage_plus_10_Percent_of_last_Month

The algorithm uses two rollup blocks, numeric constant, calculation and switch block to calculate the result.

Figure 1. Steam usage plus 10 percent of last month wire sheet



Steam Usage this month
Data Source Block
Out
Fallback In
Data a:steamUsage
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Use Request Totalize false
Aggregation
Data Filter
Rollup
Totalize
Unit Conversion

Rollup_for_this_month
Rollup Block
Out
Trend In
Rollup Sum
Time Range thisMonth
Use Request Rollup false
Use Request Time Range false

BiSwitch
Bi Switch Block
Out
In1
Operator Greater Thar
In2
True In
False In

Result
Result Block
In

Steam Usage last month
Data Source Block
Out
Fallback In
Data a:steamUsage
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Use Request Totalize false
Aggregation
Data Filter
Rollup
Totalize
Unit Conversion

Rollup_for_last_month
Rollup Block
Out
Trend In
Rollup Sum
Time Range lastMonth
Use Request Rollup false
Use Request Time Range false

NumericConstant
Numeric Constant Block
Out
Makes Trends false
Value 1.10 {ok}

BiMath
Bi Math Block
Out
In1
Operator Multiply
In2

Parent topic: Algorithm library reference

Steam_Usage_Plus_5_Percent_of_last_Month

This algorithm compares steam usage this month with usage from the previous month. It produces a true result, indicating that an exception exists, if the total steam usage for this month is greater than the total monthly usage for the previous month by more than 5%.

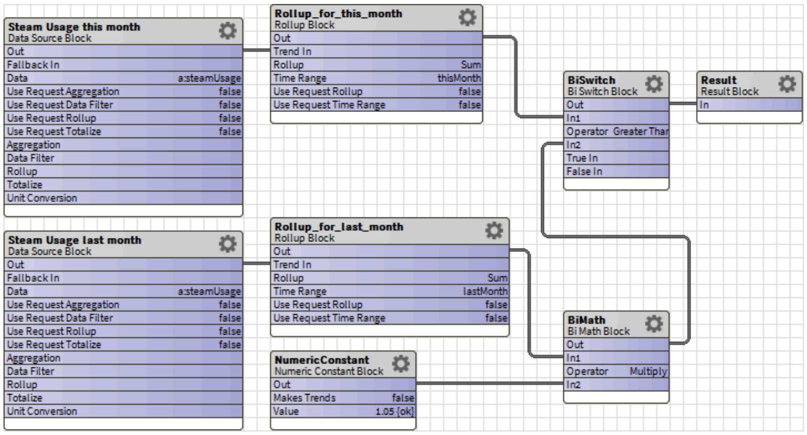
Input ORD

slot:/Algorithm/English/Steam_Usage_plus_5_Percent_of_last_Month

slot:/Algorithm/Metric/Steam_Usage_plus_5_Percent_of_last_Month

The algorithm uses two rollup blocks, numeric constant, calculation and switch block to calculate the result.

Figure 1. Steam usage plus 5 percent of last month wire sheet



Steam Usage this month
Data Source Block
Out
Fallback In
Data a:steamUsage
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Use Request Totalize false
Aggregation
Data Filter
Rollup
Totalize
Unit Conversion

Rollup_for_this_month
Rollup Block
Out
Trend In
Rollup Sum
Time Range thisMonth
Use Request Rollup false
Use Request Time Range false

BiSwitch
Bi Switch Block
Out
In1
Operator Greater Thar
In2
True In
False In

Result
Result Block
In

Steam Usage last month
Data Source Block
Out
Fallback In
Data a:steamUsage
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Use Request Totalize false
Aggregation
Data Filter
Rollup
Totalize
Unit Conversion

Rollup_for_last_month
Rollup Block
Out
Trend In
Rollup Sum
Time Range lastMonth
Use Request Rollup false
Use Request Time Range false

NumericConstant
Numeric Constant Block
Out
Makes Trends false
Value 1.05 {ok}

BiMath
Bi Math Block
Out
In1
Operator Multiply
In2

Parent topic: Algorithm library reference

Steam_Water_Usage_Minus_10_PC_of_last_Month

This algorithm compares steam water usage this month with usage from the previous month. It produces a true result, indicating that an exception exists, if the total steam water usage for this month is less than 90% of the total monthly usage for the previous month.

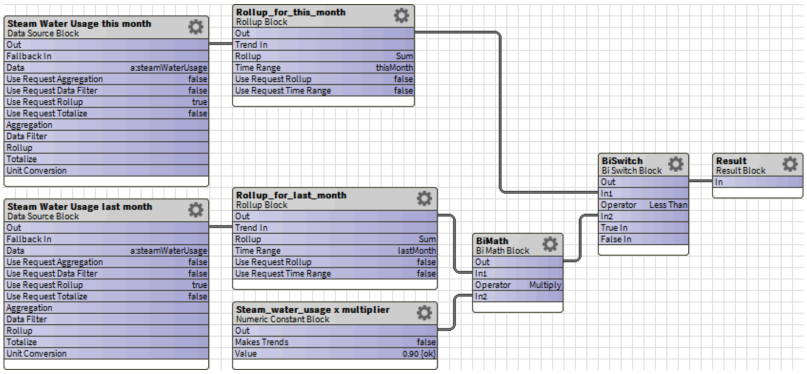
Input ORD

slot:/Algorithm/English/Steam_Water_Usage_Minus_10_PC_of_last_Month

slot:/Algorithm/Metric/Steam_Water_Usage_Minus_10_PC_of_last_Month

The algorithm uses two rollup blocks, numeric constant, calculation and switch block to calculate the result.

Figure 1. Water usage minus 10 percent of last month wire sheet



Steam Water Usage this month
Data Source Block
Out
Fallback In
Data a:steamWaterUsage
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup true
Use Request Totalize false
Aggregation
Data Filter
Rollup
Totalize
Unit Conversion

Rollup_for_this_month
Rollup Block
Out
Trend In
Rollup Sum
Time Range thisMonth
Use Request Rollup false
Use Request Time Range false

Steam Water Usage last month
Data Source Block
Out
Fallback In
Data a:steamWaterUsage
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup true
Use Request Totalize false
Aggregation
Data Filter
Rollup
Totalize
Unit Conversion

Rollup_for_last_month
Rollup Block
Out
Trend In
Rollup Sum
Time Range lastMonth
Use Request Rollup false
Use Request Time Range false

Steam_water_usage x multiplier
Numeric Constant Block
Out
Makes Trends false
Value 0.90 {ok}

BiMath
Bi Math Block
Out
In1
Operator Multiply
In2

BiSwitch
Bi Switch Block
Out
In1
Operator Less Than
In2
True In
False In

Result
Result Block
In

Parent topic: Algorithm library reference

Steam_Water_Usage_Minus_5_PC_of_last_Month

This algorithm compares steam usage this month with usage from the previous month. It produces a true result, indicating that an exception exists, if the total steam usage for this month is less than 95% of the total monthly usage for the previous month.

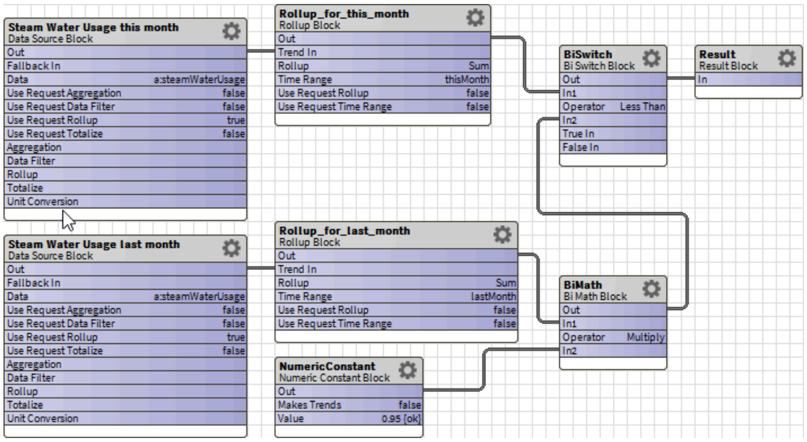
Input ORD

slot:/Algorithm/English/Steam_Water_Usage_Minus_5_PC_of_last_Month

slot:/Algorithm/Metric/Steam_Water_Usage_Minus_5_PC_of_last_Month

The algorithm uses two rollup blocks, numeric constant, calculation and switch block to calculate the result.

Figure 1. Water usage minus 5 percent of last month wire sheet



Steam Water Usage this month
Data Source Block
Out
Fallback In
Data a:steamWaterUsage
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup true
Use Request Totalize false
Aggregation
Data Filter
Rollup
Totalize
Unit Conversion

Rollup_for_this_month
Rollup Block
Out
Trend In
Rollup Sum
Time Range thisMonth
Use Request Rollup false
Use Request Time Range false

BiSwitch
Bi Switch Block
Out
In1
Operator Less Than
In2
True In
False In

Result
Result Block
In

Steam Water Usage last month
Data Source Block
Out
Fallback In
Data a:steamWaterUsage
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup true
Use Request Totalize false
Aggregation
Data Filter
Rollup
Totalize
Unit Conversion

Rollup_for_last_month
Rollup Block
Out
Trend In
Rollup Sum
Time Range lastMonth
Use Request Rollup false
Use Request Time Range false

NumericConstant
Numeric Constant Block
Out
Makes Trends false
Value 0.95 {ok}

BiMath
Bi Math Block
Out
In1
Operator Multiply
In2

Parent topic: Algorithm library reference

Steam_Water_Usage_Plus_10_PC_of_last_Month

This algorithm compares steam water usage this month with usage from the previous month. It produces a true result, indicating that an exception exists, if the total steam water usage for this month is greater than the total monthly usage for the previous month by more than 10%.

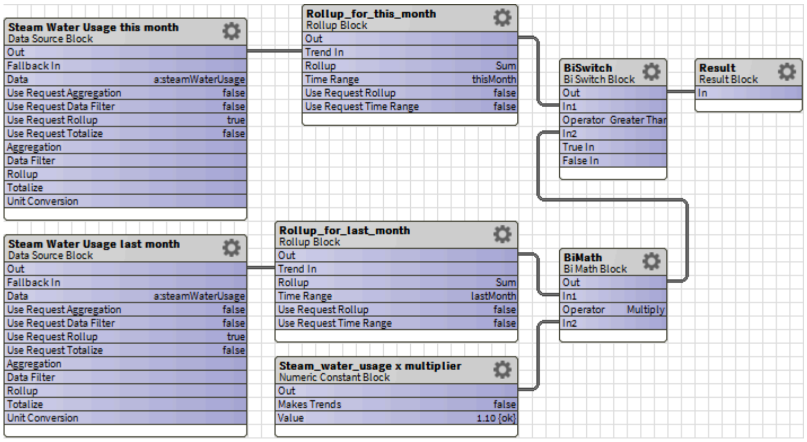
Input ORD

slot:/Algorithm/English/Steam_Water_Usage_Plus_10_PC_of_last_Month

slot:/Algorithm/Metric/Steam_Water_Usage_Plus_10_PC_of_last_Month

The algorithm uses two rollup blocks, numeric constant, calculation and switch block to calculate the result.

Figure 1. Water usage plus 10 percent of last month wire sheet



Steam Water Usage this month
Data Source Block
Out
Fallback In
Data a:steamWaterUsage
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup true
Use Request Totalize false
Aggregation
Data Filter
Rollup
Totalize
Unit Conversion

Rollup_for_this_month
Rollup Block
Out
Trend In
Rollup Sum
Time Range thisMonth
Use Request Rollup false
Use Request Time Range false

BiSwitch
Bi Switch Block
Out
In1
Operator Greater Thar
In2
True In
False In

Result
Result Block
In

Steam Water Usage last month
Data Source Block
Out
Fallback In
Data a:steamWaterUsage
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup true
Use Request Totalize false
Aggregation
Data Filter
Rollup
Totalize
Unit Conversion

Rollup_for_last_month
Rollup Block
Out
Trend In
Rollup Sum
Time Range lastMonth
Use Request Rollup false
Use Request Time Range false

BiMath
Bi Math Block
Out
In1
Operator Multiply
In2

Steam_water_usage x multiplier
Numeric Constant Block
Out
Makes Trends false
Value 1.10 {ok}

Parent topic: Algorithm library reference

Steam_Water_Usage_Plus_5_PC_of_last_Month

This algorithm compares steam water usage this month with usage from the previous month. It produces a true result, indicating that an exception exists, if the total steam water usage for this month is greater than the total monthly usage for the previous month by more than 5%.

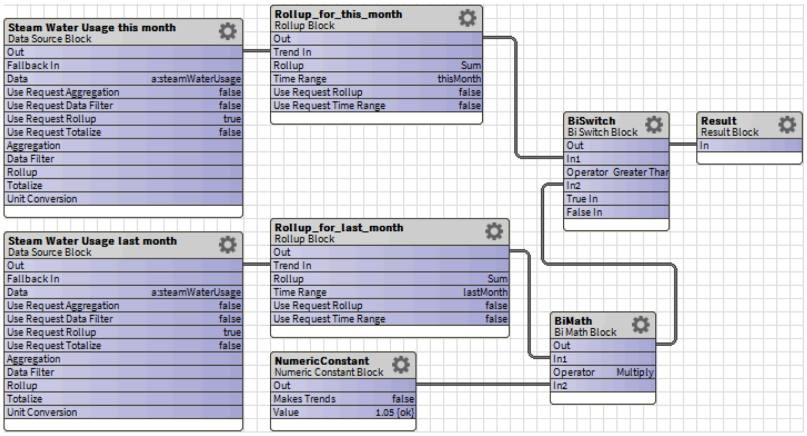
Input ORD

slot:/Algorithm/English/Steam_Water_Usage_Plus_5_PC_of_last_Month

slot:/Algorithm/Metric/Steam_Water_Usage_Plus_5_PC_of_last_Month

The algorithm uses two rollup blocks, numeric constant, calculation and switch block to calculate the result.

Figure 1. Water usage plus 5 percent of last month wire sheet



Steam Water Usage this month
Data Source Block
Out
Fallback In
Data a:steamWaterUsage
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup true
Use Request Totalize false
Aggregation
Data Filter
Rollup
Totalize
Unit Conversion

Rollup_for_this_month
Rollup Block
Out
Trend In
Rollup Sum
Time Range thisMonth
Use Request Rollup false
Use Request Time Range false

BiSwitch
Bi Switch Block
Out
In1
Operator Greater Thar
In2
True In
False In

Result
Result Block
In

Steam Water Usage last month
Data Source Block
Out
Fallback In
Data a:steamWaterUsage
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup true
Use Request Totalize false
Aggregation
Data Filter
Rollup
Totalize
Unit Conversion

Rollup_for_last_month
Rollup Block
Out
Trend In
Rollup Sum
Time Range lastMonth
Use Request Rollup false
Use Request Time Range false

NumericConstant
Numeric Constant Block
Out
Makes Trends false
Value 1.05 {ok}

BiMath
Bi Math Block
Out
In1
Operator Multiply
In2

Parent topic: Algorithm library reference

Simultaneous_Heat_Cool

This algorithm analyzes trend data and returns a true result, indicating that an exception existed, if the building was occupied, the supply fan command was on, the supply fan status was on, the heating valve position or the heating valve command was greater than 0%, and the cooling valve position or cooling valve command was greater than 0% for 15 minutes. A false result indicates that nothing was wrong.

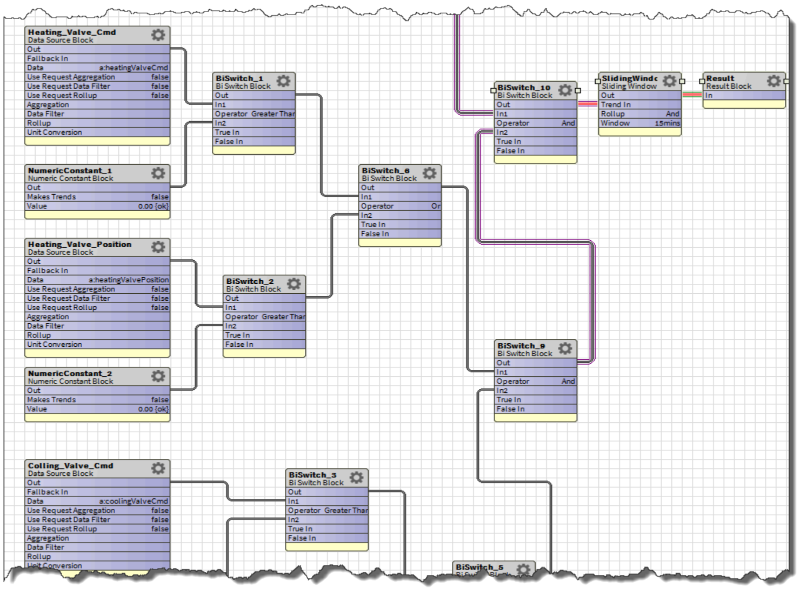
Input ORDs

slot:/Algorithm/English/Simultaneous_Heat_Cool

slot:/Algorithm/Metric/Simultaneous_Heat_Cool

Wire sheet

Figure 1. Fragment of the Simultaneous Heat Cool algorithm wire sheet



Heating_Valve_Cmd
Data Source Block
Out
Fallback In
Data a:heatingValveCmd
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Aggregation
Data Filter
Rollup
Unit Conversion

NumericConstant_1
Numeric Constant Block
Out
Makes Trends false
Value 0.00 {ok}

Heating_Valve_Position
Data Source Block
Out
Fallback In
Data a:heatingValvePosition
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Aggregation
Data Filter
Rollup
Unit Conversion

NumericConstant_2
Numeric Constant Block
Out
Makes Trends false
Value 0.00 {ok}

Colling_Valve_Cmd
Data Source Block
Out
Fallback In
Data a:coolingValveCmd
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Aggregation
Data Filter
Rollup
Unit Conversion

BiSwitch_1
Bi Switch Block
Out
In1
Operator Greater Thar
In2
True In
False In

BiSwitch_2
Bi Switch Block
Out
In1
Operator Greater Thar
In2
True In
False In

BiSwitch_3
Bi Switch Block
Out
In1
Operator Greater Thar
In2
True In
False In

BiSwitch_6
Bi Switch Block
Out
In1
Operator Or
In2
True In
False In

BiSwitch_5

BiSwitch_10
Bi Switch Block
Out
In1
Operator And
In2
True In
False In

BiSwitch_9
Bi Switch Block
Out
In1
Operator And
In2
True In
False In

SlidingWindc
Sliding Window
Out
Trend In
Rollup And
Window 15mins

Result
Result Block
In

Parent topic: Algorithm library reference

SpaceDifferentialPressureHighLow

This algorithm returns true if the differential pressure in the space (room or building) is greater than or less than the setpoint for more than 30 minutes.

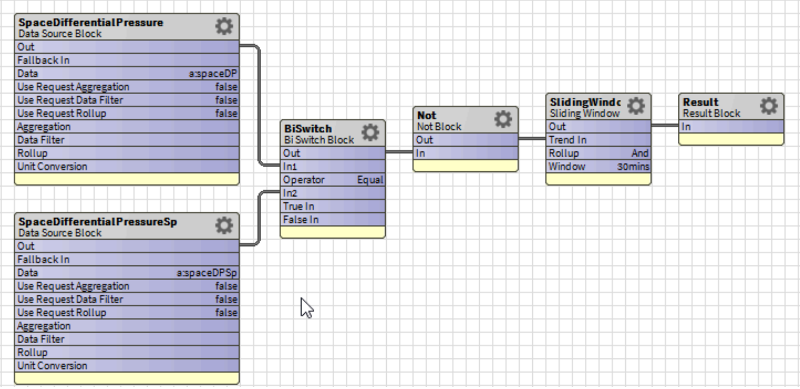
Input ORD

slot:/Algorithm/English/SpaceDifferentialPressureHighLow

slot:/Algorithm/Metric/SpaceDifferentialPressureHighLow

Wire sheet view

Figure 1. Space differential pressure wire sheet



Parent topic: Algorithm library reference

SpaceExhaustFlowIsHigh

This algorithm returns true if the space exhaust air flow is greater than the space exhaust airflow setpoint.

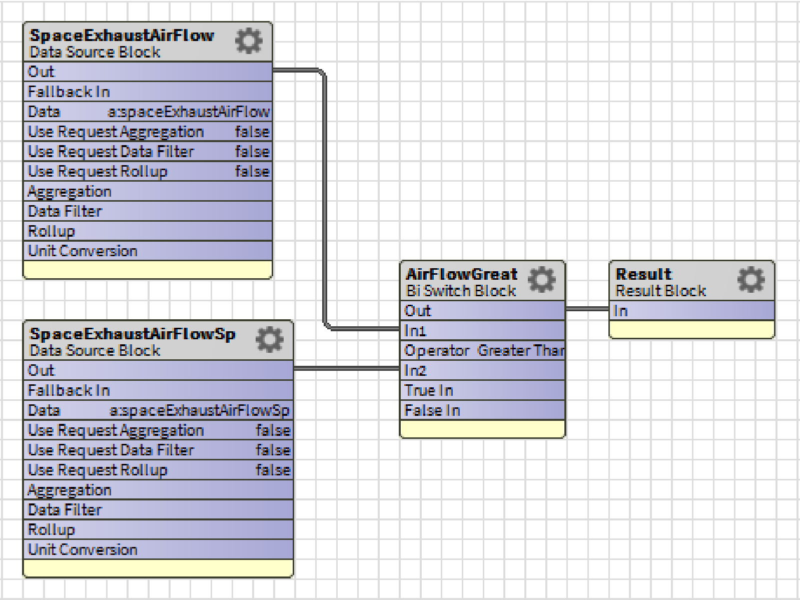
Input ORD

slot:/Algorithm/English/SpaceExhaustFlowIsHigh

slot:/Algorithm/Metric/SpaceExhaustFlowIsHigh

Wire sheet view

Figure 1. Space exhaust air flow high wire sheet



Parent topic: Algorithm library reference

SpaceSupplyFlowIsHigh

This algorithm returns true if the space supply air flow is greater than the space supply air flow setpoint.

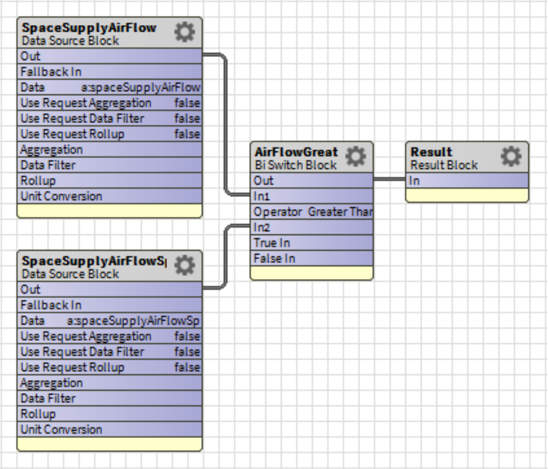
Input ORD

slot:/Algorithm/English/SpaceSupplyFlowIsHigh

slot:/Algorithm/Metric/SpaceSupplyFlowIsHigh

Wire sheet view

Figure 1. Space supply air flow high wire sheet



Parent topic: Algorithm library reference

Unscheduled_Operation

This algorithm evaluates trend data collected from an AHU unit. It determines if an AHU unit was running at an unscheduled time. It returns true if the facility is unoccupied, the command was given to turn the supply fan on, or the supply fan is on. It does not matter how long the supply fan was running. A false result indicates that the AHU ran on schedule.

ORDs

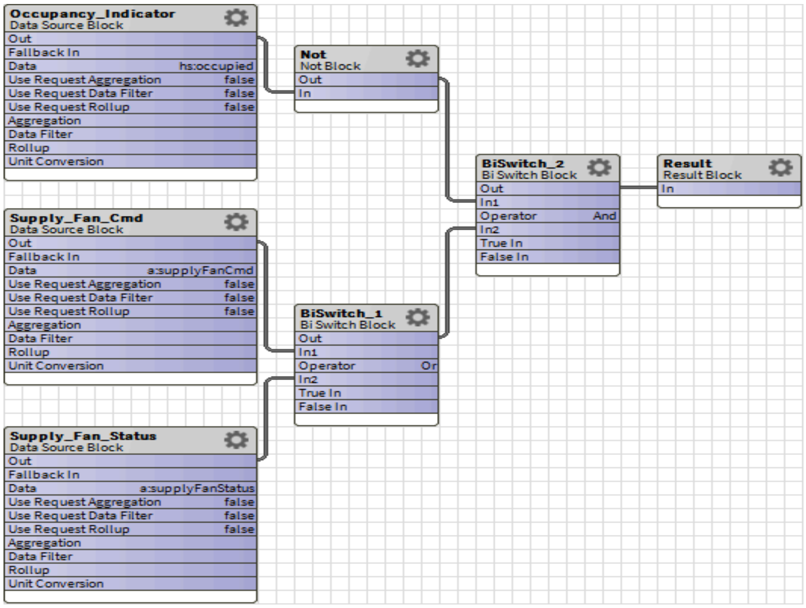
This algorithm might produce false positive results if the system activates while unoccupied to maintain unoccupied setpoints.

slot:/Algorithm/English/Unscheduled_Operation

slot:/Algorithm/English/Unscheduled_Operation

Algorithm

Figure 1. Unscheduled Operation algorithm



Occupancy_Indicator
Data Source Block
Out
Fallback In
Data hs:occupied
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Aggregation
Data Filter
Rollup
Unit Conversion

Supply_Fan_Cmd
Data Source Block
Out
Fallback In
Data a:supplyFanCmd
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Aggregation
Data Filter
Rollup
Unit Conversion

Supply_Fan_Status
Data Source Block
Out
Fallback In
Data a:supplyFanStatus
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Aggregation
Data Filter
Rollup
Unit Conversion

Not
Not Block
Out
In

BiSwitch_1
Bi Switch Block
Out
In1
Operator Or
In2
True In
False In

BiSwitch_2
Bi Switch Block
Out
In1
Operator And
In2
True In
False In

Result
Result Block
In

Parent topic: Algorithm library reference

Unscheduled_Operation_EFan

This algorithm returns true if a supply fan (EFAN) is on or its status is ok and a space (room or building) is unoccupied.

Three source blocks are required for this algorithm:

- occupied
- supplyFancommand
- supplyFanStatus

For the algorithm to work, tag your points with both a:supplyFanCmd and a:supplyFanStatus. Both blocks are required so that the algorithm can evaluate the "or" condition. Neither block can return null.

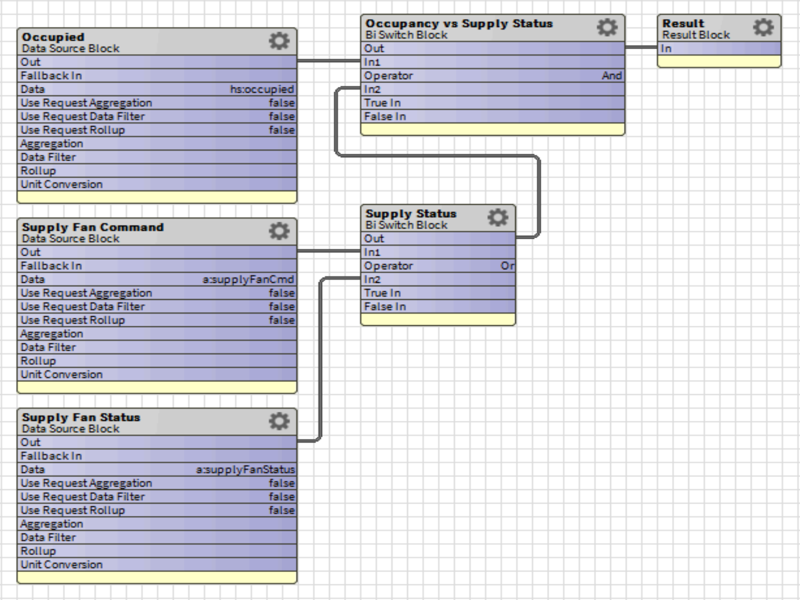
Input ORD

slot:/Algorithm/English/Unscheduled_Operation_EFan

slot:/Algorithm/Metric/Unscheduled_Operation_EFan

Wire sheet view

Figure 1. Unscheduled operation — EFan wire sheet



Occupied
Data Source Block
Out
Fallback In
Data hs:occupied
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Aggregation
Data Filter
Rollup
Unit Conversion

Supply Fan Command
Data Source Block
Out
Fallback In
Data a:supplyFanCmd
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Aggregation
Data Filter
Rollup
Unit Conversion

Supply Fan Status
Data Source Block
Out
Fallback In
Data a:supplyFanStatus
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Aggregation
Data Filter
Rollup
Unit Conversion

Occupancy vs Supply Status
Bi Switch Block
Out
In1
Operator And
In2
True In
False In

Supply Status
Bi Switch Block
Out
In1
Operator Or
In2
True In
False In

Result
Result Block
In

Parent topic: Algorithm library reference

Water_Usage_5_Percent_Less_Than_Last_Year

This algorithm evaluates trend data for water usage. It returns a true result, indicating that an exception existed, if water usage for this month was less than water usage the same month last year by more than 5%. A false result indicates that water usage was normal.

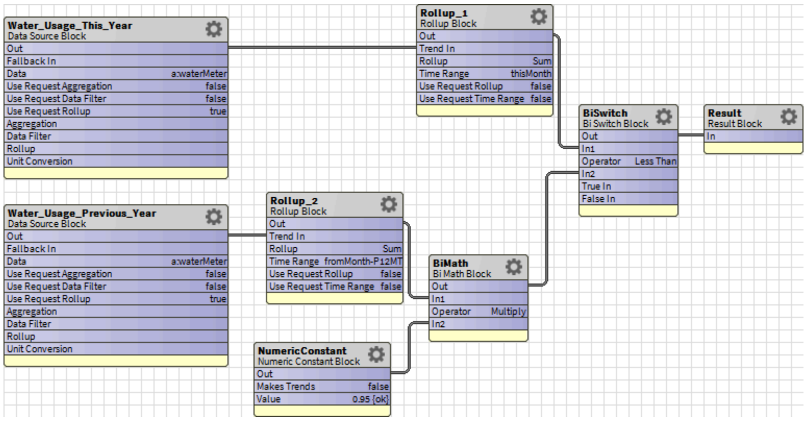
ORDs

slot:/Algorithm/English/Water_Usage_5_Percent_Less_Than_Last_Year

slot:/Algorithm/Metric/Water_Usage_5_Percent_Less_Than_Last_Year

Wire Sheet

Figure 1. Water usage algorithm



Parent topic: [Algorithm library reference]()

Water_Usage_5_Percent_More_Than_Last_Year

This algorithm evaluates trend data for water usage. It returns a true result, indicating that an exception existed, if water usage for this month was greater than water usage the same month last year by more than 5%. A false result indicates that water usage was normal.

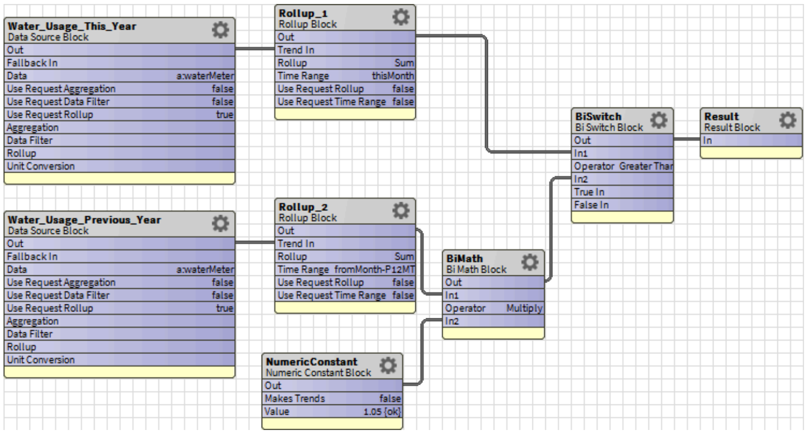
ORDs

slot:/Algorithm/English/Water_Usage_5_Percent_More_Than_Last_Year

slot:/Algorithm/Metric/Water_Usage_5_Percent_More_Than_Last_Year

Wire sheet

Figure 1. Water usage algorithm



Parent topic: [Algorithm library reference]()

WeatherNormalizedConsumption

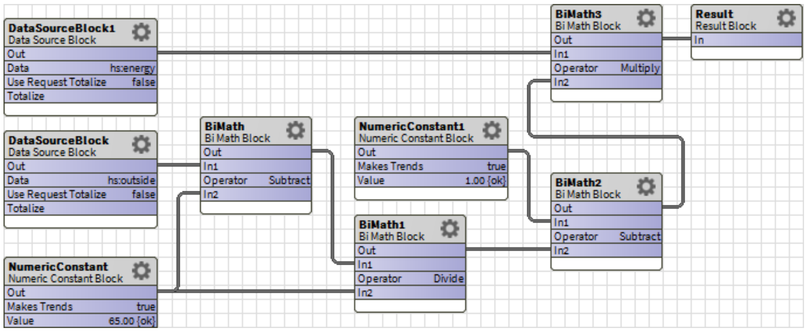
This algorithm uses three inputs to calculate normal energy consumption based on the weather.

Input ORD

slot:/Algorithm/English/WeatherNormalizedConsumption

slot:/Algorithm/Metric/WeatherNormalizedConsumption

Figure 1. WeatherNormalizedConsumption wire sheet



DataSourceBlock1
Data Source Block
Out
Data hs:energy
Use Request Totalize false
Totalize

DataSourceBlock
Data Source Block
Out
Data hs:outside
Use Request Totalize false
Totalize

NumericConstant
Numeric Constant Block
Out
Makes Trends true
Value 65.00 {ok}

BiMath
Bi Math Block
Out
In1
Operator Subtract
In2

NumericConstant1
Numeric Constant Block
Out
Makes Trends true
Value 1.00 {ok}

BiMath1
Bi Math Block
Out
In1
Operator Divide
In2

BiMath2
Bi Math Block
Out
In1
Operator Subtract
In2

BiMath3
Bi Math Block
Out
In1
Operator Multiply
In2

Result
Result Block
In

Parent topic: Algorithm library reference

WeatherNormalizedDemand

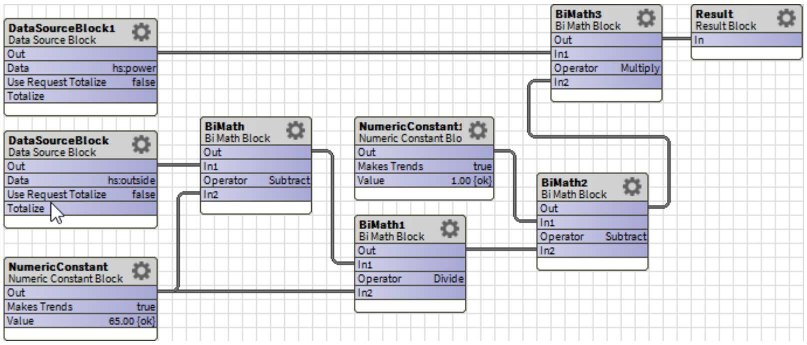
This algorithm uses three inputs to calculate normal energy demand based on the weather.

Input ORD

slot:/Algorithm/English/WeatherNormalizedDemand

slot:/Algorithm/Metric/WeatherNormalizedDemand

Figure 1. WeatherNormalizedDemand wire sheet



DataSourceBlock1
Data Source Block
Out
Data hs:power
Use Request Totalize false
Totalize

DataSourceBlock
Data Source Block
Out
Data hs:outside
Use Request Totalize false
Totalize

NumericConstant
Numeric Constant Block
Out
Makes Trends true
Value 65.00 {ok}

BiMath
Bi Math Block
Out
In1
Operator Subtract
In2

NumericConstant1
Numeric Constant Blo
Out
Makes Trends true
Value 1.00 {ok}

BiMath1
Bi Math Block
Out
In1
Operator Divide
In2

BiMath2
Bi Math Block
Out
In1
Operator Subtract
In2

BiMath3
Bi Math Block
Out
In1
Operator Multiply
In2

Result
Result Block
In

Parent topic: Algorithm library reference

Components, views and windows

The user interface includes components, views and windows, which provide the means for communicating with the system.

The Help topics include context sensitive information about each component and view, as well as information about individual windows.

- **AnalyticService (BAnalyticService)**
  This is the root container of nearly all analytical components except analytic proxy points, which are scattered throughout the station tree. The properties on this component configure global aspects of the framework.
- **Alert (BAnalyticAlert)**
  This component processes the configured algorithm (Boolean result) to determine if nodes are in an alert condition, which may also generate an alarm.
- **Algorithm (BAlgorithm)**
  This component analyzes data by managing the execution of blocks in the **Wire Sheet**, which produces a result via the result block. You can view the result of processing an algorithm in various ways including, but not limited to: a control point with an Analytic Proxy Ext, a Bound Label widget with an Analytic Binding, a chart with an Analytic Binding or a table with an Analytic Binding. Algorithms are implemented with logic blocks linked together using a **Wire Sheet**. They may combine both historical and real-time data to produce results, which may be a trend or a single value.
- **Definition (BAnalyticDataDefinition)**
  This optional component configures the default values for facets, aggregation and rollup properties. Each tag from the Haystack, Niagara or a custom dictionary requires a definition. An algorithm or a graphics binding (Px view or Ux chart) defines the tag to use for searching a hierarchy and the node at which to begin the search. Using this information the framework retrieves trend data. It then combines the retrieved data using the facets, aggregation and rollup properties defined on the definition that is associated with the tag.
- **Cyclic poller (BCyclicPoller)**
  This poller executes framework proxy points and alerts at even intervals across the time span as defined by the **Min Interval**, **Max Interval** and **Rate**.
- **Proxy extension (BAnalyticProxyExt)**
  This component resolves the configured analytic request and stores the result using its parent control point. On writable points, the proxy extension writes to priority level 16.
- **Combination (BCombination)**
  This component works behind the scenes to define how to combine multiple pieces of data. When used for aggregation, it defines how the framework combines data from multiple individual points. When used for rollups, it defines how the framework combines multiple values in a single interval of a trend.
- **Interval (BInterval)**
  This component works behind the scenes to define the period of time the framework uses to separate values in a trend (time series). Whenever an interval is specified, a rollup is required to combine all values that fall within a single interval.
- **baselineValue window**

AnalyticService (BAnalyticService)

This is the root container of nearly all analytical components except analytic proxy points, which are scattered throughout the station tree. The properties on this component configure global aspects of the framework.

Under the service container, are folders and properties for:

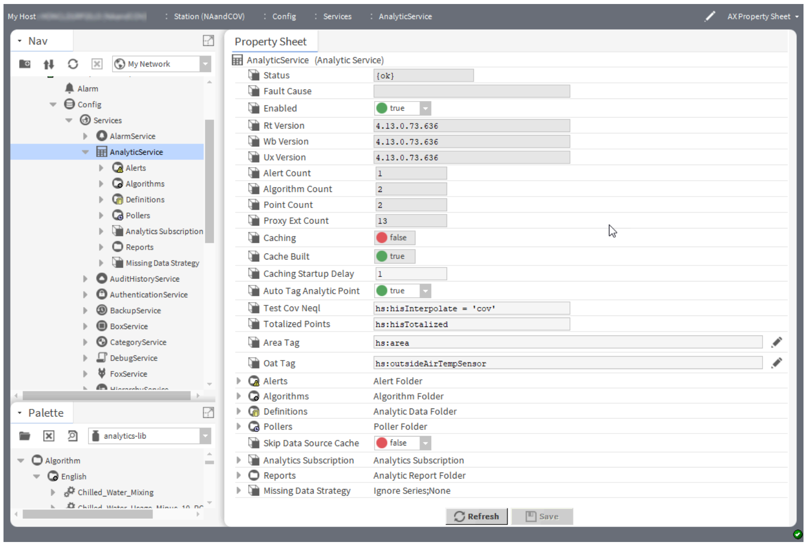
- Alerts

- Algorithms

- Definitions
- Pollers
- Analytics Subscription
- Reports contains HTML versions of seven charts that service the energy suite
- Missing Data Strategy

These folders may have sub-folders.

Figure 1. AnalyticService properties



Property Sheet

AnalyticService (Analytic Service)

Status	{ok}
Fault Cause	
Enabled	● true
Rt Version	4.13.0.73.636
Wb Version	4.13.0.73.636
Ux Version	4.13.0.73.636
Alert Count	1
Algorithm Count	2
Point Count	2
Proxy Ext Count	13
Caching	● false
Cache Built	● true
Caching Startup Delay	1
Auto Tag Analytic Point	● true
Test Cov Neql	hs:hisInterpolate = 'cov'
Totalized Points	hs:hisTotalized
Area Tag	hs:area
Oat Tag	hs:outsideAirTempSensor
Alerts	Alert Folder
Algorithms	Algorithm Folder
Definitions	Analytic Data Folder
Pollers	Poller Folder
Skip Data Source Cache	● false
Analytics Subscription	Analytics Subscription
Reports	Analytic Report Folder
Missing Data Strategy	Ignore Series;None

To view the component's **Property Sheet** in Workbench, right-click the **Config > Services > AnalyticService** container in the Nav tree, and click **Views > Property Sheet**. The same properties are available in the web UI.

Property	Value	Description
Status	read-only	Displays the current state of the component or extension. The platform connection will be in `fault` if any of the following occurs: Supervisor has no ProvisioningNwExt under its **NiagaraNetwork** (for example, it has been deleted). Supervisor is not licensed for provisioning. **NiagaraStation** is in `fault`. The station’s platform daemon rejects the platform connection’s credentials. The extension is `disabled` if its Enabled property is set to false or the ProvisioningNwExt is disabled.
Fault Cause	read-only	Indicates the reason why a system object (network, device, component, extension, etc.) is not working (in fault). This property is empty unless a fault exists.
Enabled	true or false	Activates (true) and deactivates (false) use of the object

Property	Value	Description
		(network, device, point, component, table, schedule, descriptor, etc.).
Rt Version	read-only	Displays the installed version of the Niagara Analytics Framework runtime (Rt) modules.
Wb Version	read-only	Displays the installed version of the Workbench modules.
Ux Version	read-only	Displays the installed version of the Ux chart modules.
Alert Count	read-only number	Reports the current number of alerts in the Alert folder.
Algorithm Count	read-only number	Reports the current number of algorithms in the station. Algorithms are the rules you create.
Point Count	read-only number	Reports how many analytic points you are using. Auto Tag Analytic Point configures the framework to add an `a:a` tag to each point used in each algorithm. The framework uses this tag to return this Point Count.
Proxy Ext Count	read-only number	Sums all Analytic proxy extensions used by the framework.
Caching	read-only true or false	Returns true if the framework is currently in the process of caching memory. Otherwise, returns false.
Cache Built	read-only true or false	Returns true if the framework has completed the process of caching memory. Otherwise, returns false.
Caching Startup Delay	Number (Defaults to 1 minute)	Configures the amount of time to delay before refreshing cache at station startup. Stations with a lot of connected equipment may need a longer delay before refreshing cache. You can see this delay in the console. If you add elements to the model after starting the station, just use the Refresh Cache action.
Auto Tag Analytic Point	true or false (default)	Configures the automatic assignment of an `a:a` tag to each point used in an algorithm. true does two things when rebuilding cache memory by right-clicking **AnalyticService > Actions > Refresh Cache**: • includes the point data in the algorithm result. • updates the counts when rebuilding cache memory. Knowing how many points are used by the framework is related to licensing. Auto tagging continues until it reaches the maximum number of points allowed by your license. When it reaches the maximum, it stops tagging. false does not automatically assign an `a:a` tag to each point. This does two things when rebuilding cache memory: • excludes the point data from the algorithm

Property	Value	Description
		result. - does not add the point to the counts on this **Property Sheet**. A false setting for this property may explain why algorithms may not work.
Test Cov Neql	namespace (defaults to hs:hisInterpolate = ‘cov’)	The Analytics engine uses this expression to identify cov points. The engine automatically inserts cov points using the interval mentioned in the request.
Totalized Points	text	Defines the tag used to identify points to be totaled.
Area Tag	namespace:tagname	Defines the value tag from the tag dictionary to use for the Floor Area normalization calculation on the **Average Profile** and **Load Duration** reports.
Oat Tag	namespace:tagname	Defines the outside air temperature used by the **Average Profile** and **Load Duration** reports in their Degree Day normalization calculation.
Alerts	container	Serves a a container for all diagnostics.
Algorithms	container	Serves as a container for all algorithms.
Definitions	container	Serves as a container for all Data Definitions.
Pollers	container	Serves as a container for all pollers used to execute alerts and analytic proxy points.
Skip Data Source Cache	true or false (default)	The framework automatically caches the data source in memory. If you do not want this feature, set this property to true.
Analytics Subscription	additional properties	As with other Niagara features, use of this framework is based on a subscription, which you must renew periodically. The Analytics Subscription properties configure an alarm to remind you that your subscription is about to expire. Refer to *Analytics Subscription properties*.
Reports	container	Serves as a location for analytic reports.
Missing Data Strategy	additional properties	Configures how the framework handles missing data in a series when processing analytic requests and one or more records are missing for an interval. It applies when even a single record for an interval is missing. It does not apply to value requests. Refer to Missing Data Strategy
Chart Render Capacity	additional properties	These properties are only visible after removing the hidden flag on the chartRenderCapacity slot. Refer to #analytics-AnalyticService__ChartRenderCapacityPropertiesAsThey-B9F0342D

Actions

Action	Description
Refresh Cache	Calculates data cache memory requirements without searching all hierarchies. You should invoke this action after adding new tags.
Stop Caching	Terminates caching.

Chart Render Capacity (advanced property)

This property configures the default number of records used to render data on charts and reports. The default limitations help to provide acceptable framework performance.

Important:

Increasing these values may significantly degrade framework performance.

To access the Chart Render Capacity property you must first un-check the chartRenderCapacity hidden flag on the slot sheet, then expand the AnalyticService node in Workbenchor the web UI.

Figure 2. Chart Render Capacity property as it appears in the web UI

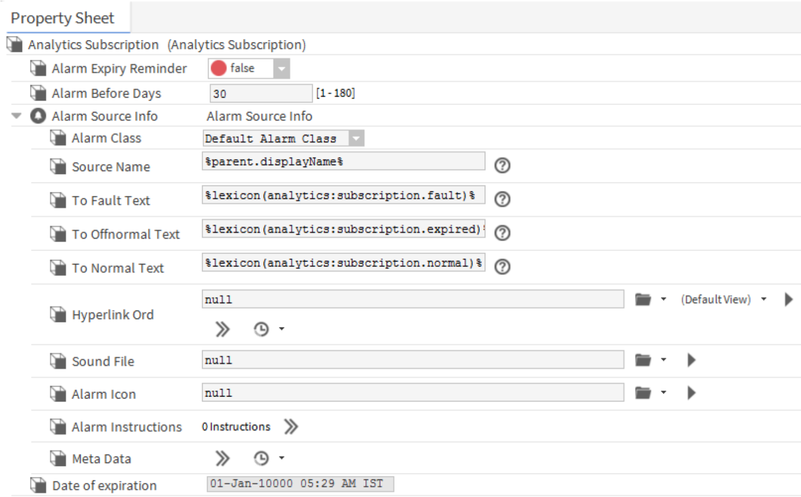
- **Analytics Subscription properties**
  This component configures alarm properties.
- **Missing Data Strategy**
  In statistics, imputation is the process of replacing missing data with substituted values. Incomplete, incorrect, inaccurate and irrelevant data are replaced, modified, or deleted. This is also known as data cleansing or data cleaning.
- **Analytic Service View**
  This view provides access to all framework features.
- **Automatic scaling of values**
  When the system scales values, it automatically shortens the value and adds a suffix. This improves the readability of large data and data with multiple decimal places in report tables.

Parent topic: Components, views and windows

Analytics Subscription properties

This component configures alarm properties.

Figure 1. Subscription properties view



Property Sheet

Analytics Subscription (Analytics Subscription)

- Alarm Expiry Reminder: false
- Alarm Before Days: 30 [1 - 180]
- Alarm Source Info: Alarm Source Info
  - Alarm Class: Default Alarm Class
  - Source Name: %parent.displayName%
  - To Fault Text: %lexicon(analytics:subscription.fault)%
  - To Offnormal Text: %lexicon(analytics:subscription.expired)%
  - To Normal Text: %lexicon(analytics:subscription.normal)%
  - Hyperlink Ord: null (Default View)
  - Sound File: null
  - Alarm Icon: null
  - Alarm Instructions: 0 Instructions
  - Meta Data
- Date of expiration: 01-Jan-10000 05:29 AM IST

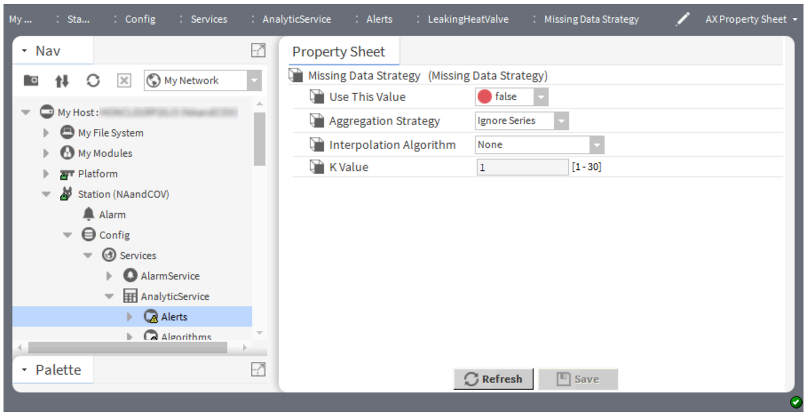
Property	Value	Description
Alarm Expiry Reminder	true or false (default)	Enables and disables the sounding of an alarm to remind you that your subscription is expiring.
Alarm Source Info	additional properties	Contains a set of properties for configuring and routing alarms when this component is the alarm source. For property descriptions, refer to the *Niagara Alarms Guide*
Alarm Before Days	number	Before the subscription expires, sets when to sound the alarm.
Date of expiration	read-only	Indicates when the subscription expires.

Parent topic: AnalyticService (BAnalyticService)

Missing Data Strategy

In statistics, imputation is the process of replacing missing data with substituted values. Incomplete, incorrect, inaccurate and irrelevant data are replaced, modified, or deleted. This is also known as data cleansing or data cleaning.

Figure 1. Missing Data Strategy properties



My ... : Sta... : Config : Services : AnalyticService : Alerts : LeakingHeatValve : Missing Data Strategy

Property Sheet

Missing Data Strategy (Missing Data Strategy)

- Use This Value: false
- Aggregation Strategy: Ignore Series
- Interpolation Algorithm: None
- K Value: 1 [1 - 30]

You may access this view from a variety of locations. For example, expand **Config > Services > AnalyticService**, right-click **Alerts**, click**Views > AX Property Sheet**, scroll down and double-click **Missing Data Strategy**.

Property	Value	Description
Use This Value	check box	Enables and disables missing data interpolation for the current value. When enabled, the framework applies this strategy to all requests. When disabled, the framework ignores this strategy for all requests.
Aggregation Strategy	drop-down list	Configures how the framework handles missing trend data (data in a series) when processing analytic requests and one or more records are missing for an interval. It applies when even a single record for an interval is missing. It does not apply to value requests. Note: If the analytic trend request specifies Interval = none, the framework ignores the Missing Data Strategy. Ignore Series ignores the entire series if any record even one interval is missing. Ignore Point ignores any missing records and aggregates the values in the existing records.
Interpolation Algorithm	additional properties	Defines the algorithm used to interpolate values for missing values (missing records). None does not interpolate values for missing records. Linear Interpolation replaces a missing record by linearly interpolating the missing value using the prior and next records on either side of the missing record. K-Nearest Neighbor is for numeric, enum and Boolean records. This strategy replaces a missing value by calculating the majority value recorded for the item’s nearest K number of neighbors.
K Value	Numeric field editor (default = 1, min =1, max = 30)	Defines the number of records used by the configured interpolation algorithm when a record is missing to calculate the interpolated value.

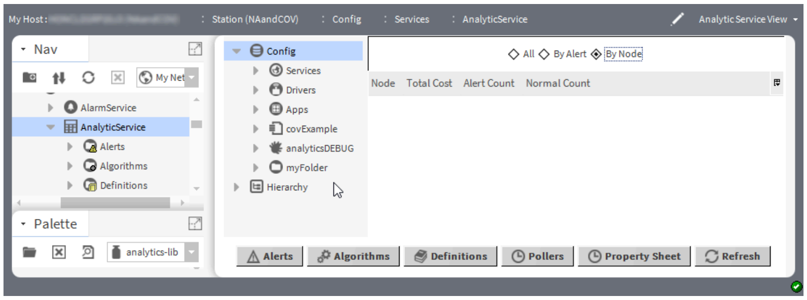
Parent topic: AnalyticService (BAnalyticService)

Analytic Service View

This view provides access to all framework features.

Radio buttons and columns

Figure 1. Analytic Service View



You access this view by expanding the **Services** folder in Workbench followed by double-clicking the **AnalyticService** node.

Radio button	Column	Description
All	Node	Identifies the node’s slot path.
	Alert	Identifies the alert’s slot path.
	Cost	Displays the calculated cost, which is based on a variety of factors.
	State	Indicates the state of the alert (normal or alert).
By Alert	Alert	Identifies the alert’s slot path.
	Total Cost	Displays the combined cost of all active alerts for this specific alert.
	Alert Count	Displays the total number of nodes in alert state.
	Normal Count	Displays the total number of nodes in a normal state.
By Node	Node	Identifies the node’s slot path.
	Total Cost	Displays the combined cost of all active alerts for this specific node.
	Alert Count	Displays the total number of alerts for this node.
	Normal Count	Displays the total number of alerts in a normal state.

Buttons

- **Alerts** opens the Analytic Alert Manager view.
- **Algorithms** opens the Algorithm Manager view.
- **Definitions** opens the Analytic Data Manager view.
- **Pollers** opens the Poller Manager view.
- **Refresh** updates the current **Analytic Service View**.

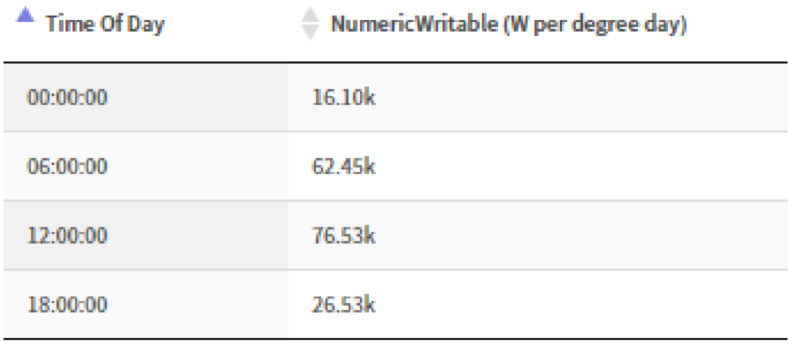
Parent topic: AnalyticService (BAnalyticService)

Automatic scaling of values

When the system scales values, it automatically shortens the value and adds a suffix. This improves the readability of large data and data with multiple decimal places in report tables.

For example, a value of 16100 scales in the table as 16.10k.

Figure 1. A table with scaled values



Time Of Day	NumericWritable (W per degree day)
00:00:00	16.10k
06:00:00	62.45k
12:00:00	76.53k
18:00:00	26.53k

Number conversion

These abbreviations are for values that are greater than 1,000.

Suffix symbol	Name	Positive orders of 10
T	trillion	1,000,000,000,000
G	billion	1,000,000,000
M	million	1,000,000
k	thousand	1,000

Decimal number conversion

These abbreviations are for values that are less than 1.

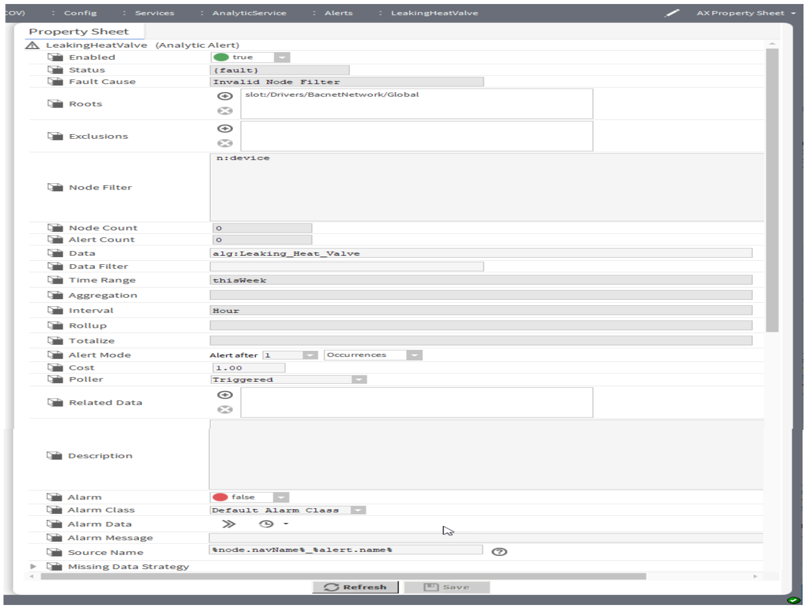
Suffix symbol	Name	Negative orders of 10
m	thousandth	0.0001
µ	millionth	0.000 001
n	billionth	0.000 000 0001
p	trillionth	0.000 000 000 001

Parent topic: AnalyticService (BAnalyticService)

Alert (BAnalyticAlert)

This component processes the configured algorithm (Boolean result) to determine if nodes are in an alert condition, which may also generate an alarm.

Figure 1. Alerts properties


COV) : Config : Services : AnalyticService : Alerts : LeakingHeatValve
AX Property Sheet
Property Sheet
Refresh
Save

LeakingHeatValve (Analytic Alert)

Enabled	● true
Status	{fault}
Fault Cause	Invalid Node Filter
Roots	slot:/Drivers/BacnetNetwork/Global
Exclusions	
Node Filter	n:device
Node Count	0
Alert Count	0
Data	alg:Leaking_Heat_Valve
Data Filter	
Time Range	thisWeek
Aggregation	
Interval	Hour
Rollup	
Totalize	
Alert Mode	Alert after 1 Occurrences
Cost	1.00
Poller	Triggered
Related Data	
Description	
Alarm	● false
Alarm Class	Default Alarm Class
Alarm Data	
Alarm Message	
Source Name	%node.navName%_%alert.name%
Missing Data Strategy	

There are multiple ways to view an alert’s properties once you expand **Config > Services > AnalyticService > Alerts**.

- Double-click the **Alerts** folder (this opens the **Analytic Alert Manager**), and double-click the alert row in the table (this opens the **Edit** window).
- Double-click the **Alerts** folder (this opens the **Analytic Alert Manager**), and double-click the **Property Sheet** icon (▣) in the Views column.
- Right-click the alert and click **Views > Property Sheet**
- Double-click the alert (this opens the **Alert Nodes View**), and click the **Property Sheet** button at the bottom of the window.

Property	Value	Description
Enabled	true or false	Activates (true) and deactivates (false) use of the object (network, device, point, component, table, schedule, descriptor, etc.).
Status	read-only	Reports the condition of the entity or process at last polling. {ok} indicates that the component is licensed and polling successfully. {down} indicates that the last check was unsuccessful, perhaps because of an incorrect property, or possibly loss of

Property	Value	Description
		network connection. {disabled} indicates that the Enable property is set to false. {fault} indicates another problem. Refer to Fault Cause for more information.
Fault Cause	read-only	Indicates the reason why a system object (network, device, component, extension, etc.) is not working (in fault). This property is empty unless a fault exists.
Roots	ORD	Beginning from the root of the Nav tree, this property defines the ORD from which to start scanning for alert conditions. The framework eliminates nodes listed in the Exclusions property before it evaluates each node for an alert condition.
Exclusions	ORD	Identifies the nodes (and their subtree) to exclude when scanning for points to monitor.
Node Filter	optional NEQL predicate	Limits which components to process the alert against in conjunction with Roots and Exclusions. Typically, you use marker tags, such as n:device, hs:ahu, hs:vav, etc. to only process the alert against a device, floor or building.
Node Count	read-only	For the selected roots, and taking into consideration the exclusions, displays the total number of nodes that are part of the calculation. Use this information to confirm that the result of a formula looks reasonable.
Alert Count	read-only number	Out of all nodes (Node Count) that could be in an alert state, this property reports the total number of nodes that are in an alert state.
Data	tag or algorithm name	Specifies the an algorithm (or tag) used to retrieve data. The purpose of an alert is to detect a fault condition (Boolean value result). In most cases, you would pick an algorithm that has a Boolean result when configuring this property for an alert. Algorithms with a numeric or enum value greater than zero are considered true and less than zero are considered false. This is how output from one algorithm becomes the input data source to another algorithm. The prefix for algorithms is alg:.

Property	Value	Description
Data Filter	optional NEQL predicate	Identifies data sources in the subtree of the request node. When a predicate accepts a node that contains the desired data, the system does not search the node’s subtree for additional values (including the root node).
Time Range	drop-down list	Defines the beginning and end of a period of time over which to count the number of occurrences or weight. If the alert depends on a certain number of occurrences or weight over time, this property is required, otherwise, it is optional.
Aggregation	value (defaults to First)	Configures the default function to apply when the analytic request combines values from multiple data sources into a single value. This applies to both value and trend requests. If aggregation is not enabled in the binding/settings window, the aggregation value defined in the Data Definition applies to all chart bindings, reports and tables. And returns the logical “and” of Boolean values. Avg returns the statistical mean, which is determined by calculating the sum of all values and dividing by the number of values. Count returns the total number or quantity of values in a combination. If you request this value on a binding in a PX view, the system counts the number of values based on the properties defined by the data source block and the algorithm’s **Property Sheet**. First returns the first value in the combination. Last returns the last value in the combination. Max returns the highest value in the combination. Median returns the value in the middle of a sorted combination—the number that separates the higher half from the lower half. Min returns the lowest value in the combination. Mode returns the statistically most frequently occurring number in the combination. Or returns the logical “or” of Boolean values. Range returns the statistical difference between the largest and smallest values in

Property	Value	Description
		the combination. Sum adds together all values in the combination resulting in a single value. Load Factor returns the average value divided by peak value. Std Dev returns the standard deviation of the values in the combination.
Interval	optional drop-down list (defaults to the optimal number of records on reports)	Refers to the BInterval component, which the framework uses to identify the time between values in a trend (time series). When specified, a rollup is required, which causes the system to combine all values that fall into a single interval. Options range from None to a Year. Above the drop-down list, the Use This Value check box turns on and off the check box next to Interval in the **Settings** window (you access this window by clicking the Edit button (✎), followed by clicking the Settings button (⚙) on the chart). The availability of this check box provides an easy way for a user to enable and disable the use of intervals in chart calculations.
Rollup	value (when configured in the Data Definition, defaults to First, when configured elsewhere, defaults to the value as defined in the Data Definition)	Configures the default function to apply when the analytic request needs to rollup records from a single data source into less granular records. If rollup is not enabled in the binding/settings window, the rollup value configured in the Data Definition applies to all chart bindings, reports and tables. And returns the logical “and” of Boolean values. Avg returns the statistical mean, which is determined by calculating the sum of all values and dividing by the number of values. Count returns the total number or quantity of values in a combination. If you request this value on a binding in a PX view, the system counts the number of values based on the properties defined by the data source block and the algorithm’s **Property Sheet**. First returns the first value in the combination. Last returns the last value in the combination. Max returns the highest value in the combination. Median returns the value in the middle of a sorted combination—the number that

Property	Value	Description
		separates the higher half from the lower half. Min returns the lowest value in the combination. Mode returns the statistically most frequently occurring number in the combination. Or returns the logical “or” of Boolean values. Range returns the statistical difference between the largest and smallest values in the combination. Sum adds together all values in the combination resulting in a single value. Std Dev calculates the standard deviation of the values in the combination. Load Factor calculates the average divided by peak (Max) value.
Totalize	true (default) or false	Turns on (true) and off (false) value accumulation. By default, the framework totalizes (accumulates) all consumption history values in charts, tables and reports. To prevent cumulative values, disable this property (set it to false).
Alert Mode	drop-down list	Defines the alert condition to use when you specify a time range. The alarm is raised within poller time when the time range is not specified. To configure an alert at a different time, specify a time range using one of these options.: Occurrences identifies the number of times an algorithm must output a result of “true” over the configured time range for there to be an alert condition. Runtime Seconds identifies the number of seconds an algorithm must maintain a result of “true” over the configured time range for there to be an alert condition. Runtime Minutes identifies the number of minutes an algorithm must maintain a result of “true” over the configured time range for there to be an alert condition. Runtime Hours identifies the number of hours an algorithm must maintain a result of “true” over the configured time range for there to be an alert condition.
Cost	read-only	Calculates the total cost by multiplying the cost of the alert by the Alert Mode. For example, if Alert Mode is Runtime Minutes, the total cost is the cost-per-

Property	Value	Description
		minute multiplied by the number of minutes the algorithm maintained a result of true. This value indicates the severity of the alert.
Poller	drop-down list	Schedules alert evaluation.
Related Data	read-only	Lists the algorithm specified by the Data property and any inputs (tag, tag group, or algorithm names). To populate this property, right-click the alert and click **Actions > Find Related Data**.
Description	string	Provides additional context regarding the alert.
Alarm	true or false (default)	Configures the alert to generate an alarm record (true) or not to generate the record (false).
Alarm Class	drop-down list (defaults to Medium)	Specifies the alarm routing option for the component. Replace provides a selection list of a local alarm classes, from which to select one to use for all alarms received from this device. Use Existing routes alarms from this remote station to any matching alarm class, that is, one with an identical name as that in each alarm record. If the program finds no local matching alarm class, it uses the station's default alarm class. Prepend adds leading text (as specified) to the incoming alarm class string, then routes it to any local matching alarm class in the station. Append adds trailing text (as specified) to the incoming alarm class string, then routes it to any local matching alarm class in the station.
Alarm Data	BFormat	Defines the data to add to the alarm data facets of the alarm record. The framework resolves the BFormats at alarm record creation time. BFormats must begin with the alert or node token to access the corresponding entity.
Alarm Message	BFormat	Creates a message value for the alarm record. The framework resolves the BFormats when requested. BFormats must begin with the alert or node token to access the corresponding entity.
Source Name	BFormat	Displays the name in an alarm record that identifies the source of the alarm.

Property	Value	Description
Missing Data Strategy	additional properties	Configures how the framework handles missing data in a series when processing analytic requests and one or more records are missing for an interval. It applies when even a single record for an interval is missing. It does not apply to value requests. Refer to Missing Data Strategy

Alarm data BFormat syntax

When the alert generates an alarm record, the framework resolves and stores in the applicable BAlarmRecord property BFormat syntaxes from the Alarm Data (facet keys with string values), Alarm Message and Source Name properties. The base object it uses for these BFormat syntaxes is either the alert or the node (BINavNode), which the alert is being processed against.

The Alarm Message's property BFormat resolves at the time the framework creates and stores the alarm record as a BFormat in the alarmData.msgText facet key. An alarm console or alarm recipient resolves the BFormat again, this time against the BAlarmRecord. Since the BFormat text resolves twice, to have it display a single percentage symbol (%) in the alarm console message text, you must enter quadruple percentage symbols (%%%%) in the Alarm Message property.

Table 1. BFormat examples

BFormat	Description
%alert.name%	Identifies the slot name of the alert object. Do not use **%alert.message%**.
%node.parent.displayName%	Identifies the display name of the parent for the node that is going into alert.
%node.name%	Resolves the name of the component in the hierarchy tree. The name of a component might include escaped characters, such as "hierarchy:/Campus/station$3a$7ch$3acafb," which may not be easily human readable.
%node.navDisplayName%	Resolves the display name of the component in the hierarchy tree, such as Building1.
%node.navOrd.resolve.component.slotPath%	Resolves the component in the station and displays the slot path, such as slot:/AnalyticsPlayground/Campus/Building1.
%node.navOrd.resolve.component.parent.displayName%	Resolves the parent component in the station and returns the display name, such as Campus. The parent of the component in the station may be different from its parent in the hierarchy tree.

Actions

Action	Description
Find Related Data	If the alert data include an algorithm, the system recursively scans the algorithm and all of its data sources and adds all IDs to the Related Data property.

- **Alerts folder**
  This folder contains alerts, which run periodically (based on pollers) to collect real-time and historical

data. It may contain other alerts folders. Double-clicking this component opens the **Analytic Alert Manager**.

- **Analytic Alert Manager view**
  This table view lists the alerts that the framework uses to monitor components in the station. Using this view you can group alerts (in folders), create and edit alerts as well as tag them. To view a table of components that are currently in alert, expand the **AnalyticService > Alerts** folder in the Nav tree and double-click the alert.
- **Alert Nodes View**
  This table lists all components that currently meet the condition(s) monitored by the selected alert.
- **New/edit alert windows**
  These windows create and edit alerts.

Parent topic: Components, views and windows

Alerts folder

This folder contains alerts, which run periodically (based on pollers) to collect real-time and historical data. It may contain other alerts folders. Double-clicking this component opens the **Analytic Alert Manager**.

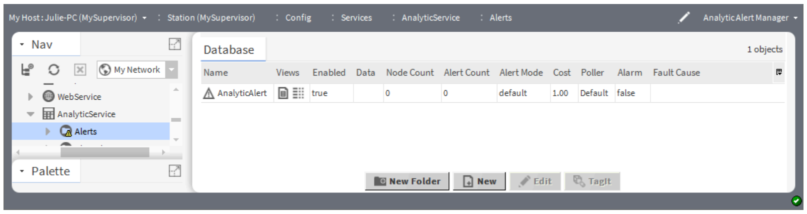
Parent topic: Alert (BAnalyticAlert)

Analytic Alert Manager view

This table view lists the alerts that the framework uses to monitor components in the station. Using this view you can group alerts (in folders), create and edit alerts as well as tag them. To view a table of components that are currently in alert, expand the **AnalyticService > Alerts** folder in the Nav tree and double-click the alert.

Columns

Figure 1. Alert Manager View



Name	Views	Enabled	Data	Node Count	Alert Count	Alert Mode	Cost	Poller	Alarm	Fault Cause
AnalyticAlert		true		0	0	default	1.00	Default	false	

After editing an alert, even if you did not change anything, the framework colors the row in the table a tan color and sets Node Count and Alert Count to zero. If you view the **Property Sheet**, Status reports `stale`. To update the view, right-click the **AnalyticService** and click **Actions > Refresh Cache**.

Column Name	Description
Name	Reports the name of the entity or logical grouping.
Views	Provides icon hyperlinks on the **AX Property Sheet** and **Wire Sheet** views to other related views . Double-clicking an icon opens the related view.
Enabled	Indicates if the network, device, point or component is active or inactive.
Data	Displays the data used by an algorithm to evaluate some fault detection logic (algorithm).
Node Count	Displays the total number of nodes that are part of the calculation. Use this information to confirm that the result

Column Name	Description
	of a formula looks reasonable.
Alert Count	Out of all nodes that can be in an alert state, reports the total number of nodes that are in an alert state.
Alert Mode	Displays the alert condition used. Displays the default after one occurrence.
Cost	Displays the total cost by multiplying the cost of the alert by the Alert Mode.
Poller	Displays which poller the framework scheduled to evaluate the alert.
Alarm	Displays if the alert is configured to generate an alarm.
Fault Cause	Indicates the reason for a fault.

Buttons

- **New Folder** creates a folder for organizing Analytic alert components.
- **New** creates a new Analytic alert.
- **Edit** opens the selected Analytic alert for updating.
- **Tagit** is disabled for Analytics.

Parent topic: Alert (BAnalyticAlert)

Alert Nodes View

This table lists all components that currently meet the condition(s) monitored by the selected alert.

Columns

You access this view by expanding the **Config** > **Services** > **AnalyticService** > **Alerts** container in the Nav tree, followed by double-clicking an alert node in the container.

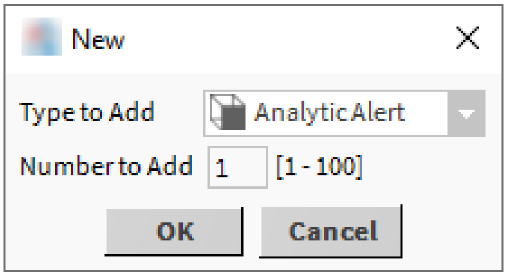
Column	Description
Node	Identifies the node in the Hierarchy or Station tree, which this alert processes.
State	Reports the state of the alert.
Cost	Reports the total cost of the alert condition.

Parent topic: Alert (BAnalyticAlert)

New/edit alert windows

These windows create and edit alerts.

Figure 1. New alert window



New

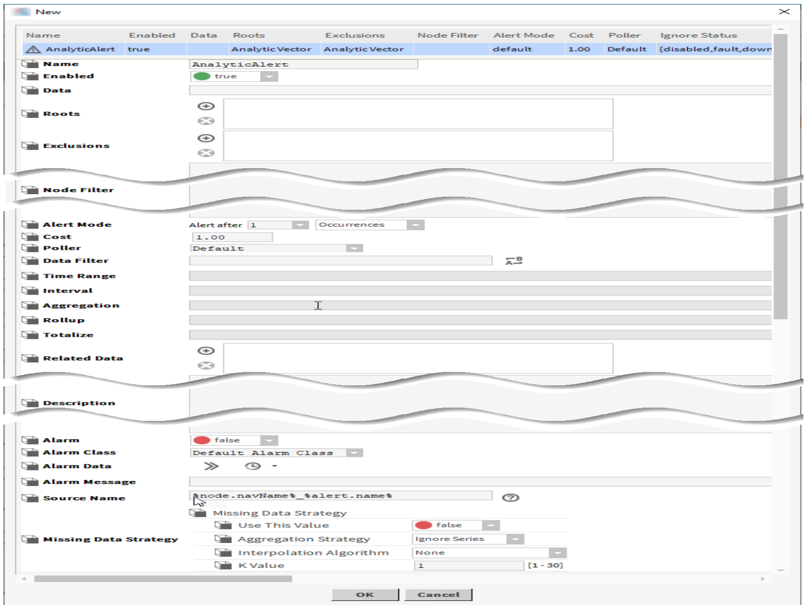
Type to Add: AnalyticAlert

Number to Add: 1 [1 - 100]

You create and edit alerts by double-clicking the Alerts node in the Nav tree and clicking **New Alert**.

Property	Value	Description
Type to Add	drop-down list	Defines the type of alert.
Number to Add	number between 1 and 100	Defines how many alerts to create.

Figure 2. New Alert properties



New

Name	Enabled	Data	Roots	Exclusions	Node Filter	Alert Mode	Cost	Poller	Ignore Status
AnalyticAlert	true		Analytic Vector	Analytic Vector		default	1.00	Default	{disabled,fault,down

Name: AnalyticAlert

Enabled: true

Data

Roots

Exclusions

Node Filter

Alert Mode: Alert after 1 Occurrences

Cost: 1.00

Poller: Default

Data Filter

Time Range

Interval

Aggregation

Rollup

Totalize

Related Data

Description

Alarm: false

Alarm Class: Default Alarm Class

Alarm Data

Alarm Message

Source Name: %node.navName%_%alert.name%

Missing Data Strategy:
- Missing Data Strategy
  - Use This Value: false
  - Aggregation Strategy: Ignore Series
  - Interpolation Algorithm: None
  - K Value: 1 [1 - 30]

The top two rows of this window display the database view of each added alert. In addition to the standard property (Enabled), these properties configure an alert.

Property	Value	Description
Name	text	Defines a descriptive name for the alert.
Data	tag or algorithm name	Specifies the an algorithm (or tag) used to retrieve data. The purpose of an alert is to detect a fault condition (Boolean value result). In most cases, you would pick an algorithm that has a Boolean result when configuring this property for an alert. Algorithms with a numeric or enum value greater than zero are considered true and less than zero are

Property	Value	Description
		considered false. This is how output from one algorithm becomes the input data source to another algorithm. The prefix for algorithms is alg:.
Roots	ORD	Beginning from the root of the Nav tree, this property defines the ORD from which to start scanning for alert conditions. The framework eliminates nodes listed in the Exclusions property before it evaluates each node for an alert condition.
Exclusions	ORD	Identifies the nodes (and their subtree) to exclude when scanning for points to monitor.
Node Filter	optional NEQL predicate	Limits which components to process the alert against in conjunction with Roots and Exclusions. Typically, you use marker tags, such as n:device, hs:ahu, hs:vav, etc. to only process the alert against a device, floor or building.
Cost	currency	Calculates the total cost by multiplying the cost of the alert by the Alert Mode. For example, if Alert Mode is Runtime Minutes, the total cost is the cost-per-minute multiplied by the number of minutes the algorithm maintained a result of true. This value indicates the severity of the alert.
Poller	drop-down list	Schedules alert evaluation.
Data Filter	optional NEQL predicate	Identifies data sources in the subtree of the request node. When a predicate accepts a node that contains the desired data, the system does not search the node's subtree for additional values (including the root node).
Time Range	drop-down list	This property is required for rollup requests (analyticRollup), trends (analyticTrend), and rollup bindings. It is optional elsewhere. It is not used if the block's Use Request Time Range property is true and the request specifies a time range. Options range from From to All.
Interval	optional drop-down list (defaults to the optimal number of records on reports)	Refers to the BInterval component, which the framework uses to identify the time between values in a trend (time series). When specified, a rollup is required, which causes the system to combine all values

Property	Value	Description
		that fall into a single interval. Options range from None to a Year. Above the drop-down list, the Use This Value check box turns on and off the check box next to Interval in the **Settings** window (you access this window by clicking the Edit button (✎), followed by clicking the Settings button (⚙) on the chart). The availability of this check box provides an easy way for a user to enable and disable the use of intervals in chart calculations.
Aggregation	value (defaults to First)	Configures the default function to apply when the analytic request combines values from multiple data sources into a single value. This applies to both value and trend requests. If aggregation is not enabled in the binding/settings window, the aggregation value defined in the Data Definition applies to all chart bindings, reports and tables. And returns the logical “and” of Boolean values. Avg returns the statistical mean, which is determined by calculating the sum of all values and dividing by the number of values. Count returns the total number or quantity of values in a combination. If you request this value on a binding in a PX view, the system counts the number of values based on the properties defined by the data source block and the algorithm’s **Property Sheet**. First returns the first value in the combination. Last returns the last value in the combination. Max returns the highest value in the combination. Median returns the value in the middle of a sorted combination—the number that separates the higher half from the lower half. Min returns the lowest value in the combination. Mode returns the statistically most frequently occurring number in the combination. Or returns the logical “or” of Boolean values. Range returns the statistical difference between the largest and smallest values in the combination.

Property	Value	Description
		Sum adds together all values in the combination resulting in a single value. Load Factor returns the average value divided by peak value. Std Dev returns the standard deviation of the values in the combination.
Rollup	value (when configured in the Data Definition, defaults to First, when configured elsewhere, defaults to the value as defined in the Data Definition)	Configures the default function to apply when the analytic request needs to rollup records from a single data source into less granular records. If rollup is not enabled in the binding/settings window, the rollup value configured in the Data Definition applies to all chart bindings, reports and tables. And returns the logical "and" of Boolean values. Avg returns the statistical mean, which is determined by calculating the sum of all values and dividing by the number of values. Count returns the total number or quantity of values in a combination. If you request this value on a binding in a PX view, the system counts the number of values based on the properties defined by the data source block and the algorithm's **Property Sheet**. First returns the first value in the combination. Last returns the last value in the combination. Max returns the highest value in the combination. Median returns the value in the middle of a sorted combination—the number that separates the higher half from the lower half. Min returns the lowest value in the combination. Mode returns the statistically most frequently occurring number in the combination. Or returns the logical "or" of Boolean values. Range returns the statistical difference between the largest and smallest values in the combination. Sum adds together all values in the combination resulting in a single value. Std Dev calculates the standard deviation of the values in the combination. Load Factor calculates the average divided

Property	Value	Description
		by peak (Max) value.
Totalize	true (default) or false	Turns on (true) and off (false) value accumulation. By default, the framework totalizes (accumulates) all consumption history values in charts, tables and reports. To prevent cumulative values, disable this property (set it to false).
Alert Mode	drop-down list	Defines the alert condition to use when you specify a time range. The alarm is raised within poller time when the time range is not specified. To configure an alert at a different time, specify a time range using one of these options.: Occurrences identifies the number of times an algorithm must output a result of “true” over the configured time range for there to be an alert condition. Runtime Seconds identifies the number of seconds an algorithm must maintain a result of “true” over the configured time range for there to be an alert condition. Runtime Minutes identifies the number of minutes an algorithm must maintain a result of “true” over the configured time range for there to be an alert condition. Runtime Hours identifies the number of hours an algorithm must maintain a result of “true” over the configured time range for there to be an alert condition.
Cost	dollars	Calculates the total cost by multiplying the cost of the alert by the Alert Mode. For example, if Alert Mode is Runtime Minutes, the total cost is the cost-per-minute multiplied by the number of minutes the algorithm maintained a result of true. This value indicates the severity of the alert.
Description	string	Provides additional context regarding the alert.
Related Data	tags	Lists the algorithm specified by the Data property and any inputs (tag, tag group, or algorithm names). To populate this property, right-click the alert and click **Actions > Find Related Data**.
Alarm	true or false (default)	Configures the alert to generate an alarm record (true) or not to generate the record (false). qA

Property	Value	Description
Alarm Class	drop-down list (defaults to Medium)	Specifies the alarm routing option for the component. Replace provides a selection list of a local alarm classes, from which to select one to use for all alarms received from this device. Use Existing routes alarms from this remote station to any matching alarm class, that is, one with an identical name as that in each alarm record. If the program finds no local matching alarm class, it uses the station's default alarm class. Prepend adds leading text (as specified) to the incoming alarm class string, then routes it to any local matching alarm class in the station. Append adds trailing text (as specified) to the incoming alarm class string, then routes it to any local matching alarm class in the station.
Alarm Data	BFormat	Defines the data to add to the alarm data facets of the alarm record. The framework resolves the BFormats at alarm record creation time. BFormats must begin with the alert or node token to access the corresponding entity.
Alarm Message	BFormat	Creates a message value for the alarm record. The framework resolves the BFormats when requested. BFormats must begin with the alert or node token to access the corresponding entity.
Missing Data Strategy	additional properties	Configures how the framework handles missing data in a series when processing analytic requests and one or more records are missing for an interval. It applies when even a single record for an interval is missing. It does not apply to value requests. Refer to Missing Data Strategy
Source Name	BFormat	Displays the name in an alarm record that identifies the source of the alarm.

Alarm data BFormat syntax

Table 1. BFormat examples

BFormat	Description
%alert.name%	Identifies the slot name of the alert object. Do not use **%alert.message%**.
%node.parent.displayName%	Identifies the display name of the parent for the node that is going into

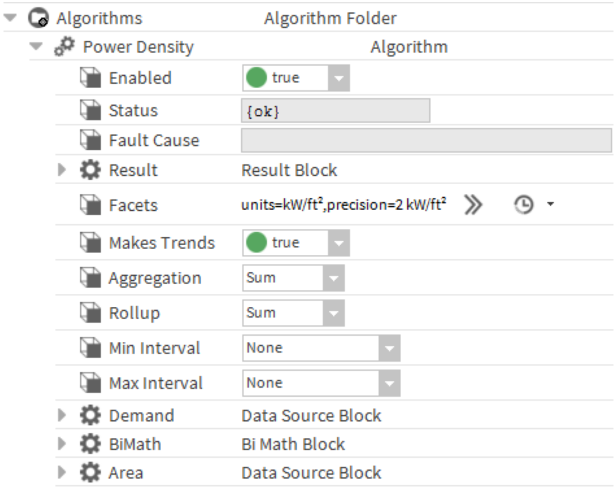
BFormat	Description
	alert.
%node.name%	Resolves the name of the component in the hierarchy tree. The name of a component might include escaped characters, such as “hierarchy:/Campus/station$3a$7ch$3acafb,” which may not be easily human readable.
%node.navDisplayName%	Resolves the display name of the component in the hierarchy tree, such as Building1.
%node.navOrd.resolve.component.slotPath%	Resolves the component in the station and displays the slot path, such as slot:/AnalyticsPlayground/Campus/Building1.
%node.navOrd.resolve.component.parent.displayName%	Resolves the parent component in the station and returns the display name, such as Campus. The parent of the component in the station may be different from its parent in the hierarchy tree.

Parent topic: Alert (BAnalyticAlert)

Algorithm (BAlgorithm)

This component analyzes data by managing the execution of blocks in the **Wire Sheet**, which produces a result via the result block. You can view the result of processing an algorithm in various ways including, but not limited to: a control point with an Analytic Proxy Ext, a Bound Label widget with an Analytic Binding, a chart with an Analytic Binding or a table with an Analytic Binding. Algorithms are implemented with logic blocks linked together using a **Wire Sheet**. They may combine both historical and real-time data to produce results, which may be a trend or a single value.

Figure 1. Example of an Algorithm properties



Algorithms	Algorithm Folder
Power Density	Algorithm
Enabled	true
Status	{ok}
Fault Cause	
Result	Result Block
Facets	units=kW/ft²,precision=2 kW/ft²
Makes Trends	true
Aggregation	Sum
Rollup	Sum
Min Interval	None
Max Interval	None
Demand	Data Source Block
BiMath	Bi Math Block
Area	Data Source Block

To view an algorithm’s **Property Sheet**, open the **Algorithm Manager** (double-click **Config > Services > AnalyticService > Algorithms**), right-click the algorithm, and click **Views > Property Sheet**.

This view contains the properties described in the table below. Each added block contains additional properties.

In addition to the standard properties (Status, Fault Cause, and Facets), these properties are unique to this component.

Property	Value	Description
Enabled	true (default) or false	Determines if the algorithm processes all requests. true processes all requests. false prevents the algorithm from processing all requests.
Result	additional properties	Is a frozen property of the algorithm and must be the final block linked in the **Wire Sheet** to output the result of the algorithm being processed.
Facets	units, precision, min, max, etc.	Configure how to display historical and calculated values, such as units or precision. A unit is a standard facet that applies to both data input and data output. You use it for viewing a point's value or algorithm's result. If a units facet is assigned, it need not match the units facet of a Data Source or Data Definition; however, it must be correct for the raw value being processed and must be convertible to the corresponding unit specified in the Data Definition or Data Source.
Makes Trends	true (default) or false	Controls the processing of trend requests. false outputs a constant value for both value and trend requests. true outputs a trend with constant values for trend requests.
Aggregation	drop-down list(defaults to First)	Configures the default aggregation function passed through the algorithm when the Data Source block's Use Request Aggregation property is set to true and the Aggregation property is not explicitly configured in the request. If aggregation is not enabled in the binding/settings window, the aggregation value defined in the Data Definition applies to all chart bindings, reports and tables. And returns the logical "and" of Boolean values. Avg returns the statistical mean, which is determined by calculating the sum of all values and dividing by the number of values. Count returns the total number or quantity of values in a combination. If you request this value on a binding in a PX view, the system counts the number of values based on the properties defined by the data source block and the algorithm's **Property Sheet**.

Property	Value	Description
		First returns the first value in the combination. Last returns the last value in the combination. Max returns the highest value in the combination. Median returns the value in the middle of a sorted combination—the number that separates the higher half from the lower half. Min returns the lowest value in the combination. Mode returns the statistically most frequently occurring number in the combination. Or returns the logical "or" of Boolean values. Range returns the statistical difference between the largest and smallest values in the combination. Sum adds together all values in the combination resulting in a single value. Load Factor returns the average value divided by peak value. Std Dev returns the standard deviation of the values in the combination.
Rollup	check box (if optional, and) drop-down list or ORD parameter (when configured in the data definition, defaults to First, when configured elsewhere, defaults to the value as defined in the Data Definition); rollup=option	Configures the default rollup function passed through the algorithm when the Data Source block Use Request Rollup property is set to true and the rollup property is not explicitly configured in the request. If rollup is not enabled in the binding/ settings window, the rollup value configured in the Data Definition applies to all chart bindings, reports and tables. And returns the logical "and" of Boolean values. Avg returns the statistical mean, which is determined by calculating the sum of all values and dividing by the number of values. Count returns the total number or quantity of values in a combination. If you request this value on a binding in a PX view, the system counts the number of values based on the properties defined by the data source block and the algorithm's **Property Sheet**. First returns the first value in the combination. Last returns the last value in the

Property	Value	Description
		combination. Max returns the highest value in the combination. Median returns the value in the middle of a sorted combination—the number that separates the higher half from the lower half. Min returns the lowest value in the combination. Mode returns the statistically most frequently occurring number in the combination. Or returns the logical “or” of Boolean values. Range returns the statistical difference between the largest and smallest values in the combination. Sum adds together all values in the combination resulting in a single value. Std Dev calculates the standard deviation of the values in the combination. Load Factor calculates the average divided by peak (Max) value.
Min Interval	drop-down with time intervals (defaults to None)	Defines the minimum wait time between processing items in the queue. The default value of zero (0) does not enforce a minimum wait time. Triggered Poller components do not use a Rate property. You should use the Min Interval property to balance CPU load by slowing down the execution of items in the queue.
Max Interval	drop-down with time intervals (defaults to None)	Defines the maximum wait time between processing items in the queue. The default value of zero (0) does not enforce a maximum wait time.

What follows in the Property Sheet are containers for the various logic blocks. Logic block properties vary depending on the type of block. Each block is documented in the *Logic blocks* chapter.

- **Algorithms folder**
  This folder contains all framework algorithms. Double-clicking this component opens the **Algorithm Manager**.
- **Algorithm Manager view**
  This table view manages the algorithms used by alerts, Px views, and standard Niagara logic. Using this view you can group algorithms (in folders), create and edit algorithms as well as tag them.
- **New/Edit algorithm windows**
  These windows create and edit algorithm properties.

Parent topic: Components, views and windows

Algorithms folder

This folder contains all framework algorithms. Double-clicking this component opens the **Algorithm Manager**.

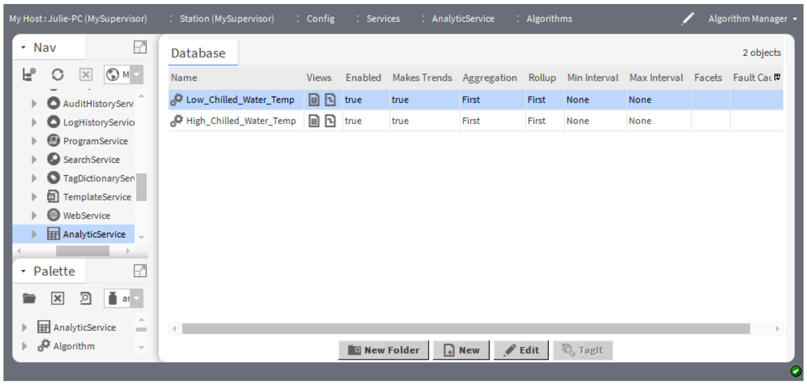
Parent topic: Algorithm (BAlgorithm)

Algorithm Manager view

This table view manages the algorithms used by alerts, Px views, and standard Niagara logic. Using this view you can group algorithms (in folders), create and edit algorithms as well as tag them.

Columns

Figure 1. Algorithm Manager view



Name	Views	Enabled	Makes Trends	Aggregation	Rollup	Min Interval	Max Interval	Facets	Fault Cau
Low_Chilled_Water_Temp		true	true	First	First	None	None		
High_Chilled_Water_Temp		true	true	First	First	None	None		

Column Name	Description
Name	Reports the name of the algorithm.
Views	Provides icon hyperlinks on the **AX Property Sheet** and **Wire Sheet** views to other related views . Double-clicking an icon opens the related view.
Enabled	Indicates if the network, device, point or component is active or inactive.
Makes Trends	Displays whether the algorithm processes trend requests.
Aggregation	Displays the mathematical function used to combine data from multiple data sources.
Rollup	Displays the mathematical function to be used to combine data from a single source.
Min Interval	Displays the minimum allowed interval when requesting a trend.
Max Interval	Displays the maximum allowed interval when requesting a trend.
Facets	Displays the configured facets.
Fault Cause	Indicates the reason for a fault.

Buttons

- **New Folder** creates a new algorithm folder for organizing algorithm components.
- **New** creates a new algorithm.

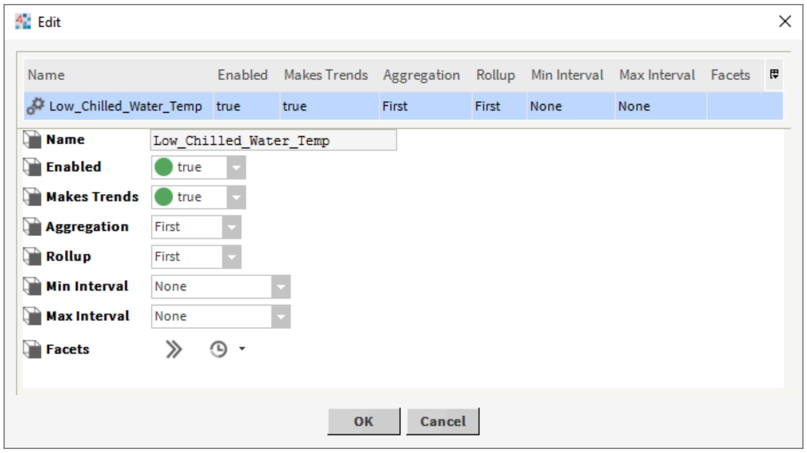
- **Edit** opens the algorithm for updating.
- **Tagit** is disabled for Analytics.

Parent topic: Algorithm (BAlgorithm)

New/Edit algorithm windows

These windows create and edit algorithm properties.

Figure 1. Edit algorithm window



Edit

Name	Enabled	Makes Trends	Aggregation	Rollup	Min Interval	Max Interval	Facets
Low_Chilled_Water_Temp	true	true	First	First	None	None	

Name: Low_Chilled_Water_Temp

Enabled: true

Makes Trends: true

Aggregation: First

Rollup: First

Min Interval: None

Max Interval: None

Facets

This window opens when you click **Edit** in the **Algorithm Manager** view.

Property	Value	Description
Name	text	Provides a unique name for each algorithm.
Enabled	true or false	Activates (true) and deactivates (false) use of the object (network, device, point, component, table, schedule, descriptor, etc.).
Makes Trends	true (default) or false	Controls the processing of trend requests. false outputs a constant value for both value and trend requests. true outputs a trend with constant values for trend requests.
Aggregation	drop-down list (defaults to First)	Configures the default aggregation function passed through the algorithm when the Data Source block Use Request Aggregation property is set to true and the Aggregation property is not explicitly configured in the request. If aggregation is not enabled in the binding/settings window, the aggregation value defined in the Data Definition applies to all chart bindings, reports and tables.

Property	Value	Description
		And returns the logical “and” of Boolean values. Avg returns the statistical mean, which is determined by calculating the sum of all values and dividing by the number of values. Count returns the total number or quantity of values in a combination. If you request this value on a binding in a PX view, the system counts the number of values based on the properties defined by the data source block and the algorithm’s **Property Sheet**. First returns the first value in the combination. Last returns the last value in the combination. Max returns the highest value in the combination. Median returns the value in the middle of a sorted combination—the number that separates the higher half from the lower half. Min returns the lowest value in the combination. Mode returns the statistically most frequently occurring number in the combination. Or returns the logical “or” of Boolean values. Range returns the statistical difference between the largest and smallest values in the combination. Sum adds together all values in the combination resulting in a single value. Load Factor returns the average value divided by peak value. Std Dev returns the standard deviation of the values in the combination.
Rollup	drop-down list (defaults to First)	Configures the default function to apply when the analytic request needs to rollup records from a single data source into less granular records. If rollup is not enabled in the binding/ settings window, the rollup value configured in the Data Definition applies to all chart bindings, reports and tables. And returns the logical “and” of Boolean values. Avg returns the statistical mean, which is determined by calculating the sum of all values and dividing by the number of

Property	Value	Description
		values. Count returns the total number or quantity of values in a combination. If you request this value on a binding in a PX view, the system counts the number of values based on the properties defined by the data source block and the algorithm's **Property Sheet**. First returns the first value in the combination. Last returns the last value in the combination. Max returns the highest value in the combination. Median returns the value in the middle of a sorted combination—the number that separates the higher half from the lower half. Min returns the lowest value in the combination. Mode returns the statistically most frequently occurring number in the combination. Or returns the logical "or" of Boolean values. Range returns the statistical difference between the largest and smallest values in the combination. Sum adds together all values in the combination resulting in a single value. Std Dev calculates the standard deviation of the values in the combination. Load Factor calculates the average divided by peak (Max) value.
Min Interval	drop-down list (defaults to None)	Defines the minimum wait time between processing items in the queue. The default value of zero (0) does not enforce a minimum wait time. Triggered Poller components do not use a Rate property. You should use the Min Interval property to balance CPU load by slowing down the execution of items in the queue.
Max Interval	drop-down list (defaults to None)	Defines the maximum wait time between processing items in the queue. The default value of zero (0) does not enforce a maximum wait time.
Facets	Units, precison, min, max, etc.	Configure how to display historical and calculated values, such as units or precision. A unit is a standard facet that applies to

Property	Value	Description
		both data input and data output. You use it for viewing a point's value or algorithm's result. If a units facet is assigned, it need not match the units facet of a Data Source or Data Definition; however, it must be correct for the raw value being processed and must be convertible to the corresponding unit specified in the Data Definition or Data Source.

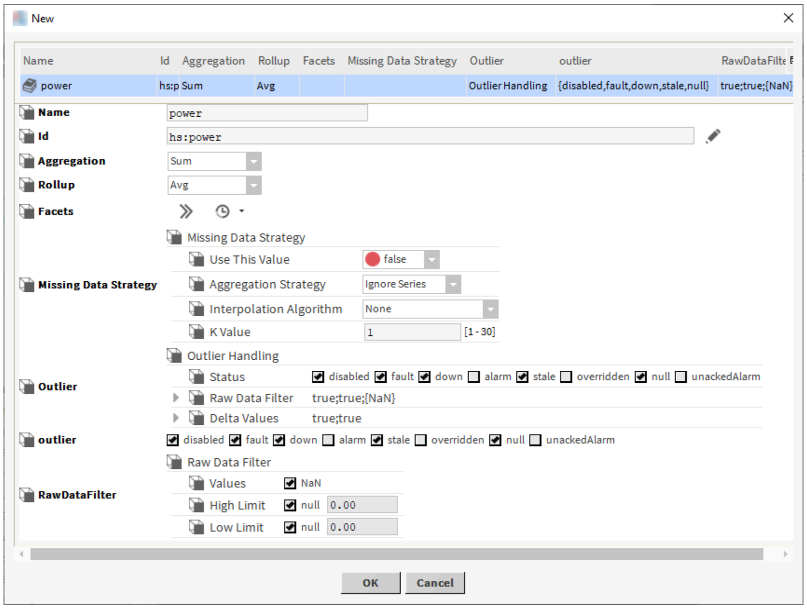
Parent topic: Algorithm (BAlgorithm)

Definition (BAnalyticDataDefinition)

This optional component configures the default values for facets, aggregation and rollup properties. Each tag from the Haystack, Niagara or a custom dictionary requires a definition. An algorithm or a graphics binding (Px view or Ux chart) defines the tag to use for searching a hierarchy and the node at which to begin the search. Using this information the framework retrieves trend data. It then combines the retrieved data using the facets, aggregation and rollup properties defined on the definition that is associated with the tag.

Data Definition components simplify analytic setting up an application by configuring typical properties in one place, which are then applied as defaults throughout the application. When you create a Data Definition for a specific data item, you can still override its properties for a specific analytic request in the Alert, Analytic Proxy Ext or Binding.

Figure 1. Example of a Data Definition properties



Name	Id	Aggregation	Rollup	Facets	Missing Data Strategy	Outlier	outlier	RawDataFilte
power	hs:p	Sum	Avg			Outlier Handling	{disabled,fault,down,stale,null}	true;true;{NaN}

Name	power	
Id	hs:power	
Aggregation	Sum	
Rollup	Avg	
Facets		
Missing Data Strategy	Missing Data Strategy	
	Use This Value	● false
	Aggregation Strategy	Ignore Series
	Interpolation Algorithm	None
	K Value	1 [1 - 30]
Outlier	Outlier Handling	
	Status	☑ disabled ☑ fault ☑ down □ alarm ☑ stale □ overridden ☑ null □ unackedAlarm
	Raw Data Filter	true;true;{NaN}
	Delta Values	true;true
outlier	☑ disabled ☑ fault ☑ down □ alarm ☑ stale □ overridden ☑ null □ unackedAlarm	
RawDataFilter	Raw Data Filter	
	Values	☑ NaN
	High Limit	☑ null 0.00
	Low Limit	☑ null 0.00

To view a definition's **Property Sheet**, double-click open the **Analytic Data Manager** (double-click **Config > Services > AnalyticService > Definitions**) and click **New**.

Property	Value	Description
Name	text	Assigns a unique name to each Data Definition component.
Id	namespace:name	Configures a fully-qualified tag name or tag group name that identifies specific data in the station, such as hs:zoneAirTempSensor to identify Zone Temp sensors. namespace is the name of a tag dictionary. name is the tag name used to collect point output in a hierarchy. Tagging a Data Definition automatically associates the properties defined by the definition with the tag that shares the same name.
Aggregation	drop-down list (defaults to First)	Configures the default function to apply when the analytic request needs to combine values from multiple data sources into a single value. This applies to both value and trend requests. If aggregation is not enabled in the binding/settings window, the aggregation value defined in the Data Definition applies to all chart bindings, reports and tables. And returns the logical “and” of Boolean values. Avg returns the statistical mean, which is determined by calculating the sum of all values and dividing by the number of values. Count returns the total number or quantity of values in a combination. If you request this value on a binding in a PX view, the system counts the number of values based on the properties defined by the data source block and the algorithm’s **Property Sheet**. First returns the first value in the combination. Last returns the last value in the combination. Max returns the highest value in the combination. Median returns the value in the middle of a sorted combination—the number that separates the higher half from the lower half. Min returns the lowest value in the combination. Mode returns the statistically most frequently occurring number in the combination. Or returns the logical “or” of Boolean values. Range returns the statistical difference

Property	Value	Description
		between the largest and smallest values in the combination. Sum adds together all values in the combination resulting in a single value. Load Factor returns the average value divided by peak value. Std Dev returns the standard deviation of the values in the combination.
Rollup	drop-down list (defaults to First)	Configures the default function to apply when the analytic request needs to rollup records from a single data source into less granular records. This typically only applies to trend request, but may also apply to value requests where the Id is an algorithm that contains a block like Runtime or Sliding Window, which processes a trend request. If rollup is not enabled in the binding/ settings window, the rollup value configured in the Data Definition applies to all chart bindings, reports and tables. And returns the logical “and” of Boolean values. Avg returns the statistical mean, which is determined by calculating the sum of all values and dividing by the number of values. Count returns the total number or quantity of values in a combination. If you request this value on a binding in a PX view, the system counts the number of values based on the properties defined by the data source block and the algorithm’s **Property Sheet**. First returns the first value in the combination. Last returns the last value in the combination. Max returns the highest value in the combination. Median returns the value in the middle of a sorted combination—the number that separates the higher half from the lower half. Min returns the lowest value in the combination. Mode returns the statistically most frequently occurring number in the combination. Or returns the logical “or” of Boolean values. Range returns the statistical difference between the largest and smallest values in

Property	Value	Description
		the combination. Sum adds together all values in the combination resulting in a single value. Std Dev calculates the standard deviation of the values in the combination. Load Factor calculates the average divided by peak (Max) value.
Facets	units, precision, min, max, etc.	Clicking the chevron to the right of this property o pens a standard **Config Facets** window. If no facets are defined, these values default to the default facets configured in the tag associated with the point. The facets you configure here override the default facets.
Missing Data Strategy, Use This Value	check box	Enables and disables missing data interpolation for the current value. When enabled, the framework applies this strategy to all requests. When enabled, the framework applies this strategy to all requests.
Missing Data Strategy, Aggregation Strategy	drop-down list	Configures how the framework handles missing trend data (data in a series) when processing analytic requests and one or more records are missing for an interval. It applies when even a single record for an interval is missing. It does not apply to value requests. Note: If the analytic trend request specifies Interval = none, the framework ignores the Missing Data Strategy. Ignore Series ignores the entire series if any record even one interval is missing. Ignore Point ignores any missing records and aggregates the values in the existing records.
Missing Data Strategy, Interpolation Algorithm	drop-down list	Defines the algorithm used to interpolate values for missing values (missing records). None does not interpolate values for missing records. Linear Interpolation replaces a missing record by linearly interpolating the missing value using the prior and next records on either side of the missing record. K-Nearest Neighbor is for numeric, enum and Boolean records. This strategy replaces a missing value by calculating the majority value recorded for the item’s nearest K number of neighbors.

Property	Value	Description
Missing Data Strategy, K Value	Numeric field editor (default = 1, min =1, max = 30)	Defines the number of records used by the configured interpolation algorithm when a record is missing to calculate the interpolated value.
Outlier, Status	Status check boxes (disabled, fault, down, stale, and null are checked by default)	Configures filtering behavior to remove records from a dataset based on the status flags or value of each record. Status values include: disabled, fault, down, alarm, stale, overridden, null, unackedAlarm and NaN (Not a Number. This is another way, similar to the InvalidValueFilter block, to filter records based on bad status conditions. If you check all boxes, the framework filters out all records except those with a status of {ok}, which is always enabled. If you check no box, the framework filters out no records based on status. You may configure additional properties for High Limit and Low Limit (defaults to null check box selected, which does not enforce a limit). This filtering does not apply to value requests. Algorithm blocks may perform additional filtering based on statuses or values. After the framework filters out the records with invalid data, use a missing data strategy to interpolate valid data.
Outlier, RawDataFilter, High Limit	null check box (defaults to checked) or numeric value	Optionally, defines a number above which a data value (an outlier) should be excluded from an analytic calculation. Setting a limit is similar to using a RangeFilter block in an algorithm where values greater than the high limit are excluded from being processed and using an InvalidValueFilter to filter NaN numbers without the benefit of also filtering infinite values. A use case might be that you have a sensor with a range of 0-150 deg F, any readings above 150 would be anomalies or suspect values, which need to filtered out.
Outlier, RawDataFilter, Low Limit	null check box (defaults to checked) or numeric value	Optionally, defines a number below which a data value (an outlier) should be excluded from an analytic calculation. Setting a limit is similar to using a RangeFilter block in an algorithm where values lower than the low limit are excluded from being processed and using an InvalidValueFilter to filter NaN numbers without the benefit of also filtering infinite values. A use case might be that you have a sensor with a range of 0-150 deg F, any

Property	Value	Description
		readings below zero would be anomalies or suspect values, which need to filtered out.
Outlier, Delta Values, High Limit	null check box (defaults to checked) or numeric value	Sets a high limit that applies when Analytics is calculating a delta value. You might have a history for electrical energy consumption (KWH) that is totalized, which means that an ever increasing value is being logged. Analytics gets the delta values (difference between each record) to show the electrical consumption for a period like 15 minutes or a day. The High Limit property would filter out a calculated delta value that exceeds the configured high limit.
Outlier, Delta Values, Low Limit	null check box (defaults to checked) or numeric value	Sets a low limit that applies when Analytics is calculating a delta value. You might have a history for electrical energy consumption (KWH) that is totalized, which means that an ever increasing value is being logged. Analytics gets the delta values (difference between each record) to show the electrical consumption for a period like 15 minutes or a day. The Low Limit property would filter out a calculated delta value that is lower than the configured low limit.
outlier		These properties are duplicate properties of the Outlier Handling properties above. You do not need to configure them.
RawDataFilter, Values		
RawDataFilter, High Limit		
RawDataFilter, Low Limit		

- **Definitions Folder**
  This folder under the **AnalyticService** organizes Analytic Data Definition components. These components contain the Data Definitions that are associated with tags. Double-clicking this folder opens the **Analytic Data Manager**.
- **Outlier Handling**
  These properties configure which values to treat as outliers and how to determine their values.
- **Analytic Data Manager view**
  This view lists the Data Definitions for each data ID, such as real energy, zone temperature, air flow quantity, set point, phase A amperage, phase B amperage, etc. and displays property configurations for each.
- **New/Edit Definition windows**
  These windows create and edit Data Definition properties.

Parent topic: Components, views and windows

Definitions Folder

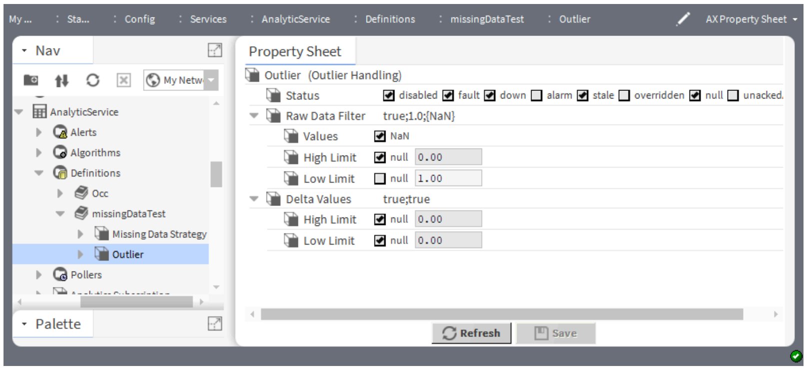
This folder under the **AnalyticService** organizes Analytic Data Definition components. These components contain the Data Definitions that are associated with tags. Double-clicking this folder opens the **Analytic Data Manager**.

Parent topic: Definition (BAnalyticDataDefinition)

Outlier Handling

These properties configure which values to treat as outliers and how to determine their values.

Figure 1. Outlier properties



Outlier (Outlier Handling)
- Status ☑ disabled ☑ fault ☑ down □ alarm ☑ stale □ overridden ☑ null □ unacked.
- Raw Data Filter true;1.0;{NaN}
  - Values ☑ NaN
  - High Limit ☑ null 0.00
  - Low Limit □ null 1.00
- Delta Values true;true
  - High Limit ☑ null 0.00
  - Low Limit ☑ null 0.00

To access this property sheet expand **Config > Services > AnalyticService > Definitions**, expand a definition and double-click **Outlier**.

Property	Value	Description
Outlier, Status	Status check boxes (disabled, fault, down, stale, and null are checked by default)	Configures filtering behavior to remove records from a dataset based on the status flags or value of each record. Status values include: disabled, fault, down, alarm, stale, overridden, null, unackedAlarm and NaN (Not a Number. This is another way, similar to the InvalidValueFilter block, to filter records based on bad status conditions. If you check all boxes, the framework filters out all records except those with a status of {ok}, which is always enabled. If you check no box, the framework filters out no records based on status. You may configure additional properties for High Limit and Low Limit (defaults to null check box selected, which does not enforce a limit). This filtering does not apply to value requests. Algorithm blocks may perform additional filtering based on statuses or values. After the framework filters out the records with invalid data, use a missing data strategy to interpolate valid data.

Property	Value	Description
Outlier, RawDataFilter, High Limit	null check box (defaults to checked) or numeric value	Optionally, defines a number above which a data value (an outlier) should be excluded from an analytic calculation. Setting a limit is similar to using a RangeFilter block in an algorithm where values greater than the high limit are excluded from being processed and using an InvalidValueFilter to filter NaN numbers without the benefit of also filtering infinite values. A use case might be that you have a sensor with a range of 0-150 deg F, any readings above 150 would be anomalies or suspect values, which need to filtered out.
Outlier, RawDataFilter, Low Limit	null check box (defaults to checked) or numeric value	Optionally, defines a number below which a data value (an outlier) should be excluded from an analytic calculation. Setting a limit is similar to using a RangeFilter block in an algorithm where values lower than the low limit are excluded from being processed and using an InvalidValueFilter to filter NaN numbers without the benefit of also filtering infinite values. A use case might be that you have a sensor with a range of 0-150 deg F, any readings below zero would be anomalies or suspect values, which need to filtered out.
Outlier, Delta Values, High Limit	null check box (defaults to checked) or numeric value	Sets a high limit that applies when Analytics is calculating a delta value. You might have a history for electrical energy consumption (KWH) that is totalized, which means that an ever increasing value is being logged. Analytics gets the delta values (difference between each record) to show the electrical consumption for a period like 15 minutes or a day. The High Limit property would filter out a calculated delta value that exceeds the configured high limit.
Outlier, Delta Values, Low Limit	null check box (defaults to checked) or numeric value	Sets a low limit that applies when Analytics is calculating a delta value. You might have a history for electrical energy consumption (KWH) that is totalized, which means that an ever increasing value is being logged. Analytics gets the delta values (difference between each record) to show the electrical consumption for a period like 15 minutes or a day. The Low Limit property would filter out a calculated delta value that is lower than the configured low limit.

- **Outlier, Raw Data Filter**

These properties define high and low values as outliers to be removed from the data set.

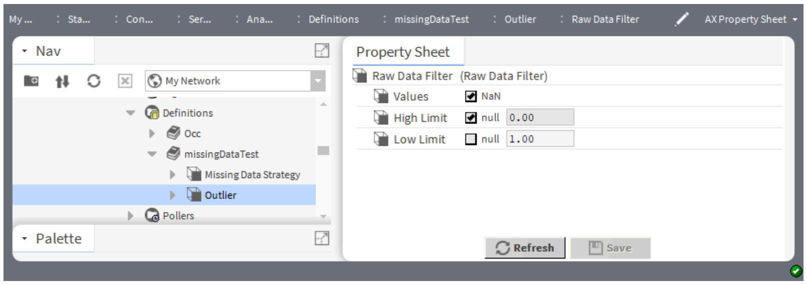
- **Outlier, Delta Values**
  These properties filter data from a data set based on a change in value.

Parent topic: Definition (BAnalyticDataDefinition)

Outlier, Raw Data Filter

These properties define high and low values as outliers to be removed from the data set.

Figure 1. Outlier, Raw Data Filter properties



Raw Data Filter (Raw Data Filter)

Values	☑ NaN	
High Limit	☑ null	0.00
Low Limit	□ null	1.00

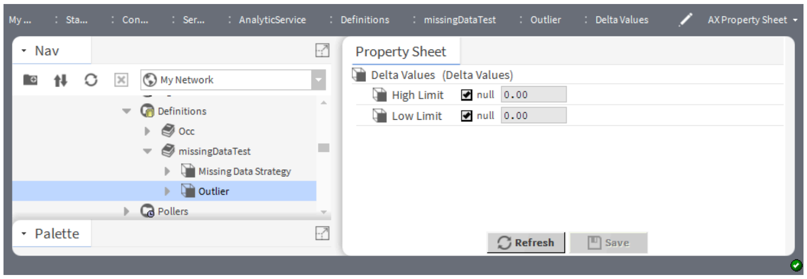
Property	Value	Description
Values	check box for NaN (not a number)	Excludes values that are not numbers.
High Limit	null check box (defaults to checked) or numeric value	Optionally, defines a number above which a data value (an outlier) should be excluded from an analytic calculation. Setting a limit is similar to using a RangeFilter block in an algorithm where values greater than the high limit are excluded from being processed and using an InvalidValueFilter to filter NaN numbers without the benefit of also filtering infinite values. A use case might be that you have a sensor with a range of 0-150 deg F, any readings above 150 would be anomalies or suspect values, which need to filtered out.
Low Limit	null check box (defaults to checked) or numeric value	Optionally, defines a number below which a data value (an outlier) should be excluded from an analytic calculation. Setting a limit is similar to using a RangeFilter block in an algorithm where values lower than the low limit are excluded from being processed and using an InvalidValueFilter to filter NaN numbers without the benefit of also filtering infinite values. A use case might be that you have a sensor with a range of 0-150 deg F, any readings below zero would be anomalies or suspect values, which need to filtered out.

Parent topic: Outlier Handling

Outlier, Delta Values

These properties filter data from a data set based on a change in value.

Figure 1. Outlier, Delta Values properties



Delta Values (Delta Values)
High Limit ☑ null 0.00
Low Limit ☑ null 0.00

Property	Value	Description
High Limit	null check box (defaults to checked) or numeric value	Optionally, defines a number above which a data value (an outlier) should be excluded from an analytic calculation. Setting a limit is similar to using a RangeFilter block in an algorithm where values greater than the high limit are excluded from being processed and using an InvalidValueFilter to filter NaN numbers without the benefit of also filtering infinite values. A use case might be that you have a sensor with a range of 0-150 deg F, any readings above 150 would be anomalies or suspect values, which need to filtered out.
Low Limit	null check box (defaults to checked) or numeric value	Optionally, defines a number below which a data value (an outlier) should be excluded from an analytic calculation. Setting a limit is similar to using a RangeFilter block in an algorithm where values lower than the low limit are excluded from being processed and using an InvalidValueFilter to filter NaN numbers without the benefit of also filtering infinite values. A use case might be that you have a sensor with a range of 0-150 deg F, any readings below zero would be anomalies or suspect values, which need to filtered out.

Parent topic: Outlier Handling

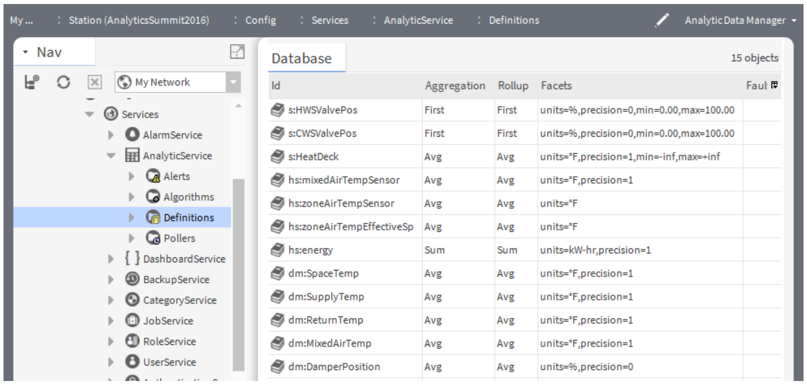
Analytic Data Manager view

This view lists the Data Definitions for each data ID, such as real energy, zone temperature, air flow quantity, set point, phase A amperage, phase B amperage, etc. and displays property configurations for each.

Columns

Each Analytic Data Folder has an Analytic Data Manager view assigned as its primary view. This view allows creating additional nested Analytic Data Folder components to organize Analytic Data Definition components in groups instead of maintaining all components in a flat tree. This view also allows creating, editing and deleting Analytic Data Definition components.

Figure 1. Analytic Data Manager view



Id	Aggregation	Rollup	Facets	Faul
s:HWSValvePos	First	First	units=%,precision=0,min=0.00,max=100.00	
s:CWSValvePos	First	First	units=%,precision=0,min=0.00,max=100.00	
s:HeatDeck	Avg	Avg	units=°F,precision=1,min=-inf,max=+inf	
hs:mixedAirTempSensor	Avg	Avg	units=°F,precision=1	
hs:zoneAirTempSensor	Avg	Avg	units=°F	
hs:zoneAirTempEffectiveSp	Avg	Avg	units=°F	
hs:energy	Sum	Sum	units=kW-hr,precision=1	
dm:SpaceTemp	Avg	Avg	units=°F,precision=1	
dm:SupplyTemp	Avg	Avg	units=°F,precision=1	
dm:ReturnTemp	Avg	Avg	units=°F,precision=1	
dm:MixedAirTemp	Avg	Avg	units=°F,precision=1	
dm:DamperPosition	Avg	Avg	units=%,precision=0	

To open this view, double-click **Services > AnalyticService > Definitions**.

Column Name	Description
Id	Displays the tag name.
Aggregation	Displays the mathematical function used to combine data from multiple data sources.
Rollup	Displays the mathematical function used to combine data from single data source.
Facets	Display the configured facets.
Fault Cause	Indicates the reason for a fault.
Missing Data Strategy	If enabled, displays the configured strategy, such as Ignore Series;K Nearest Neighbour;1.
Outlier	Displays all status bit check boxes, which are selected to be filtered.
outlier	Shows the status bits.
RawDataFilter	Displays filter properties, such as 800.0;0.0;{NaN}.

Buttons

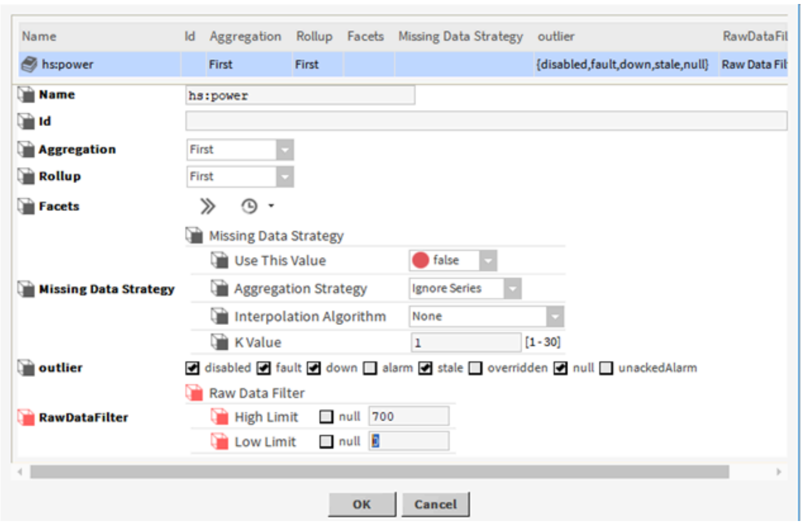
- **New Folder** creates a new folder for organizing Data Definition components.
- **New** creates a new Data Definition component.
- **Edit** opens the Data Definition component for updating.
- **Tagit** is disabled for Analytics.

Parent topic: Definition (BAnalyticDataDefinition)

New/Edit Definition windows

These windows create and edit Data Definition properties.

Figure 1. New definition window



Name	Id	Aggregation	Rollup	Facets	Missing Data Strategy	outlier	RawDataFil
hs:power		First	First			{disabled,fault,down,stale,null}	Raw Data Fil

Name: hs:power

Id:

Aggregation: First

Rollup: First

Facets:

Missing Data Strategy:
- Missing Data Strategy
  - Use This Value: ● false
  - Aggregation Strategy: Ignore Series
  - Interpolation Algorithm: None
  - K Value: 1 [1 - 30]

outlier: ☑ disabled ☑ fault ☑ down □ alarm ☑ stale □ overridden ☑ null □ unackedAlarm

RawDataFilter:
- Raw Data Filter
  - High Limit: □ null 700
  - Low Limit: □ null 0

This window opens when you click **New** on the **Analytic Data Manager** view.

Property	Value	Description
Name	text	Provides a descriptive name for the definition.
Id	tag	Configures a fully-qualified (namespace:tagName), tag name or tag group name that identifies some specific data in the station, such as hs:zoneAirTempSensor to identify Zone Temp sensors.
Aggregation	drop-down list (defaults to First)	Configures the default function to apply when the analytic request combines values from multiple data sources into a single value. This applies to both value and trend requests. If aggregation is not enabled in the binding/settings window, the aggregation value defined in the Data Definition applies to all chart bindings, reports and tables. And returns the logical "and" of Boolean values. Avg returns the statistical mean, which is determined by calculating the sum of all values and dividing by the number of values. Count returns the total number or quantity of values in a combination. If you request this value on a binding in a PX view, the system counts the number of values based on the properties defined by the data source block and the algorithm's **Property Sheet**. First returns the first value in the

Property	Value	Description
		combination. Last returns the last value in the combination. Max returns the highest value in the combination. Median returns the value in the middle of a sorted combination—the number that separates the higher half from the lower half. Min returns the lowest value in the combination. Mode returns the statistically most frequently occurring number in the combination. Or returns the logical "or" of Boolean values. Range returns the statistical difference between the largest and smallest values in the combination. Sum adds together all values in the combination resulting in a single value. Load Factor returns the average value divided by peak value. Std Dev returns the standard deviation of the values in the combination.
Rollup	drop-down list (defaults to First)	Configures the default function to apply when the analytic request needs to rollup records from a single data source into less granular records. If rollup is not enabled in the binding/ settings window, the rollup value configured in the Data Definition applies to all chart bindings, reports and tables. And returns the logical "and" of Boolean values. Avg returns the statistical mean, which is determined by calculating the sum of all values and dividing by the number of values. Count returns the total number or quantity of values in a combination. If you request this value on a binding in a PX view, the system counts the number of values based on the properties defined by the data source block and the algorithm's **Property Sheet**. First returns the first value in the combination. Last returns the last value in the combination. Max returns the highest value in the combination.

Property	Value	Description
		Median returns the value in the middle of a sorted combination—the number that separates the higher half from the lower half. Min returns the lowest value in the combination. Mode returns the statistically most frequently occurring number in the combination. Or returns the logical “or” of Boolean values. Range returns the statistical difference between the largest and smallest values in the combination. Sum adds together all values in the combination resulting in a single value. Std Dev calculates the standard deviation of the values in the combination. Load Factor calculates the average divided by peak (Max) value.
Facets	units, precision, min, max, etc.	Configure how to display historical and calculated values, such as units or precision. A unit is a standard facet that applies to both data input and data output. You use it for viewing a point’s value or algorithm’s result. If a units facet is assigned, it need not match the units facet of a Data Source or Data Definition; however, it must be correct for the raw value being processed and must be convertible to the corresponding unit specified in the Data Definition or Data Source.
Missing Data Strategy	additional properties	Configures how the framework handles missing data in a series when processing analytic requests and one or more records are missing for an interval. It applies when even a single record for an interval is missing. It does not apply to value requests.
Outlier (Outlier Handling)	Status check boxes (disabled, fault, down, stale, and null are checked by default)	Configures filtering behavior to remove records from a dataset based on the status flags or value of each record. Status values include: disabled, fault, down, alarm, stale, overridden, null, unackedAlarm and NaN (Not a Number. This is another way, similar to the InvalidValueFilter block, to filter records based on bad status conditions. If you check all boxes, the framework filters out all records except those with a status of {ok}, which is always enabled.

Property	Value	Description
		If you check no box, the framework filters out no records based on status. You may configure additional properties for High Limit and Low Limit (defaults to null check box selected, which does not enforce a limit). This filtering does not apply to value requests. Algorithm blocks may perform additional filtering based on statuses or values. After the framework filters out the records with invalid data, use a missing data strategy to interpolate valid data.
RawDataFilter, High Limit	null check box (defaults to checked) or numeric value	Optionally, defines a number above which a data value (an outlier) should be excluded from an analytic calculation. Setting a limit is similar to using a RangeFilter block in an algorithm where values greater than the high limit are excluded from being processed and using an InvalidValueFilter to filter NaN numbers without the benefit of also filtering infinite values. A use case might be that you have a sensor with a range of 0-150 deg F, any readings above 150 would be anomalies or suspect values, which need to filtered out.
RawDataFilter, Low Limit	null check box (defaults to checked) or numeric value	Optionally, defines a number below which a data value (an outlier) should be excluded from an analytic calculation. Setting a limit is similar to using a RangeFilter block in an algorithm where values lower than the low limit are excluded from being processed and using an InvalidValueFilter to filter NaN numbers without the benefit of also filtering infinite values. A use case might be that you have a sensor with a range of 0-150 deg F, any readings below zero would be anomalies or suspect values, which need to filtered out.
Delta Value, High Limit	null check box (defaults to checked) or numeric value	Processes a high-limit delta value from a totalized value.
Delta Value, Low Limit	null check box (defaults to checked) or numeric value	Processes a low-limit delta value from a totalized value.

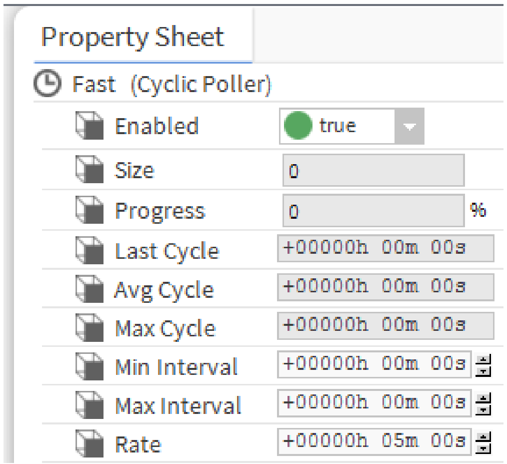
Parent topic: Definition (BAnalyticDataDefinition)

Cyclic poller (BCyclicPoller)

This poller executes framework proxy points and alerts at even intervals across the time span as defined by the **Min Interval**, **Max Interval** and **Rate**.

Analytics pollers are different from Niagara interval triggers. Analytic pollers combine all the things you want the poller to do and attempts to accomplish your polling tasks (Size) within the Rate as defined on this **Property Sheet**. A poller attempts to schedule each component's execution during the period to spread out the work in an effort to minimize CPU impact, whereas a trigger executes all linked components at the same time and the station processes queued items as quickly as possible.

Figure 1. Example of a cyclic poller view



Property Sheet

Fast (Cyclic Poller)

Enabled ● true

Size 0

Progress 0 %

Last Cycle +00000h 00m 00s

Avg Cycle +00000h 00m 00s

Max Cycle +00000h 00m 00s

Min Interval +00000h 00m 00s

Max Interval +00000h 00m 00s

Rate +00000h 05m 00s

To view a cyclic poller **Property Sheet**, open the **Poller Manager** (double-click **Config > Services > AnalyticService > Pollers**), right-click a poller of type **Cyclic Poller**, and click **Views > Property Sheet**.

In addition to the standard property (Enabled), this component provides these properties.

Property	Value	Description
Size	read-only number	Indicates the number of alerts and Analyitc Proxy Ext components, which the poller is managing.
Progress	read-only percentage	Indicates the progress through the current cycle.
Last Cycle	read-only time	Displays the amount of time taken to run the last instance of the cycle.
Avg Cycle	read-only time	Displays the average amount of time required to run an instance of the cycle.
Max cycle	read-only time	Displays the maximum amount of time taken to run an instance of the cycle.
Min Interval	hours minutes seconds	Defines the minimum wait time between processing items in the queue. The default value of zero (0) does not enforce a minimum wait time. Triggered Poller components do not use a Rate property. You should use the Min Interval property to balance CPU load by slowing down the execution of items in the queue. For example, a queue size of 87344 with a

Property	Value	Description
		Min Interval of 2 seconds creates a cycle that lasts a little over 48 hours.
Max Interval	hours minutes seconds	Defines the maximum wait time between processing items in the queue. The default value of zero (0) does not enforce a maximum wait time.
Rate	number	Establishes polling frequency by defining the amount of time between cyclic polls.

Poller examples

A cyclic poller tries to balance the execution by dividing the Rate by the number of items in the queue. The following table explains three poller examples.

Items in the queue	Rate	Min Interval	Max Interval	Execution
6	30	0	0	The poller attempts to execute an item once every five seconds, such that it executes all six items once every 30 seconds.
30	30	2	0	The poller needs to execute an item once every 1 second, but Min Interval = 2 seconds causing the poller to wait for 2 seconds between each item. The result: the poller executes all 30 items once every 1 minute, which is longer than the configured rate.
2	30	0	5	The poller needs to execute an item once every 15 seconds, but Max Interval = 5 seconds so the poller waits only 5 seconds between each item. The result: the poller executes the 2 items once every 30 seconds, but the items in the queue execute closer together, only 5 seconds apart instead of 30 seconds apart.

- **Triggered poller (BTriggeredPoller)**
  This poller executes or alerts analytic proxy points when you invoke its execute action.
- **Pollers folder**
  The **Pollers** folder contains poller objects, which define how frequently the framework executes alert and Analyitc Proxy Ext components. Double-clicking this component opens the **Poller Manager**.
- **Poller Manager view**
  This table view lists the pollers that execute alerts and Analytic Proxy Extension components.
- **New/Edit poller window**
  This window creates a new poller.

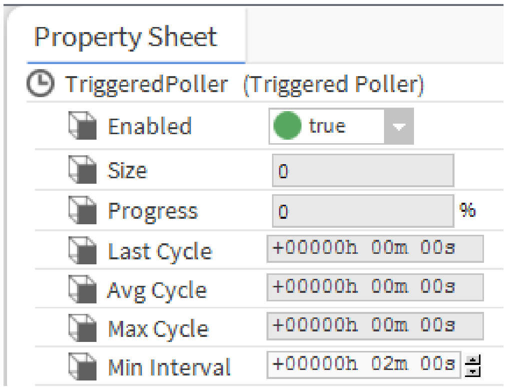
Parent topic: Components, views and windows

Triggered poller (BTriggeredPoller)

This poller executes or alerts analytic proxy points when you invoke its execute action.

To view a triggered poller **Property Sheet**, open the **Poller Manager** (double-click **Config > Services > AnalyticService > Pollers**), right-click a poller of type **Triggered Poller**, and click **Views > Property Sheet**.

Figure 1. Example of a triggered poller properties



Property Sheet

TriggeredPoller (Triggered Poller)

Enabled ● true

Size 0

Progress 0 %

Last Cycle +00000h 00m 00s

Avg Cycle +00000h 00m 00s

Max Cycle +00000h 00m 00s

Min Interval +00000h 02m 00s

In addition to the standard property (Enabled), this component provides these properties.

Property	Value	Description
Size	read-only number	Indicates the number of alerts and Analytic Proxy Ext components, which the poller is managing.
Progress	read-only percentage	Indicates the progress through the current cycle.
Last Cycle	read-only time	Displays the amount of time taken to run the last instance of the cycle.
Avg Cycle	read-only time	Displays the average amount of time required to run an instance of the cycle.
Max cycle	read-only time	Displays the maximum amount of time taken to run an instance of the cycle.
Min Interval	hours minutes seconds	Defines the minimum wait time between processing items in the queue. The default value of zero (0) does not enforce a minimum wait time. Triggered Poller components do not use a Rate property. You should use the Min Interval property to balance CPU load by slowing down the execution of items in the queue. For example, a queue size of 87344 with a Min Interval of 2 seconds creates a cycle that lasts a little over 48 hours.

Actions

Execute executes each item exactly once. This action has no effect if you invoke it during an execution. You can link this action to a trigger schedule as well as to the cycle complete topic of other triggered pollers.

Parent topic: Cyclic poller (BCyclicPoller)

Pollers folder

The **Pollers** folder contains poller objects, which define how frequently the framework executes alert and

Analyitc Proxy Ext components. Double-clicking this component opens the **Poller Manager**.

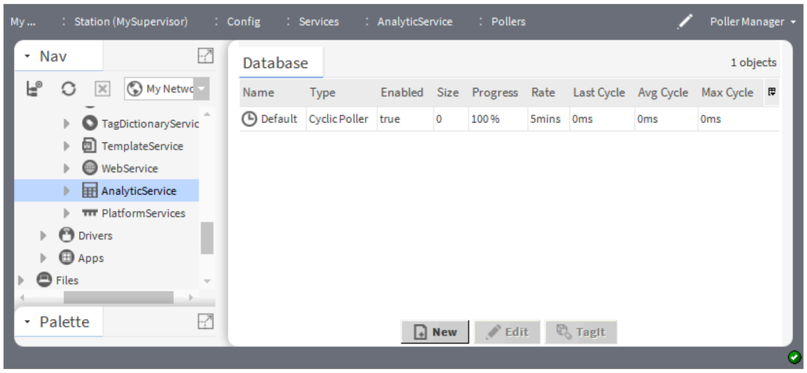
Parent topic: Cyclic poller (BCyclicPoller)

Poller Manager view

This table view lists the pollers that execute alerts and Analytic Proxy Extension components.

Columns

Figure 1. Poller Manager view



Name	Type	Enabled	Size	Progress	Rate	Last Cycle	Avg Cycle	Max Cycle
Default	Cyclic Poller	true	0	100 %	5mins	0ms	0ms	0ms

You access this view by double-clicking the **Pollers** folder under **Config > Services > AnalyticService** in the Nav tree.

Column Name	Description
Name	Reports the name of the poller.
Type	Displays the type of poller.
Enabled	Indicates if the network, device, point or component is active or inactive.
Size	Indicates the number of alerts and Analyitc Proxy Ext components, which the poller is managing.
Progress	Displays the progress of the current cycle.
Rate	Displays the polling frequency by defining the amount of time between cyclic polls.
Last Cycle	Displays the amount of time taken to run the last instance of the cycle.
Avg Cycle	Displays the average amount of time required to run an instance of the cycle.
Max cycle	Displays the maximum amount of time taken to run an instance of the cycle.

Buttons

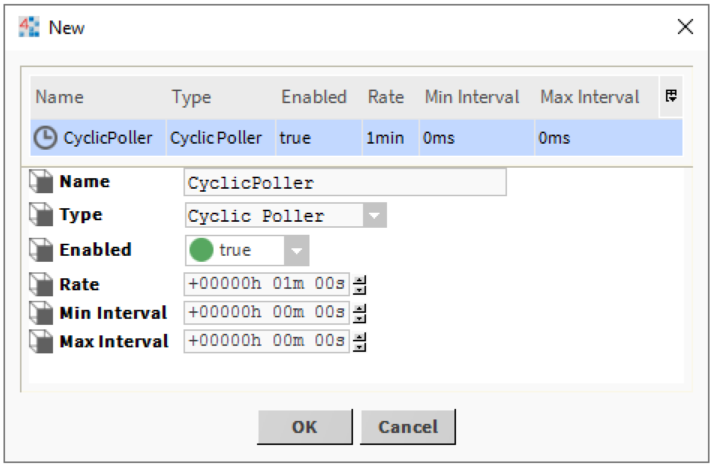
- **New** creates a new cyclic or triggered poller.
- **Edit** opens the poller for updating.
- **Tagit** is disabled for Analytics.

Parent topic: Cyclic poller (BCyclicPoller)

New/Edit poller window

This window creates a new poller.

Figure 1. New Poller window



Name	Type	Enabled	Rate	Min Interval	Max Interval
CyclicPoller	Cyclic Poller	true	1min	0ms	0ms

Name	CyclicPoller
Type	Cyclic Poller
Enabled	true
Rate	+00000h 01m 00s
Min Interval	+00000h 00m 00s
Max Interval	+00000h 00m 00s

This window opens when you click **New** on the **Poller Manager** view.

Property	Value	Description
Name	text	Provides a unique name to refer to the poller.
Type	drop-down list	Selects the type of poller.
Enabled	true or false	Activates (true) and deactivates (false) use of the object (network, device, point, component, table, schedule, descriptor, etc.).
Rate	hours minutes seconds	Establishes polling frequency by defining the amount of time between cyclic polls.
Min Interval	hours minutes seconds	Defines the minimum wait time between processing items in the queue. The default value of zero (0) does not enforce a minimum wait time. Triggered Poller components do not use a Rate property. You should use the Min Interval property to balance CPU load by slowing down the execution of items in the queue. For example, a queue size of 87344 with a Min Interval of 2 seconds creates a cycle that lasts a little over 48 hours.
Max Interval	hours minutes seconds	Defines the maximum wait time between processing items in the queue. The default value of zero (0) does not enforce a

Property	Value	Description
		maximum wait time.

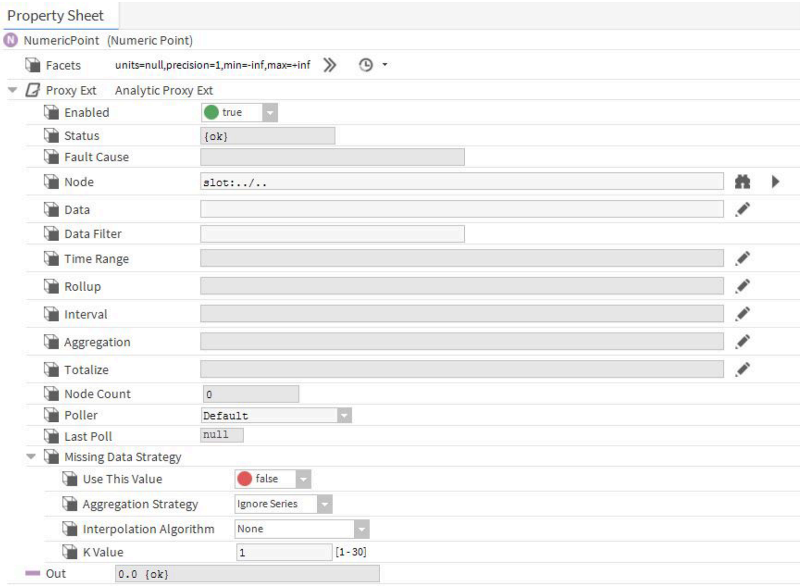
Parent topic: Cyclic poller (BCyclicPoller)

Proxy extension (BAnalyticProxyExt)

This component resolves the configured analytic request and stores the result using its parent control point. On writable points, the proxy extension writes to priority level 16.

There are read-only and writable control points (Numeric, Boolean, Enum and String) with Analytic Proxy Extensions located in the **Points** folder of the analytics palette. The **Property Sheets** for configuring the Analytic Proxy Extensions associated with framework points are what set framework points apart from standard numeric, Boolean, enum and string points. When you drag these points to any location in a station, the Analytic Proxy Extension comes along with the point.

Figure 1. Example of an Analytic Proxy extension



Property Sheet

NumericPoint (Numeric Point)

Facets	units=null,precision=1,min=-inf,max=+inf
Proxy Ext	Analytic Proxy Ext
Enabled	● true
Status	{ok}
Fault Cause	
Node	slot:../..
Data	
Data Filter	
Time Range	
Rollup	
Interval	
Aggregation	
Totalize	
Node Count	0
Poller	Default
Last Poll	null
Missing Data Strategy	
Use This Value	● false
Aggregation Strategy	Ignore Series
Interpolation Algorithm	None
K Value	1 [1-30]
Out	0.0 {ok}

In addition to the standard properties (Enabled, Status, and Fault Cause), this component provides these properties.

Properties	Value	Description
Node	ORD (defaults to a relative ORD slot:../.. that refers to the parent of the control point)	Defines the ORD to the desired slot.
Data	tag or algorithm name	Specifies the an algorithm (or tag) used to retrieve data. The purpose of an alert is to detect a fault condition (Boolean value result). In most cases, you would pick an algorithm that has a Boolean result when configuring this property for an alert. Algorithms with a numeric or enum value greater than zero

Properties	Value	Description
		are considered true and less than zero are considered false. This is how output from one algorithm becomes the input data source to another algorithm. The prefix for algorithms is `alg:`.
Data Filter	optional NEQL predicate	Identifies data sources in the subtree of the request node. When a predicate accepts a node that contains the desired data, the system does not search the node's subtree for additional values (including the root node).
Time Range (optional)	drop-down list	Defines the beginning and end of a period of time during which the analytic request processes historical data.
Interval	optional drop-down list (defaults to the optimal number of records on reports)	Refers to the BInterval component, which the framework uses to identify the time between values in a trend (time series). When specified, a rollup is required, which causes the system to combine all values that fall into a single interval. Options range from None to a Year. Above the drop-down list, the Use This Value check box turns on and off the check box next to Interval in the **Settings** window (you access this window by clicking the Edit button (✎), followed by clicking the Settings button (⚙) on the chart). The availability of this check box provides an easy way for a user to enable and disable the use of intervals in chart calculations.
Rollup	value (when configured in the Data Definition, defaults to First, when configured elsewhere, defaults to the value as defined in the Data Definition)	Configures the default function to apply when the analytic request needs to rollup records from a single data source into less granular records. If rollup is not enabled in the binding/ settings window, the rollup value configured in the Data Definition applies to all chart bindings, reports and tables. And returns the logical "and" of Boolean values. Avg returns the statistical mean, which is determined by calculating the sum of all values and dividing by the number of values. Count returns the total number or quantity of values in a combination. If you request this value on a binding in a PX view, the system counts the number of values based on the properties defined by the data source block and the algorithm's **Property**

Properties	Value	Description
		Sheet. First returns the first value in the combination. Last returns the last value in the combination. Max returns the highest value in the combination. Median returns the value in the middle of a sorted combination—the number that separates the higher half from the lower half. Min returns the lowest value in the combination. Mode returns the statistically most frequently occurring number in the combination. Or returns the logical "or" of Boolean values. Range returns the statistical difference between the largest and smallest values in the combination. Sum adds together all values in the combination resulting in a single value. Std Dev calculates the standard deviation of the values in the combination. Load Factor calculates the average divided by peak (Max) value.
Aggregation	value (defaults to First)	Configures the default function to apply when the analytic request combines values from multiple data sources into a single value. This applies to both value and trend requests. If aggregation is not enabled in the binding/settings window, the aggregation value defined in the Data Definition applies to all chart bindings, reports and tables. And returns the logical "and" of Boolean values. Avg returns the statistical mean, which is determined by calculating the sum of all values and dividing by the number of values. Count returns the total number or quantity of values in a combination. If you request this value on a binding in a PX view, the system counts the number of values based on the properties defined by the data source block and the algorithm's **Property Sheet**. First returns the first value in the combination. Last returns the last value in the

Properties	Value	Description
		combination. Max returns the highest value in the combination. Median returns the value in the middle of a sorted combination—the number that separates the higher half from the lower half. Min returns the lowest value in the combination. Mode returns the statistically most frequently occurring number in the combination. Or returns the logical "or" of Boolean values. Range returns the statistical difference between the largest and smallest values in the combination. Sum adds together all values in the combination resulting in a single value. Load Factor returns the average value divided by peak value. Std Dev returns the standard deviation of the values in the combination.
Poller	name	Identifies which poller manages execution of the proxy extension.
Last Poll	read-only	Indicates the date and time that the system polled this point successfully.
Missing Data Strategy	additional properties	Configures how the framework handles missing data in a series when processing analytic requests and one or more records are missing for an interval. It applies when even a single record for an interval is missing. It does not apply to value requests.

Parent topic: Components, views and windows

Combination (BCombination)

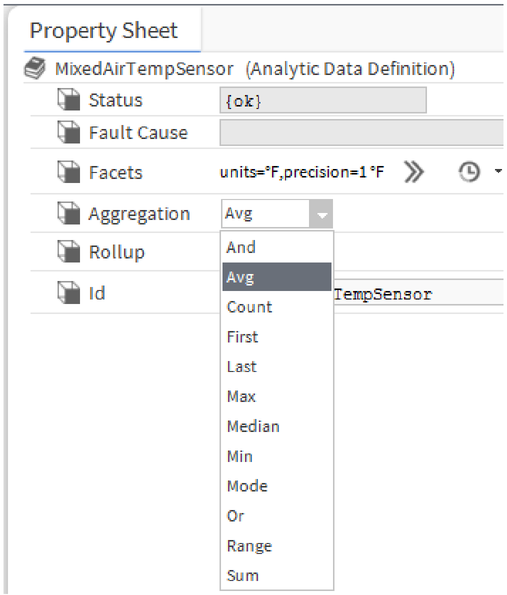
This component works behind the scenes to define how to combine multiple pieces of data. When used for aggregation, it defines how the framework combines data from multiple individual points. When used for rollups, it defines how the framework combines multiple values in a single interval of a trend.

A combination excludes data with an invalid status, unless everything being combined has an invalid status. Invalid status values are: disabled, down, fault, stale and null.

Aggregation and rollup options

This component is responsible for the list of options to configure the **Aggregation** and **Rollup** properties on various components and widgets in the framework.

Figure 1. Aggregation drop-down list on an Analytic Data Definition



MixedAirTempSensor (Analytic Data Definition)

Status	{ok}
Fault Cause	
Facets	units=°F,precision=1 °F
Aggregation	Avg
Rollup	
Id	TempSensor

To view one of these **Property Sheet**s, open the **Analytic Data Manager** (double-click **Config > Services > AnalyticService > Definitions**), right-click a definition, and click **Views > Property Sheet**.

Combination values

Value	Description
And	Logical “and” of Boolean values.
Avg	The statistical mean, which is determined by adding all of the values together and dividing by the number of values.
Count	The number or quantity of values in a combination.
First	The first value in a combination.
Last	The last value in a combination.
Max	The largest value in a numeric combination.
Median	The value in the middle of a sorted numeric combination. It is the number that separates the higher half from the lower half.
Min	The smallest value in a numeric combination.
Mode	Statistically, the most frequently-occurring number in the combination.
Or	Logical “or” of Boolean values.
Range	Statistically, the difference between the largest and smallest values in the combination.
Sum	The result of adding together all values in the combination.
Load factor	The measure of utilization rate.

Value	Description
Standard deviation	The standard deviation calculated for all values.

Parent topic: Components, views and windows

Interval (BInterval)

This component works behind the scenes to define the period of time the framework uses to separate values in a trend (time series). Whenever an interval is specified, a rollup is required to combine all values that fall within a single interval.

Default Interval

When a request does not specify the default interval, the system calculates one based on the time range:

- If the time range is >= one year, the interval is one month.
- If the time range is >= one month, the interval is one day.
- If the time range is >= one week, the interval is six hours.
- If the time range is >= one day, the interval is fifteen minutes.
- If the time range is >= twelve hours, the interval is five minutes.
- If none of the conditions above match, the interval is one minute.

Values

- None, that is, no interval. This effectively defines the interval as a single millisecond.
- Second
- Five Seconds
- Ten Seconds
- Fifteen Seconds
- Thirty Seconds
- Minute
- Five Minutes
- Fifteen Minutes
- Twenty Minutes
- Thirty Minutes
- Hour
- Two hours

- Three Hours
- Four Hours
- Six Hours
- Twelve Hours
- Day
- Week
- Month
- Quarter, that is, three months
- Year

Parent topic: Components, views and windows

Logic blocks

These blocks consist of logic that are used to analyze both real-time values and trend data. You access block properties by double-clicking the block on the **Wire Sheet** or the block name in the Nav tree.

- **Data Source (BDataSourceBlock)**
  This block supplies data to an algorithm. In addition to algorithms, a request for data may also come from a variety of sources, such as graphics bindings and alerts.
- **Result block**
  This block receives the result of the algorithm calculation.
- **Constant blocks**
  These components output a primitive value (Boolean, Enum, Numeric or String). Constant blocks support both value and trend requests. They either use the Time Range and Interval properties from the trend request to generate a constant trend with timestamps in the time range at the configured interval and the configured constant value, or a constant trend based on the value of the Make Trends or Makes Trends property.
- **Filter blocks**
  These components provide practical limits on input and output values.
- **General blocks**
  These objects are grouped as miscellaneous blocks because they do not fit into any of the other object categories.
- **Math blocks**
  These components provide basic mathematical expressions for use in the framework formulas.
- **Switch blocks**
  A switch evaluates a Boolean condition to select one of two possible inputs.

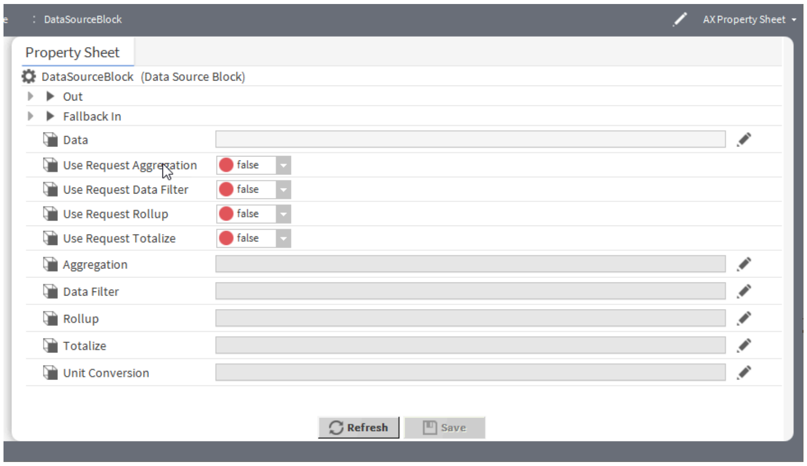
Data Source (BDataSourceBlock)

This block supplies data to an algorithm. In addition to algorithms, a request for data may also come from a variety of sources, such as graphics bindings and alerts.

This block supports both real-time values and trend requests if the underlying data support them.

This block can override some properties of the request. In the process, it generates a modified request for data and supplies the results to the blocks that are linked to its out property.

Figure 1. Data Source Block properties



DataSourceBlock (Data Source Block)

Out	
Fallback In	
Data	
Use Request Aggregation	false
Use Request Data Filter	false
Use Request Rollup	false
Use Request Totalize	false
Aggregation	
Data Filter	
Rollup	
Totalize	
Unit Conversion	

To view these properties, double-click the block on the **Wire Sheet** or the block name in the Nav tree.

Property	Value	Description
Out	read-only value slot	Displays the value or trend resolved for the block.
Fallback In	value slot	Links from the output of other logic blocks or data sources to supply an alternate input source. If the data source is not available for a node, the framework uses this alternate input. The input could be another data source object like this one, or it could be something, such as a constant. The framework does not use the Fallback Input if this data source is available.
Use Request Aggregation	true or false (default)	Determines the aggregation method to use. While the Data Definition associated with each tag defines the default aggregation method, an incoming request from a data source block can override this default aggregation method. true causes the framework to use the function defined by the Aggregation property, if defined in the request. false causes the framework to use the algorithm’s Aggregation property.
Use Request Data Filter	true or false (default)	Determines the data filter method to use. true uses the Data Filter property defined in the request. false uses the Data Filter value as defined in the block. If the Data Filter on the **Data Source Block** is not defined, it has no effect on the result of processing the request.
Use Request Rollup	true or false (default)	Determines the request rollup method to use. The Data Definition associated with each tag defines the default method to roll up data. An incoming request from a data source, rollup, sliding window or today builder block may override this default method. true causes the framework to use the function defined by the Rollup property if defined in the request, else the algorithm’s Rollup property. false causes the framework to use the function defined by the block’s Rollup property if defined, else the Data Definition’s Rollup property if a Data Definition exists for the tag, else the default First function.
Use Request Totalize	true or false (default)	Determines the function to use when calculating a total. true causes the **Data Source Block** to use

Property	Value	Description
		the value of the Totalize property if defined in the request, else it uses Totalize = true. false causes the block to use the value of the block’s Totalize property when the Totalize property is configured. If the Totalize property on the block is not defined, the block uses the default value (true).
Aggregation	value (defaults to First)	Configures the default function used to combine values from multiple data sources into a single value when the block's Use Request Aggregation property is true. This applies to both value and trend requests. If aggregation is not enabled in the binding/settings window, the aggregation value defined in the Data Definition applies to all chart bindings, reports and tables. And returns the logical “and” of Boolean values. Avg returns the statistical mean, which is determined by calculating the sum of all values and dividing by the number of values. Count returns the total number or quantity of values in a combination. If you request this value on a binding in a PX view, the system counts the number of values based on the properties defined by the data source block and the algorithm’s **Property Sheet**. First returns the first value in the combination. Last returns the last value in the combination. Max returns the highest value in the combination. Median returns the value in the middle of a sorted combination—the number that separates the higher half from the lower half. Min returns the lowest value in the combination. Mode returns the statistically most frequently occurring number in the combination. Or returns the logical “or” of Boolean values. Range returns the statistical difference between the largest and smallest values in the combination. Sum adds together all values in the combination resulting in a single value. Load Factor returns the average value

Property	Value	Description
		divided by peak value. Std Dev returns the standard deviation of the values in the combination.
Data Filter	optional NEQL predicate (property) or ORD parameter (dataFilter=predicate)	Identifies data sources in the subtree of the request node. When a predicate accepts a node that contains the desired data, the system does not search the node's subtree for additional values (including the root node).
Rollup	check box (if optional, and) drop-down list or ORD parameter (when configured in the data definition, defaults to First, when configured elsewhere, defaults to the value as defined in the Data Definition); rollup=option	Configures the default rollup function passed through the algorithm when the Data Source block Use Request Rollup property is set to true. If rollup is not enabled in the binding/ settings window, the rollup value configured in the Data Definition applies to all chart bindings, reports and tables. And returns the logical "and" of Boolean values. Avg returns the statistical mean, which is determined by calculating the sum of all values and dividing by the number of values. Count returns the total number or quantity of values in a combination. If you request this value on a binding in a PX view, the system counts the number of values based on the properties defined by the data source block and the algorithm's **Property Sheet**. First returns the first value in the combination. Last returns the last value in the combination. Max returns the highest value in the combination. Median returns the value in the middle of a sorted combination—the number that separates the higher half from the lower half. Min returns the lowest value in the combination. Mode returns the statistically most frequently occurring number in the combination. Or returns the logical "or" of Boolean values. Range returns the statistical difference between the largest and smallest values in the combination. Sum adds together all values in the combination resulting in a single value.

Property	Value	Description
		Std Dev calculates the standard deviation of the values in the combination. Load Factor calculates the average divided by peak (Max) value.
Totalize	true or false (default)	Determines if the totalized or delta values should be calculated.
Unit conversion	optional	If set, the system converts values supplied using the units defined in the data policy of the data source to the units defined by this property.

Processing with Use Request Aggregation

When Use Request Aggrigation is false and Aggregation is not configured, the block uses the aggregation function defined in the applicable Data Definition. If there is no Data Definition, the block uses the default First function. When Use Request Aggrigation is false and Aggregation property is configured, the block uses the Aggregation property value. When Use Request Aggregation is true, the block uses the aggregation function defined in the request (Analytic Proxy Ext, Analytic Binding, etc.), unless the request does not specify the aggregation function, in which case the block uses the Algorithm's Aggregation property value.

The following table may help to determine when each value applies.

Data Definition Aggregation	Data Source Block Use Request Aggregation	Data Source Block Aggregation	Request Aggregation	Algorithm Aggregation	Actual Aggregation Function
Unconfigured	False	Unconfigured	Sum	Max	Defaults to First
Last	False	Unconfigured	Sum	Max	Last
Last	False	**Avg**	Sum	Max	Avg
Last	True	Avg	**Sum**	Max	Sum
Last	True	Avg	Unconfigured	**Max**	Max

Processing with Use Request Rollup

When Use Request Rollup is false and Rollup is not configured, the block uses the rollup function defined in the applicable Data Definition. If there is no Data Definition, the block uses the default First function. When Use Request Rollup is false and Rollup property is configured, the block uses the Rollup property value. When Use Request Rollup is true, the block uses the rollup function defined in the request (Analytic Proxy Ext, Analytic Binding, etc.), unless the request does not specify the rollup function in which case the block uses the Algorithm's Rollup property value.

The following table may help to determine when each value applies.

Data Definition Rollup	Data Source Block Use Request Rollup	Data Source Block Rollup	Request Rollup	Algorithm Rollup	Actual Rollup Function
Unconfigured	False	Unconfigured	Sum	Max	Defaults to First
Last	False	Unconfigured	Sum	Max	Last
Last	False	**Avg**	Sum	Max	Avg
Last	True	Avg	**Sum**	Max	Sum
Last	True	Avg	Unconfigured	**Max**	Max

Parent topic: Logic blocks

Result block

This block receives the result of the algorithm calculation.

Parent topic: Logic blocks

Constant blocks

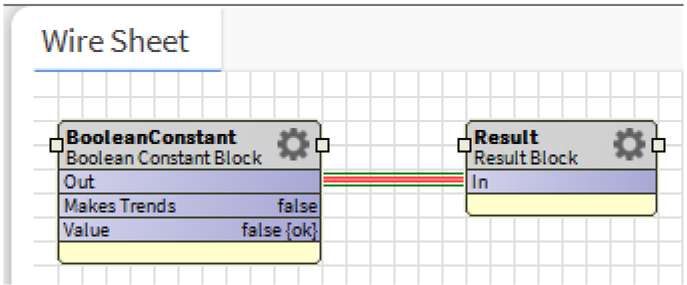
These components output a primitive value (Boolean, Enum, Numeric or String). Constant blocks support both value and trend requests. They either use the Time Range and Interval properties from the trend request to generate a constant trend with timestamps in the time range at the configured interval and the configured constant value, or a constant trend based on the value of the Make Trends or Makes Trends property.

The four constants return data as follows:

- A **Boolean Constant Block** outputs a Boolean value of either True (1) or False (0).
- An **Enum Constant Block** outputs an Enumerated value. The Enum Range property configures either a frozen or user-defined range (ordinal and tag values pairs), which applies to the Value property.
- A **Numeric Constant Block** outputs a numeric value.
- A **String Constant Block** outputs a string value.

Consider an algorithm named BooleanConstant.

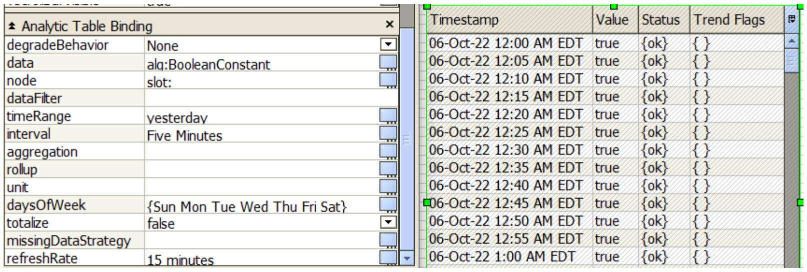
Figure 1. BooleanConstant algorithm



This algorithm has a **Boolean Constant Block** with Makes Trends = false (the default) and Value = true. The Out slot on the **Boolean Constant Block** is linked to a **Result Block**.

Next, consider a Px configured as follows:

Figure 2. Px Analytic Table Binding for the BooleanConstant algorithm



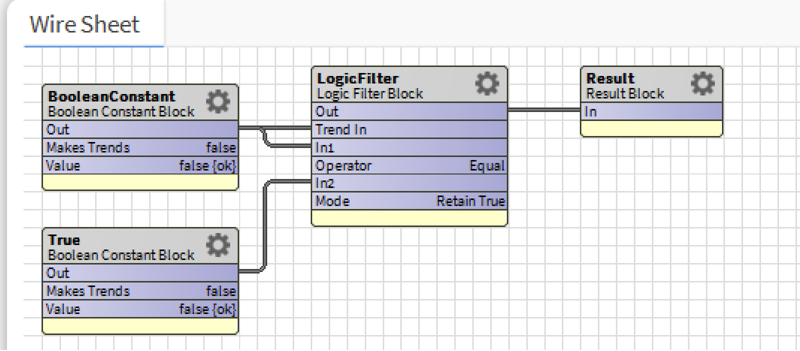
Analytic Table Binding	
degradeBehavior	None
data	alg:BooleanConstant
node	slot:
dataFilter	
timeRange	yesterday
interval	Five Minutes
aggregation	
rollup	
unit	
daysOfWeek	{Sun Mon Tue Wed Thu Fri Sat}
totalize	false
missingDataStrategy	
refreshRate	15 minutes

Timestamp	Value	Status	Trend Flags
06-Oct-22 12:00 AM EDT	true	{ok}	{ }
06-Oct-22 12:05 AM EDT	true	{ok}	{ }
06-Oct-22 12:10 AM EDT	true	{ok}	{ }
06-Oct-22 12:15 AM EDT	true	{ok}	{ }
06-Oct-22 12:20 AM EDT	true	{ok}	{ }
06-Oct-22 12:25 AM EDT	true	{ok}	{ }
06-Oct-22 12:30 AM EDT	true	{ok}	{ }
06-Oct-22 12:35 AM EDT	true	{ok}	{ }
06-Oct-22 12:40 AM EDT	true	{ok}	{ }
06-Oct-22 12:45 AM EDT	true	{ok}	{ }
06-Oct-22 12:50 AM EDT	true	{ok}	{ }
06-Oct-22 12:55 AM EDT	true	{ok}	{ }
06-Oct-22 1:00 AM EDT	true	{ok}	{ }

The px is configured with a Bound Table and an **Analytic Table Binding** configured with data = `alg:BooleanConstant`, timeRange = yesterday, interval = Five Minutes. The **Boolean Constant Block** handles the trend request by generating a constant trend with five-minute-interval records all of which have values set to true for yesterday.

Now, consider another algorithm named BooleanConstantLogicFilter.

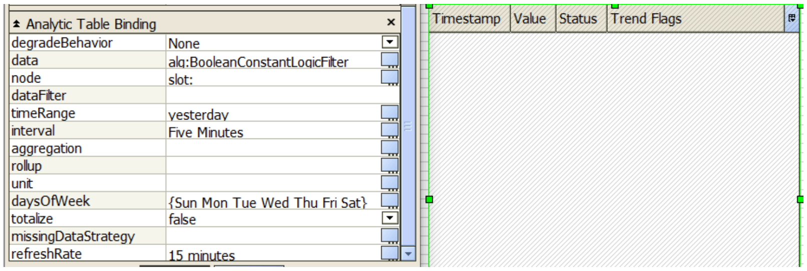
Figure 3. BooleanConstantLogicFilter algorithm



This algorithm includes a **Logic Filter Block** and a second **Boolean Constant Block**.

The **Logic Filter Block** requires the block linked into the Trend In slot to provide a trend, but in this case the Boolean Constant does not generate a trend to pass via the link to the downstream block in the algorithm.

Figure 4. No trend data returned



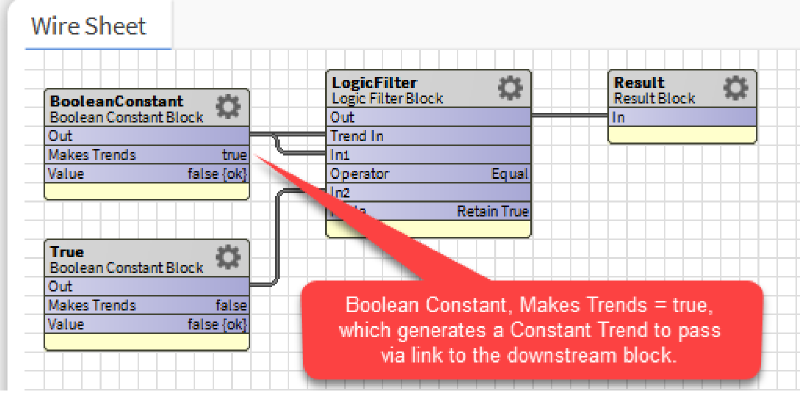
Analytic Table Binding	
degradeBehavior	None
data	alg:BooleanConstantLogicFilter
node	slot:
dataFilter	
timeRange	yesterday
interval	Five Minutes
aggregation	
rollup	
unit	
daysOfWeek	{Sun Mon Tue Wed Thu Fri Sat}
totalize	false
missingDataStrategy	
refreshRate	15 minutes

Timestamp	Value	Status	Trend Flags

The BooleanConstantLogicFilter algorithm returns no data from the trend request.

But when the Makes Trends property on the **Boolean Constant Block** is set to true, the algorithm generates a constant trend.

Figure 5. Makes Trends property set to true on the Boolean Constant Block



BooleanConstant: Makes Trends true; Value false {ok}

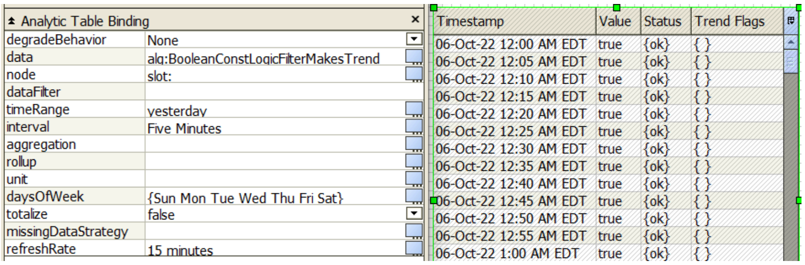
LogicFilter: Operator Equal; Retain True

True: Makes Trends false; Value false {ok}

Boolean Constant, Makes Trends = true, which generates a Constant Trend to pass via link to the downstream block.

The **Boolean Constant Block** passes the trend data to the **Logic Filter Block** Trend In slot causing the BooleanConstLogicFilterMakesTrend algorithm to return a result for yesterday.

Figure 6. Constant trend data returned by the BooleanConstLogicFilterMakesTrend algorithm



Analytic Table Binding

degradeBehavior	None
data	alg:BooleanConstLogicFilterMakesTrend
node	slot:
dataFilter	
timeRange	yesterday
interval	Five Minutes
aggregation	
rollup	
unit	
daysOfWeek	{Sun Mon Tue Wed Thu Fri Sat}
totalize	false
missingDataStrategy	
refreshRate	15 minutes

Timestamp	Value	Status	Trend Flags
06-Oct-22 12:00 AM EDT	true	{ok}	{ }
06-Oct-22 12:05 AM EDT	true	{ok}	{ }
06-Oct-22 12:10 AM EDT	true	{ok}	{ }
06-Oct-22 12:15 AM EDT	true	{ok}	{ }
06-Oct-22 12:20 AM EDT	true	{ok}	{ }
06-Oct-22 12:25 AM EDT	true	{ok}	{ }
06-Oct-22 12:30 AM EDT	true	{ok}	{ }
06-Oct-22 12:35 AM EDT	true	{ok}	{ }
06-Oct-22 12:40 AM EDT	true	{ok}	{ }
06-Oct-22 12:45 AM EDT	true	{ok}	{ }
06-Oct-22 12:50 AM EDT	true	{ok}	{ }
06-Oct-22 12:55 AM EDT	true	{ok}	{ }
06-Oct-22 1:00 AM EDT	true	{ok}	{ }

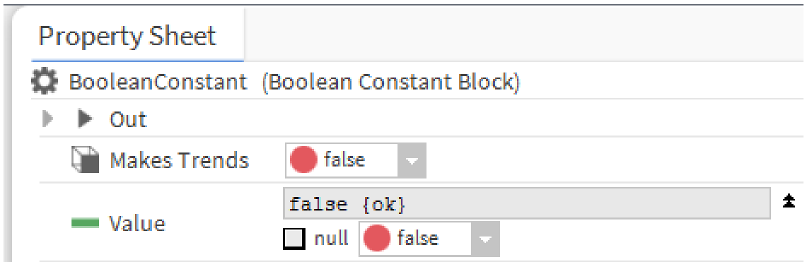
- **Boolean constant (BBooleanConstantBlock)**
  This constant block outputs a Boolean value of True and False.
- **Enum constant (BEnumConstantBlock)**
  This constant block outputs the value of an enum (enumerated type).
- **Enum facets window**
  Enum facets define the possible enumerated states (operating range) of the component. Each state pairs an integer with a value (value-to-text or ordinal-tag). A point’s **Facets** slot is blank until you edit the Enum Range properties and supply the ordinal-tag values.
- **Numeric constant (BNumericConstantBlock)**
  This constant block outputs a numeric value.
- **String constant (BStringConstantBlock)**
  This constant block outputs a text value.

Parent topic: Logic blocks

Boolean constant (BBooleanConstantBlock)

This constant block outputs a Boolean value of True and False.

Figure 1. Boolean Constant properties



Property Sheet

BooleanConstant (Boolean Constant Block)

Out

Makes Trends ● false

Value false {ok}

□ null ● false

To view these properties, double-click the block on the **Wire Sheet** or the block name in the Nav tree.

Property	Value	Description
Out	read-only value slot	Outputs a constant value or constant trend based on the value of the Make Trends or Makes Trends property.
Makes Trends	true or false (default)	Controls the processing of trend requests. false outputs a constant value for both value and trend requests. true outputs a trend with constant values for trend requests.
Value	true or false	Sets the value for the record. This value can be set on the **Property Sheet** or the **Set** action. To view the null check box, click the down arrows to the right of the **Value** property.

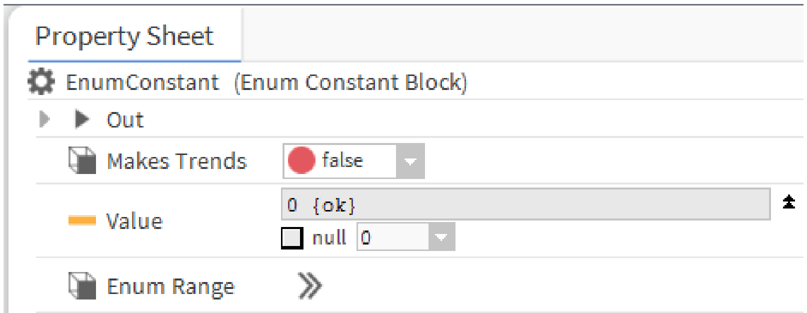
Parent topic: Constant blocks

Enum constant (BEnumConstantBlock)

This constant block outputs the value of an enum (enumerated type).

Property sheet

Figure 1. Enum Constant properties



Property Sheet

EnumConstant (Enum Constant Block)

Out

Makes Trends ● false

Value 0 {ok}

□ null 0

Enum Range

To view these properties, double-click the block on the **Wire Sheet** or the block name in the Nav tree.

Property	Value	Description
Out	read-only value slot	Outputs a constant value or constant trend

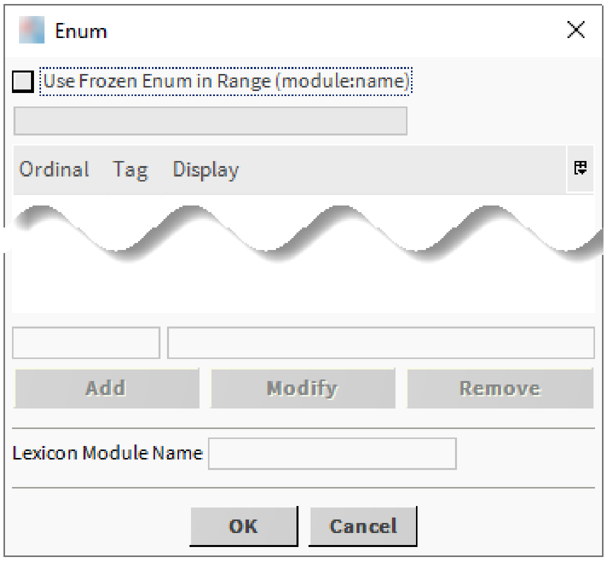
Property	Value	Description
		based on the value of the Make Trends or Makes Trends property.
Makes Trends	true or false (default)	Controls the processing of trend requests. false outputs a constant value for both value and trend requests. true outputs a trend with constant values for trend requests.
Value	text	Sets the value for the record. This value can be set on the **Property Sheet** or the **Set** action. To view the null check box, click the down arrows to the right of the **Value** property.
Enum Range	additional window	Defines enum facets. Click the chevron to the right of this property to open the Enum facets window.

Parent topic: Constant blocks

Enum facets window

Enum facets define the possible enumerated states (operating range) of the component. Each state pairs an integer with a value (value-to-text or ordinal-tag). A point's **Facets** slot is blank until you edit the Enum Range properties and supply the ordinal-tag values.

Figure 1. Enum facets window



Property	Value	Description
Use Frozen Enum in Range (module:name)	module:name	Identifies well-known enumerations as defined in various installed modules. When you enable this option, the system provides a drop-down list from which to select ordinal-tag pairs by module and enumeration type.

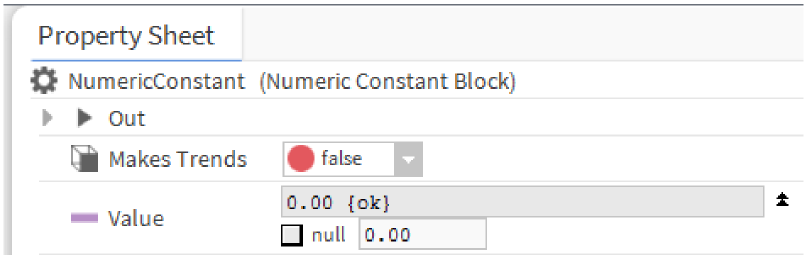
Property	Value	Description
Ordinal column and box	a unique integer	Sets the integer. The box to the left above the **Add** button is where you enter the integer.
Tag column and box	unique text	Sets the tag. The box to the right of the ordinal box is where you enter the tag.
Display column	text	Shows how the enum value appears in the system.
Lexicon Module Name	text	Identifies the name of a lexicon that contains enum facets. When tag strings match lexicon keys, the system displays the lexicon strings (values) for the ordinals instead of the tag text.

Parent topic: Constant blocks

Numeric constant (BNumericConstantBlock)

This constant block outputs a numeric value.

Figure 1. Numeric Constant



Property Sheet

NumericConstant (Numeric Constant Block)

Out

Makes Trends: false

Value: 0.00 {ok}

□ null 0.00

To view these properties, double-click the block on the **Wire Sheet** or the block name in the Nav tree.

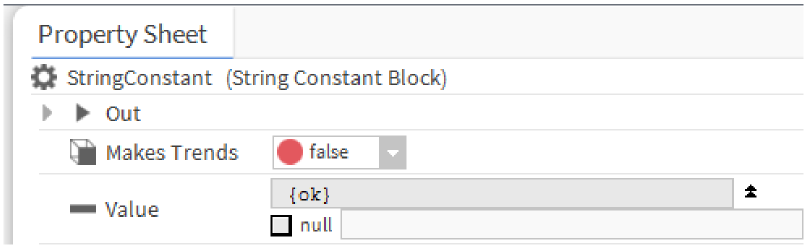
Property	Value	Description
Out	read-only value slot	Outputs a constant value or constant trend based on the value of the Make Trends or Makes Trends property.
Makes Trends	true or false (default)	Controls the processing of trend requests. false outputs a constant value for both value and trend requests. true outputs a trend with constant values for trend requests.
Value	number to two decimal places	Sets the value for the record. This value can be set on the **Property Sheet** or the **Set** action. To view the null check box, click the down arrows to the right of the **Value** property.

Parent topic: Constant blocks

String constant (BStringConstantBlock)

This constant block outputs a text value.

Figure 1. String Constant properties



Property Sheet

StringConstant (String Constant Block)

Out

Makes Trends ● false

Value {ok}

□ null

To view these properties, double-click the block on the **Wire Sheet** or the block name in the Nav tree.

Property	Value	Description
Out	read-only value slot	Outputs a constant value or constant trend based on the value of the Make Trends or Makes Trends property.
Makes Trends	true or false (default)	Controls the processing of trend requests. false outputs a constant value for both value and trend requests. true outputs a trend with constant values for trend requests.
Value	text	Sets the value for the record. This value can be set on the **Property Sheet** or the **Set** action. To view the null check box, click the down arrows to the right of the **Value** property.

Parent topic: Constant blocks

Filter blocks

These components provide practical limits on input and output values.

Histories collect data either on change of value or at some fixed interval. Unless the framework filters values prior to recording history records, it is possible the history records might contain values that are anomalies, such as infinite values, values that are not numbers (NaN, Not a Number), or erroneously high or low values. Analysis of historical data might be affected by those anomalies or there may be a need to simply exclude certain records based on the record’s value or timestamp. Filter blocks in algorithms ensure that the algorithm processes only valid data.

Analytics supports five filter blocks. The following table summarizes when each block applies.

Table 1. Filter block usage

Block	Value Request	Trend Request	Notes
Deadband Filter	No	Yes	

Block	Value Request	Trend Request	Notes
Invalid Value Filter	No	Yes	
Logic Filter	No	Yes	
Range Filter	No	Yes	
Time Filter	Yes	Yes	The timestamp for a value request is the current station time. If the timestamp does not pass the enabled criterion, the block returns a value with a null status.

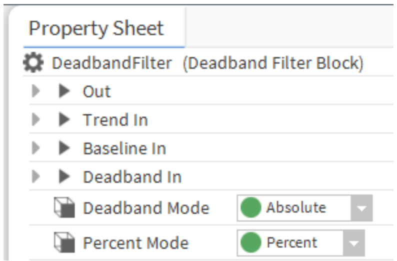
- **Deadband Filter (BDeadbandFilter)**
  This block removes numeric values that are within a configured range. It is similar to the range filter, which removes values outside the configured range, except that a deadband filter uses inverted logic. The block only supports analytic trend requests. Value requests always return a value with a null status.
- **Invalid Value filter (BInvalidValueFilter)**
  This component evaluates an input and outputs only the valid values. A value is considered invalid if it is NaN (Not a Number), it is an infinite number (+inf or -inf), or its status is down, disabled, in fault, null or stale.
- **Logic filter (BLogicFilter)**
  This block evaluates the In1 and In2 slot values using the configured Boolean expression. It only supports analytic trend requests.
- **Range filter (BRangeFilterBlock)**
  This block filters numeric values that are outside the configured range. The Range Filter (filters values outside the range) is similar to the Deadband Filter (filters out values within the range) with inverted logic. The block only supports analytic trend requests. Value requests always return a value with a null status.
- **Time Filter (BTimeFilterBlock)**
  This block filters out values whose timestamp does not match one or more configured criteria. The Time Filter block is primarily intended for analytic trend requests, but also supports analytic value requests. The timestamp for an analytic value request is the current station time. If the analytic value request does not pass the enabled criteria, the request returns a resulting value with a null status.

Parent topic: Logic blocks

Deadband Filter (BDeadbandFilter)

This block removes numeric values that are within a configured range. It is similar to the range filter, which removes values outside the configured range, except that a deadband filter uses inverted logic. The block only supports analytic trend requests. Value requests always return a value with a null status.

Figure 1. Deadband filter properties



Property Sheet

DeadbandFilter (Deadband Filter Block)

- Out
- Trend In
- Baseline In
- Deadband In
- Deadband Mode: Absolute
- Percent Mode: Percent

To view these properties, double-click the block on the **Wire Sheet** block or the block name in the Nav tree.

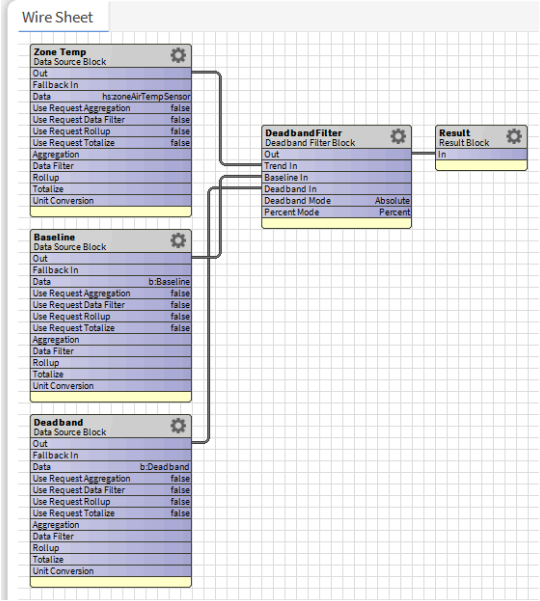
Property	Value	Description
Out	read-only value slot	Outputs the filtered value.
Trend In	required value slot	Links from the output of other logic blocks or data sources to supply trend data to the current logic block.
Baseline In	required numeric value slot	Links from the output of other logic blocks or data sources to supply a trend. This trend provides a baseline value upon which to apply a deadband.
Deadband In	required numeric value slot	Links from the output of other logic blocks or data sources to supply one or more input values that represent a dead or neutral zone around a baseline.
Deadband Mode	drop-down list	Determines what the engine uses to establish the deadband. Absolute, configures a specific value to use as the deadband. Percent, configures the deadband as a percentage.
Percent Mode	drop-down list	Determines the form in which the constant appears if **Deadband Mode** is a percentage. Percent treats the value as a percent (0-100). For example, value of 30% would be represented as 30. Decimal treats the value as a decimal number (0-1). For example, a value of 30% would be represented as .3.

How the framework configures a deadband range

The Baseline In, Deadband In, Deadband Mode and Percent Mode properties on a **Deadband Filter Block** determine the range of values removed. The Deadband Mode property determines if the Deadband In value is an absolute value or a percentage. The Percent Mode property determines if the percentage value is expressed as a decimal (0-1) or as a percentage (0-100).

Consider three numeric points with numeric interval histories for Zone Temp tagged with `hs:zoneAirTempSensor` and `a:a` marker tags, a baseline value of 72 (`b:Baseline` and `a:a` marker tags) and a deadband value of 5 (`b:Deadband` and `a:a` marker tags). The algorithm looks like this:

Figure 2. Deadband algorithm



Wire Sheet

Zone Temp
Data Source Block
Out
Fallback In
Data hs:zoneAirTempSensor
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Use Request Totalize false
Aggregation
Data Filter
Rollup
Totalize
Unit Conversion

Baseline
Data Source Block
Out
Fallback In
Data b:Baseline
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Use Request Totalize false
Aggregation
Data Filter
Rollup
Totalize
Unit Conversion

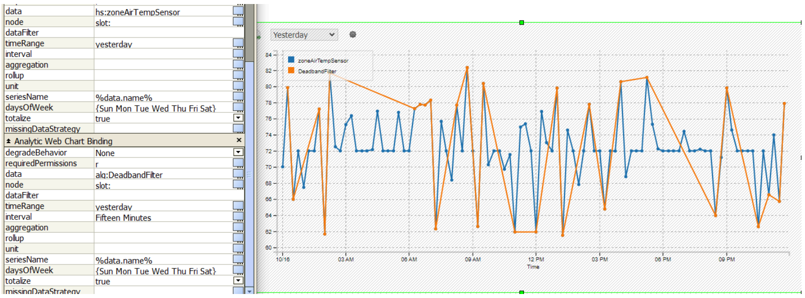
Deadband
Data Source Block
Out
Fallback In
Data b:Deadband
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Use Request Totalize false
Aggregation
Data Filter
Rollup
Totalize
Unit Conversion

DeadbandFilter
Deadband Filter Block
Out
Trend In
Baseline In
Deadband In
Deadband Mode Absolute
Percent Mode Percent

Result
Result Block
In

A Px with an Analytic Web Chart displays the unfiltered hs:zoneAirTempSensor values compared to the alg:DeadbandFilter, which removes any records where the value is within the calculated 67-77 range.

Figure 3. Chart comparing air temp values compared to the deadband filter



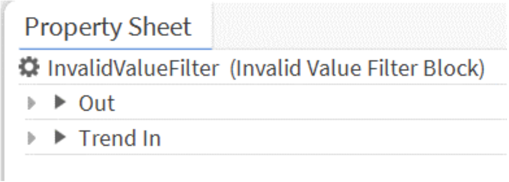
data	hs:zoneAirTempSensor
node	slot:
dataFilter	
timeRange	yesterday
interval	
aggregation	
rollup	
unit	
seriesName	%data.name%
daysOfWeek	{Sun Mon Tue Wed Thu Fri Sat}
totalize	true
missingDataStrategy	
Analytic Web Chart Binding	
degradeBehavior	None
requiredPermissions	r
data	alg:DeadbandFilter
node	slot:
dataFilter	
timeRange	yesterday
interval	Fifteen Minutes
aggregation	
rollup	
unit	
seriesName	%data.name%
daysOfWeek	{Sun Mon Tue Wed Thu Fri Sat}
totalize	true
missingDataStrategy	

Parent topic: Filter blocks

Invalid Value filter (BInvalidValueFilter)

This component evaluates an input and outputs only the valid values. A value is considered invalid if it is NaN (Not a Number), it is an infinite number (+inf or -inf), or its status is down, disabled, in fault, null or stale.

Figure 1. Invalid Value Filter properties



Property Sheet

InvalidValueFilter (Invalid Value Filter Block)

▶ Out

▶ Trend In

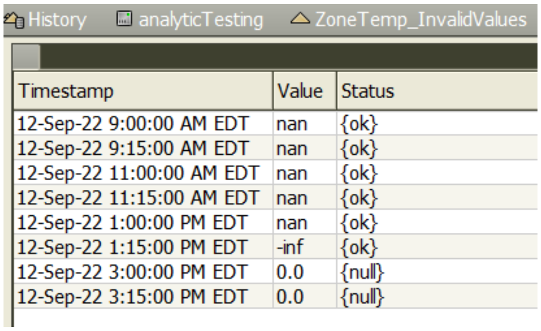
To view these properties, double-click the block on the **Wire Sheet** block or the block name in the Nav tree.

Property	Value	Description
Out	read-only value slot	Outputs a filtered value.
Trend In	required value slot	Links from the output of other logic blocks or data sources to supply trend data to the current logic block.

Examples

Consider a numeric history with records that contain an invalid value (NaN, +inf or -inf) or null status.

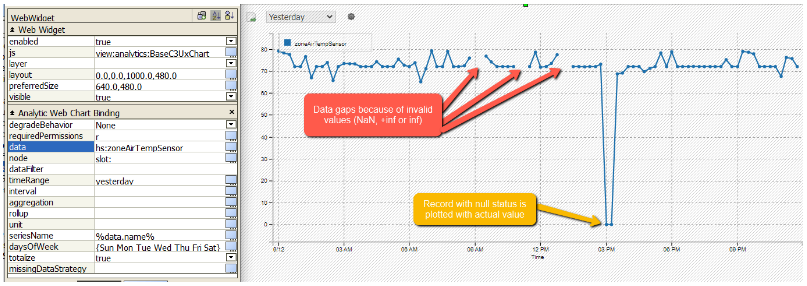
Figure 2. History table that contains invalid values



Timestamp	Value	Status
12-Sep-22 9:00:00 AM EDT	nan	{ok}
12-Sep-22 9:15:00 AM EDT	nan	{ok}
12-Sep-22 11:00:00 AM EDT	nan	{ok}
12-Sep-22 11:15:00 AM EDT	nan	{ok}
12-Sep-22 1:00:00 PM EDT	nan	{ok}
12-Sep-22 1:15:00 PM EDT	-inf	{ok}
12-Sep-22 3:00:00 PM EDT	0.0	{null}
12-Sep-22 3:15:00 PM EDT	0.0	{null}

The Analytic Web Chart does not plot record values, which are NaN or infinite (+inf or -inf), but does plot record values with the other status settings, including those with a null status.

Figure 3. Invalid values in a chart



WebWidget

Web Widget

enabled	true
js	view:analytics:BaseC3UxChart
layer	
layout	0.0,0.0,1000.0,480.0
preferredSize	640.0,480.0
visible	true

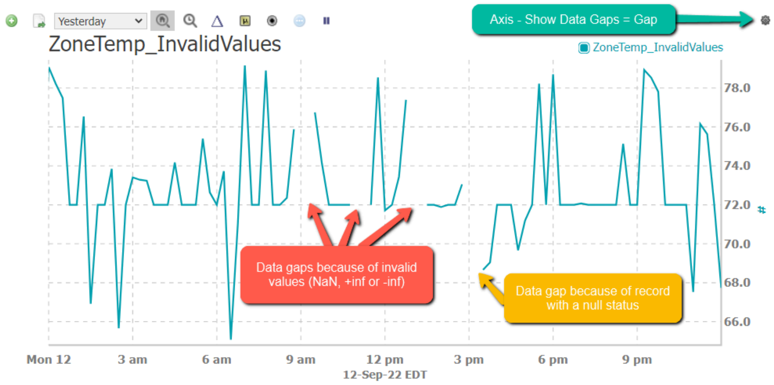
Analytic Web Chart Binding

degradeBehavior	None
requiredPermissions	r
data	hs:zoneAirTempSensor
node	slot:
dataFilter	
timeRange	yesterday
interval	
aggregation	
rollup	
unit	
seriesName	%data.name%
daysOfWeek	{Sun Mon Tue Wed Thu Fri Sat}
totalize	true
missingDataStrategy	

Yesterday
zoneAirTempSensor
80
70
60
50
40
30
20
10
0
9/12
03 AM
06 AM
09 AM
12 PM
03 PM
06 PM
09 PM
Time
Data gaps because of invalid values (NaN, +inf or inf)
Record with null status is plotted with actual value

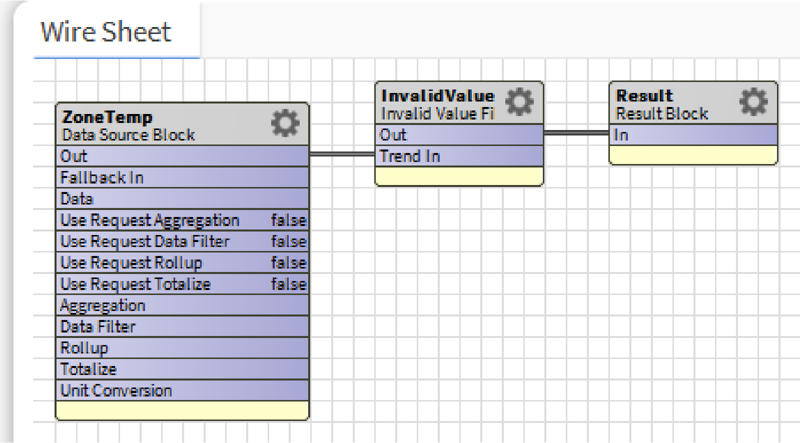
This is similar to how the standard Web Chart functions when the **Axis > Show Data Gaps** property equals gap; however, the standard Web Chart displays a gap for any record with a null status.

Figure 4. Gaps shown for any record with a null status



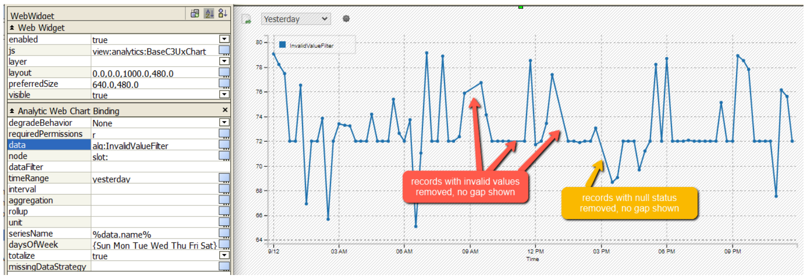
An algorithm can use an **Invalid Value Filter Block** to filter out trend records with invalid values or statuses.

Figure 5. An algorithm with an Invalid Value Filter Block



The Analytic Web Chart using the `alg:InvalidValueFilter` algorithm does not display gaps at the timestamps where records have been filtered out.

Figure 6. A chart that no longer shows invalid value gaps


WebWidget

Web Widget

enabled	true
js	view:analytics:BaseC3UxChart
layer	
layout	0.0,0.0,1000.0,480.0
preferredSize	640.0,480.0
visible	true

Analytic Web Chart Binding

degradeBehavior	None
requiredPermissions	r
data	alg:InvalidValueFilter
node	slot:
dataFilter	
timeRange	yesterday
interval	
aggregation	
rollup	
unit	
seriesName	%data.name%
daysOfWeek	{Sun Mon Tue Wed Thu Fri Sat}
totalize	true
missingDataStrategy	

Yesterday
InvalidValueFilter
80
78
76
74
72
70
68
66
64
records with invalid values removed, no gap shown
records with null status removed, no gap shown
9/12
03 AM
06 AM
09 AM
12 PM
03 PM
06 PM
09 PM
Time

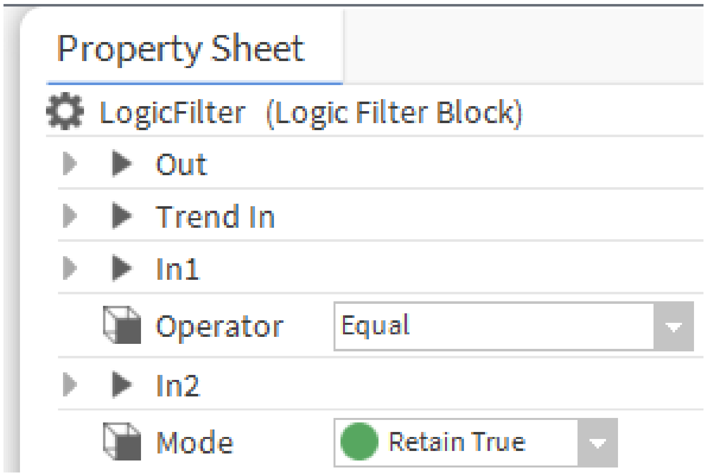
Parent topic: Filter blocks

Logic filter (BLogicFilter)

This block evaluates the In1 and In2 slot values using the configured Boolean expression. It only supports analytic trend requests.

The Mode property configures which Boolean condition (true or false) causes the trend input value to be filtered out. When this property is set to Retain True and the Boolean expression evaluates to false, the block filters the trend value out of the result. Trend In is not required to be one of the inputs used for the Boolean expression. The status of the In1 and In2 slots are merged with the status of the trend input record and applied to the resulting value.

Figure 1. Logic Filter properties



Property Sheet

LogicFilter (Logic Filter Block)

- Out
- Trend In
- In1
- Operator: Equal
- In2
- Mode: ● Retain True

To view these properties, double-click the block on the **Wire Sheet** block or the block name in the Nav tree.

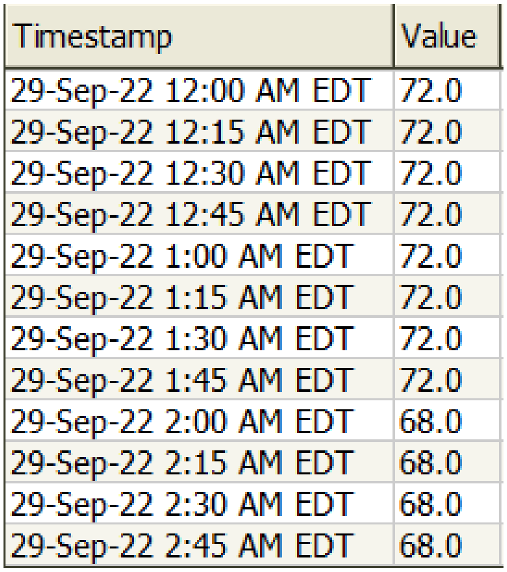
Property	Value	Description
Out	read-only value slot	Outputs a filtered value from the Trend In.
Trend In	optional value slot	Links from the output of other logic blocks or data sources to supply trend data to the current logic block. This property is not required to be one of the inputs used for the Boolean expression. The framework merges the status of the In1 and In2 slots with the status of the Trend In record and applies all three to the resulting value.
In 1	operand	Defines the first operand used in the expression. Links from the output of other logic blocks or data sources to supply one or more input values to another logic block.
Operator	operator	Defines the function to perform.
In 2	operand	Defines the second operand used in the expression. Links from the output of other logic blocks or data sources to supply one or more input values to a logic block.
Mode	Retain False (default) or Retain True	Configures which Boolean condition (true or false) filters the trend input value. Retain True filters the trend value out of the result when the Boolean expression evaluates to false.

Property	Value	Description
		Retain False does the opposite. It filters the trend value out of the result when the Boolean expression evaluates to true.

Example

Consider a ZoneTemp with `hs:zoneAirTempSensor` and `a:a` marker tags and with history logged at 15–minute intervals.

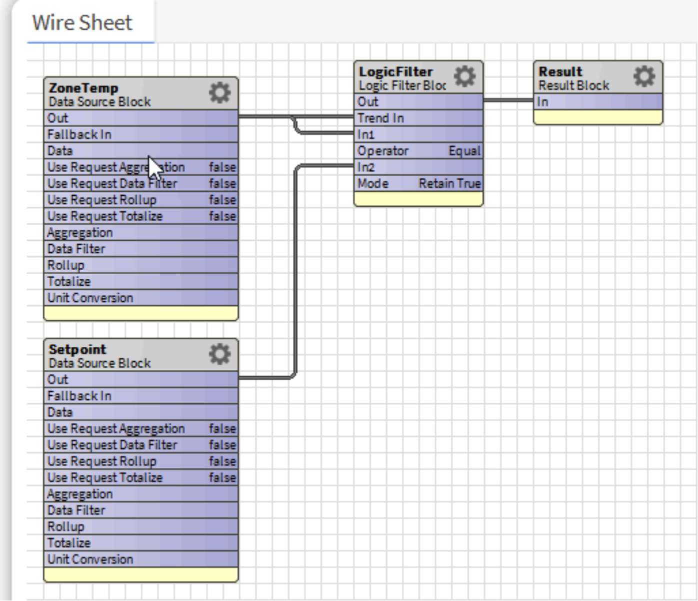
Figure 2. History table with logged values



Timestamp	Value
29-Sep-22 12:00 AM EDT	72.0
29-Sep-22 12:15 AM EDT	72.0
29-Sep-22 12:30 AM EDT	72.0
29-Sep-22 12:45 AM EDT	72.0
29-Sep-22 1:00 AM EDT	72.0
29-Sep-22 1:15 AM EDT	72.0
29-Sep-22 1:30 AM EDT	72.0
29-Sep-22 1:45 AM EDT	72.0
29-Sep-22 2:00 AM EDT	68.0
29-Sep-22 2:15 AM EDT	68.0
29-Sep-22 2:30 AM EDT	68.0
29-Sep-22 2:45 AM EDT	68.0

An algorithm with a **Logic Filter Block** evaluates whether the ZoneTemp trend value is greater than a Setpoint (`hs:sp`) value, and filters records from the result when the Boolean condition is false.

Figure 3. Algorithm with a Logic Filter



Wire Sheet

ZoneTemp
Data Source Block

Out	
Fallback In	
Data	
Use Request Aggregation	false
Use Request Data Filter	false
Use Request Rollup	false
Use Request Totalize	false
Aggregation	
Data Filter	
Rollup	
Totalize	
Unit Conversion	

Setpoint
Data Source Block

Out	
Fallback In	
Data	
Use Request Aggregation	false
Use Request Data Filter	false
Use Request Rollup	false
Use Request Totalize	false
Aggregation	
Data Filter	
Rollup	
Totalize	
Unit Conversion	

LogicFilter
Logic Filter Bloc

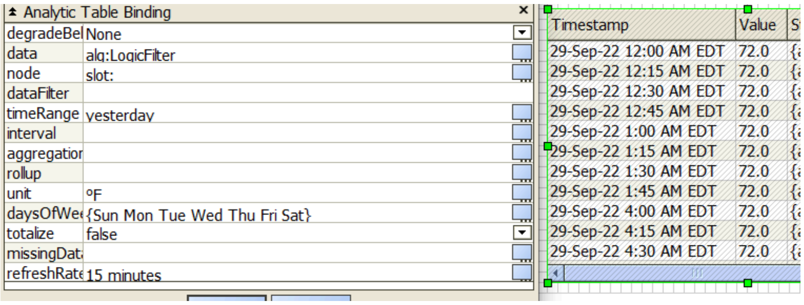
Out	
Trend In	
In1	
Operator	Equal
In2	
Mode	Retain True

Result
Result Block

In	

The resulting table shows the valid record values.

Figure 4. Record values after the Logic Filter removed records



Analytic Table Binding

degradeBel	None
data	alg:LogicFilter
node	slot:
dataFilter	
timeRange	yesterday
interval	
aggregatior	
rollup	
unit	°F
daysOfWe	{Sun Mon Tue Wed Thu Fri Sat}
totalize	false
missingDat	
refreshRat	15 minutes

Timestamp	Value	S
29-Sep-22 12:00 AM EDT	72.0	{
29-Sep-22 12:15 AM EDT	72.0	{
29-Sep-22 12:30 AM EDT	72.0	{
29-Sep-22 12:45 AM EDT	72.0	{
29-Sep-22 1:00 AM EDT	72.0	{
29-Sep-22 1:15 AM EDT	72.0	{
29-Sep-22 1:30 AM EDT	72.0	{
29-Sep-22 1:45 AM EDT	72.0	{
29-Sep-22 4:00 AM EDT	72.0	{
29-Sep-22 4:15 AM EDT	72.0	{
29-Sep-22 4:30 AM EDT	72.0	{

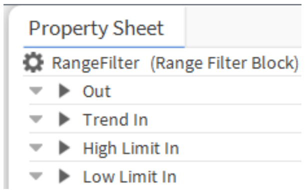
Parent topic: Filter blocks

Range filter (BRangeFilterBlock)

This block filters numeric values that are outside the configured range. The Range Filter (filters values outside the range) is similar to the Deadband Filter (filters out values within the range) with inverted logic. The block only supports analytic trend requests. Value requests always return a value with a null status.

Values are considered in range if they are greater than or equal to the low limit and less than or equal to the high limit. Only trend requests are supported, value requests return null. The status of the resulting value is the combination of the in, high limit, and the low limit values.

Figure 1. Range Filter properties



Property Sheet

RangeFilter (Range Filter Block)

- Out
- Trend In
- High Limit In
- Low Limit In

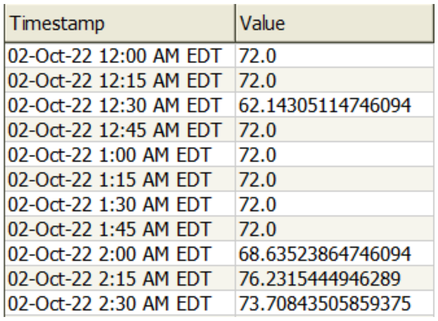
To view these properties, double-click the block on the **Wire Sheet** block or the block name in the Nav tree.

Property	Value	Description
Out	read-only value slot	Outputs a filtered value.
Trend In	value slot	Links from the output of other logic blocks or data sources to supply trend data to the current logic block.
High Limit In	numeric value slot	Links from the output of other logic blocks or data sources to supply one or more values that define the maximum valid value. The framework merges status of the High Limit In and Low Limit In slots with the status of the Trend In record and applies all three to the resulting value.
Low Limit In	numeric value slot	Links from the output of other logic blocks or data sources to supply one or more input values to define a minimum valid value.

Examples

Consider a ZoneTemp with `hs:zoneAirTempSensor` and `a:a` marker tags, and history logged at 15–minute intervals.

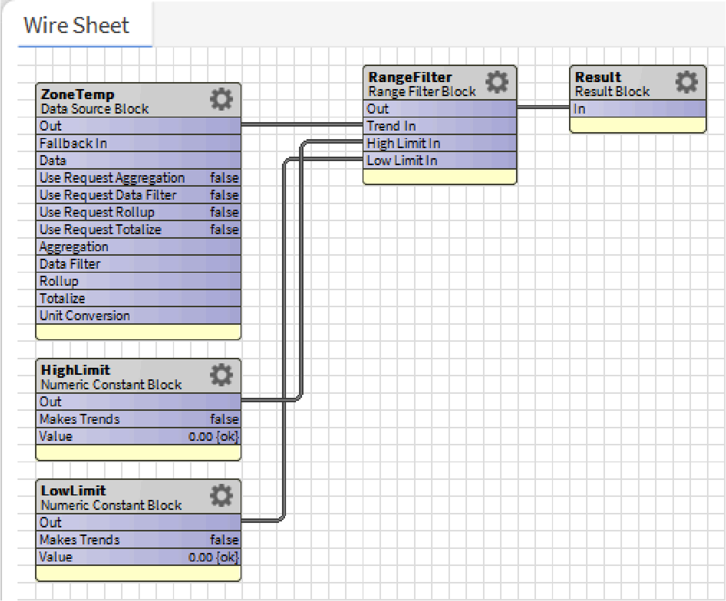
Figure 2. History table includes out of range values



Timestamp	Value
02-Oct-22 12:00 AM EDT	72.0
02-Oct-22 12:15 AM EDT	72.0
02-Oct-22 12:30 AM EDT	62.14305114746094
02-Oct-22 12:45 AM EDT	72.0
02-Oct-22 1:00 AM EDT	72.0
02-Oct-22 1:15 AM EDT	72.0
02-Oct-22 1:30 AM EDT	72.0
02-Oct-22 1:45 AM EDT	72.0
02-Oct-22 2:00 AM EDT	68.63523864746094
02-Oct-22 2:15 AM EDT	76.2315444946289
02-Oct-22 2:30 AM EDT	73.70843505859375

An algorithm with a **Range Filter Block** evaluates whether the ZoneTemp trend value is greater than the High Limit value or less than the Low Limit value, and filters those records from the result.

Figure 3. Algorithm with a Range Filter


Wire Sheet

ZoneTemp
Data Source Block
Out
Fallback In
Data
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Use Request Totalize false
Aggregation
Data Filter
Rollup
Totalize
Unit Conversion

HighLimit
Numeric Constant Block
Out
Makes Trends false
Value 0.00 {ok}

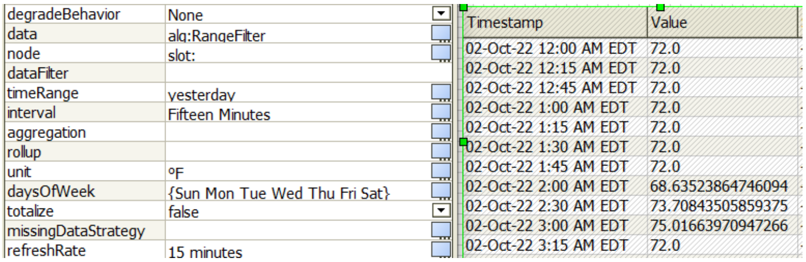
LowLimit
Numeric Constant Block
Out
Makes Trends false
Value 0.00 {ok}

RangeFilter
Range Filter Block
Out
Trend In
High Limit In
Low Limit In

Result
Result Block
In

The resulting table shows only the record values that are within the range.

Figure 4. Record values after the Logic Filter removed records



degradeBehavior	None
data	alg:RangeFilter
node	slot:
dataFilter	
timeRange	yesterday
interval	Fifteen Minutes
aggregation	
rollup	
unit	°F
daysOfWeek	{Sun Mon Tue Wed Thu Fri Sat}
totalize	false
missingDataStrategy	
refreshRate	15 minutes

Timestamp	Value
02-Oct-22 12:00 AM EDT	72.0
02-Oct-22 12:15 AM EDT	72.0
02-Oct-22 12:45 AM EDT	72.0
02-Oct-22 1:00 AM EDT	72.0
02-Oct-22 1:15 AM EDT	72.0
02-Oct-22 1:30 AM EDT	72.0
02-Oct-22 1:45 AM EDT	72.0
02-Oct-22 2:00 AM EDT	68.63523864746094
02-Oct-22 2:30 AM EDT	73.70843505859375
02-Oct-22 3:00 AM EDT	75.01663970947266
02-Oct-22 3:15 AM EDT	72.0

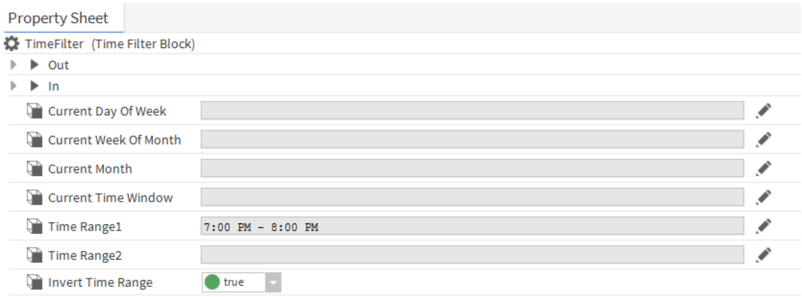
Parent topic: Filter blocks

Time Filter (BTimeFilterBlock)

This block filters out values whose timestamp does not match one or more configured criteria. The Time Filter block is primarily intended for analytic trend requests, but also supports analytic value requests. The timestamp for an analytic value request is the current station time. If the analytic value request does not pass the enabled criteria, the request returns a resulting value with a null status.

In general, the timestamp must pass all of the enabled criteria for a value to be returned. If both Time Range 1 and Time Range 2 are configured, the timestamp must pass only one of the time range criteria. This is an ‘or’ gate that combines Time Range 1 and Time Range 2 into a single criterion with all of the other enabled criteria.

Figure 1. Time Filter properties



Property Sheet

TimeFilter (Time Filter Block)

- Out
- In

Current Day Of Week	
Current Week Of Month	
Current Month	
Current Time Window	
Time Range1	7:00 PM - 8:00 PM
Time Range2	
Invert Time Range	● true

To view these properties, double-click on the **Wire Sheet** block or the block name in the Nav tree.

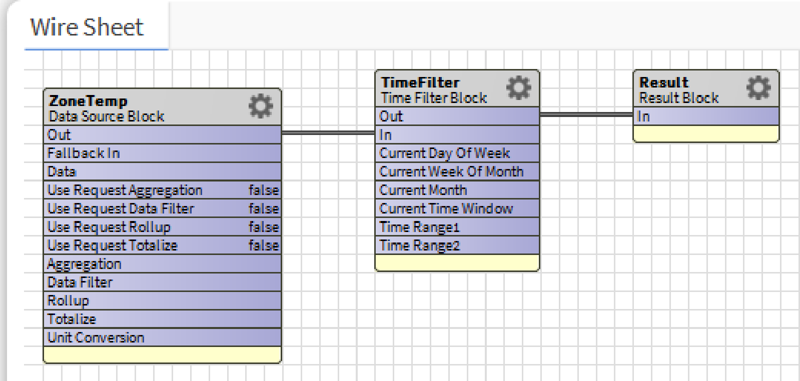
Property	Value	Description
Out	read-only value slot	Outputs a filtered value.
In	read-only value slot	Links from the output of other logic blocks or data sources to supply trend data to the current logic block.
Current Day of Week	true or false (default)	Determines if the timestamp must match the real-time day of the week. true requires the day of week from the timestamp being evaluated to match the day of the week for the station’s current time. For example, if true, if the station’s current time day of week is Monday, and if the trend request time range is last month, the result will contain data from all Mondays in the previous month. false permits the resulting value to include any day of the week.
Current Week of Month	true or false (default)	Determines if the timestamp must match the real-time week of the month. true requires the week of month from the timestamp being evaluated to match the week of month from the station’s current time. For example, if true, the station’s current time week of month is 2nd week, and the trend request time range is previous three months, the result will only contain data from the 2nd week of the three previous months. false permits the resulting value to contain data from any week of the month.
Current Month	true or false (default)	Determines if the timestamp must match the real-time month. true requires the month of year from the timestamp being evaluated to match the month of year for the station’s current time. If true, the station’s current month of

Property	Value	Description
		the year is September 2025, and the trend request time range is previous two years, the result will only contain data from September 2023 and September 2024. false permits the resulting value to contain data from any month.
Current Time Window	hours minutes seconds	Sets a time of day for evaluation. If configured, the timestamp being evaluated must fall within the configured time range of the current day. For example, if Current Time Window is 2 hours, the station's current time is 3:31 PM, and the trend request time range is today, the result will only contain records whose timestamp is between 1:31 PM and 3:31 PM of today.
Time Range 1 and Time Range 2	Start Time and End Time	Defines a range for time of day. If configured, the timestamp being evaluated must fall within the configured time range. If you configure both Time Range 1 and Time Range 2 the timestamp being evaluated must fall within one of the time range criteria and all of the other enabled criteria. An 'or' gate can combine Time Range 1 and Time Range 2 into a single criterion. For example, if Time Range 1 is 6:00 AM - 12:00 PM and Time Range 2 is 5:00 PM - 9:00 PM, and the trend request time range is last week, the result will contain records from any day during last week where the timestamp is in the range of 6:00 AM - 12:00 PM or 5:00 PM - 9:00 PM.
Invert Time Range	true or false (default)	Excludes from the output the time range given in Time Range 1 and Time Range 2. Note: If the time range spans to the next day, configure End Time for Time Range 1 as 11:59 PM and not 12 AM.

Example

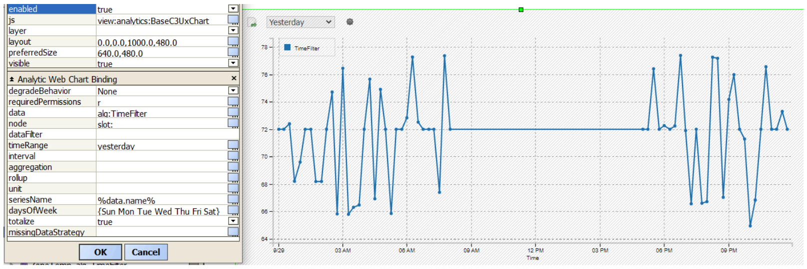
Consider an algorithm with a **Time Filter Block** configured with Time Range 1 = 8:00 AM - 5:00 PM and Invert Time Range = true. The Invert Time Range = true causes the algorithm to filter out any values whose timestamp is in the range of 8:00 AM - 5:00 PM for any day.

Figure 2. Time range algorithm



This type of algorithm is useful for analyzing data during an unoccupied or off peak time period. It creates a chart like the following.

Figure 3. Chart filtered by time range



enabled	true
js	view:analytics:BaseC3UxChart
layer	
layout	0.0,0.0,1000.0,480.0
preferredSize	640.0,480.0
visible	true

Analytic Web Chart Binding

degradeBehavior	None
requiredPermissions	r
data	alg:TimeFilter
node	slot:
dataFilter	
timeRange	yesterday
interval	
aggregation	
rollup	
unit	
seriesName	%data.name%
daysOfWeek	{Sun Mon Tue Wed Thu Fri Sat}
totalize	true
missingDataStrategy	

Parent topic: Filter blocks

General blocks

These objects are grouped as miscellaneous blocks because they do not fit into any of the other object categories.

The following table identifies if the block relates to a value request or a trend request.

Block	Value Request	Trend Request	Notes
Consumption To Demand	Yes	Yes	A value request processes a trend request for the previous 1 hour to calculate the time difference from the last trend record.
Demand To Delta Consumption	Yes	Yes	A value request processes a trend request for the previous 1 hour to calculate the time difference from the last trend record.
Intersection	Yes	Yes	
Interval Count	No	Yes	
Logic Folder	Yes	Yes	This container organizes a subset of logic and is linked using its Out slot to other blocks in the parent algorithm. It handles both value and trend requests.
Not	Yes	Yes	
Psychrometric	Yes	Yes	

Block	Value Request	Trend Request	Notes
Request Overrides	Yes	Yes	
Rollup	Yes	Yes	A value request processes a trend request using the Rollup block’s Time Range property and returns a single value by applying the value request’s Rollup property if Use Request Rollup is true, otherwise by applying the Rollup block’s Rollup property.
Runtime	Yes	Yes	A trend request resets an accumulated runtime value to zero when the next record is inactive. This prevents rolling up runtime values, such as daily or weekly totals. A value request processes a trend request using the Runtime block's Time Range property and returns a single value, which is the accumulated active time.
Sliding Window	Yes	Yes	A value request processes a trend request using the Sliding Window block's Window property to calculate a time range ending with the current time.
Timer Range Offset	Yes	Yes	A value request processes a trend request with a configured time range offset and returns a single value.
Timestamp Offset	Yes	Yes	
Today Builder	Yes	Yes	A value request processes a trend request with the time range configured on the block and applies the rollup function on the block to return a single value.
Value Duration	Yes	Yes	A value request processes a trend request using the search limit value for the time range. A trend request ignores the time range in the request and uses the search limit value for the time range.
Value Map	Yes	Yes	

- **Debug (BDebugBlock)**
  This block provides tools to debug algorithms. To provide debug information for value and trend requests, you link this block in line between other blocks in an algorithm.
- **Consumption To Demand (BConsumptionToDemandBlock)**
  This block converts consumption values into demand values for both value and trend requests. This block supports both trend and value requests.
- **Demand to Delta Consumption (BDemandToConsumptionBlock)**
  This block converts demand values into consumption values for both Analytic Value and Analytic Trend Requests.
- **Intersection (BIntersectionBlock)**
  This component combines two trends. It includes only rows with identical timestamps from both inputs. This object was designed for the scenario where the timestamps from a filtered trend are needed to extract values from another trend. The block supports both value and trend requests, although the intent of this block is for processing trends.
- **Interval count (BIntervalCountBlock)**
  This block does not have an input slot. It automatically detects the time range and interval from the analytic trend request and outputs the number of intervals for the given time range. The purpose of this object is to distribute a value, such as cost among all rows in a trend being displayed in a chart or table.
- **Logic folder (LogicFolder)**
  This folder block organizes other analytic blocks and logic in a sub folder of an algorithm. It is similar to an **Algorithm** in that both have a default **Result Block**, but the **Result Block** is linked to the Out slot of

the **Logic Folder**. If an algorithm contains a lot of blocks in its **Wire Sheet**, it may be useful to organize subsets of the blocks based on functionality.

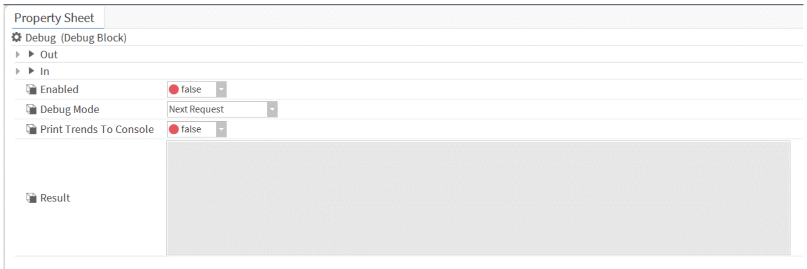
- **Not (BNotBlock)**
  This component outputs the Boolean opposite of the In property. It contains no properties to be configured. It supports both trend and value requests. The **Not Block** may be useful for determining off peak times for energy reporting or for scheduling related Analytics and fault detection.
- **Psychrometric (BPsychrometricBlock)**
  This block calculates various properties of moist air given the temperature (Fahrenheit or Celsius) and humidity (% relative humidity) inputs. It supports both trend and value requests. The Mode property determines the calculation to perform: Dew Point Temperature, Enthalpy (default), Saturation Pressure, Vapor Pressure or Wet Bulb Temperature.
- **Request overrides (BRequestOverridesBlock)**
  This block may override the Aggregation, Interval, Rollup and Time Range properties in the analytics request. It supports both trend and value requests.
- **Rollup (BRollupBlock)**
  This block combines all values of a trend based on the configured Rollup function and Interval property from the request. It supports both trend and value requests.
- **Runtime (BRuntimeBlock)**
  This block accumulates the amount of time the input is true (on or running). It supports both trend and value requests.
- **SlidingWindow (BSlidingWindowBlock)**
  This block produces a trend where each value represents a rollup of all the prior values for a duration of time, referred to as the window. The block only requires a single value within the window to produce a result.
- **Time range offset (BTimeRangeOffsetBlock)**
  This block adjusts the time range, which the analytic trend request may have specified. The resulting time range is an absolute value of two fixed timestamps. This is important when you have additional time offsets in an algorithm. The **Time Range Offset Block** is often used in conjunction with a **Timestamp Offset Block** to perform period-over-period comparisons, such as month over month (MOM) or year over year (YOY).
- **Timestamp offset (BTimestampOffsetBlock)**
  This block adjusts the timestamp of a history record as it is processed by the algorithm. It is often used in conjunction with a **Time Range Offset** block to perform period-over-period comparisons, such as month over month (MOM) or year over year (YOY).
- **Today Builder (BDayBuilderBlock)**
  This block generates a trend that uses past historical data to represent “today.” The system converts timestamps by changing the date to today while preserving the time of day. When there are multiple values for the same timestamp, the rollup determines how to combine them. This block supports both trend and value requests.
- **Value duration (BValueDurationBlock)**
  This block determines how long the input has been at the current value. Both value and trend requests are supported.
- **Value Map (BValueMapBlock)**
  This block represents a map of key value pairs, which define substitution values where you configure the data type (Numeric, String, Boolean, Enum or Other) for both the key and value objects. There are frozen properties associated with the default key value pair and dynamic BValueMapEntry properties for any user defined objects. This block supports both trend and value requests, and retains the timestamp and status of input values.

Parent topic: Logic blocks

Debug (BDebugBlock)

This block provides tools to debug algorithms. To provide debug information for value and trend requests, you link this block in line between other blocks in an algorithm.

Figure 1. Debug Block properties



Property Sheet

Debug (Debug Block)

- Out
- In

Enabled	● false
Debug Mode	Next Request
Print Trends To Console	● false
Result	

To view these properties, double-click the block on the **Wire Sheet** or the block name in the Nav tree.

Property	Value	Description
Enabled	true or false	Turns use of this block on and off. true causes the **Debug Block** to processes requests based on the Debug Mode property and updates the Results property. false disables processing. Regardless of the Enabled property value, the **Debug Block** always passes the unmodified value or trend data through to the downstream linked blocks.
Debug Mode	See Debug mode check boxes	Controls specific results so that you can better understand and explain the results you are receiving.
Print Trend to Console	true or false	Controls the trend rows that appear in the console. When true, the framework prints every row of a trend to the console where you can view each row in the **Application Director**.
Results	data	Provides details about the request parameters and, for value requests, the resulting value.

Debug mode check boxes

Debug mode check box	Description
All Requests	When enabled, updates the result field for every request.
All Trend Requests	When enabled, updates the result field with trend requests only.
All Value Requests	When enabled, updates the result field for value requests only.
Next Request	When enabled, updates the result field for the next request, then disables itself.
Next Trend Request	When enabled, updates the result field for the next trend request, then disables itself.
Next Value Request	When enabled, updates the result field for the next value request, then disables itself.

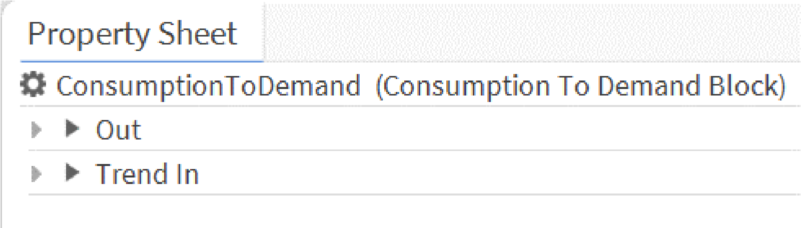
Parent topic: General blocks

Consumption To Demand (BConsumptionToDemandBlock)

This block converts consumption values into demand values for both value and trend requests. This block supports both trend and value requests.

Value requests require access to trend data as their consumption data source. Even though the value request does not include a time range, this block requests the previous one (1) hour of trend data from the input source and searches for the last trend record prior to the current time. It calculates demand by taking the real time value and using the time delta from the last record in the underlying trend.

Figure 1. Consumption to Demand properties



To view these properties, double-click the block on the **Wire Sheet** or the block name in the Nav tree.

Property	Value	Description
Out	read-only value slot	Outputs the demand value calculated from the Trend In consumption value.
Trend In	required value slot	Links from the output of other logic blocks or data sources to supply trend data to the current logic block.

Example

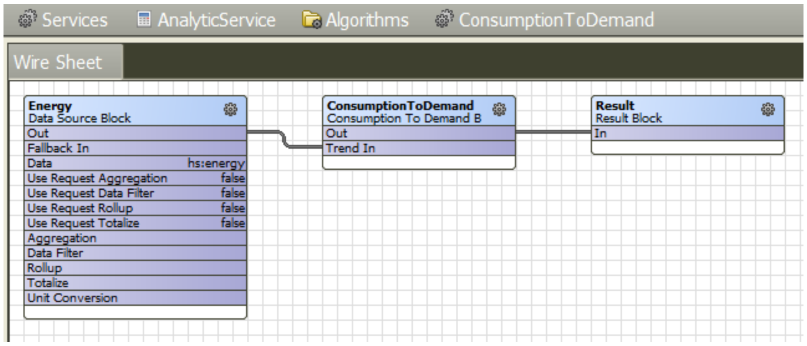
Consider an energy point (KWH) with a 15-minute-interval history. The current value of the point is 240 at 1:28:49 PM and the last record in the applicable history is from 1:15:00 PM.

Inputs: there are 3,600,000 milliseconds in one hour and the time difference is 13 minutes 49 seconds.

Calculations: (13 minutes * 60,000 milliseconds/minute) + (49 seconds * 1,000 milliseconds/second) = 829,000 milliseconds

demand = consumption * (milliseconds in one hour / time difference milliseconds) = 240 KWH * (3,600,000 / 829,000) = 1042 KW

Figure 2. Consumption To Demand algorithm



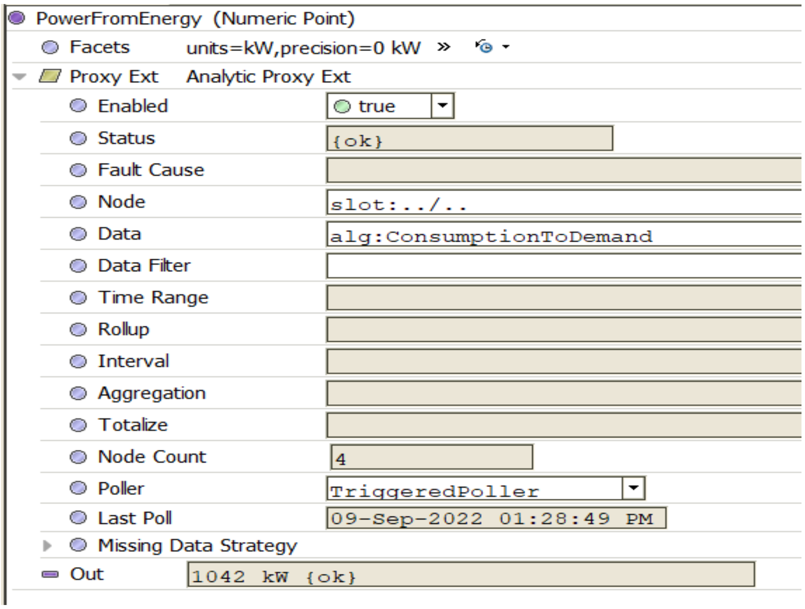
Energy
Data Source Block
Out
Fallback In
Data hs:energy
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Use Request Totalize false
Aggregation
Data Filter
Rollup
Totalize
Unit Conversion

ConsumptionToDemand
Consumption To Demand B
Out
Trend In

Result
Result Block
In

The screen capture shows the blocks used in the algorithm. The following **Property Sheet** shows how the **Proxy Ext** properties are configured.

Figure 3. Consumption To Demand properties

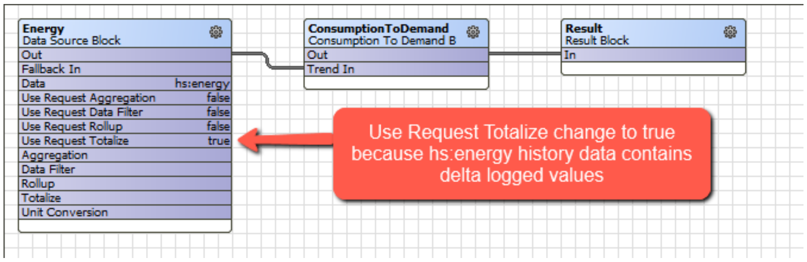


PowerFromEnergy (Numeric Point)

Facets	units=kW,precision=0 kW
Proxy Ext	Analytic Proxy Ext
Enabled	true
Status	{ok}
Fault Cause	
Node	slot:../..
Data	alg:ConsumptionToDemand
Data Filter	
Time Range	
Rollup	
Interval	
Aggregation	
Totalize	
Node Count	4
Poller	TriggeredPoller
Last Poll	09-Sep-2022 01:28:49 PM
Missing Data Strategy	
Out	1042 kW {ok}

For a trend request, if the history data are logged as delta values (delta consumption per interval instead of a totalized, ever increasing value), the totalize property in the request must be true. Otherwise the block calculates the delta of the values, which are already delta-logged. Note, the default totalize property value might be true for the binding but the **DataSourceBlock**’s Use Request Totalize and Totalize properties are false by default. You must either set the Totalize property on the **DataSourceBlock** or change the Use Request Totalize property to true and also ensure the binding’s Totalize property is true.

Figure 4. Consumption To Demand algorithm configuration



Energy
Data Source Block
Out
Fallback In
Data hs:energy
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Use Request Totalize true
Aggregation
Data Filter
Rollup
Totalize
Unit Conversion

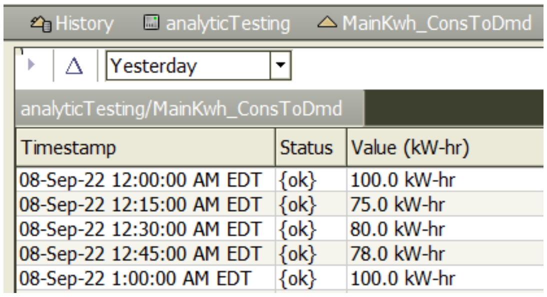
ConsumptionToDemand
Consumption To Demand B
Out
Trend In

Result
Result Block
In

Use Request Totalize change to true because hs:energy history data contains delta logged values

Now, consider another example in which an energy consumption history logs at 15-minute intervals with delta values as follows:

Figure 5. Energy consumption history example

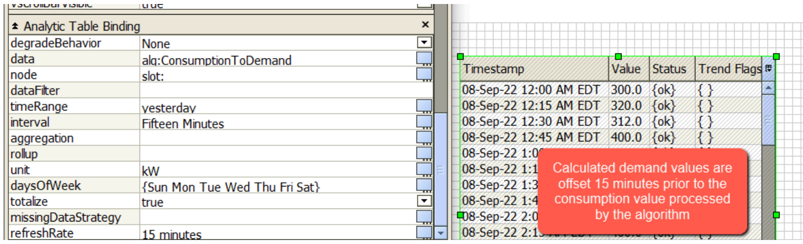


analyticTesting/MainKwh_ConsToDmd

Timestamp	Status	Value (kW-hr)
08-Sep-22 12:00:00 AM EDT	{ok}	100.0 kW-hr
08-Sep-22 12:15:00 AM EDT	{ok}	75.0 kW-hr
08-Sep-22 12:30:00 AM EDT	{ok}	80.0 kW-hr
08-Sep-22 12:45:00 AM EDT	{ok}	78.0 kW-hr
08-Sep-22 1:00:00 AM EDT	{ok}	100.0 kW-hr

The algorithm processes a consumption value of 75.0 KW-hr at 12:15:00 AM and returns a demand value of 300.0 KW at 12:00 AM. The demand value is offset backwards one 15-minute interval indicating what would have been the demand from 12:00 AM - 12:15 AM. This offset results in the 75.0 KW-hr consumption recorded at 12:15 AM.

Figure 6. Calculated demand values offset



Analytic Table Binding

degradeBehavior	None
data	alg:ConsumptionToDemand
node	slot:
dataFilter	
timeRange	yesterday
interval	Fifteen Minutes
aggregation	
rollup	
unit	kW
daysOfWeek	{Sun Mon Tue Wed Thu Fri Sat}
totalize	true
missingDataStrategy	
refreshRate	15 minutes

Timestamp	Value	Status	Trend Flags
08-Sep-22 12:00 AM EDT	300.0	{ok}	{ }
08-Sep-22 12:15 AM EDT	320.0	{ok}	{ }
08-Sep-22 12:30 AM EDT	312.0	{ok}	{ }
08-Sep-22 12:45 AM EDT	400.0	{ok}	{ }
08-Sep-22 1:0			
08-Sep-22 1:1			
08-Sep-22 1:3			
08-Sep-22 1:4			
08-Sep-22 2:0			
08-Sep-22 2:1			

Calculated demand values are offset 15 minutes prior to the consumption value processed by the algorithm

Note: For the specific use case above where consumption is delta logged at 15-minute intervals, the calculated demand should be four times the consumption value because the calculation is consumption * (60 minutes / 15 minutes) or simplified as consumption * 4.

Parent topic: General blocks

Demand to Delta Consumption (BDemandToConsumptionBlock)

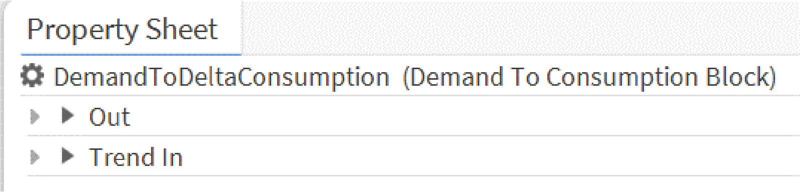
This block converts demand values into consumption values for both Analytic Value and Analytic Trend Requests.

The framework supports both trend and value requests. Value requests require access to trend data as their demand data source. Even though a value request does not include a time range, the block requests the previous 1 hour of trend data for the input source and searches for the last trend record prior to the current time.

Value requests require access to trend data as their demand data source. Even though a value request does not include a time range, the block requests the previous 1 hour of trend data for the input source and searches for the last trend record prior to the current time.

With value requests, the framework calculates consumption by taking the real-time value and using the time delta from the last record in the underlying trend.

Figure 1. Demand to Delta Consumption properties



Property Sheet

DemandToDeltaConsumption (Demand To Consumption Block)

- Out
- Trend In

To view these properties, double-click the block on the **Wire Sheet** or the block name in the Nav tree.

Property	Value	Description
Out	read-only value slot	Outputs a calculated consumption value from the Trend In demand value.
Trend In	required value slot	Links from the output of other logic blocks or data sources to supply trend data to the current logic block.

Example

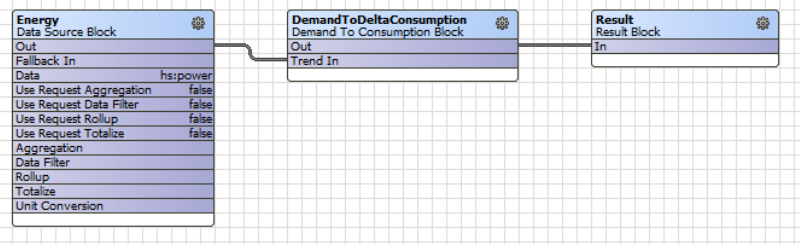
Consider a power point (KW) with a 15-minute interval history. The current value of the point is 240 at 10:07:10 AM and the last record in the applicable history is from 10:00:00 AM.

Inputs: there are 3,600,000 milliseconds in one hour and the time difference is 7 minutes 10 seconds.

Calculations: (7 minutes * 60,000 milliseconds/minute) + (10 seconds * 1,000 milliseconds/second) = 430,000 milliseconds

consumption = demand * (time difference milliseconds / milliseconds in one hour) = 240 KW * (430,000 / 3,600,000) = 28.7 KWH

Figure 2. Demand to Delta Consumption algorithm



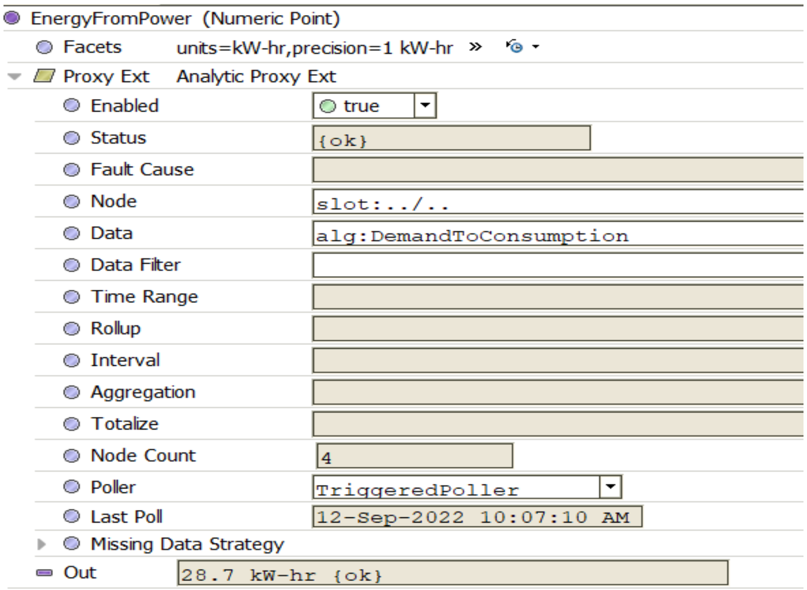
Energy
Data Source Block
Out
Fallback In
Data hs:power
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Use Request Totalize false
Aggregation
Data Filter
Rollup
Totalize
Unit Conversion

DemandToDeltaConsumption
Demand To Consumption Block
Out
Trend In

Result
Result Block
In

The screen capture shows the blocks used in the algorithm. The following **Property Sheet** shows how the **Proxy Ext** properties are configured.

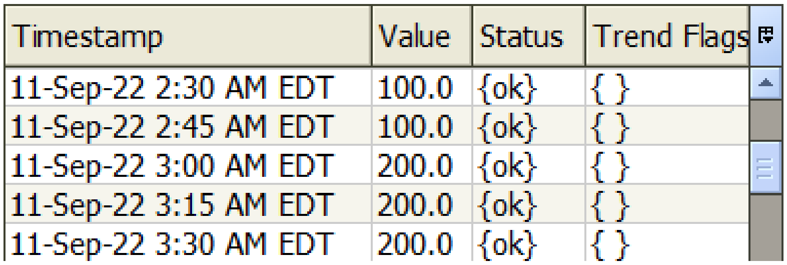
Figure 3. Demand to Delta Consumption properties



EnergyFromPower (Numeric Point)	
Facets	units=kW-hr,precision=1 kW-hr
Proxy Ext	Analytic Proxy Ext
Enabled	true
Status	{ok}
Fault Cause	
Node	slot:../..
Data	alg:DemandToConsumption
Data Filter	
Time Range	
Rollup	
Interval	
Aggregation	
Totalize	
Node Count	4
Poller	TriggeredPoller
Last Poll	12-Sep-2022 10:07:10 AM
Missing Data Strategy	
Out	28.7 kW-hr {ok}

Consider an electrical power history logged at 15-minute intervals with the following values:

Figure 4. Electrical power history values

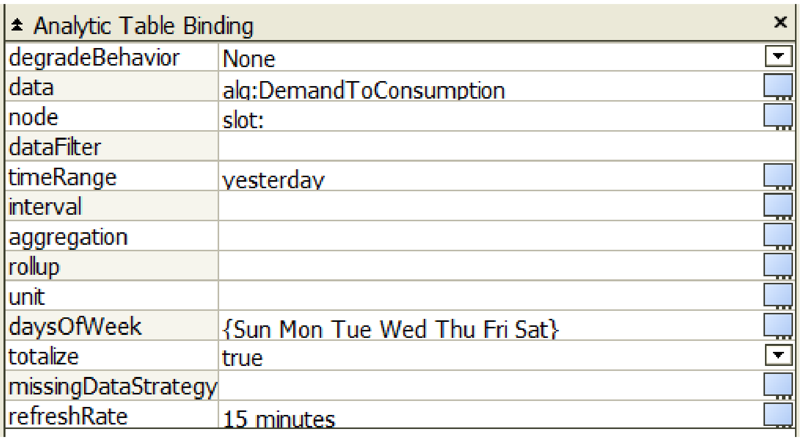


Timestamp	Value	Status	Trend Flags
11-Sep-22 2:30 AM EDT	100.0	{ok}	{ }
11-Sep-22 2:45 AM EDT	100.0	{ok}	{ }
11-Sep-22 3:00 AM EDT	200.0	{ok}	{ }
11-Sep-22 3:15 AM EDT	200.0	{ok}	{ }
11-Sep-22 3:30 AM EDT	200.0	{ok}	{ }

The algorithm processes a demand value of 200.0 KW at 3:00 AM and returns a delta consumption value of 50.0 KWH at 2:45 AM. It offsets the consumption value backwards by one 15-minute interval indicating this would have been the consumption from 2:45 AM - 3:00 AM. The result is 200.0 KW of demand recorded at 3:00 AM.

A Px view might be configured with a table widget that uses an Analytic Table Binding configured as below.

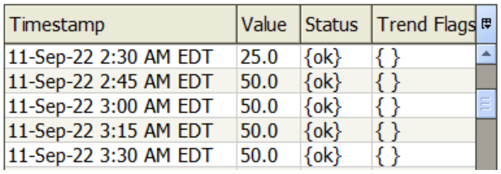
Figure 5. Analytic Table Binding properties



Analytic Table Binding	
degradeBehavior	None
data	alg:DemandToConsumption
node	slot:
dataFilter	
timeRange	yesterday
interval	
aggregation	
rollup	
unit	
daysOfWeek	{Sun Mon Tue Wed Thu Fri Sat}
totalize	true
missingDataStrategy	
refreshRate	15 minutes

The algorithm processes a demand value of 200.0 KW at 3:00 AM. The result generates the following table.

Figure 6. Table result



Timestamp	Value	Status	Trend Flags
11-Sep-22 2:30 AM EDT	25.0	{ok}	{ }
11-Sep-22 2:45 AM EDT	50.0	{ok}	{ }
11-Sep-22 3:00 AM EDT	50.0	{ok}	{ }
11-Sep-22 3:15 AM EDT	50.0	{ok}	{ }
11-Sep-22 3:30 AM EDT	50.0	{ok}	{ }

The algorithm returns a delta consumption value of 50.0 KWH at 2:45 AM. This consumption value is offset backwards one 15-minute interval indicating that 50.0 KWH would have been the consumption from 2:45 AM - 3:00 AM to result in the 200.0 KW demand being recorded at 3:00 AM.

Note: For this specific use case where demand is logged at 15-minute intervals, the calculated demand should be one-quarter the demand value because the calculation is `demand * (15 minutes / 60 minutes)` or simplified as `demand * 0.25`.

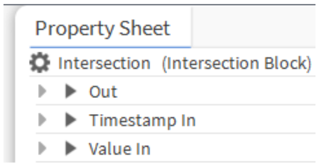
Parent topic: General blocks

Intersection (BIntersectionBlock)

This component combines two trends. It includes only rows with identical timestamps from both inputs. This object was designed for the scenario where the timestamps from a filtered trend are needed to extract values from another trend. The block supports both value and trend requests, although the intent of this block is for processing trends.

This block evaluates whether the Value In timestamp matches the Timestamp In timestamp. If they match, the algorithm passes the Value In to the Out. For a trend request the block filters out records where the two input timestamps do not match.

Figure 1. Intersection properties



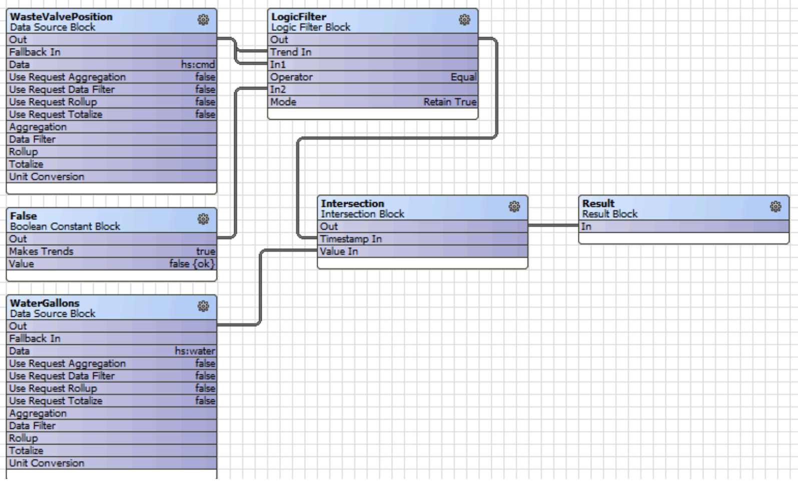
To view these properties, double-click the block on the **Wire Sheet** or the block name in the Nav tree.

Property	Value	Description
Out	read-only value slot	Outputs the filtered trend records from the Value In.
Timestamp In	value slot	Links from the output of other logic blocks or data sources to identify timestamps to use from this trend to filter records from the Value In trend.
Value In	value slot	Links from the output of other logic blocks or data sources to identify values to use from this trend.

Examples

An algorithm might use this block with a **Logic Filter** block that only passes a trend record when the waste valve position is closed (not dumping some liquid product to waste) into an **Intersection** block as the Timestamp In. The **Intersection** block filters out any trend records from the **Water Gallons** data source linked into the Value In whenever there is not a record with the matching timestamp from the **Logic Filter** linked to the Timestamp In. In this case, both data sources have the same interval (15 minutes) but the **Waste Valve Position** data is potentially filtered so the result of the algorithm will likely have less than 96 records in a single day, meaning no record whenever the Waste Value Position is open.

Figure 2. Intersection algorithm



In the next example, the two data sources have different intervals (day interval vs. 15 minute interval). The **Intersection** block normalizes (overrides) the 15-minute interval to return daily intervals to evaluate. In this case, there would be only a single daily record at 12:00 am for both the Timestamp In and Value In slots so the block passes all of the records. The **Intersection** block is not really filtering, but the function of aligning the intervals and timestamps is something a **Time** filter cannot do on its own, at least not with needing to process records at 15-minute intervals and rollup the results to daily values.

This example has one history that is time filtered for three shifts and then intersected to create a web chart. Each shift shows on the same chart.

Shift 1 is from 7:00 am - 4:00 pm

Shift 2 is from 4:00 pm - 12:00 am (midnight)

Shift 3 is from 12:00 am (midnight) - 7:00 am

The **Wire Sheets** for the three shifts follow.

Figure 3. Shift 1 algorithm

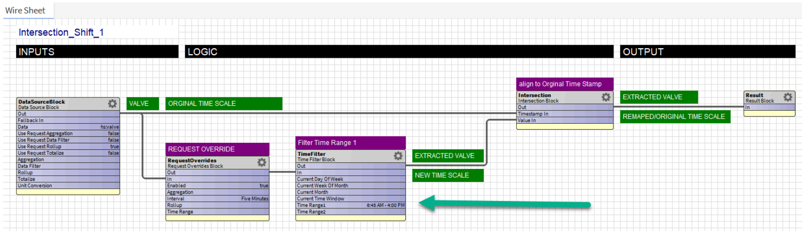
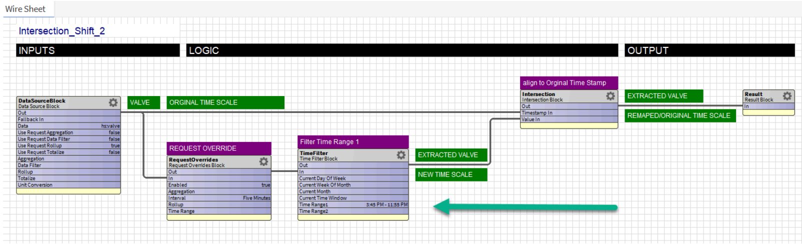


Figure 4. Shift 2 algorithm



DataSourceBlock
Data Source Block
Out
Fallback In
Data hs:valve
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup true
Use Request Totalize false
Aggregation
Data Filter
Rollup
Totalize
Unit Conversion

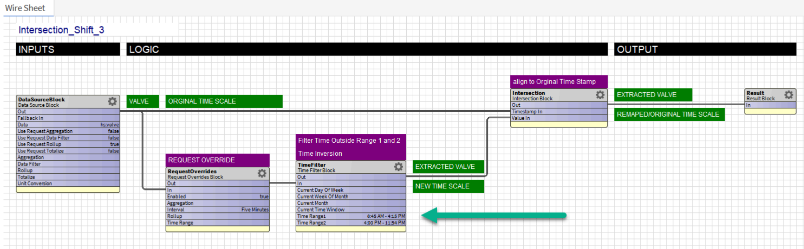
RequestOverrides
Request Overrides Block
Out
In
Enabled true
Aggregation
Interval Five Minutes
Rollup
Time Range

TimeFilter
Time Filter Block
Out
In
Current Day Of Week
Current Week Of Month
Current Month
Current Time Window
Time Range1 3:45 PM - 11:55 PM
Time Range2

Intersection
Intersection Block
Out
Timestamp In
Value In

Result
Result Block
In

Figure 5. Shift 3 algorithm



DataSourceBlock
Data Source Block
Out
Fallback In
Data hs:valve
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup true
Use Request Totalize false
Aggregation
Data Filter
Rollup
Totalize
Unit Conversion

RequestOverrides
Request Overrides Block
Out
In
Enabled true
Aggregation
Interval Five Minutes
Rollup
Time Range

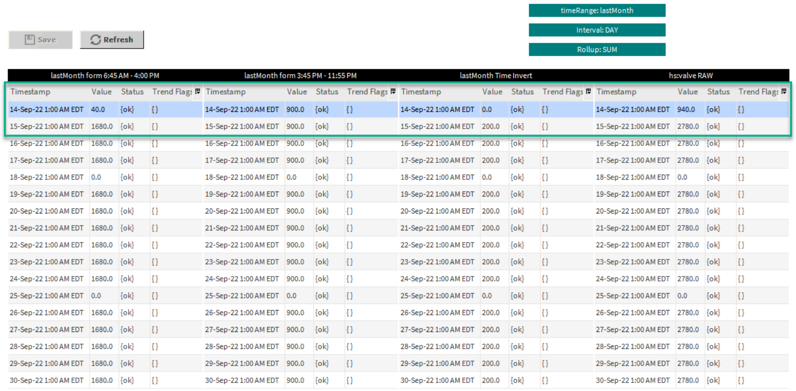
TimeFilter
Time Filter Block
Out
In
Current Day Of Week
Current Week Of Month
Current Month
Current Time Window
Time Range1 6:45 AM - 4:15 PM
Time Range2 4:00 PM - 11:54 PM

Intersection
Intersection Block
Out
Timestamp In
Value In

Result
Result Block
In

The following screen capture shows the data processed by the algorithms.

Figure 6. Data source



timeRange: lastMonth

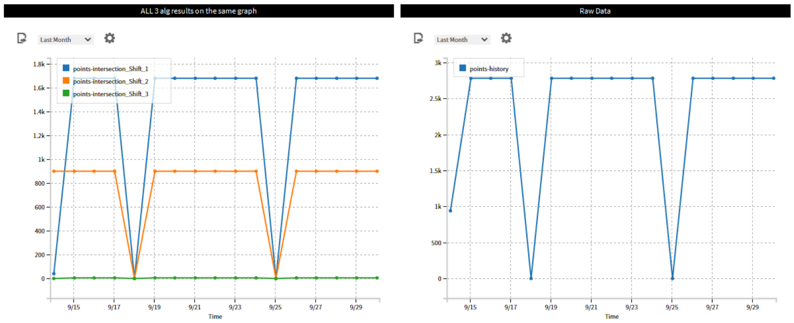
Interval: DAY

Rollup: SUM

lastMonth form 6:45 AM - 4:00 PM				lastMonth form 3:45 PM - 11:55 PM				lastMonth Time Invert				hs:valve RAW			
Timestamp	Value	Status	Trend Flags	Timestamp	Value	Status	Trend Flags	Timestamp	Value	Status	Trend Flags	Timestamp	Value	Status	Trend Flags
14-Sep-22 1:00 AM EDT	40.0	{ok}	{}	14-Sep-22 1:00 AM EDT	900.0	{ok}	{}	14-Sep-22 1:00 AM EDT	0.0	{ok}	{}	14-Sep-22 1:00 AM EDT	940.0	{ok}	{}
15-Sep-22 1:00 AM EDT	1680.0	{ok}	{}	15-Sep-22 1:00 AM EDT	900.0	{ok}	{}	15-Sep-22 1:00 AM EDT	200.0	{ok}	{}	15-Sep-22 1:00 AM EDT	2780.0	{ok}	{}
16-Sep-22 1:00 AM EDT	1680.0	{ok}	{}	16-Sep-22 1:00 AM EDT	900.0	{ok}	{}	16-Sep-22 1:00 AM EDT	200.0	{ok}	{}	16-Sep-22 1:00 AM EDT	2780.0	{ok}	{}
17-Sep-22 1:00 AM EDT	1680.0	{ok}	{}	17-Sep-22 1:00 AM EDT	900.0	{ok}	{}	17-Sep-22 1:00 AM EDT	200.0	{ok}	{}	17-Sep-22 1:00 AM EDT	2780.0	{ok}	{}
18-Sep-22 1:00 AM EDT	0.0	{ok}	{}	18-Sep-22 1:00 AM EDT	0.0	{ok}	{}	18-Sep-22 1:00 AM EDT	0.0	{ok}	{}	18-Sep-22 1:00 AM EDT	0.0	{ok}	{}
19-Sep-22 1:00 AM EDT	1680.0	{ok}	{}	19-Sep-22 1:00 AM EDT	900.0	{ok}	{}	19-Sep-22 1:00 AM EDT	200.0	{ok}	{}	19-Sep-22 1:00 AM EDT	2780.0	{ok}	{}
20-Sep-22 1:00 AM EDT	1680.0	{ok}	{}	20-Sep-22 1:00 AM EDT	900.0	{ok}	{}	20-Sep-22 1:00 AM EDT	200.0	{ok}	{}	20-Sep-22 1:00 AM EDT	2780.0	{ok}	{}
21-Sep-22 1:00 AM EDT	1680.0	{ok}	{}	21-Sep-22 1:00 AM EDT	900.0	{ok}	{}	21-Sep-22 1:00 AM EDT	200.0	{ok}	{}	21-Sep-22 1:00 AM EDT	2780.0	{ok}	{}
22-Sep-22 1:00 AM EDT	1680.0	{ok}	{}	22-Sep-22 1:00 AM EDT	900.0	{ok}	{}	22-Sep-22 1:00 AM EDT	200.0	{ok}	{}	22-Sep-22 1:00 AM EDT	2780.0	{ok}	{}
23-Sep-22 1:00 AM EDT	1680.0	{ok}	{}	23-Sep-22 1:00 AM EDT	900.0	{ok}	{}	23-Sep-22 1:00 AM EDT	200.0	{ok}	{}	23-Sep-22 1:00 AM EDT	2780.0	{ok}	{}
24-Sep-22 1:00 AM EDT	1680.0	{ok}	{}	24-Sep-22 1:00 AM EDT	900.0	{ok}	{}	24-Sep-22 1:00 AM EDT	200.0	{ok}	{}	24-Sep-22 1:00 AM EDT	2780.0	{ok}	{}
25-Sep-22 1:00 AM EDT	0.0	{ok}	{}	25-Sep-22 1:00 AM EDT	0.0	{ok}	{}	25-Sep-22 1:00 AM EDT	0.0	{ok}	{}	25-Sep-22 1:00 AM EDT	0.0	{ok}	{}
26-Sep-22 1:00 AM EDT	1680.0	{ok}	{}	26-Sep-22 1:00 AM EDT	900.0	{ok}	{}	26-Sep-22 1:00 AM EDT	200.0	{ok}	{}	26-Sep-22 1:00 AM EDT	2780.0	{ok}	{}
27-Sep-22 1:00 AM EDT	1680.0	{ok}	{}	27-Sep-22 1:00 AM EDT	900.0	{ok}	{}	27-Sep-22 1:00 AM EDT	200.0	{ok}	{}	27-Sep-22 1:00 AM EDT	2780.0	{ok}	{}
28-Sep-22 1:00 AM EDT	1680.0	{ok}	{}	28-Sep-22 1:00 AM EDT	900.0	{ok}	{}	28-Sep-22 1:00 AM EDT	200.0	{ok}	{}	28-Sep-22 1:00 AM EDT	2780.0	{ok}	{}
29-Sep-22 1:00 AM EDT	1680.0	{ok}	{}	29-Sep-22 1:00 AM EDT	900.0	{ok}	{}	29-Sep-22 1:00 AM EDT	200.0	{ok}	{}	29-Sep-22 1:00 AM EDT	2780.0	{ok}	{}
30-Sep-22 1:00 AM EDT	1680.0	{ok}	{}	30-Sep-22 1:00 AM EDT	900.0	{ok}	{}	30-Sep-22 1:00 AM EDT	200.0	{ok}	{}	30-Sep-22 1:00 AM EDT	2780.0	{ok}	{}

The following is a chart produced by the algorithms.

Figure 7. Chart



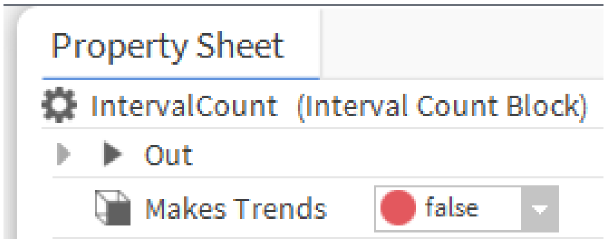
Parent topic: General blocks

Interval count (BIntervalCountBlock)

This block does not have an input slot. It automatically detects the time range and interval from the analytic trend request and outputs the number of intervals for the given time range. The purpose of this object is to distribute a value, such as cost among all rows in a trend being displayed in a chart or table.

If the **Interval Count** block needs to pass the results as a trend via the Out slot linked to a downstream block, you must change the block's Makes Trends property its default of false to true.

Figure 1. Interval Count properties



Property Sheet

IntervalCount (Interval Count Block)

Out

Makes Trends ● false

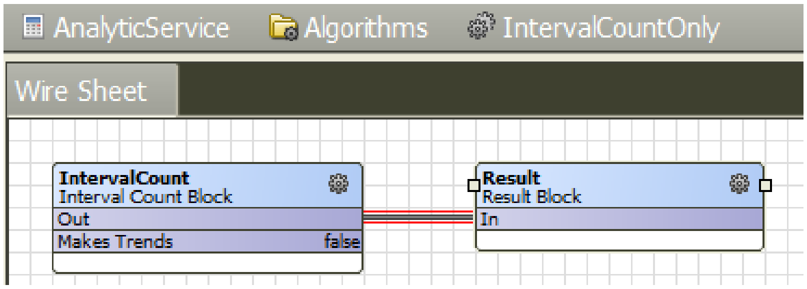
To view these properties, double-click the block on the **Wire Sheet** or the block name in the Nav tree.

Property	Value	Description
Out	read-only value slot	Outputs a constant value of constant trend based on the value of Make Trends or Makes Trends property, where the value is the calculated interval count.
Makes Trends	true or false (default)	Controls the processing of trend requests. false outputs a constant value for both value and trend requests. true outputs a trend with constant values for trend requests.

Example

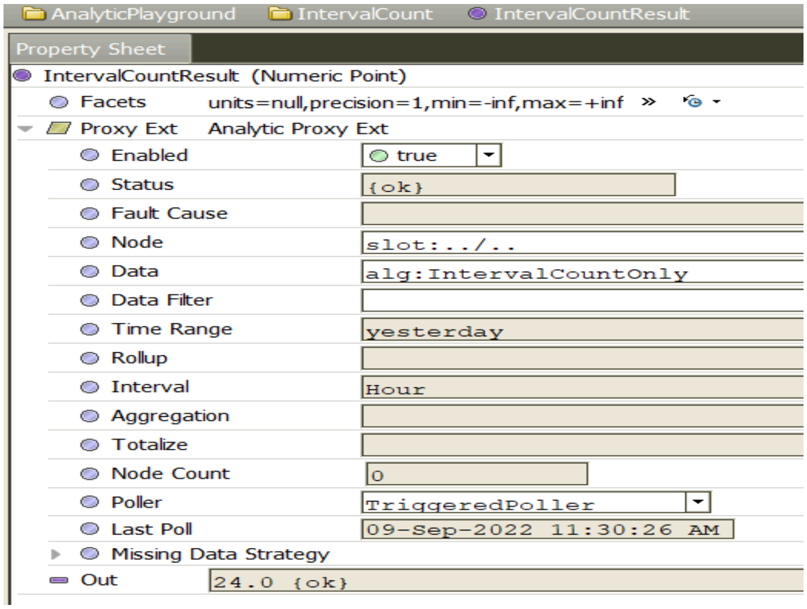
Consider a basic algorithm with no **DataSourceBlock** where the **Interval Count** block is linked directly to a **Result** block.

Figure 2. Interval Count algorithm



A numeric point with an Analytic Proxy Ext that is configured with Data = alg:IntervalCountOnly, time range = yesterday and interval = hour returns a value of 24, indicating there are 24 hourly intervals in the time range of yesterday.

Figure 3. Interval Count configuration

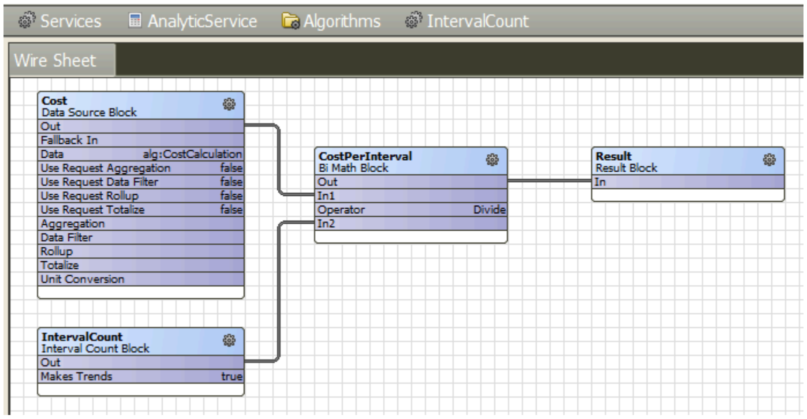


IntervalCountResult (Numeric Point)

Facets	units=null,precision=1,min=-inf,max=+inf
Proxy Ext	Analytic Proxy Ext
Enabled	true
Status	{ok}
Fault Cause	
Node	slot:../..
Data	alg:IntervalCountOnly
Data Filter	
Time Range	yesterday
Rollup	
Interval	Hour
Aggregation	
Totalize	
Node Count	0
Poller	TriggeredPoller
Last Poll	09-Sep-2022 11:30:26 AM
Missing Data Strategy	
Out	24.0 {ok}

Consider a more complex algorithm with a data source named **Cost**, which runs another algorithm to calculate a cost for the requested time range. The **Interval Count** component calculates the number of intervals in the requested time range, which calculates and displays the cost per interval in a chart or table.

Figure 4. Cost for requested time range algorithm



Cost
Data Source Block

Out	
Fallback In	
Data	alg:CostCalculation
Use Request Aggregation	false
Use Request Data Filter	false
Use Request Rollup	false
Use Request Totalize	false
Aggregation	
Data Filter	
Rollup	
Totalize	
Unit Conversion	

CostPerInterval
Bi Math Block

Out	
In1	
Operator	Divide
In2	

Result
Result Block

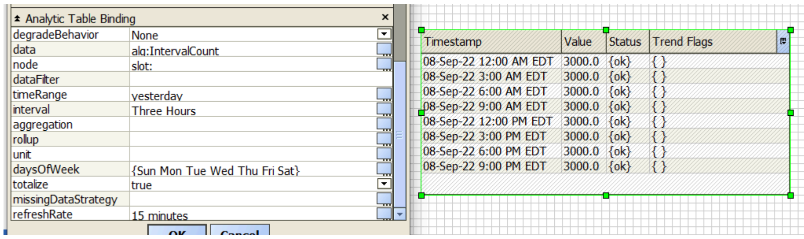
In

IntervalCount
Interval Count Block

Out	
Makes Trends	true

Assuming the **Cost** data source (alg:CostCalculation) returns a value of 24,000 the **Analytic Table Binding** requests a time range of yesterday (24 hours) with an interval of three hours (8 intervals in the 24 hour time range); the algorithm distributes the cost of 24,000 evenly to each row in the table.

Figure 5. Analytic Table Binding and output table



Analytic Table Binding

degradeBehavior	None
data	alg:IntervalCount
node	slot:
dataFilter	
timeRange	yesterday
interval	Three Hours
aggregation	
rollup	
unit	
daysOfWeek	{Sun Mon Tue Wed Thu Fri Sat}
totalize	true
missingDataStrategy	
refreshRate	15 minutes

Timestamp	Value	Status	Trend Flags
08-Sep-22 12:00 AM EDT	3000.0	{ok}	{ }
08-Sep-22 3:00 AM EDT	3000.0	{ok}	{ }
08-Sep-22 6:00 AM EDT	3000.0	{ok}	{ }
08-Sep-22 9:00 AM EDT	3000.0	{ok}	{ }
08-Sep-22 12:00 PM EDT	3000.0	{ok}	{ }
08-Sep-22 3:00 PM EDT	3000.0	{ok}	{ }
08-Sep-22 6:00 PM EDT	3000.0	{ok}	{ }
08-Sep-22 9:00 PM EDT	3000.0	{ok}	{ }

Parent topic: General blocks

Logic folder (LogicFolder)

This folder block organizes other analytic blocks and logic in a sub folder of an algorithm. It is similar to an **Algorithm** in that both have a default **Result Block**, but the **Result Block** is linked to the Out slot of the **Logic Folder**. If an algorithm contains a lot of blocks in its **Wire Sheet**, it may be useful to organize subsets of the blocks based on functionality.

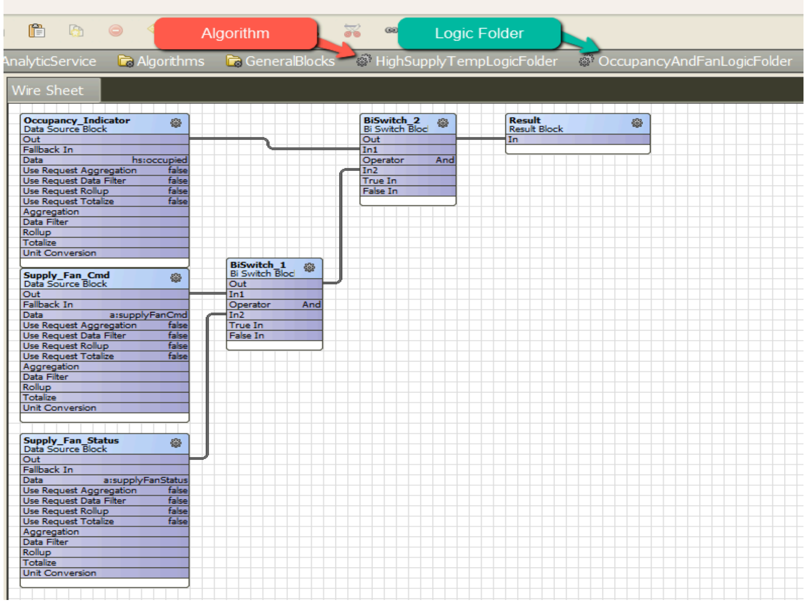
To view these properties, double-click the block on the **Wire Sheet** or the block name in the Nav tree.

Property	Value	Description
Out	read-only value slot	Outputs a constant value.
Result	value	Displays the logical result.

Example

A **Logic Folder** named **OccupancyAndFanLogicFolder** is nested within the **HighSupplyTempLogicFolder** of an algorithm, and contains a subset of Boolean logic evaluating if the equipment is occupied, the supply fan command is on, and the supply fan status indicates on.

Figure 1. OccupancyAndFanLogicFolder algorithm



Occupancy_Indicator
Data Source Block
Out
Fallback In
Data hs:occupied
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Use Request Totalize false
Aggregation
Data Filter
Rollup
Totalize
Unit Conversion

Supply_Fan_Cmd
Data Source Block
Out
Fallback In
Data a:supplyFanCmd
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Use Request Totalize false
Aggregation
Data Filter
Rollup
Totalize
Unit Conversion

Supply_Fan_Status
Data Source Block
Out
Fallback In
Data a:supplyFanStatus
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Use Request Totalize false
Aggregation
Data Filter
Rollup
Totalize
Unit Conversion

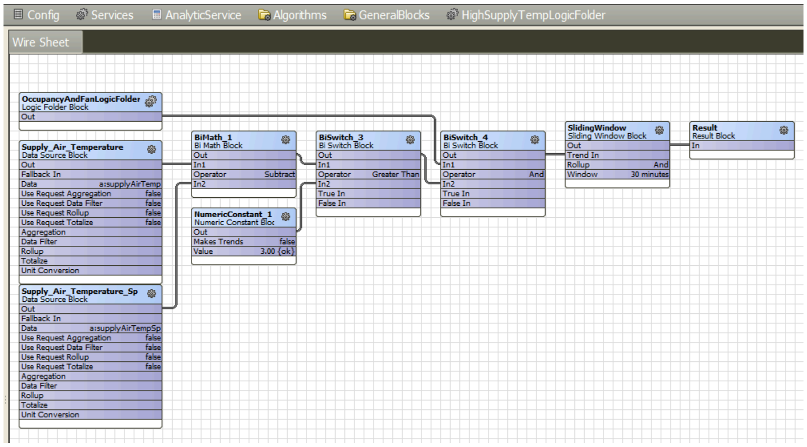
BiSwitch_1
Bi Switch Bloc
Out
In1
Operator And
In2
True In
False In

BiSwitch_2
Bi Switch Bloc
Out
In1
Operator And
In2
True In
False In

Result
Result Block
In

The Result block from the **OccupancyAndFanLogicFolder** outputs the result value through its Out slot, which can be linked to other blocks in the parent **HighSupplyTempLogicFolder** algorithm.

Figure 2. HighSupplyTempLogicFolder

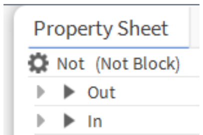


Parent topic: General blocks

Not (BNotBlock)

This component outputs the Boolean opposite of the In property. It contains no properties to be configured. It supports both trend and value requests. The **Not Block** may be useful for determining off peak times for energy reporting or for scheduling related Analytics and fault detection.

Figure 1. Not properties

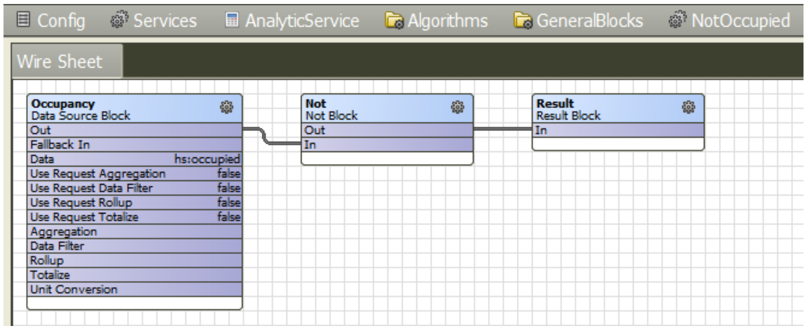


To view these properties, double-click the block on the **Wire Sheet** or the block name in the Nav tree.

Slot	Value	Description
Out	read-only value slot	Outputs the Boolean opposite of the In.
In	required value slot	Links from the output of other logic blocks or data sources to supply trend data to the current logic block.

Example

Figure 2. Not block algorithm

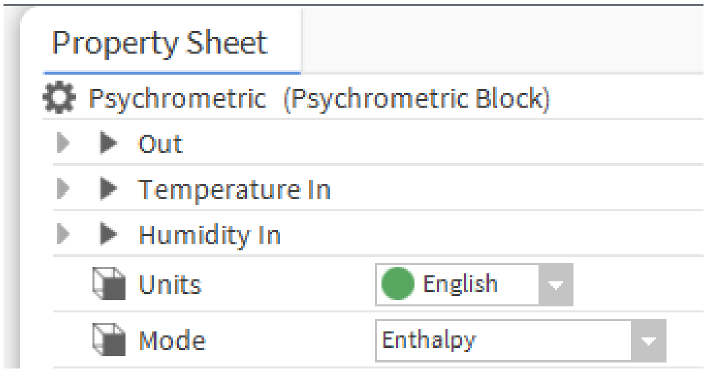


Parent topic: General blocks

Psychrometric (BPsychrometricBlock)

This block calculates various properties of moist air given the temperature (Fahrenheit or Celsius) and humidity (% relative humidity) inputs. It supports both trend and value requests. The Mode property determines the calculation to perform: Dew Point Temperature, Enthalpy (default), Saturation Pressure, Vapor Pressure or Wet Bulb Temperature.

Figure 1. Psychometric properties



Property Sheet

Psychrometric (Psychrometric Block)

- Out
- Temperature In
- Humidity In

Units: English

Mode: Enthalpy

To view these properties, double-click the block on the **Wire Sheet** or the block name in the Nav tree.

Property	Value	Description
Out	read-only value slot	Outputs a calculated value based on the Temperature In, Humidity In values and the Mode properties.
Temperature In	required value slot	Links from the output of other logic blocks or data sources to supply degrees of temperature. Note: Make sure the Units property matches the units of this property.
Humidity In	required value slot	Links from the output of other logic blocks or data sources to supply the percentage of humidity. Humidity In values are expressed as 0-100% regardless of the Units property value.
Units	drop-down list	Identifies the units: Metric or English (default) for both Temperature In and Out. The unit the system uses to display temperature values depends on this property and on your selection for Mode. Refer to Metric and English units.
Mode	drop-down list	Identifies the psychrometric calculation to perform: Dewpoint Temp, Enthalpy (default), Saturation Pressure, Vapor Pressure or Wetbulb Temp.

Metric and English units

Use this table to determine the correct units for inputs and outputs based on the configuration of the Mode and Units properties.

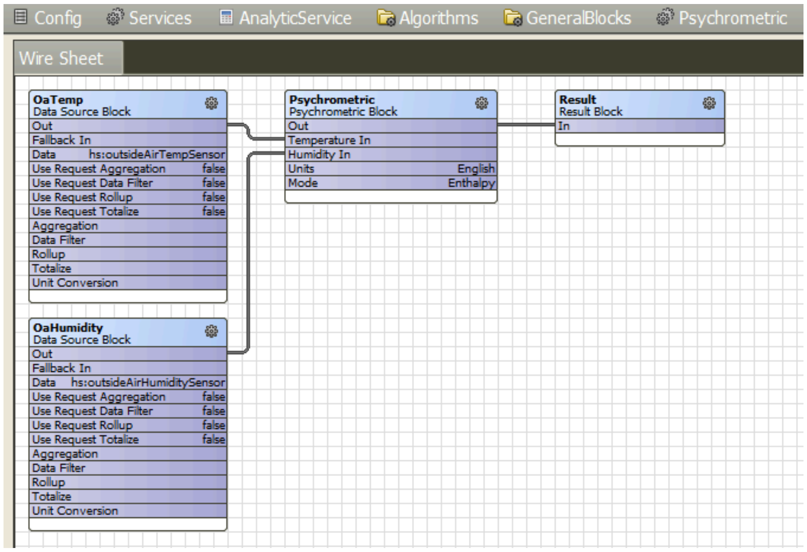
Table 1. Mode units in Metric and English

Mode property	Units = Metric	Units = English
Dewpoint Temp (Dewpoint Temperature)	°C (degrees Celsius)	°F (degrees Fahrenheit)
Enthalpy	kj/kg (kilo joule per kilo gram)	BTU/lb (British thermal unit per pound)
Saturation Pressure	kPa (kilo Pascal)	lbs (pounds)
Vapor Pressure	hPa (hectopascals)	inHg or in.Hg (inch of mercury)
Wetbulb Temp (Wetbulb Temperature)	°C (degrees Celsius)	°F (degrees Fahrenheit)

Example

An algorithm may be configured with **DataSourceBlock**s for temperature (hs:outsideAirTempSensor) and humidity (hs:outsideAirHumiditySensor), which are linked to the **Psychrometric** block inputs.

Figure 2. Psychrometric block algorithm



OaTemp
Data Source Block
Out
Fallback In
Data hs:outsideAirTempSensor
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Use Request Totalize false
Aggregation
Data Filter
Rollup
Totalize
Unit Conversion

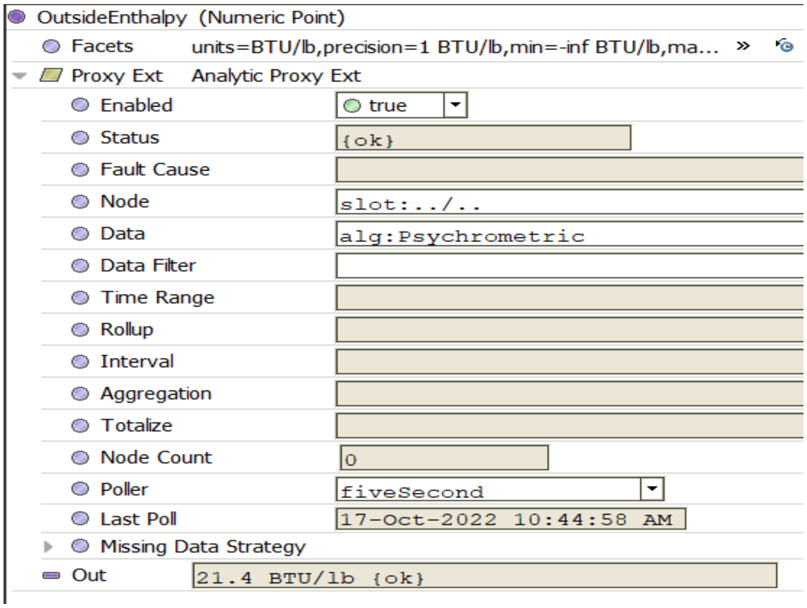
OaHumidity
Data Source Block
Out
Fallback In
Data hs:outsideAirHumiditySensor
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Use Request Totalize false
Aggregation
Data Filter
Rollup
Totalize
Unit Conversion

Psychrometric
Psychrometric Block
Out
Temperature In
Humidity In
Units English
Mode Enthalpy

Result
Result Block
In

A numeric point with an analytic proxy extension may be used to calculate real time enthalpy values using numeric points in the station with outside air temperature (hs:outsideAirTempSensor and a:a marker tags) and outside air humidity (hs:outsideAirHumiditySensor and a:a marker tags).

Figure 3. Psychrometric configuration



OutsideEnthalpy (Numeric Point)

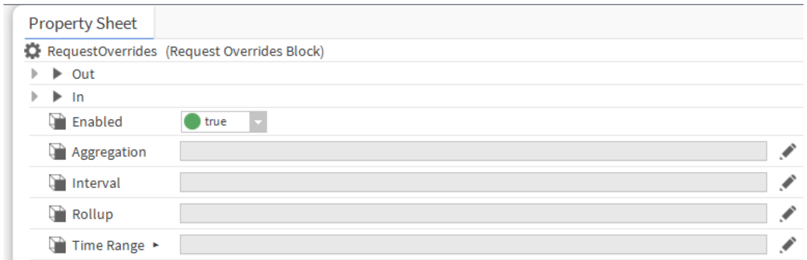
Facets	units=BTU/lb,precision=1 BTU/lb,min=-inf BTU/lb,ma...
Proxy Ext	Analytic Proxy Ext
Enabled	true
Status	{ok}
Fault Cause	
Node	slot:../..
Data	alg:Psychrometric
Data Filter	
Time Range	
Rollup	
Interval	
Aggregation	
Totalize	
Node Count	0
Poller	fiveSecond
Last Poll	17-Oct-2022 10:44:58 AM
Missing Data Strategy	
Out	21.4 BTU/lb {ok}

Parent topic: General blocks

Request overrides (BRequestOverridesBlock)

This block may override the Aggregation, Interval, Rollup and Time Range properties in the analytics request. It supports both trend and value requests.

Figure 1. Request Overrides properties



Property Sheet

RequestOverrides (Request Overrides Block)

Out	
In	
Enabled	true
Aggregation	
Interval	
Rollup	
Time Range	

To view these properties, double-click the block on the **Wire Sheet** or the block name in the Nav tree.

Property	Value	Description
Out	read-only value slot	Outputs the In value.
In	required value slot	Links from the output of other logic blocks or data sources to supply trend data to the current logic block.
Aggregation	check box (if optional, and) drop-down list (property, defaults to First) or ORD parameter (aggregation=option)	Configures the default function used to combine values from multiple data sources into a single value when the block's Use Request Aggregation property is true. This applies to both value and trend requests. If aggregation is not enabled in the binding/settings window, the aggregation value defined in the Data Definition applies

Property	Value	Description
		to all chart bindings, reports and tables. And returns the logical “and” of Boolean values. Avg returns the statistical mean, which is determined by calculating the sum of all values and dividing by the number of values. Count returns the total number or quantity of values in a combination. If you request this value on a binding in a PX view, the system counts the number of values based on the properties defined by the data source block and the algorithm’s **Property Sheet**. First returns the first value in the combination. Last returns the last value in the combination. Max returns the highest value in the combination. Median returns the value in the middle of a sorted combination—the number that separates the higher half from the lower half. Min returns the lowest value in the combination. Mode returns the statistically most frequently occurring number in the combination. Or returns the logical “or” of Boolean values. Range returns the statistical difference between the largest and smallest values in the combination. Sum adds together all values in the combination resulting in a single value. Load Factor returns the average value divided by peak value. Std Dev returns the standard deviation of the values in the combination.
Interval	optional drop-down list (defaults to the optimal number of records on reports); `interval=option`	Refers to the BInterval component, which the framework uses to identify the time between values in a trend (time series). When specified, a rollup is required, which causes the system to combine all values that fall into a single interval. Options range from None to a Year. Above the drop-down list, the Use This Value check box turns on and off the check box next to Interval in the **Settings** window (you access this window by clicking the Edit button (✎), followed by clicking the

Property	Value	Description
		Settings button (⚙) on the chart). The availability of this check box provides an easy way for a user to enable and disable the use of intervals in chart calculations.
Rollup	check box (if optional, and) drop-down list or ORD parameter (when configured in the data definition, defaults to First, when configured elsewhere, defaults to the value as defined in the Data Definition); rollup=option	Configures the default rollup function passed through the algorithm when the block's Use Request Rollup property is set to true. If rollup is not enabled in the binding/ settings window, the rollup value configured in the Data Definition applies to all chart bindings, reports and tables. And returns the logical "and" of Boolean values. Avg returns the statistical mean, which is determined by calculating the sum of all values and dividing by the number of values. Count returns the total number or quantity of values in a combination. If you request this value on a binding in a PX view, the system counts the number of values based on the properties defined by the data source block and the algorithm's **Property Sheet**. First returns the first value in the combination. Last returns the last value in the combination. Max returns the highest value in the combination. Median returns the value in the middle of a sorted combination—the number that separates the higher half from the lower half. Min returns the lowest value in the combination. Mode returns the statistically most frequently occurring number in the combination. Or returns the logical "or" of Boolean values. Range returns the statistical difference between the largest and smallest values in the combination. Sum adds together all values in the combination resulting in a single value. Std Dev calculates the standard deviation of the values in the combination. Load Factor calculates the average divided by peak (Max) value.
Time Range	value (defaults to Today)	Defines the time period over which to

Property	Value	Description
		combine the data in a rollup. This property is required for rollup requests (analyticRollup), trends (analyticTrend), and rollup bindings. It is optional elsewhere. It is not used if the block's Use Request Time Range property is true and the request specifies a time range. Options range from From to All. Time Range defaults to Today, which causes the framework to return a point's current value.

Processing with Use Request Rollup

When Use Request Rollup is false and Rollup is not configured, the block uses the rollup function defined in the applicable Data Definition. If there is no Data Definition, the block uses the default First function. When Use Request Rollup is false and Rollup property is configured, the block uses the Rollup property value. When Use Request Rollup is true, the block uses the rollup function defined in the request (Analytic Proxy Ext, Analytic Binding, etc.), unless the request does not specify the rollup function in which case the block uses the Algorithm's Rollup property value.

The following table may help to determine when each value applies.

Data Definition Rollup	Data Source Block Use Request Rollup	Data Source Block Rollup	Request Rollup	Algorithm Rollup	Actual Rollup Function
Unconfigured	False	Unconfigured	Sum	Max	Defaults to First
Last	False	Unconfigured	Sum	Max	Last
Last	False	**Avg**	Sum	Max	Avg
Last	True	Avg	**Sum**	Max	Sum
Last	True	Avg	Unconfigured	**Max**	Max

Parent topic: General blocks

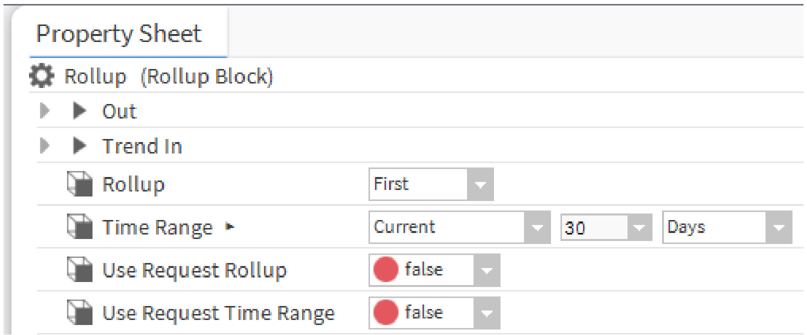
Rollup (BRollupBlock)

This block combines all values of a trend based on the configured Rollup function and Interval property from the request. It supports both trend and value requests.

For a value request the block's Time Range property is used to request trend data and returns a single value by applying the value request's Rollup property if the block's Use Request Rollup property is true, otherwise by applying the block's Rollup property.

This block does not modify the Rollup property in the request, meaning that if the block's Use Request Rollup is false it uses the function defined by the block's Rollup property but other downstream linked blocks use the original Rollup property from the request.

Figure 1. Rollup properties



Property Sheet

Rollup (Rollup Block)

- Out
- Trend In
- Rollup: First
- Time Range ▸: Current 30 Days
- Use Request Rollup: ● false
- Use Request Time Range: ● false

To view these properties, double-click the block on the **Wire Sheet** or the block name in the Nav tree.

Property	Value	Description
Out	read-only value slot	Outputs a rolled up value based on the configured Rollup function and Interval property from the request.
Trend In	required value slot	Links from the output of other logic blocks or data sources to supply trend data to the current logic block.
Rollup	check box (if optional, and) drop-down list or ORD parameter (when configured in the data definition, defaults to First, when configured elsewhere, defaults to the value as defined in the Data Definition); rollup=option	Configures the default Rollup function used by the block when the block’s Use Request Rollup property is set to false. If rollup is not enabled in the binding/ settings window, the rollup value configured in the Data Definition applies to all chart bindings, reports and tables. And returns the logical “and” of Boolean values. Avg returns the statistical mean, which is determined by calculating the sum of all values and dividing by the number of values. Count returns the total number or quantity of values in a combination. If you request this value on a binding in a PX view, the system counts the number of values based on the properties defined by the data source block and the algorithm’s **Property Sheet**. First returns the first value in the combination. Last returns the last value in the combination. Max returns the highest value in the combination. Median returns the value in the middle of a sorted combination—the number that separates the higher half from the lower half. Min returns the lowest value in the

Property	Value	Description
		combination. Mode returns the statistically most frequently occurring number in the combination. Or returns the logical “or” of Boolean values. Range returns the statistical difference between the largest and smallest values in the combination. Sum adds together all values in the combination resulting in a single value. Std Dev calculates the standard deviation of the values in the combination. Load Factor calculates the average divided by peak (Max) value.
Time Range	range of options	Defines the time period over which to combine the data in a rollup. This property is required for rollup requests (analyticRollup), trends (analyticTrend), and rollup bindings. It is optional elsewhere. It is not used if the block’s Use Request Time Range property is true and the request specifies a time range. Options range from From to All. Time Range defaults to Today, which causes the framework to return a point’s current value.
Use Request Rollup	true or false (default)	Determines the rollup function to use. The data definition associated with each tag defines the default function for rolling up data. true causes the framework to use the function defined by the incoming request. false causes the framework to use the function defined in the block’s Rollup property.
Use Request Time Range	true or false (default)	Determines the time range for rolling up data. true causes the framework to use the time range defined by the incoming request. false causes the framework to use the time range defined in the block’s Time Range property.

Processing with Use Request Rollup

When Use Request Rollup is false and Rollup is not configured, the block uses the rollup function defined in the applicable Data Definition. If there is no Data Definition, the block uses the default First function. When Use Request Rollup is false and Rollup property is configured, the block uses the Rollup property value. When Use

Request Rollup is true, the block uses the rollup function defined in the request (Analytic Proxy Ext, Analytic Binding, etc.), unless the request does not specify the rollup function in which case the block uses the Algorithm’s Rollup property value.

The following table may help to determine when each value applies.

Data Definition Rollup	Data Source Block Use Request Rollup	Data Source Block Rollup	Request Rollup	Algorithm Rollup	Actual Rollup Function
Unconfigured	False	Unconfigured	Sum	Max	Defaults to First
Last	False	Unconfigured	Sum	Max	Last
Last	False	**Avg**	Sum	Max	Avg
Last	True	Avg	**Sum**	Max	Sum
Last	True	Avg	Unconfigured	**Max**	Max

Parent topic: General blocks

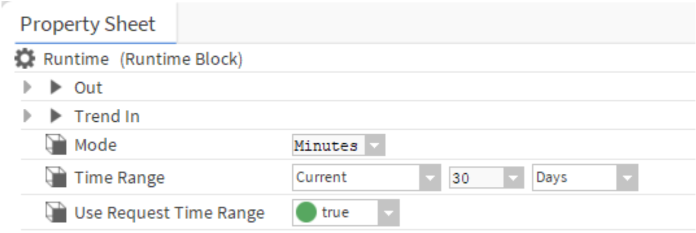
Runtime (BRuntimeBlock)

This block accumulates the amount of time the input is true (on or running). It supports both trend and value requests.

For value requests, the block processes a trend request using the block’s Time Range property. If the input to Trend In has a trend available, the block calculates the total amount of time the value was true, otherwise, it returns a zero value.

For trend requests, when the Use Request Time Range is true (the default value), the block uses the Request Rime Range property, otherwise the block uses its Time Range property. Each row in the result represents the accumulation of true time since last false (off not running) record.

Figure 1. Runtime properties



Property Sheet

Runtime (Runtime Block)

Out

Trend In

Mode: Minutes

Time Range: Current, 30, Days

Use Request Time Range: true

To view these properties, double-click the block on the **Wire Sheet** or the block name in the Nav tree.

Property	Value	Description
Out	read-only value slot	Outputs the calculated runtime value.
Trend In	required value slot	Links from the output of other logic blocks or data sources to supply trend data to the current logic block.
Mode	drop-down list defaults to Minutes	Configures what time unit (milliseconds,

Property	Value	Description
		seconds, minutes or hours) to apply to the accumulated values.
Time Range	range of options (defaults to Current 30 Days)	Defines the time period over which to calculate the accumulated runtime. Defines the beginning and end of a period of time over which to count the number of occurrences or weight. If the alert depends on a certain number of occurrences or weight over time, this property is required, otherwise, it is optional.
Use Request Time Range	true or false (default)	Determines the time range for rolling up data. true causes the framework to use the time range defined by the incoming request. false causes the framework to use the time range defined in the block's Time Range property.

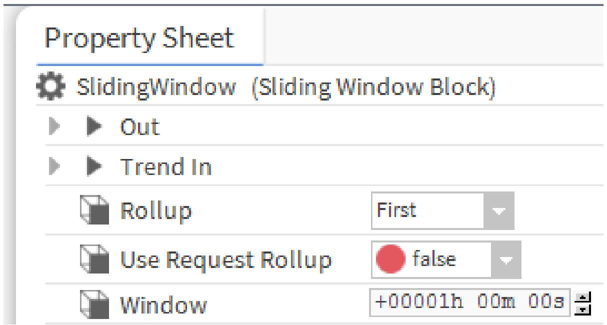
Parent topic: General blocks

SlidingWindow (BSlidingWindowBlock)

This block produces a trend where each value represents a rollup of all the prior values for a duration of time, referred to as the window. The block only requires a single value within the window to produce a result.

For value requests, the block processes a trend request using the block's Window property value to calculate a time range, where the time range start = current station time - window and time range end = current station time. In this case, you would expect the window to likely contain multiple records.

For trend requests, the block iterates through each of the records keeping track of the previous records within the window. It calculates the time range as it evaluates each record where time range start = current record timestamp - window and time range end = current record timestamp. When a trend request processes the first record there are no available past records within the window so the request passes over the first record value, but as it processes subsequent records, it likely processes multiple records whose timestamps are within the calculated window.



Property Sheet

SlidingWindow (Sliding Window Block)

- Out
- Trend In
- Rollup: First
- Use Request Rollup: false
- Window: +00001h 00m 00s

To view these properties, double-click the block on the **Wire Sheet** or the block name in the Nav tree.

Property	Value	Description
Out	read-only value slot	Outputs a trend where each value represents a rollup of all the prior values for a duration of time as defined by the Window property.
Trend In	required value slot	Links from the output of other logic blocks or data sources to supply trend data to the current logic block.
Rollup	check box (if optional, and) drop-down list or ORD parameter (when configured in the data definition, defaults to First, when configured elsewhere, defaults to the value as defined in the Data Definition); rollup=option	For trend requests, configures the rollup function used when the block’s Use Request Rollup property is false or the Rollup property is not explicitly configured in the request. For value requests, always uses the block’s Rollup property regardless of the value of the block’s Use Request Rollup property or whether the Rollup property is explicitly configured in the request. If rollup is not enabled in the binding/ settings window, the rollup value configured in the Data Definition applies to all chart bindings, reports and tables. And returns the logical “and” of Boolean values. Avg returns the statistical mean, which is determined by calculating the sum of all values and dividing by the number of values. Count returns the total number or quantity of values in a combination. If you request this value on a binding in a PX view, the system counts the number of values based on the properties defined by the data source block and the algorithm’s **Property Sheet**. First returns the first value in the combination. Last returns the last value in the combination. Max returns the highest value in the combination. Median returns the value in the middle of a sorted combination—the number that separates the higher half from the lower half. Min returns the lowest value in the combination. Mode returns the statistically most frequently occurring number in the combination. Or returns the logical “or” of Boolean values. Range returns the statistical difference

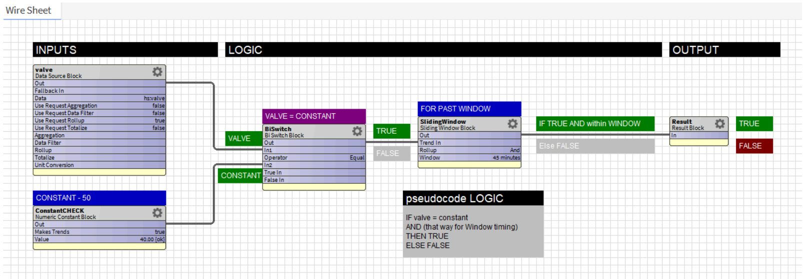
Property	Value	Description
		between the largest and smallest values in the combination. Sum adds together all values in the combination resulting in a single value. Std Dev calculates the standard deviation of the values in the combination. Load Factor calculates the average divided by peak (Max) value.
Use Request Rollup	true or false (default)	Determines the rollup function to use for trend requests. true causes the block to use the function defined by the incoming request. If Time Range is not configured in the request, the block uses its Time Range property. false causes the block to use the function defined in the block's Rollup property.
Window	hours minutes seconds milliseconds (defaults to 1 hour)	Configures the duration of time used in the rollup calculation.

Examples

When processing a trend request, this block:

- does not wait until the window of time is full of records before outputting a result. The first record that it processes may generate a fault condition even though the block is only considering a single record. Once the algorithm iterates through enough records, based on the interval configured for history records, request interval, or SWB window time, the block will likely be evaluating multiple records. A trend request could detect a nuisance fault condition if the first few records meet the criteria before getting the full window time.
- subtracts one millisecond from the window value before calculating the earliest record time. This makes it exclude the earliest record that most people might expect to be included. For example, if the window is 30 minutes, the interval is 15 minutes and the current record timestamp is 11:00 am, the block only considers the 10:45 am and 11:00 am records because the 10:30 am record is one millisecond before the calculated time.

Figure 1. Sliding Window algorithm



valve
Data Source Block
Out
Fallback In
Data hs:valve
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup true
Use Request Totalize false
Aggregation
Data Filter
Rollup
Totalize
Unit Conversion

ConstantCHECK
Numeric Constant Block
Out
Makes Trends true
Value 40.00 {ok}

BiSwitch
Bi Switch Block
Out
In1
Operator Equal
In2
True In
False In

SlidingWindow
Sliding Window Block
Out
Trend In
Rollup And
Window 45 minutes

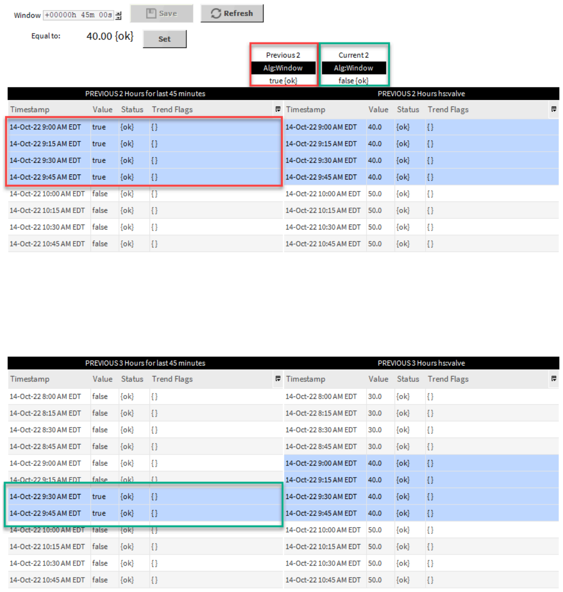
Result
Result Block
In

pseudocode LOGIC

```
IF valve = constant
AND (that way for Window timing)
THEN TRUE
ELSE FALSE
```

The screen captures illustrate the **Sliding Window** behavior where the initial record value is 40 so the **BiSwitch** output is true and the **Sliding Window** Rollup = And, so it passes a true output even though there is only one record at that point in the 45 minute window.

Figure 2. Sliding Window example data



Window +00000h 45m 00s

Equal to: 40.00 {ok}

PREVIOUS 2 Hours for last 45 minutes

Timestamp	Value	Status	Trend Flags
14-Oct-22 9:00 AM EDT	true	{ok}	{}
14-Oct-22 9:15 AM EDT	true	{ok}	{}
14-Oct-22 9:30 AM EDT	true	{ok}	{}
14-Oct-22 9:45 AM EDT	true	{ok}	{}
14-Oct-22 10:00 AM EDT	false	{ok}	{}
14-Oct-22 10:15 AM EDT	false	{ok}	{}
14-Oct-22 10:30 AM EDT	false	{ok}	{}
14-Oct-22 10:45 AM EDT	false	{ok}	{}

PREVIOUS 2 Hours hs:valve

Timestamp	Value	Status	Trend Flags
14-Oct-22 9:00 AM EDT	40.0	{ok}	{}
14-Oct-22 9:15 AM EDT	40.0	{ok}	{}
14-Oct-22 9:30 AM EDT	40.0	{ok}	{}
14-Oct-22 9:45 AM EDT	40.0	{ok}	{}
14-Oct-22 10:00 AM EDT	50.0	{ok}	{}
14-Oct-22 10:15 AM EDT	50.0	{ok}	{}
14-Oct-22 10:30 AM EDT	50.0	{ok}	{}
14-Oct-22 10:45 AM EDT	50.0	{ok}	{}

PREVIOUS 3 Hours for last 45 minutes

Timestamp	Value	Status	Trend Flags
14-Oct-22 8:00 AM EDT	false	{ok}	{}
14-Oct-22 8:15 AM EDT	false	{ok}	{}
14-Oct-22 8:30 AM EDT	false	{ok}	{}
14-Oct-22 8:45 AM EDT	false	{ok}	{}
14-Oct-22 9:00 AM EDT	false	{ok}	{}
14-Oct-22 9:15 AM EDT	false	{ok}	{}
14-Oct-22 9:30 AM EDT	true	{ok}	{}
14-Oct-22 9:45 AM EDT	true	{ok}	{}
14-Oct-22 10:00 AM EDT	false	{ok}	{}
14-Oct-22 10:15 AM EDT	false	{ok}	{}
14-Oct-22 10:30 AM EDT	false	{ok}	{}
14-Oct-22 10:45 AM EDT	false	{ok}	{}

PREVIOUS 3 Hours hs:valve

Timestamp	Value	Status	Trend Flags
14-Oct-22 8:00 AM EDT	30.0	{ok}	{}
14-Oct-22 8:15 AM EDT	30.0	{ok}	{}
14-Oct-22 8:30 AM EDT	30.0	{ok}	{}
14-Oct-22 8:45 AM EDT	30.0	{ok}	{}
14-Oct-22 9:00 AM EDT	40.0	{ok}	{}
14-Oct-22 9:15 AM EDT	40.0	{ok}	{}
14-Oct-22 9:30 AM EDT	40.0	{ok}	{}
14-Oct-22 9:45 AM EDT	40.0	{ok}	{}
14-Oct-22 10:00 AM EDT	50.0	{ok}	{}
14-Oct-22 10:15 AM EDT	50.0	{ok}	{}
14-Oct-22 10:30 AM EDT	50.0	{ok}	{}
14-Oct-22 10:45 AM EDT	50.0	{ok}	{}

Processing with Use Request Rollup

When Use Request Rollup is false and Rollup is not configured, the block uses the rollup function defined in the applicable Data Definition. If there is no Data Definition, the block uses the default First function. When Use Request Rollup is false and Rollup property is configured, the block uses the Rollup property value. When Use Request Rollup is true, the block uses the rollup function defined in the request (Analytic Proxy Ext, Analytic Binding, etc.), unless the request does not specify the rollup function in which case the block uses the Algorithm's Rollup property value.

The following table may help to determine when each value applies.

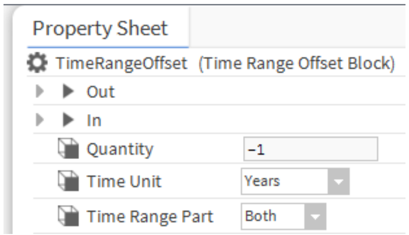
Data Definition Rollup	Data Source Block Use Request Rollup	Data Source Block Rollup	Request Rollup	Algorithm Rollup	Actual Rollup Function
Unconfigured	False	Unconfigured	Sum	Max	Defaults to First
Last	False	Unconfigured	Sum	Max	Last
Last	False	**Avg**	Sum	Max	Avg
Last	True	Avg	**Sum**	Max	Sum
Last	True	Avg	Unconfigured	**Max**	Max

Parent topic: General blocks

Time range offset (BTimeRangeOffsetBlock)

This block adjusts the time range, which the analytic trend request may have specified. The resulting time range is an absolute value of two fixed timestamps. This is important when you have additional time offsets in an algorithm. The **Time Range Offset Block** is often used in conjunction with a **Timestamp Offset Block** to perform period-over-period comparisons, such as month over month (MOM) or year over year (YOY).

Figure 1. Time Range Offset Block properties



Property Sheet

TimeRangeOffset (Time Range Offset Block)

- Out
- In
- Quantity: -1
- Time Unit: Years
- Time Range Part: Both

To view these properties, double-click the block on the **Wire Sheet** or the block name in the Nav tree.

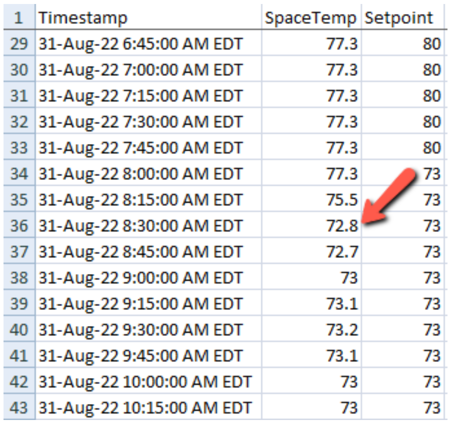
Property	Value	Description
Out	read-only value slot	Outputs the trend data with the time range offset based on the block's configuration.
In	required value slot	Links from the output of other logic blocks or data sources to supply trend data to the current logic block.
Quantity	positive or negative number (defaults to -1)	Defines the numeric amount to offset. When Quantity is negative, the framework adjusts the time range further into the past. When Quantity is positive, the framework adjusts the time range further into the future.
Time Unit	drop-down list; defaults to Years	Defines the unit to use for the offset from Years to Seconds.
Time Range Part	drop-down list; defaults to Both	Defines which parts of the time range to adjust: only the Start, only the End, or Both the Start and End timestamps. You may need to set this property to Start when this block's Quantity is negative and the **Timestamp Offset Block**'s Quantity is positive. You may need to set this property to End when both Quantitys, this block's Quantity and the **Timestamp Offset Block**'s Quantity, are positive.

Examples

Sometimes a time lag occurs between cause and effect. Changing a setpoint for a process does not have an immediate effect on the actual process variable. For example, changing a zone temperature setpoint does not immediately change the actual zone temperature, rather it might take 15 to 30 minutes for the zone temperature to reach the new setpoint. To facilitate easier comparison in these situations, you can use a **Timestamp Offset Block** to shift the timestamps for either the setpoint history or process variable history by the expected lag.

Consider these data:

Figure 2. Timestamp data

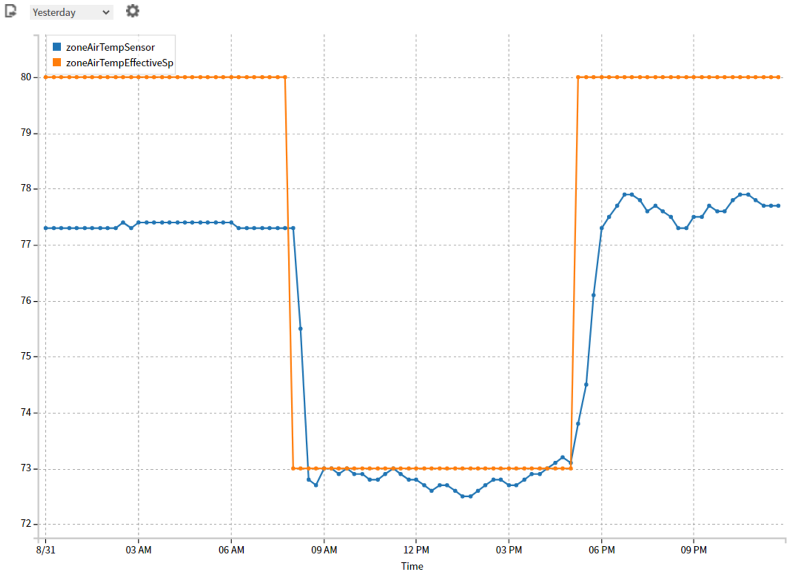


1	Timestamp	SpaceTemp	Setpoint
29	31-Aug-22 6:45:00 AM EDT	77.3	80
30	31-Aug-22 7:00:00 AM EDT	77.3	80
31	31-Aug-22 7:15:00 AM EDT	77.3	80
32	31-Aug-22 7:30:00 AM EDT	77.3	80
33	31-Aug-22 7:45:00 AM EDT	77.3	80
34	31-Aug-22 8:00:00 AM EDT	77.3	73
35	31-Aug-22 8:15:00 AM EDT	75.5	73
36	31-Aug-22 8:30:00 AM EDT	72.8	73
37	31-Aug-22 8:45:00 AM EDT	72.7	73
38	31-Aug-22 9:00:00 AM EDT	73	73
39	31-Aug-22 9:15:00 AM EDT	73.1	73
40	31-Aug-22 9:30:00 AM EDT	73.2	73
41	31-Aug-22 9:45:00 AM EDT	73.1	73
42	31-Aug-22 10:00:00 AM EDT	73	73
43	31-Aug-22 10:15:00 AM EDT	73	73

The table shows a setpoint change from 80 to 73 at 8:00 am. The actual space temp lags by about 30 minutes to reach the new setpoint.

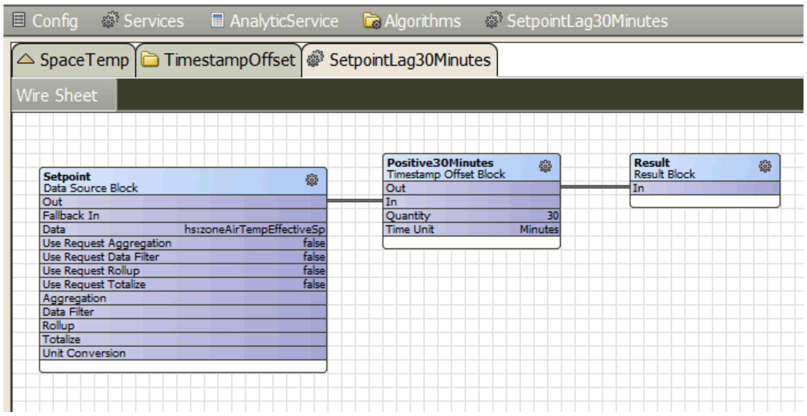
An Analytic Web Chart (`hs:zoneAirTempSensor` and `hs:zoneAirTempEffectiveSp`) shows the data with the lag.

Figure 3. Temperature data shown with a lag



This algorithm, which uses a **Timestamp Offset Block**, can shift the timestamps of the setpoint history forward by 30 minutes.

Figure 4. Timestamp Offset Block algorithm with a positive shift



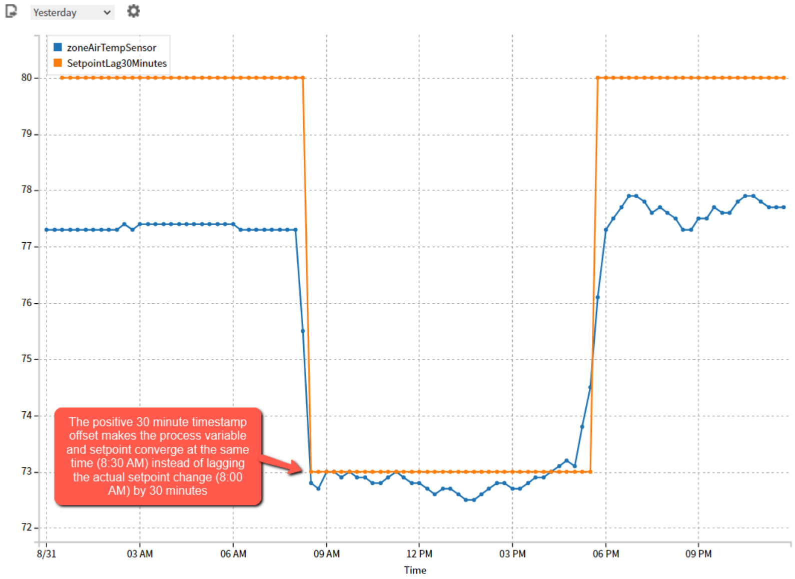
Setpoint
Data Source Block
Out
Fallback In
Data hs:zoneAirTempEffectiveSp
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Use Request Totalize false
Aggregation
Data Filter
Rollup
Totalize
Unit Conversion

Positive30Minutes
Timestamp Offset Block
Out
In
Quantity 30
Time Unit Minutes

Result
Result Block
In

An updated algorithm, alg:SetpointLag30Minutes uses the **Timestamp Offset Block** to shift the timestamps.

Figure 5. Timestamp Offset Block chart showing the timestamp positive shift



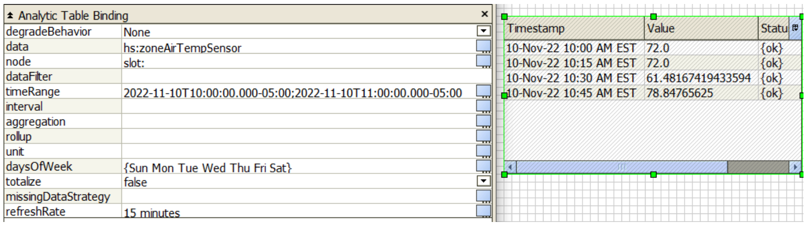
The positive 30 minute timestamp offset makes the process variable and setpoint converge at the same time (8:30 AM) instead of lagging the actual setpoint change (8:00 AM) by 30 minutes

Compared to the previous Analytic Web Chart, the orange line records are now shifted forward by 30 minutes.

When using a **Timestamp Offset Block** in an algorithm and the quantity is positive, the timestamps are shifted forward in time. The algorithm has logic that filters any records from the result whose timestamp is after the end of the time range in the request. The **Timestamp Offset Block** does not adjust the start and end times of the time range in the request, so it may seem like there are fewer than expected or even no records in the result.

Consider a Numeric Point for Zone Temp with hs:zoneAirTempSensor and a:a marker tags. A bound table with an **Analytic Table Binding** using data = hs:zoneAirTempSensor and a time range to show data between 10 am and 11 am on a single day displays the expected four records.

Figure 6. Timestamp Offset Block result showing four records



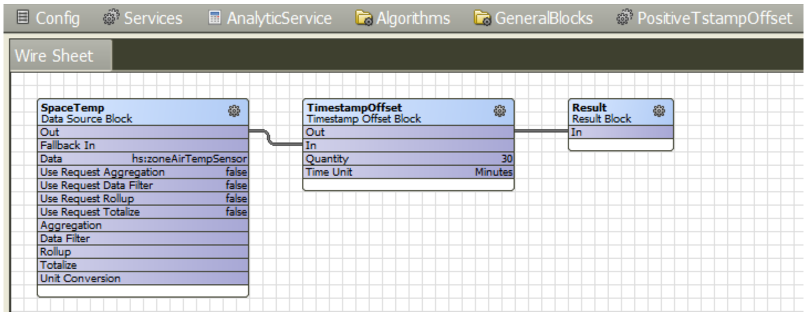
Analytic Table Binding

degradeBehavior	None
data	hs:zoneAirTempSensor
node	slot:
dataFilter	
timeRange	2022-11-10T10:00:00.000-05:00;2022-11-10T11:00:00.000-05:00
interval	
aggregation	
rollup	
unit	
daysOfWeek	{Sun Mon Tue Wed Thu Fri Sat}
totalize	false
missingDataStrategy	
refreshRate	15 minutes

Timestamp	Value	Statu
10-Nov-22 10:00 AM EST	72.0	{ok}
10-Nov-22 10:15 AM EST	72.0	{ok}
10-Nov-22 10:30 AM EST	61.48167419433594	{ok}
10-Nov-22 10:45 AM EST	78.84765625	{ok}

Consider an algorithm with a data source of hs:zoneAirTempSensor that is linked to a **Timestamp Offset Block** with a positive 30 minute offset.

Figure 7. Timestamp Offset Block algorithm with a positive 30–minute offset



Config | Services | AnalyticService | Algorithms | GeneralBlocks | PositiveTstampOffset

Wire Sheet

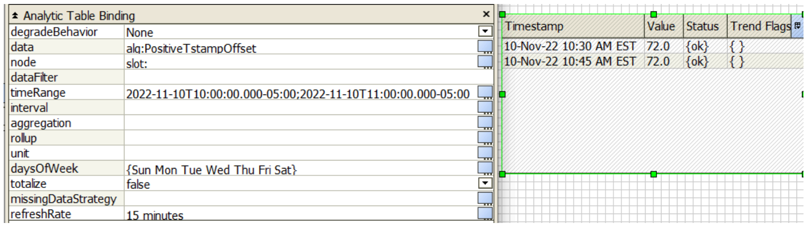
SpaceTemp
Data Source Block
- Out
- Fallback In
- Data: hs:zoneAirTempSensor
- Use Request Aggregation: false
- Use Request Data Filter: false
- Use Request Rollup: false
- Use Request Totalize: false
- Aggregation
- Data Filter
- Rollup
- Totalize
- Unit Conversion

TimestampOffset
Timestamp Offset Block
- Out
- In
- Quantity: 30
- Time Unit: Minutes

Result
Result Block
- In

A Bound Table with an **Analytic Table Binding** using data = alg:PositiveTstampOffset and the same time range to show data between 10 am and 11 am on a single day only returns two records.

Figure 8. Timestamp Offset Block example showing only two records



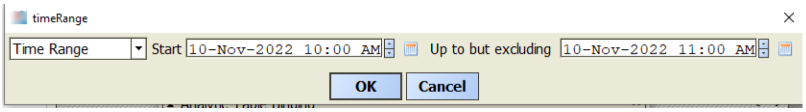
Analytic Table Binding

degradeBehavior	None
data	alg:PositiveTstampOffset
node	slot:
dataFilter	
timeRange	2022-11-10T10:00:00.000-05:00;2022-11-10T11:00:00.000-05:00
interval	
aggregation	
rollup	
unit	
daysOfWeek	{Sun Mon Tue Wed Thu Fri Sat}
totalize	false
missingDataStrategy	
refreshRate	15 minutes

Timestamp	Value	Status	Trend Flags
10-Nov-22 10:30 AM EST	72.0	{ok}	{ }
10-Nov-22 10:45 AM EST	72.0	{ok}	{ }

The other two records have timestamps of 11:00 AM and 11:15 AM, which are after the end time (exclusive) of the requested time range.

Figure 9. Time Range configuration

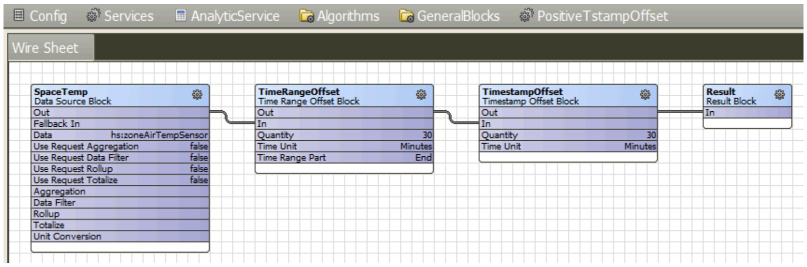


timeRange

Time Range | Start 10-Nov-2022 10:00 AM | Up to but excluding 10-Nov-2022 11:00 AM

If the **Timestamp Offset Block** was configured for positive 60 minutes, there would have been no records in the result. In this case you can add a **Time Range Offset Block** to the algorithm to adjust the end time of the requested time range in the future by the same amount of time that the Timestamp Offset block applies.

Figure 10. Timestamp Range Offset Block added to the algorithm



SpaceTemp Data Source Block

Out	
Fallback In	
Data	hs:zoneAirTempSensor
Use Request Aggregation	false
Use Request Data Filter	false
Use Request Rollup	false
Use Request Totalize	false
Aggregation	
Data Filter	
Rollup	
Totalize	
Unit Conversion	

TimeRangeOffset Time Range Offset Block

Out	
In	
Quantity	30
Time Unit	Minutes
Time Range Part	End

TimestampOffset Timestamp Offset Block

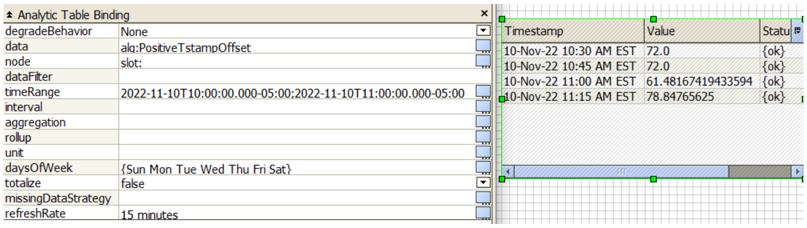
Out	
In	
Quantity	30
Time Unit	Minutes

Result Result Block

In

Using this updated alg:PositiveTstampOffset, the result displays all expected records with their adjusted timestamps.

Figure 11. Timestamp Offset Block example showing all expected records



Analytic Table Binding

degradeBehavior	None
data	alg:PositiveTstampOffset
node	slot:
dataFilter	
timeRange	2022-11-10T10:00:00.000-05:00;2022-11-10T11:00:00.000-05:00
interval	
aggregation	
rollup	
unit	
daysOfWeek	{Sun Mon Tue Wed Thu Fri Sat}
totalize	false
missingDataStrategy	
refreshRate	15 minutes

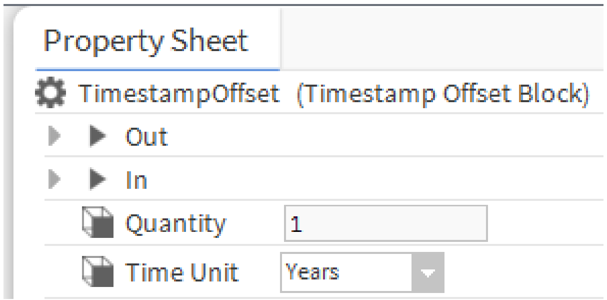
Timestamp	Value	Statu
10-Nov-22 10:30 AM EST	72.0	{ok}
10-Nov-22 10:45 AM EST	72.0	{ok}
10-Nov-22 11:00 AM EST	61.48167419433594	{ok}
10-Nov-22 11:15 AM EST	78.84765625	{ok}

Parent topic: General blocks

Timestamp offset (BTimestampOffsetBlock)

This block adjusts the timestamp of a history record as it is processed by the algorithm. It is often used in conjunction with a **Time Range Offset** block to perform period-over-period comparisons, such as month over month (MOM) or year over year (YOY).

Figure 1. Timestamp Offset properties



Property Sheet

TimestampOffset (Timestamp Offset Block)

- Out
- In
- Quantity: 1
- Time Unit: Years

To view these properties, double-click the block on the **Wire Sheet** or the block name in the Nav tree.

Property	Value	Description
Out	read-only value slot	Outputs the value from In with the timestamp offset based on the block's configuration.
In	required value slot	Links from the output of other logic blocks or data sources to supply trend data to the current logic block.
Quantity	number (defaults to 1)	Defines the numeric amount to offset. Use

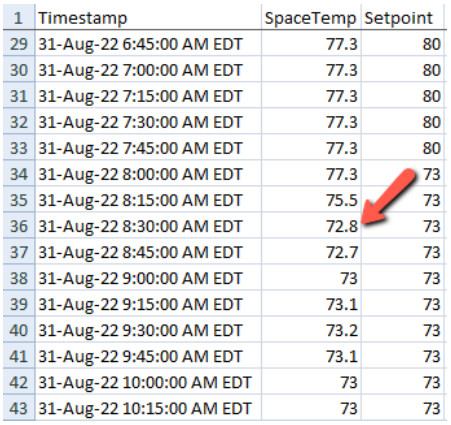
Property	Value	Description
		negative numbers to go back in time. In an algorithm, if Quantity is positive, this block shifts the timestamps forward in time and filters out any records from the algorithm's result whose timestamp is after the end of the time range in the request. This block does not adjust the start or end times of the time range in the request; therefore, it is possible to exclude the forward shifted records from the result. When using this block with a positive Quantity, it may be necessary to use a Time Range Offset block to offset only the End timestamp by the same Quantity and Time Unit, or to offset the Start timestamp by the negative quantity and same Time Unit.
Time Unit	drop-down list (defaults to Years)	Defines the unit to use for the offset from Years to Seconds.

Examples

Sometimes a time lag occurs between cause and effect. Changing a setpoint for a process does not have an immediate effect on the actual process variable. For example, changing a zone temperature setpoint does not immediately change the actual zone temperature, rather it might take 15 to 30 minutes for the zone temperature to reach the new setpoint. To facilitate easier comparison in these situations, you can use a **Timestamp Offset Block** to shift the timestamps for either the setpoint history or process variable history by the expected lag.

Consider these data:

Figure 2. Timestamp data

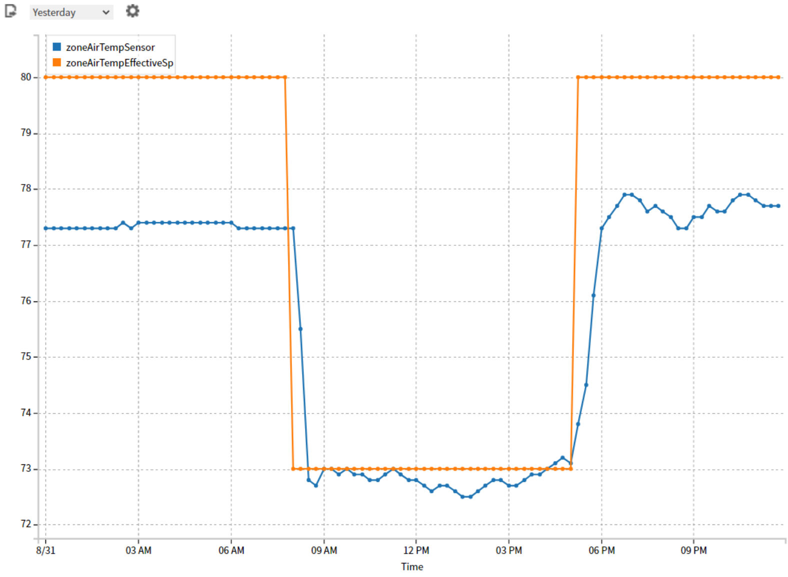


1	Timestamp	SpaceTemp	Setpoint
29	31-Aug-22 6:45:00 AM EDT	77.3	80
30	31-Aug-22 7:00:00 AM EDT	77.3	80
31	31-Aug-22 7:15:00 AM EDT	77.3	80
32	31-Aug-22 7:30:00 AM EDT	77.3	80
33	31-Aug-22 7:45:00 AM EDT	77.3	80
34	31-Aug-22 8:00:00 AM EDT	77.3	73
35	31-Aug-22 8:15:00 AM EDT	75.5	73
36	31-Aug-22 8:30:00 AM EDT	72.8	73
37	31-Aug-22 8:45:00 AM EDT	72.7	73
38	31-Aug-22 9:00:00 AM EDT	73	73
39	31-Aug-22 9:15:00 AM EDT	73.1	73
40	31-Aug-22 9:30:00 AM EDT	73.2	73
41	31-Aug-22 9:45:00 AM EDT	73.1	73
42	31-Aug-22 10:00:00 AM EDT	73	73
43	31-Aug-22 10:15:00 AM EDT	73	73

The table shows a setpoint change from 80 to 73 at 8:00 am. The actual space temp lags by about 30 minutes to reach the new setpoint.

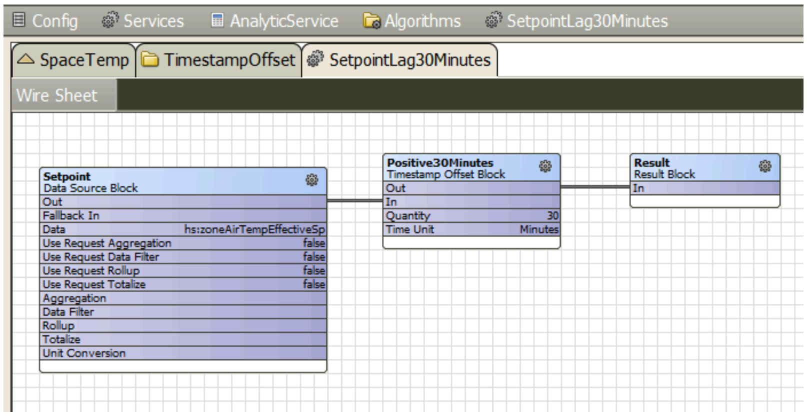
An Analytic Web Chart (`hs:zoneAirTempSensor` and `hs:zoneAirTempEffectiveSp`) shows the data with the lag.

Figure 3. Temperature data shown with a lag



This algorithm, which uses a **Timestamp Offset Block**, can shift the timestamps of the setpoint history forward by 30 minutes.

Figure 4. Timestamp Offset Block algorithm with a positive shift


Config | Services | AnalyticService | Algorithms | SetpointLag30Minutes

SpaceTemp | TimestampOffset | SetpointLag30Minutes

Wire Sheet

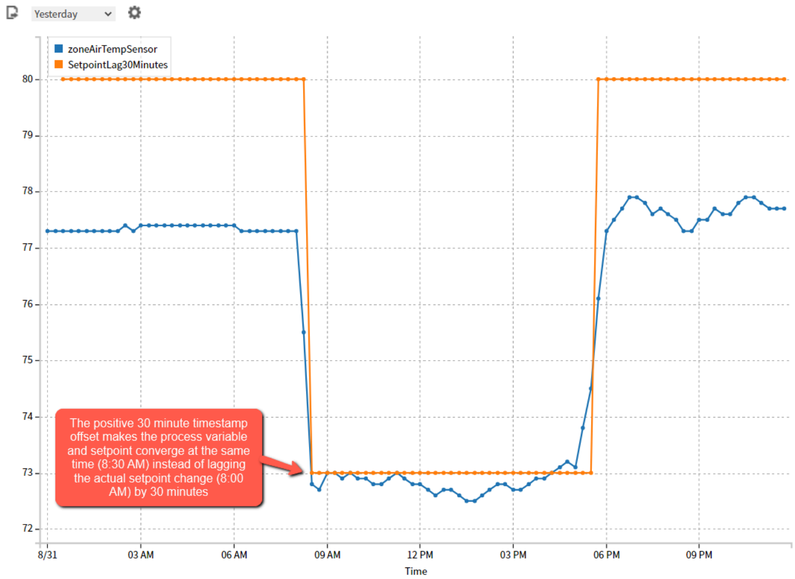
Setpoint
Data Source Block
Out
Fallback In
Data hs:zoneAirTempEffectiveSp
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Use Request Totalize false
Aggregation
Data Filter
Rollup
Totalize
Unit Conversion

Positive30Minutes
Timestamp Offset Block
Out
In
Quantity 30
Time Unit Minutes

Result
Result Block
In

An updated algorithm, alg:SetpointLag30Minutes uses the **Timestamp Offset Block** to shift the timestamps.

Figure 5. Timestamp Offset Block chart showing the timestamp positive shift



The positive 30 minute timestamp offset makes the process variable and setpoint converge at the same time (8:30 AM) instead of lagging the actual setpoint change (8:00 AM) by 30 minutes

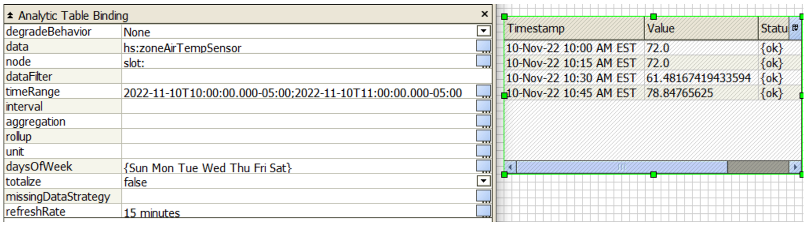
Compared to the previous Analytic Web Chart, the orange line records are now shifted forward by 30 minutes.

When using a **Timestamp Offset Block** in an algorithm and the quantity is positive, the timestamps are shifted forward in time. The algorithm has logic that filters any records from the result whose timestamp is after the end of the time range in the request. The **Timestamp Offset Block** does not adjust the start and end times of the time range in the request, so it may seem like there are fewer than expected or even no records in the result (the forward shifted records may be excluded from the result).

When using the **Timestamp Offset Block** with a positive Quantity it may be necessary to use a **Time Range Offset Block** to offset only the End timestamp by the same Quantity and Time Unit, or to offset the Start timestamp by the negative quantity and same Time Unit.

Consider a Numeric Point for Zone Temp with hs:zoneAirTempSensor and a:a marker tags. A bound table with an **Analytic Table Binding** using data = hs:zoneAirTempSensor and a time range to show data between 10 am and 11 am on a single day displays the expected four records.

Figure 6. Timestamp Offset Block result showing four records



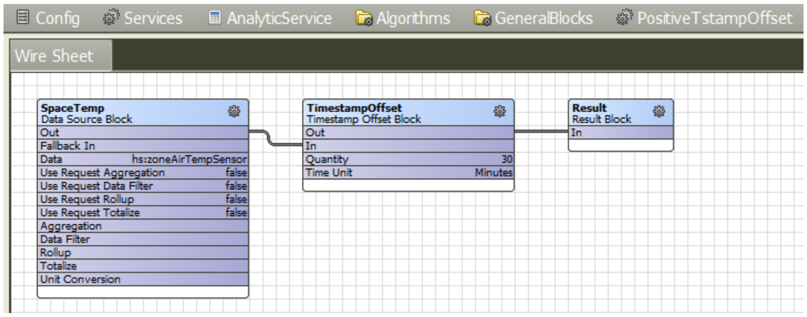
Analytic Table Binding

degradeBehavior	None
data	hs:zoneAirTempSensor
node	slot:
dataFilter	
timeRange	2022-11-10T10:00:00.000-05:00;2022-11-10T11:00:00.000-05:00
interval	
aggregation	
rollup	
unit	
daysOfWeek	{Sun Mon Tue Wed Thu Fri Sat}
totalize	false
missingDataStrategy	
refreshRate	15 minutes

Timestamp	Value	Statu
10-Nov-22 10:00 AM EST	72.0	{ok}
10-Nov-22 10:15 AM EST	72.0	{ok}
10-Nov-22 10:30 AM EST	61.48167419433594	{ok}
10-Nov-22 10:45 AM EST	78.84765625	{ok}

Consider an algorithm with a data source of hs:zoneAirTempSensor that is linked to a **Timestamp Offset Block** with a positive 30 minute offset.

Figure 7. Timestamp Offset Block algorithm with a positive 30–minute offset



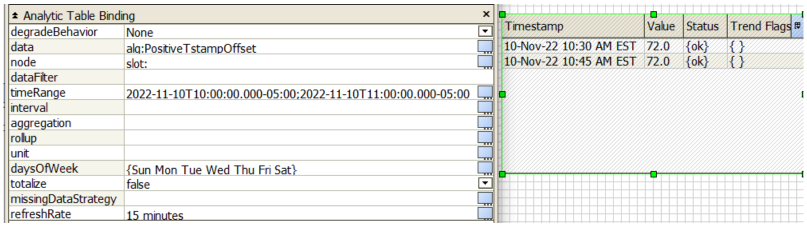
SpaceTemp
Data Source Block
Out
Fallback In
Data hs:zoneAirTempSensor
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Use Request Totalize false
Aggregation
Data Filter
Rollup
Totalize
Unit Conversion

TimestampOffset
Timestamp Offset Block
Out
In
Quantity 30
Time Unit Minutes

Result
Result Block
In

A Bound Table with an **Analytic Table Binding** using data = alg:PositiveTstampOffset and the same time range to show data between 10 am and 11 am on a single day only returns two records.

Figure 8. Timestamp Offset Block example showing only two records



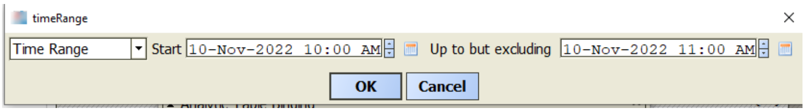
Analytic Table Binding

degradeBehavior	None
data	alg:PositiveTstampOffset
node	slot:
dataFilter	
timeRange	2022-11-10T10:00:00.000-05:00;2022-11-10T11:00:00.000-05:00
interval	
aggregation	
rollup	
unit	
daysOfWeek	{Sun Mon Tue Wed Thu Fri Sat}
totalize	false
missingDataStrategy	
refreshRate	15 minutes

Timestamp	Value	Status	Trend Flags
10-Nov-22 10:30 AM EST	72.0	{ok}	{ }
10-Nov-22 10:45 AM EST	72.0	{ok}	{ }

The other two records have timestamps of 11:00 AM and 11:15 AM, which are after the end time (exclusive) of the requested time range.

Figure 9. Time Range configuration

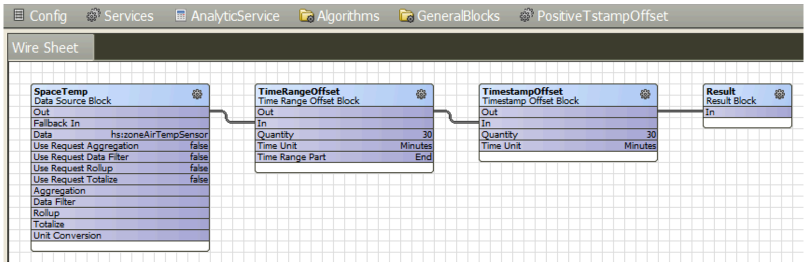


timeRange

Time Range Start 10-Nov-2022 10:00 AM Up to but excluding 10-Nov-2022 11:00 AM

If the **Timestamp Offset Block** was configured for positive 60 minutes, there would have been no records in the result. In this case you can add a **Time Range Offset Block** to the algorithm to adjust the end time of the requested time range in the future by the same amount of time that the Timestamp Offset block applies.

Figure 10. Timestamp Range Offset Block added to the algorithm



SpaceTemp
Data Source Block
Out
Fallback In
Data hs:zoneAirTempSensor
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Use Request Totalize false
Aggregation
Data Filter
Rollup
Totalize
Unit Conversion

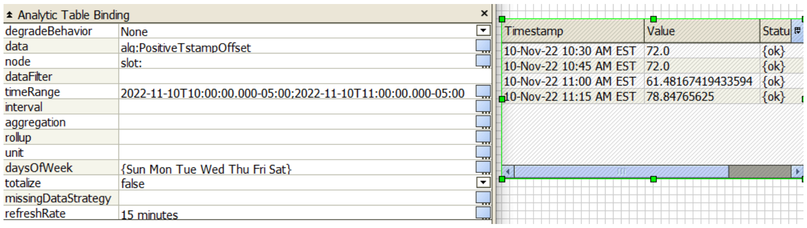
TimeRangeOffset
Time Range Offset Block
Out
In
Quantity 30
Time Unit Minutes
Time Range Part End

TimestampOffset
Timestamp Offset Block
Out
In
Quantity 30
Time Unit Minutes

Result
Result Block
In

Using this updated alg:PositiveTstampOffset, the result displays all expected records with their adjusted timestamps.

Figure 11. Timestamp Offset Block example showing all expected records



Analytic Table Binding

degradeBehavior	None
data	alg:PositiveTstampOffset
node	slot:
dataFilter	
timeRange	2022-11-10T10:00:00.000-05:00;2022-11-10T11:00:00.000-05:00
interval	
aggregation	
rollup	
unit	
daysOfWeek	{Sun Mon Tue Wed Thu Fri Sat}
totalize	false
missingDataStrategy	
refreshRate	15 minutes

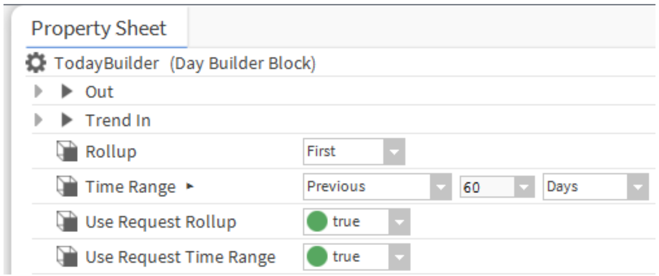
Timestamp	Value	Statu
10-Nov-22 10:30 AM EST	72.0	{ok}
10-Nov-22 10:45 AM EST	72.0	{ok}
10-Nov-22 11:00 AM EST	61.48167419433594	{ok}
10-Nov-22 11:15 AM EST	78.84765625	{ok}

Parent topic: General blocks

Today Builder (BDayBuilderBlock)

This block generates a trend that uses past historical data to represent "today." The system converts timestamps by changing the date to today while preserving the time of day. When there are multiple values for the same timestamp, the rollup determines how to combine them. This block supports both trend and value requests.

For value requests the block processes a trend request using the block's Time Range property and returns a single value. The block uses the function defined by the Rollup property defined in the request, else it uses the function defined by the block's Rollup property.
Figure 1. Today Builder properties



Property Sheet

TodayBuilder (Day Builder Block)

Out	
Trend In	
Rollup	First
Time Range	Previous 60 Days
Use Request Rollup	● true
Use Request Time Range	● true

To view these properties, double-click the block on the **Wire Sheet** or the block name in the Nav tree.

Property	Value	Description
Out	read-only value slot	Outputs a trend or value calculated from the In value.
Trend In	required value slot	Links from the output of other logic blocks or data sources to supply trend data to the current logic block.
Rollup	check box (if optional, and) drop-down list or ORD parameter (when configured in the data definition, defaults to First, when configured elsewhere, defaults to the value as defined in the Data Definition); rollup=option	Configures the rollup function used by the block if the block's Use Request Rollup property is set to false or the Rollup property is not explicitly configured in the request. If rollup is not enabled in the binding/ settings window, the rollup value configured in the Data Definition applies to all chart bindings, reports and tables. And returns the logical "and" of Boolean values. Avg returns the statistical mean, which is determined by calculating the sum of all values and dividing by the number of values.

Property	Value	Description
		Count returns the total number or quantity of values in a combination. If you request this value on a binding in a PX view, the system counts the number of values based on the properties defined by the data source block and the algorithm's **Property Sheet**. First returns the first value in the combination. Last returns the last value in the combination. Max returns the highest value in the combination. Median returns the value in the middle of a sorted combination—the number that separates the higher half from the lower half. Min returns the lowest value in the combination. Mode returns the statistically most frequently occurring number in the combination. Or returns the logical "or" of Boolean values. Range returns the statistical difference between the largest and smallest values in the combination. Sum adds together all values in the combination resulting in a single value. Std Dev calculates the standard deviation of the values in the combination. Load Factor calculates the average divided by peak (Max) value.
Time Range	range of options (defaults to Previous 60 Days)	Defines the time period over which to combine the data in a rollup. This property is required for rollup requests (analyticRollup), trends (analyticTrend), and rollup bindings. It is optional elsewhere. It is not used if the block's Use Request Time Range property is true and the request specifies a time range. Options range from From to All.
Use Request Rollup	true (default) or false	Determines the rollup function to use. true causes the block to use the function defined by the incoming request. If the Rollup property is not defined in the request, the block uses the function defined by its Rollup property. false causes the block to use the function defined by its Rollup property.

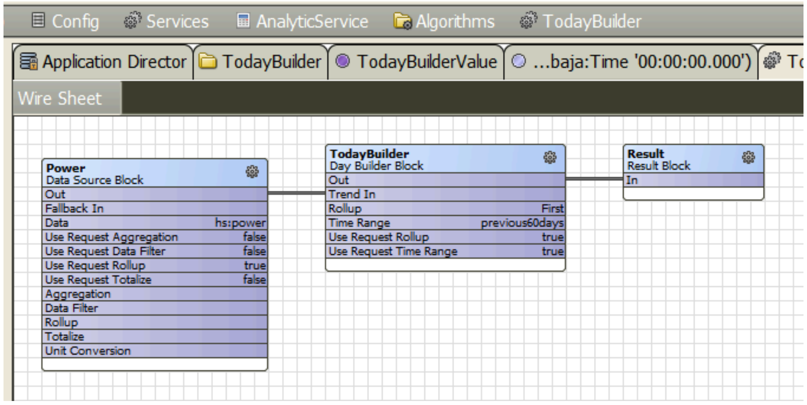
Property	Value	Description
Use Request Time Range	true (default) or false	Determines the time range for rolling up data. true causes the block to use the time range defined by the incoming request. Value requests always use the block’s Time Range property. false causes the block to use the time range defined its Time Range property.

Example

This block is similar in function to using a combination of a **Time Range Offset Block** and a **Timestamp Offset Block**. The **Today Builder Block** always requests a time range of historical data and modifies the timestamps by changing the date to today while preserving the time of day. If the requested time range spans more than one day, the block rolls up the records with matching times of day.

For a trend request, the default interval is 15 minutes, which causes the algorithm to return the requested historical data with 15-minute interval timestamps that show today’s date. This block is most useful in an algorithm that compares today’s values vs. yesterday’s values, or today’s values vs. the values one week ago, where some historical data prior to today’s data can be compared against today’s data.

Figure 2. Day Builder Block trend request algorithm



Power
Data Source Block

Out	
Fallback In	
Data	hs:power
Use Request Aggregation	false
Use Request Data Filter	false
Use Request Rollup	true
Use Request Totalize	false
Aggregation	
Data Filter	
Rollup	
Totalize	
Unit Conversion	

TodayBuilder
Day Builder Block

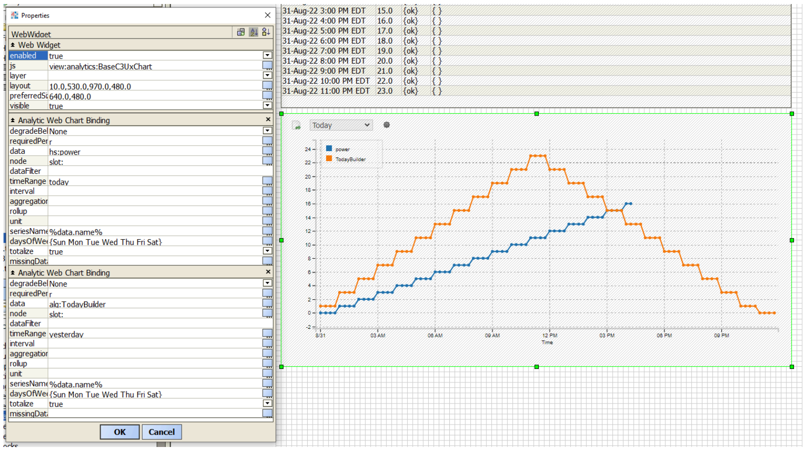
Out	
Trend In	
Rollup	First
Time Range	previous60days
Use Request Rollup	true
Use Request Time Range	true

Result
Result Block

In	

This algorithm produces the following chart.

Figure 3. Day Builder Block chart



Properties

WebWidget

Web Widget

enabled	true
js	view:analytics:BaseC3UxChart
layer	
layout	10.0,530.0,970.0,480.0
preferredSi	640.0,480.0
visible	true

Analytic Web Chart Binding

degradeBel	None
requiredPer	r
data	hs:power
node	slot:
dataFilter	
timeRange	today
interval	
aggregatior	
rollup	
unit	
seriesName	%data.name%
daysOfWee	{Sun Mon Tue Wed Thu Fri Sat}
totalize	true
missingDat	

Analytic Web Chart Binding

degradeBel	None
requiredPer	r
data	alg:TodayBuilder
node	slot:
dataFilter	
timeRange	yesterday
interval	
aggregatior	
rollup	
unit	
seriesName	%data.name%
daysOfWee	{Sun Mon Tue Wed Thu Fri Sat}
totalize	true
missingDat	

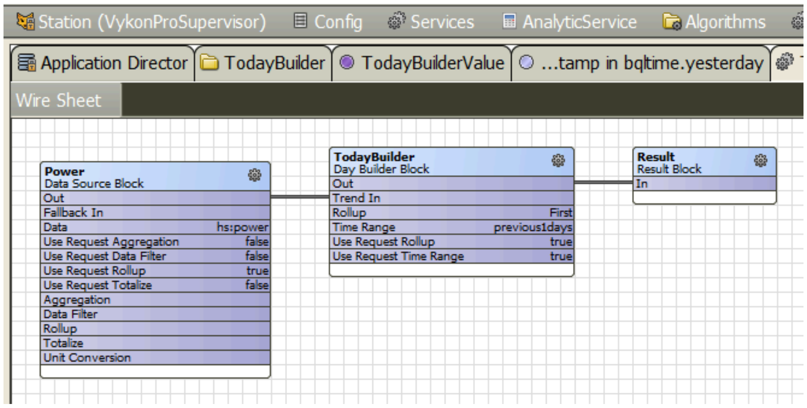
OK Cancel

31-Aug-22 3:00 PM EDT	15.0	{ok}	{ }
31-Aug-22 4:00 PM EDT	16.0	{ok}	{ }
31-Aug-22 5:00 PM EDT	17.0	{ok}	{ }
31-Aug-22 6:00 PM EDT	18.0	{ok}	{ }
31-Aug-22 7:00 PM EDT	19.0	{ok}	{ }
31-Aug-22 8:00 PM EDT	20.0	{ok}	{ }
31-Aug-22 9:00 PM EDT	21.0	{ok}	{ }
31-Aug-22 10:00 PM EDT	22.0	{ok}	{ }
31-Aug-22 11:00 PM EDT	23.0	{ok}	{ }

For a value request, the algorithm only returns a single value by rolling up the data from the requested time range using the configured rollup function.

Note that for a value request, the **Today Builder Block** uses the value of its Time Range property regardless of the value of its Use Request Time Range property because a value request does not include a Time Range. A **Today Builder block** may use the rollup function from the value request if the its Use Request Rollup property is true (the default setting).

Figure 4. Day Builder Block value request algorithm



Power
Data Source Block

Out	
Fallback In	
Data	hs:power
Use Request Aggregation	false
Use Request Data Filter	false
Use Request Rollup	true
Use Request Totalize	false
Aggregation	
Data Filter	
Rollup	
Totalize	
Unit Conversion	

TodayBuilder
Day Builder Block

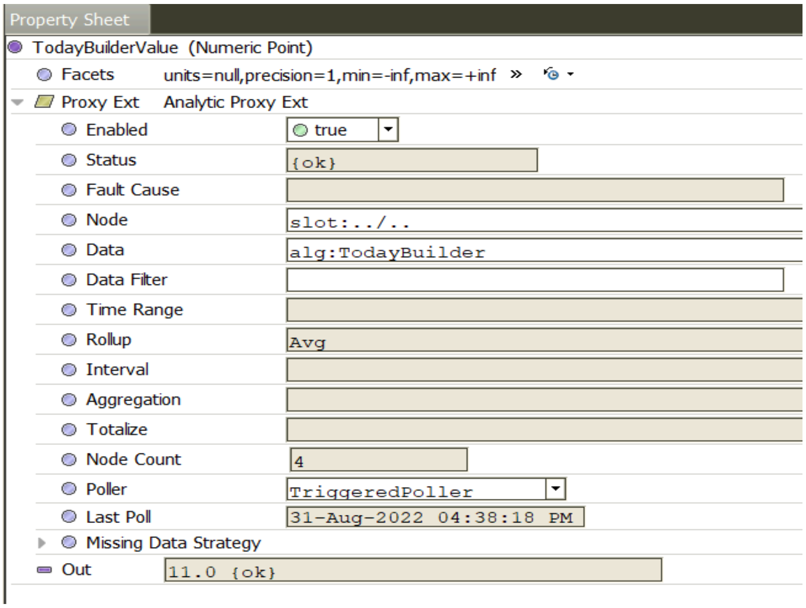
Out	
Trend In	
Rollup	First
Time Range	previous1days
Use Request Rollup	true
Use Request Time Range	true

Result
Result Block

In	

The following screen capture shows the configuration of the numeric point.

Figure 5. Numeric point configuration properties



Property Sheet

TodayBuilderValue (Numeric Point)

Facets	units=null,precision=1,min=-inf,max=+inf
Proxy Ext	Analytic Proxy Ext
Enabled	true
Status	{ok}
Fault Cause	
Node	slot:../..
Data	alg:TodayBuilder
Data Filter	
Time Range	
Rollup	Avg
Interval	
Aggregation	
Totalize	
Node Count	4
Poller	TriggeredPoller
Last Poll	31-Aug-2022 04:38:18 PM
Missing Data Strategy	
Out	11.0 {ok}

Processing with Use Request Rollup

When Use Request Rollup is false and Rollup is not configured, the block uses the rollup function defined in the applicable Data Definition. If there is no Data Definition, the block uses the default First function. When Use Request Rollup is false and Rollup property is configured, the block uses the Rollup property value. When Use Request Rollup is true, the block uses the rollup function defined in the request (Analytic Proxy Ext, Analytic Binding, etc.), unless the request does not specify the rollup function in which case the block uses the Algorithm's Rollup property value.

The following table may help to determine when each value applies.

Data Definition Rollup	Data Source Block Use Request Rollup	Data Source Block Rollup	Request Rollup	Algorithm Rollup	Actual Rollup Function
Unconfigured	False	Unconfigured	Sum	Max	Defaults to First
Last	False	Unconfigured	Sum	Max	Last
Last	False	**Avg**	Sum	Max	Avg
Last	True	Avg	**Sum**	Max	Sum
Last	True	Avg	Unconfigured	**Max**	Max

Parent topic: General blocks

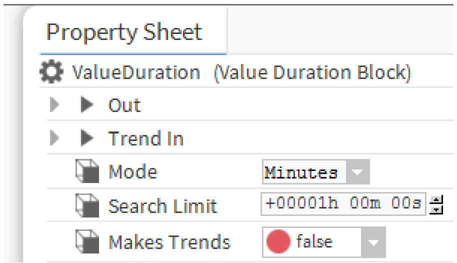
Value duration (BValueDurationBlock)

This block determines how long the input has been at the current value. Both value and trend requests are supported.

A trend request results in values that reflect the duration of the current value within the trend. A trend request typically yields a sawtooth output, where the accumulated duration value resets to zero and each value in the trend data changes.

For a value request, the Search Limit property (defaults to 1 hour) defines the time range for the block to process a trend request and returns a single value that reflects the duration of the current value in the trend.

Figure 1. Value Duration properties



Property Sheet

ValueDuration (Value Duration Block)

Out

Trend In

Mode Minutes

Search Limit +00001h 00m 00s

Makes Trends false

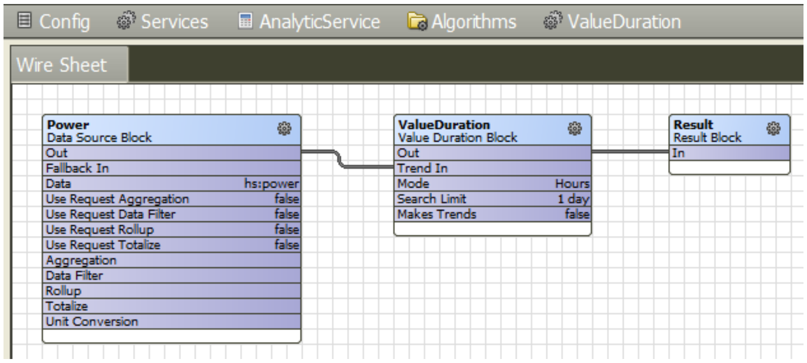
To view these properties, double-click the block on the **Wire Sheet** or the block name in the Nav tree.

Property	Value	Description
Out	read-only value slot	Outputs a constant value of constant trend from the Trend In property based on the value of the Make Trends or Makes Trends property.
Trend In	required value slot	Links from the output of other logic blocks or data sources to supply trend data to the current logic block.
Mode	drop-down list defaults to Minutes (defaults to 1 hour)	Configures what time unit (milliseconds, seconds, minutes or hours) to apply to the accumulated values.
Search Limit	hours minutes seconds miliseconds (defaults to 1 hour)	Defines how far back in time to search. The system subtracts this value from the current time to create a time range for value requests.
Makes Trends	true or false (default)	Controls the processing of trend requests. false outputs a constant value for both value and trend requests. true outputs a trend with constant values for trend requests.

Example

For a value request, the Search Limit property (defaults to 1 hour) defines the time range of data to query and the Mode property (defaults to Minutes) defines the time interval for the result. Consider an algorithm with a **Value Duration Block** configured with Mode = Hours and Search Limit = 24 hours.

Figure 2. Value Duration Block algorithm



Config › Services › AnalyticService › Algorithms › ValueDuration

Wire Sheet

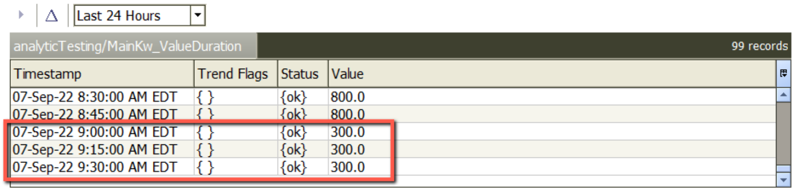
Power
Data Source Block
Out
Fallback In
Data: hs:power
Use Request Aggregation: false
Use Request Data Filter: false
Use Request Rollup: false
Use Request Totalize: false
Aggregation
Data Filter
Rollup
Totalize
Unit Conversion

ValueDuration
Value Duration Block
Out
Trend In
Mode: Hours
Search Limit: 1 day
Makes Trends: false

Result
Result Block
In

The history for the hs:power data source shows the value has been 300 since 9:00 AM.

Figure 3. Resulting data



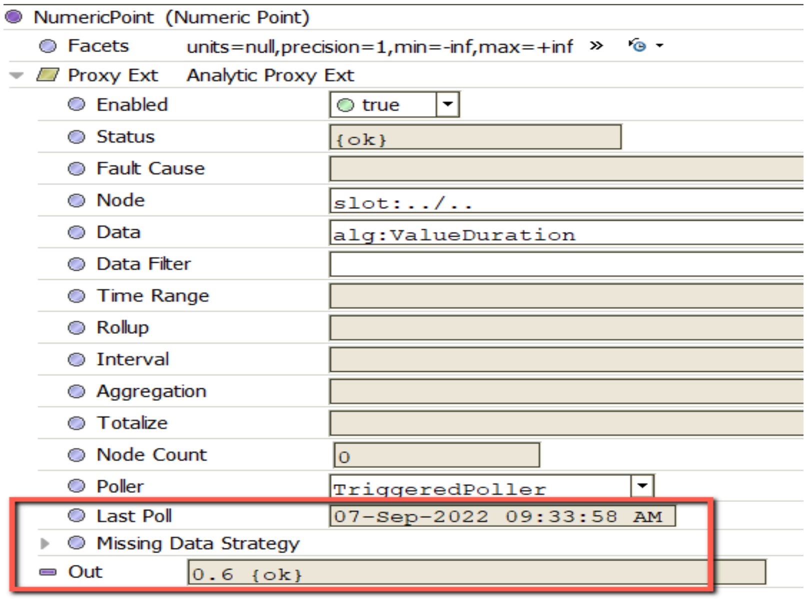
Last 24 Hours

analyticTesting/MainKw_ValueDuration — 99 records

Timestamp	Trend Flags	Status	Value
07-Sep-22 8:30:00 AM EDT	{ }	{ok}	800.0
07-Sep-22 8:45:00 AM EDT	{ }	{ok}	800.0
07-Sep-22 9:00:00 AM EDT	{ }	{ok}	300.0
07-Sep-22 9:15:00 AM EDT	{ }	{ok}	300.0
07-Sep-22 9:30:00 AM EDT	{ }	{ok}	300.0

A **Numeric Point** for an analytic proxy extension configured with Data = alg:ValueDuration polled at 9:33 AM returns a result of 0.6 hours. The value of 300 has been the same since 9:00 AM; 33 minutes converted to hours is 0.6 hours.

Figure 4. Numeric Point configuration

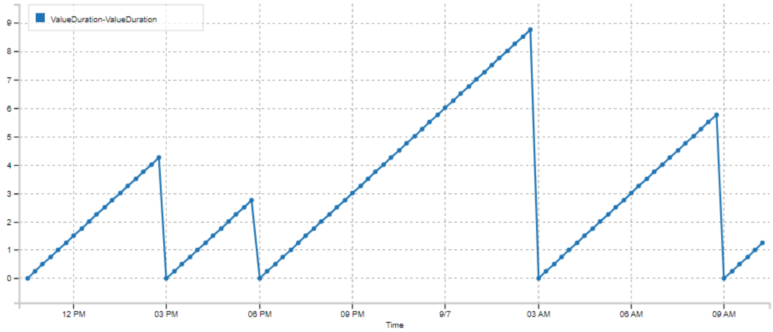


NumericPoint (Numeric Point)

Facets	units=null,precision=1,min=-inf,max=+inf
Proxy Ext	Analytic Proxy Ext
Enabled	true
Status	{ok}
Fault Cause	
Node	slot:../..
Data	alg:ValueDuration
Data Filter	
Time Range	
Rollup	
Interval	
Aggregation	
Totalize	
Node Count	0
Poller	TriggeredPoller
Last Poll	07-Sep-2022 09:33:58 AM
Missing Data Strategy	
Out	0.6 {ok}

Analytic charts and tables display the accumulated value duration times instead of the raw values from the data source history.

Figure 5. Chart showing the accumulated value



If the **Value Duration Block** needs to pass the results as a trend via the Out slot linked to a downstream block, change the Makes Trends property from its default, false, to true.

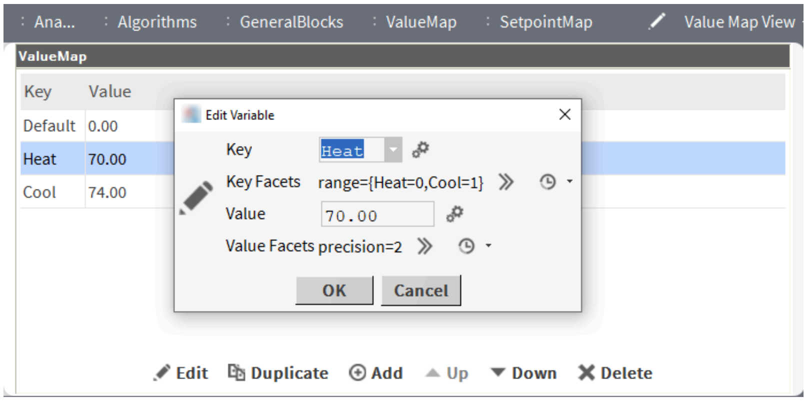
Parent topic: General blocks

Value Map (BValueMapBlock)

This block represents a map of key value pairs, which define substitution values where you configure the data type (Numeric, String, Boolean, Enum or Other) for both the key and value objects. There are frozen properties associated with the default key value pair and dynamic BValueMapEntry properties for any user defined objects. This block supports both trend and value requests, and retains the timestamp and status of input values.

The **Value Map View** (default view) on this block displays the current map and allows adding, editing, duplicating, deleting and reordering dynamic objects in the map. For requests where the In value does not match the Key property of any of the map entries, the Unmapped Mode property configures what, if any, value is returned.

Figure 1. Value Map view



Key	Value
Default	0.00
Heat	70.00
Cool	74.00

Key: Heat

Key Facets: range={Heat=0,Cool=1}

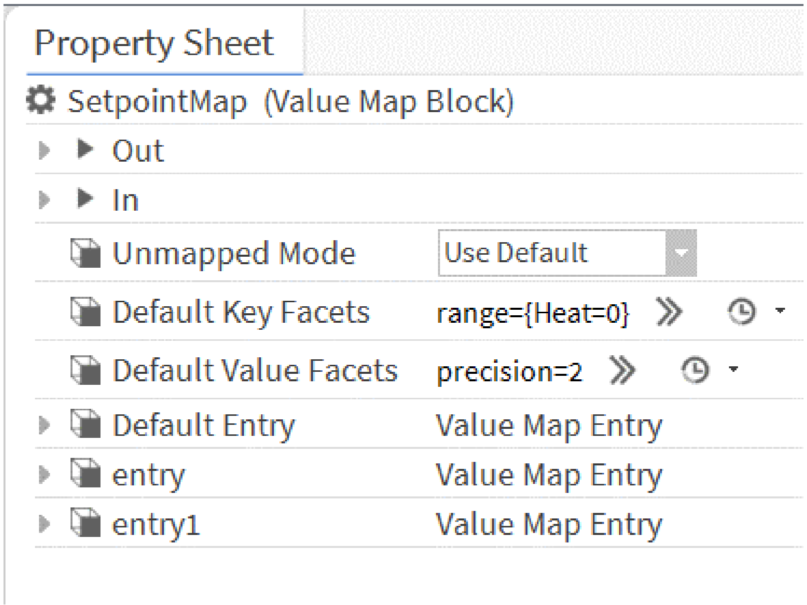
Value: 70.00

Value Facets: precision=2

Table 1. Columns

Column	Description
Key	Displays the data type for the key: Numeric, String, Boolean, Enum, or Other.
Value	Displays the data type for the value: Numeric, String, Boolean, Enum, or Other

Figure 2. Value Map properties



Property Sheet

SetpointMap (Value Map Block)

Out	
In	
Unmapped Mode	Use Default
Default Key Facets	range={Heat=0}
Default Value Facets	precision=2
Default Entry	Value Map Entry
entry	Value Map Entry
entry1	Value Map Entry

The default key facets and default value facets in this screen capture have been changed from the defaults, and there are entry and entry1 dynamic properties that correspond to the **Value Map View** above.

To view these properties, double-click the block on the **Wire Sheet** or the block name in the Nav tree.

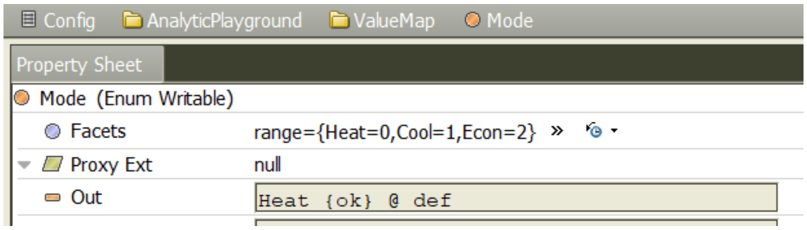
Property	Value	Description
Out	read-only value slot	Outputs a mapped value by evaluating the In value against the entryX items in the map.
In	required value slot	Links from the output of other logic blocks or data sources to supply trend data to the current logic block.
Unmapped Mode	drop-down list	Configures the behavior when the In value does not match the Key property of any entryX item in the map. Pass Through uses the value of the data source. Use Default uses the value from the Default Entry in the map. Filter Out for a value request returns the data source value with a null status. For a trend request, excludes the record from the result.
Default Key Facets	BFacets (defaults to blank range and precision = 2)	Applies facets to the key property for any user defined map entry. The actual facets for each map entry key may be customized.
Default Value Facets	BFacets (defaults to blank range and precision = 2)	Applies facets to the value property for any user defined map entry. The actual facets for each map entry value may be customized.

Property	Value	Description
Default Entry	Frozen property (defaults to key type String, key = “Default”, value type Numeric, value = 0.00)	Defines the value to use when Unmapped Mode is set to Use Default and the In value being evaluated does not match any of the entryX property keys. You cannot delete this property value. The Key must remain “Default”, but you can edit the Value (property and type) and the Value Facets as needed.
entryX where X is a sequential number for each user defined map entry	Value Map Entry	Sets up dynamic user defined map entries. When creating a new entry, the Default Key Facets property sets the Key Facets and Value Facets. You may customize these facets. The block evaluates the In value against the Key property for each entryX item in the map to determine if the In value matches the block outputs of the Value property from the entryX item with the matching key.

Example

Consider a BEnumWritable point with an Enum Range as shown in the capture, and a:a and b:HtgClgMode marker tags applied.

Figure 3. Mode property configured on an Enum Writable point



Mode (Enum Writable)

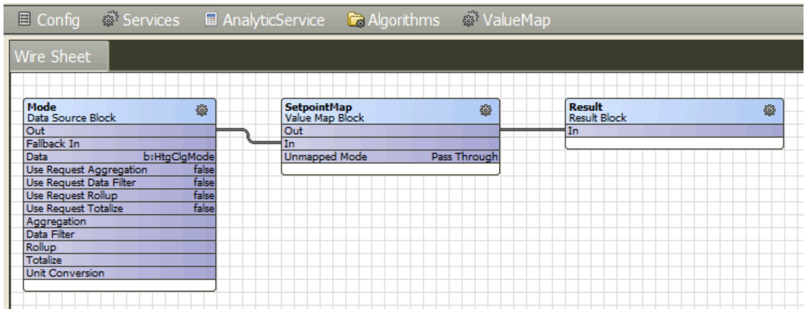
Facets range={Heat=0,Cool=1,Econ=2}

Proxy Ext null

Out Heat {ok} @ def

To determine a setpoint for some fault detection logic, you could use an algorithm with a Value Map Block based on the value of the Mode point.

Figure 4. Value Map Block algorithm



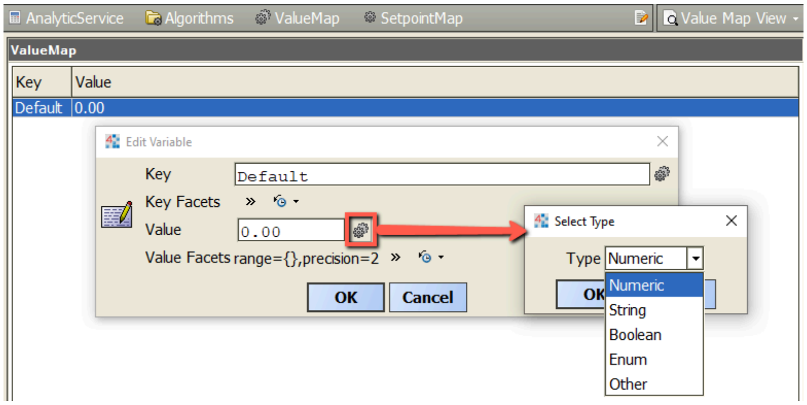
Mode
Data Source Block
Out
Fallback In
Data b:HtgClgMode
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Use Request Totalize false
Aggregation
Data Filter
Rollup
Totalize
Unit Conversion

SetpointMap
Value Map Block
Out
In
Unmapped Mode Pass Through

Result
Result Block
In

You use the **Value Map View** to configure the object type and value for the default map entry.

Figure 5. Edit Variable window



Key	Value
Default	0.00

Key Default

Key Facets

Value 0.00

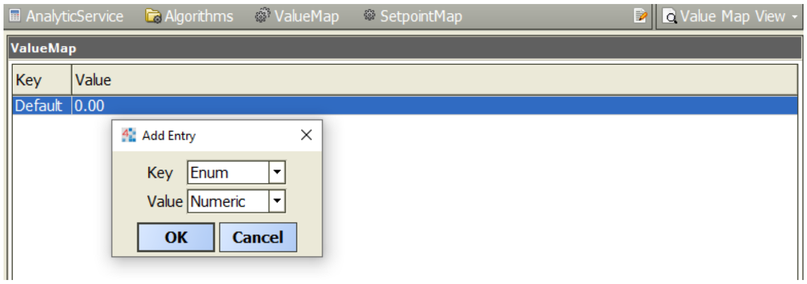
Value Facets range={},precision=2

Type Numeric

Clicking the gear icon on the right side of the Value property opens a **Select Type** field editor for configuring the object type (Numeric, String, Boolean, Enum or Other). Selecting the type makes it possible to configure the object's value.

The **Value Map View** has controls at the bottom of the view to edit, duplicate, add, delete and reorder map entries.

Figure 6. Add Entry window



Key	Value
Default	0.00

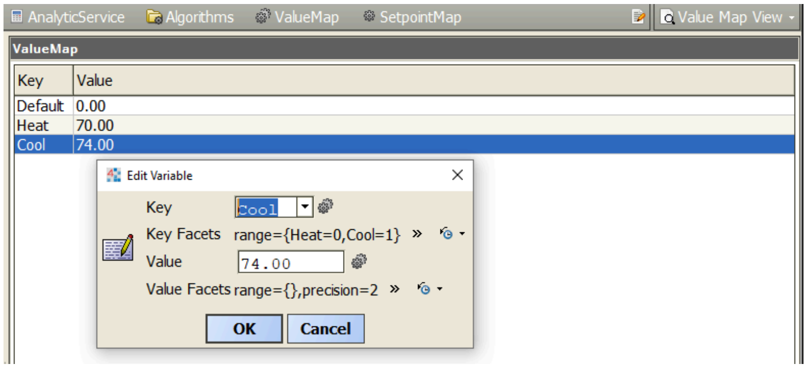
Key Enum

Value Numeric

To create a new map entry, click the **Add** button, then specify the desired object type for both the Key and Value. In this case, the data source is b:HtgClgMode, which is an Enum Writable point. The Key type should be Enum. The desired value is a number, so the Value type should be Numeric.

The framework creates the map entry with default values for the configured types.

Figure 7. Edit Variable window



Key	Value
Default	0.00
Heat	70.00
Cool	74.00

Key Cool

Key Facets range={Heat=0,Cool=1}

Value 74.00

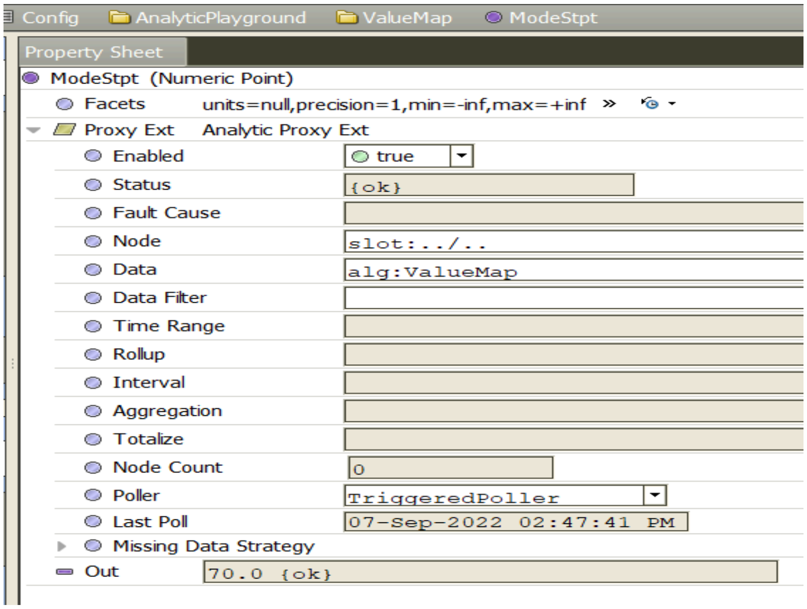
Value Facets range={},precision=2

To launch the **Edit Variable** window for configuring Key and Value object type, facets and values, double click the map entry or select the map entry and click the **Edit** button.

Typically, the Key facets should match the point facets for the data source. In this case, the facets are purposefully set without the ordinal = 2 and tag = Econ items in the range.

The **Mode** point's value is currently ordinal = 0 and tag = Heat. A numeric point with an analytic proxy extension named ModeStpt generates an analytic value request for the alg:ValueMap.

Figure 8. ModeStpt configuration



ModeStpt (Numeric Point)

Facets units=null,precision=1,min=-inf,max=+inf

Proxy Ext Analytic Proxy Ext

Enabled true

Status {ok}

Fault Cause

Node slot:../..

Data alg:ValueMap

Data Filter

Time Range

Rollup

Interval

Aggregation

Totalize

Node Count 0

Poller TriggeredPoller

Last Poll 07-Sep-2022 02:47:41 PM

Missing Data Strategy

Out 70.0 {ok}

Since the Enum tag value is Heat, the **Value Map Block** returns the numeric value of `70.0` from the matching key value pair in the map.

You can configure the **Value Map Block**'s Unmapped Mode property as either Pass Through, Use Default or Filter Out. When the data source value does not match any of the keys in the map, the block uses the value of the Unmapped Mode property to determine what value to return.

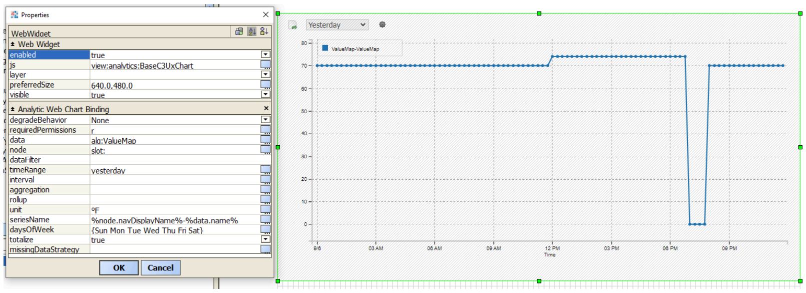
The following table summarizes the result for each request type based on the value of Mode.

Table 2. Mode results

Mode	Condition	Value Request	Trend Request
Pass Through	Value does not match any key	The value of the data source	The value of the data source
Use Default	Value does not match any key	The value from the default key in the map	The value from the default key in the map
Filter Out	Value does not match any key	a BValue with a null status	Excluded from the result

A trend request used with an Analytic Chart or Table returns results for each timestamp.

Figure 9. Value Map Block chart



Properties

WebWidget

Web Widget

enabled	true
js	view:analytics:BaseC3UxChart
layer	
preferredSize	640.0,480.0
visible	true

Analytic Web Chart Binding

degradeBehavior	None
requiredPermissions	r
data	alg:ValueMap
node	slot:
dataFilter	
timeRange	yesterday
interval	
aggregation	
rollup	
unit	°F
seriesName	%node.navDisplayName%-%data.name%
daysOfWeek	{Sun Mon Tue Wed Thu Fri Sat}
totalize	true
missingDataStrategy	

Parent topic: General blocks

Math blocks

These components provide basic mathematical expressions for use in the framework formulas.

This table summarizes how the math blocks impact value and trend requests.

Block	Value Request	Trend Request
Bi Math	Yes	Yes
Uni Math	Yes	Yes

- **BiMath (BBiMathBlock)**
  This component provides a two-operand math expression. This block requires both operands (inputs) and supports both value and trend requests.
- **Uni math (BUniMathBlock)**
  This block provides a single-operand math expression that corresponds to the functions in java.lang.Math. It supports both trend and value requests. If the expression results in NaN or Infinity, the block sets the null status flag.

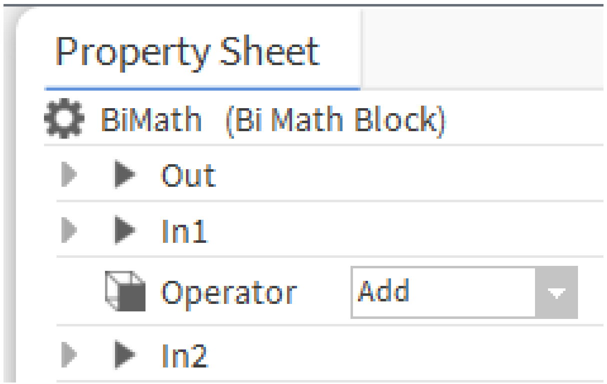
Parent topic: Logic blocks

BiMath (BBiMathBlock)

This component provides a two-operand math expression. This block requires both operands (inputs) and supports both value and trend requests.

For trend requests, at least one of the operands must be a trend.

Figure 1. BiMath properties



Property Sheet

BiMath (Bi Math Block)

Out

In1

Operator: Add

In2

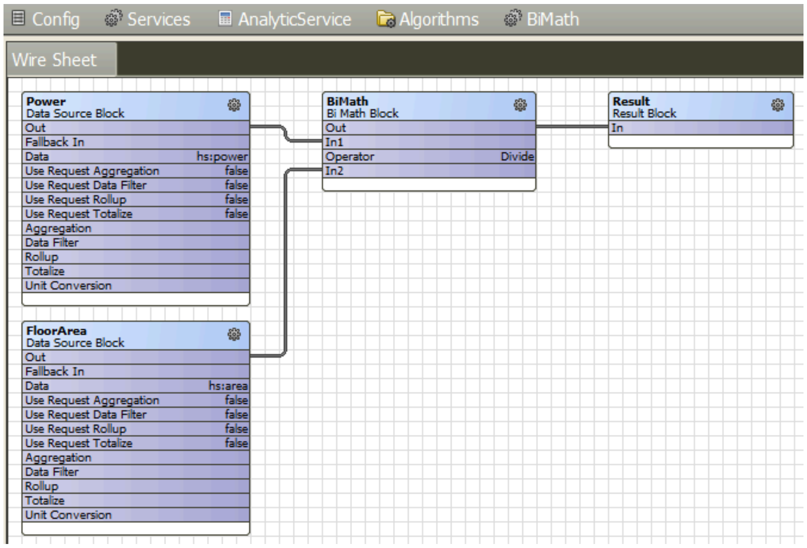
To view these properties, double-click the block on the **Wire Sheet** or the block name in the Nav tree.

Property	Value	Description
Out	read-only value slot	Outputs the result of the two inputs using the configured function.
In 1	operand	Defines the first operand used in the expression. Links from the output of other logic blocks or data sources to supply one or more input values to another logic block.
Operator	drop-down list (defaults to Add)	Defines the function to perform. Add sums In1 and In2. Subtract calculates the difference between In1 and In2. Multiply calculates the product of In1 and In2. Divide divides In1 (dividend) by In2 (divisor) Modulo Outputs the remainder after dividing In1 by In2. Exponent raises In1 (base) to the power of In2 (exponent). Min calculates the minimum value between In1 and In2. Max calculates the maximum value between In1 and In2. Avg calculates the average of In1 and In2.
In 2	operand	Defines the second operand used in the expression. Links from the output of other logic blocks or data sources to supply one or more input values to a logic block.

Example

Consider a numeric point with a history for electrical power identified by `hs:power` and `a:a` marker tags, and a second numeric point for floor area with a value of 12,500 identified by `hs:area = 12,500` and `a:a` marker tags. An algorithm with a **Bi Math Block** may calculate power density (kw/ft^2) by dividing the `hs:power` data source by the hs:area data source.

Figure 2. Bi Math algorithm



Power
Data Source Block
Out
Fallback In
Data hs:power
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Use Request Totalize false
Aggregation
Data Filter
Rollup
Totalize
Unit Conversion

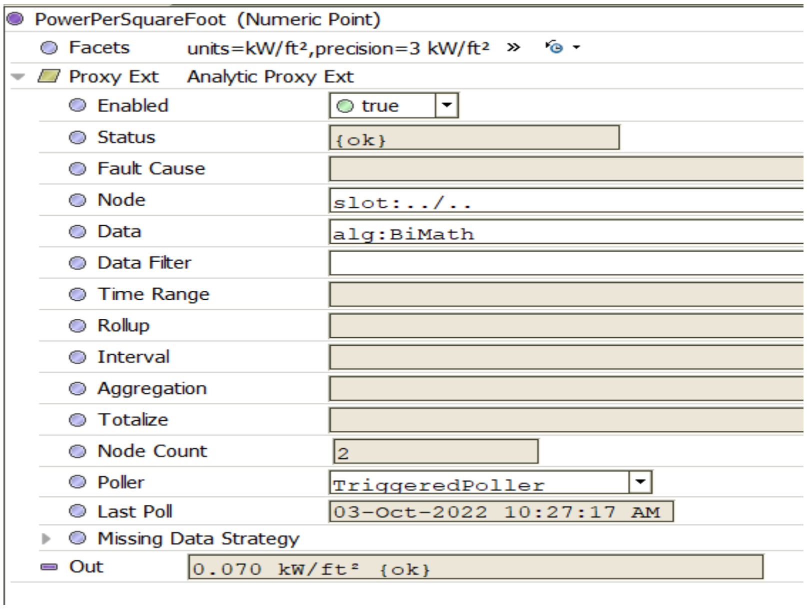
FloorArea
Data Source Block
Out
Fallback In
Data hs:area
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Use Request Totalize false
Aggregation
Data Filter
Rollup
Totalize
Unit Conversion

BiMath
Bi Math Block
Out
In1
Operator Divide
In2

Result
Result Block
In

A control point with an Analytic Proxy Ext configured to use the alg:BiMath algorithm displays the calculated power density value. Power value = 857.5 KW, Floor Area = 12,500 ft^2: 857.5 / 12,500 = 0.07 KW/ft^2. The result displays in the point's Out slot.

Figure 3. Result



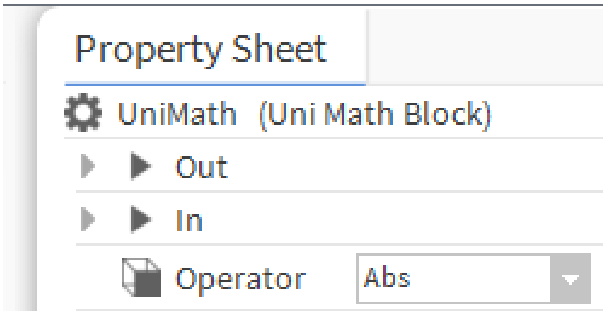
PowerPerSquareFoot (Numeric Point)
Facets units=kW/ft²,precision=3 kW/ft²
Proxy Ext Analytic Proxy Ext
Enabled true
Status {ok}
Fault Cause
Node slot:../..
Data alg:BiMath
Data Filter
Time Range
Rollup
Interval
Aggregation
Totalize
Node Count 2
Poller TriggeredPoller
Last Poll 03-Oct-2022 10:27:17 AM
Missing Data Strategy
Out 0.070 kW/ft² {ok}

Parent topic: Math blocks

Uni math (BUniMathBlock)

This block provides a single-operand math expression that corresponds to the functions in java.lang.Math. It supports both trend and value requests. If the expression results in NaN or Infinity, the block sets the null status flag.

Figure 1. UniMath properties


Property Sheet

UniMath (Uni Math Block)

Out

In

Operator Abs

To view these properties, double-click the block on the **Wire Sheet** or the block name in the Nav tree.

Property	Value	Description
Out	read-only value slot	Outputs the configured Operator applied to the In value.
In	required value slot	Links from the output of other logic blocks or data sources to supply trend data to the current logic block.
Operator	drop-down list (defaults to Abs)	Defines the function to perform. Abs outputs the absolute value of inA. Acos outputs the ArcCosine. Asin outputs the ArcSine. Ceil outputs the smallest (closest to negative infinity) floating-point value that is greater than or equal to the In value and is equal to a mathematical integer. Cos outputs the Cosine. Exp outputs Euler's number e raised to the power of In value. Floor maps a real number to the smallest following integer. Log returns the natural logarithm (base e) of the In value. Negate outputs the negative value of the In value. Round outputs the In value rounded to a mathematical integer. Sin outputs the Sine. Sqrt outputs the square root of the In value. Tangent outputs the tangent of the In value. To Degrees converts the In value from radians to degrees. To Radians converts the In value from degrees to radians.

Parent topic: Math blocks

Switch blocks

A switch evaluates a Boolean condition to select one of two possible inputs.

Table 1. Switch block usage

Block	Value Request	Trend Request	Notes
Bi Switch	Yes	Yes	
Cov Switch	Yes	Yes	Value requests process a trend request using the Value Search Limit property to calculate a time range prior to the station's current time, then find the last trend record in the time range and compares the Test In value against the last trend record value.
Deadband Switch	Yes	Yes	
Invalid Value Switch	Yes	Yes	
Range Switch	Yes	Yes	
Transition Switch	Yes	Yes	Value requests compare the Test In value against the Target In value. If the values are equal, the block processes a trend request using the Value Search Limit property to calculate a time range prior to the station's current time, then finds the last trend record in the time range and compares its Test In value against the last trend record value.
Uni Switch	Yes	Yes	

- **BiSwitch (BBiSwitchBlock)**
  This block evaluates the values of In1 and In2 using the configured Boolean operator and outputs either a True In or False In value. The block supports both value and trend requests.
- **CovSwitch (BCoVSwitchBlock)**
  This block determines if the value of Test In has changed from its prior value by comparing both value and status. It supports both value and trend requests.
- **DeadbandSwitch (BDeadbandSwitch)**
  This block evaluates the Test In value to determine whether the value is within a range of values. The block supports both value and trend requests.
- **Invalid Value Switch (BInvalidValueSwitchBlock)**
  This block evaluates a numeric value of Test In to determine whether its value is valid. A value is considered to be invalid if it is NaN (not a number), infinity, or if it is a BStatusValue with any of the following statuses: disabled, down, fault, stale or null. This block supports both value and trend requests.
- **Range switch (BRangeSwitchBlock)**
  This block evaluates whether the value of Test In is inside the configured range and outputs either a value for True In or a value for False In. This block supports both value and trend requests.
- **TransitionSwitch (BTransitionSwitchBlock)**
  This block determines whether the Test In value (both value and status) equals the Target In value and whether the Test In value has changed value or status from its prior value. This block supports both value and trend requests.
- **UniSwitch (BUniSwitchBlock)**
  This block evaluates a Boolean input, Test In, and outputs either the True In or False In value. This block supports both value and trend requests.

Parent topic: Logic blocks

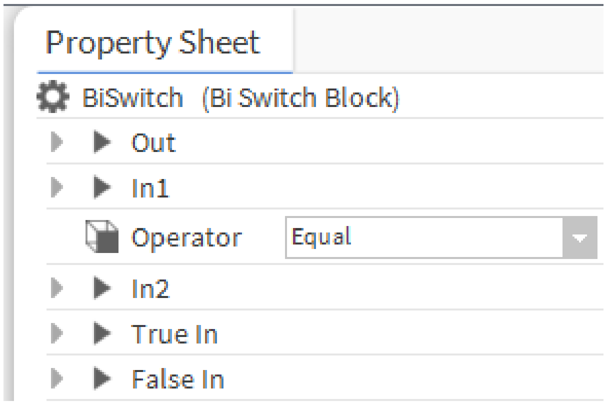
BiSwitch (BBiSwitchBlock)

This block evaluates the values of In1 and In2 using the configured Boolean operator and outputs either a True In or False In value. The block supports both value and trend requests.

If the True In slot is not linked, the block uses a default true value of `(1)`. If the False In slot is not linked, it uses a default false value of `(0)`.

For trend requests, at least one of the inputs (In1 or In2) must map to a trend.

Figure 1. BiSwitch properties



Property Sheet

BiSwitch (Bi Switch Block)

- Out
- In1
- Operator: Equal
- In2
- True In
- False In

To view these properties, double-click the block on the **Wire Sheet** or the block name in the Nav tree.

Property	Value	Description
Out	read-only value slot	Outputs either the True In or False In value based on the evaluating the Boolean condition.
In 1	operand	Defines the first operand used in the expression. Links from the output of other logic blocks or data sources to supply one or more input values to another logic block.
Operator	drop-down list (defaults to Equals)	Defines the function to perform. Equals indicates that In1 = In2. Greater Than indicates that In1 > In2. Greater Than Equal To indicates that In1 ≥ In2. Less Than indicates that In1 < In2. Equal To indicates that In1 = In2. Not Equal indicates that In1 ≠In2. And Or
In 2	operand	Defines the second operand used in the expression. Links from the output of other logic blocks or data sources to supply one or more input values to a logic block.

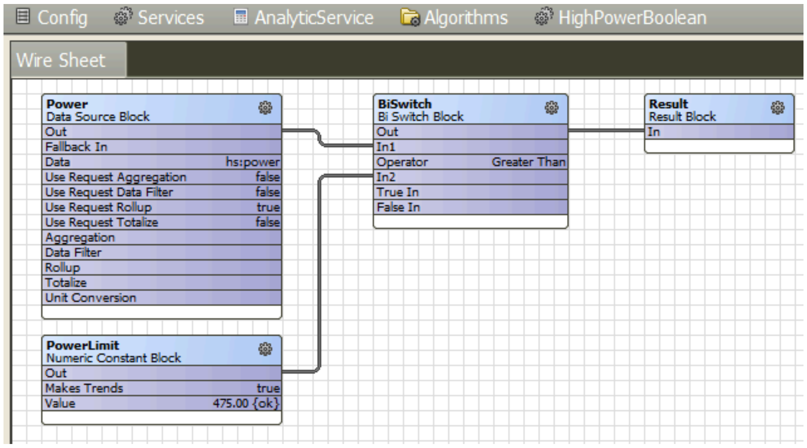
Property	Value	Description
True In	optional value slot (defaults to `True` if unlinked)	Links from the output of other logic blocks or data sources to supply one or more valid values to the logic block.
False In	optional value slot (defaults to `False` if unlinked)	Links from the output of other logic blocks or data sources to supply one or more invalid values to a logic block.

Example

This block supports the Boolean operator functions including: Equals, Greater Than, Greater Than Equal To, Less Than, Less Than Equal To, Not Equal, And, Or.

Consider a numeric point with an interval history for electrical power marked by hs:power and a:a marker tags. To identify when the electrical power exceeds some high limit (fault detection) an algorithm with a **Bi Swtich Block** may be used. In this case, since the True In and False In slots on the block are not linked, the block outputs the default values of either true (1) or false (0).

Figure 2. Bi Switch Block algorithm



Power
Data Source Block
Out
Fallback In
Data hs:power
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup true
Use Request Totalize false
Aggregation
Data Filter
Rollup
Totalize
Unit Conversion

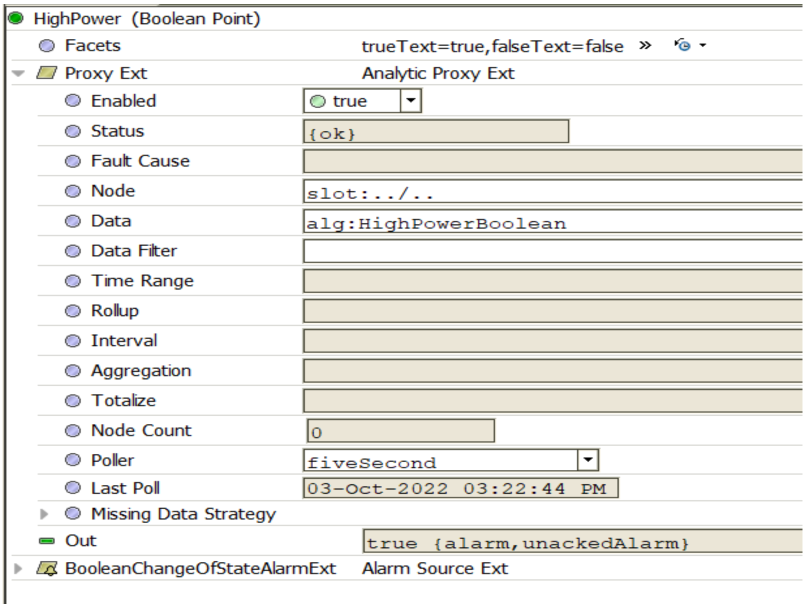
PowerLimit
Numeric Constant Block
Out
Makes Trends true
Value 475.00 {ok}

BiSwitch
Bi Switch Block
Out
In1
Operator Greater Than
In2
True In
False In

Result
Result Block
In

The alg:HighPowerBoolean would be particularly useful with a Boolean Point using an Analytic Proxy Ext in conjunction with a standard BooleanChangeOfStateAlarmExt for fault detection.

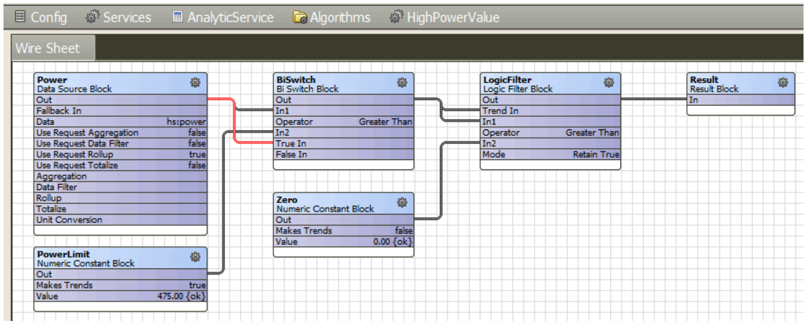
Figure 3. HighPower Boolean Point



HighPower (Boolean Point)	
Facets	trueText=true,falseText=false
Proxy Ext	Analytic Proxy Ext
Enabled	true
Status	{ok}
Fault Cause	
Node	slot:../..
Data	alg:HighPowerBoolean
Data Filter	
Time Range	
Rollup	
Interval	
Aggregation	
Totalize	
Node Count	0
Poller	fiveSecond
Last Poll	03-Oct-2022 03:22:44 PM
Missing Data Strategy	
Out	true {alarm,unackedAlarm}
BooleanChangeOfStateAlarmExt	Alarm Source Ext

You could duplicate and modify the above alg:HighPowerBoolean as below in alg:HighPowerValue.

Figure 4. HighPower Value



Power
Data Source Block
Out
Fallback In
Data hs:power
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup true
Use Request Totalize false
Aggregation
Data Filter
Rollup
Totalize
Unit Conversion

PowerLimit
Numeric Constant Block
Out
Makes Trends true
Value 475.00 {ok}

BiSwitch
Bi Switch Block
Out
In1
Operator Greater Than
In2
True In
False In

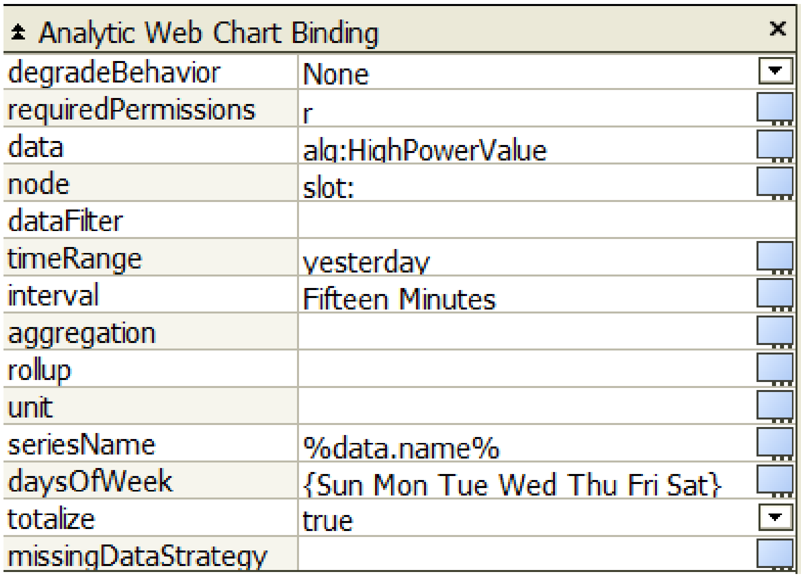
Zero
Numeric Constant Block
Out
Makes Trends false
Value 0.00 {ok}

LogicFilter
Logic Filter Block
Out
Trend In
In1
Operator Greater Than
In2
Mode Retain True

Result
Result Block
In

The **Power** data source links to both the In1 and True In slots on the **Bi Switch Block**, which makes the block either pass the actual power value when the condition is true or a false (0) value when the condition is false. The downstream **Logic Filter Block** is configured with Mode = Retain True so it only passes values greater than zero, meaning the result set only includes the trend records with a value greater than 475 KW.

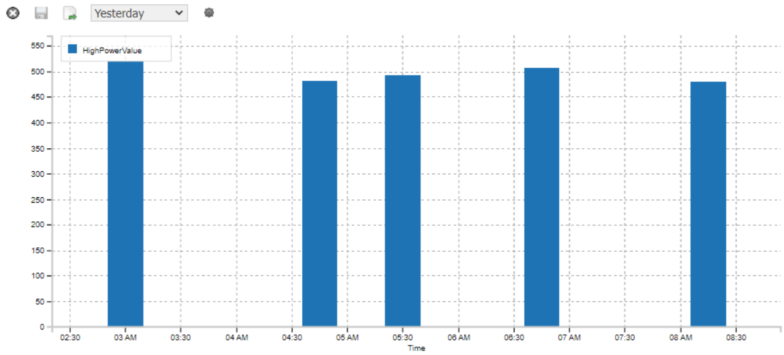
Figure 5. Analytic Web Chart Binding



Analytic Web Chart Binding

degradeBehavior	None
requiredPermissions	r
data	alg:HighPowerValue
node	slot:
dataFilter	
timeRange	yesterday
interval	Fifteen Minutes
aggregation	
rollup	
unit	
seriesName	%data.name%
daysOfWeek	{Sun Mon Tue Wed Thu Fri Sat}
totalize	true
missingDataStrategy	

Figure 6. Chart



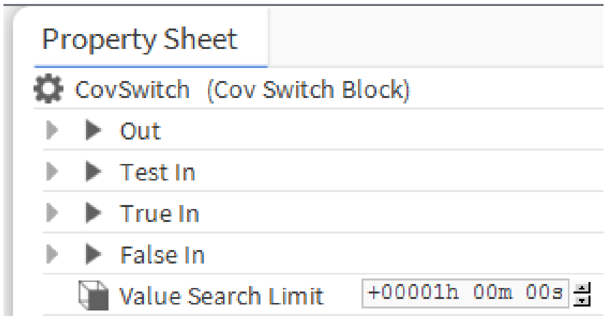
Parent topic: Switch blocks

CovSwitch (BCoVSwitchBlock)

This block determines if the value of Test In has changed from its prior value by comparing both value and status. It supports both value and trend requests.

For value requests, the block requests a trend from Test In using the Value Search Limit property as the time range prior to the current station time, and compares the last record from the trend to the Test In value. The block outputs either the True In or False In value depending on whether the value or status has changed. If the True In slot is not linked a default true value, the block uses `(1)`. If the False In slot is not linked, the block uses the default false value, `(0)`.

Figure 1. CovSwitch properties



Property Sheet

CovSwitch (Cov Switch Block)

- Out
- Test In
- True In
- False In
- Value Search Limit +00001h 00m 00s

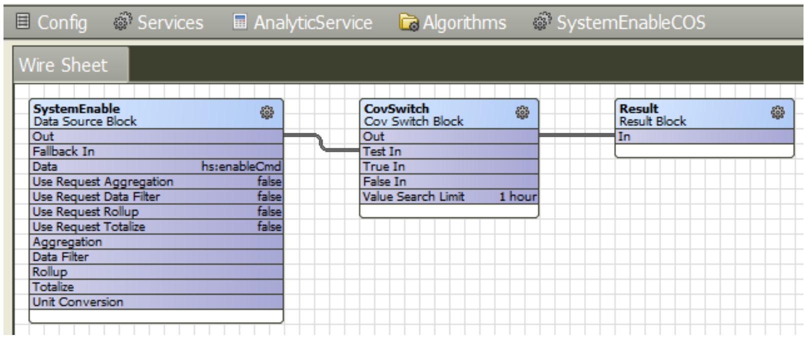
To view these properties, double-click the block on the **Wire Sheet** or the block name in the Nav tree.

Property	Value	Description
Out	read-only value slot	Outputs either the True In or False In value based on evaluating the Boolean condition.
Test In	value slot	Links from the output of other logic blocks or data sources to supply one or more input values to be tested for validity in the logic block.
True In	optional value slot (defaults to `True` if unlinked)	Links from the output of other logic blocks or data sources to supply one or more valid values to the logic block.
False In	optional value slot (defaults to `False` if unlinked)	Links from the output of other logic blocks or data sources to supply one or more invalid values to a logic block.
Value Search Limit	hours minutes seconds (defaults to 1 hour)	Defines how far back in time to request a trend from the Test In (input) for value requests. The system compares the last record of this trend to the result of a value request on the Test In to determine if there was a change of value or status.

Example

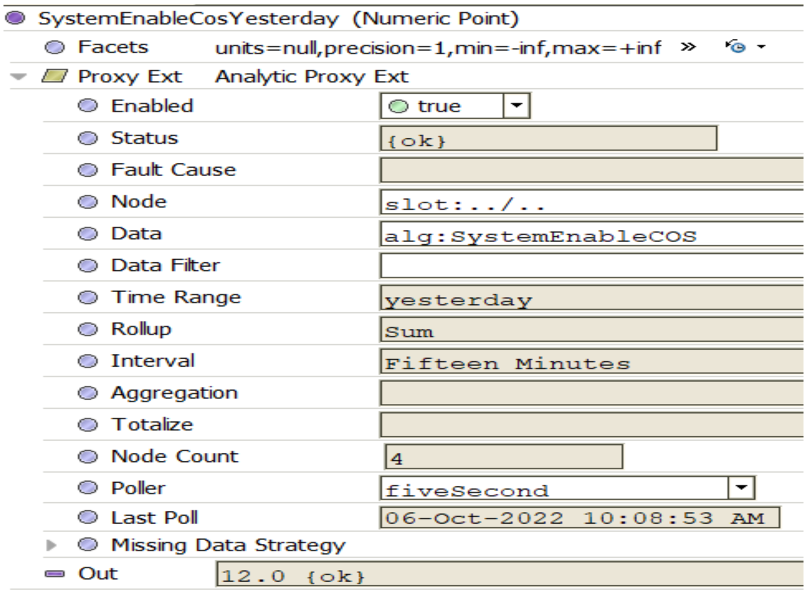
Consider a Boolean Point with either a COV or interval history for enabling a system with `hs:enableCmd` and `a:a` marker tags. An algorithm with a **Cov Switch Block** can determine when the system enable command changes state.

Figure 2. Algorithm with a CovSwitch block



A numeric point with an Analytic Proxy Ext is configured with Data = alg:SystemEnableCOS, Time Range = yesterday, Rollup = Sum and Interval = Fifteen Minutes.

Figure 3. Algorithm result



SystemEnableCosYesterday (Numeric Point)

Facets	units=null,precision=1,min=-inf,max=+inf
Proxy Ext	Analytic Proxy Ext
Enabled	true
Status	{ok}
Fault Cause	
Node	slot:../..
Data	alg:SystemEnableCOS
Data Filter	
Time Range	yesterday
Rollup	Sum
Interval	Fifteen Minutes
Aggregation	
Totalize	
Node Count	4
Poller	fiveSecond
Last Poll	06-Oct-2022 10:08:53 AM
Missing Data Strategy	
Out	12.0 {ok}

The algorithm result is the sum of the true (1) and false (0) values during the time range, which indicates how many times the system enable command changed state from false to true or from true to false.

The **Transition Switch Block** is similar to the **Cov Switch Block** with the added functionality to determine when the Test In value changes to a specific state. You could use it to count only the false to true changes, if needed.

Parent topic: Switch blocks

DeadbandSwitch (BDeadbandSwitch)

This block evaluates the Test In value to determine whether the value is within a range of values. The block supports both value and trend requests.

If the value is within the range, the output reports the True In value, otherwise the output reports the False In value. If the True In slot is not linked, the block reports the default true numeric value one (1). If the False In slot is not linked, the block reports a default false numeric value of zero (0).

For trend requests, at least one of the inputs (In1 or In2) must map to a trend.

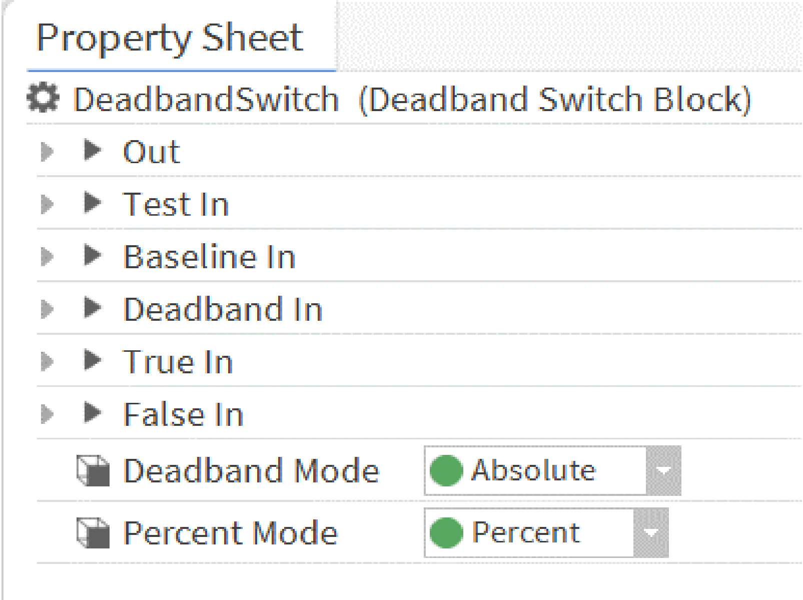
The **Deadband Switch Block** is similar to the **Range Switch Block**, except the high and low limits are calculated from a baseline and deadband instead of configuring independent high and low limits.

The range of values is determined based on the Baseline In, Deadband In, Deadband Mode and Percent Mode properties. Deadband Mode determines if the Deadband In value is an absolute value or a percentage. Percent Mode determines if the percent value is expressed as a decimal (0-1) or a percentage (0-100).

The following table provides an example of how all the properties work together to configure a deadband range.

Table 1. Deadband configuration

Baseline In	Deadband In	Deadband Mode	Percent Mode	Range Min	Range Max	Calculated Range
72	5	Absolute	N/A	Baseline — Deadband	Baseline — Deadband	67–77
72	5	Percent	Percent	Baseline — (Baseline * (Deadband/100))	Baseline — (Baseline * (Deadband/100))	68.4–75.6
72	0.05	Percent	Decimal	Baseline — (Baseline * Deadband)	Baseline — (Baseline * Deadband)	68.4–75.6



Property Sheet

DeadbandSwitch (Deadband Switch Block)

- Out
- Test In
- Baseline In
- Deadband In
- True In
- False In

Deadband Mode: Absolute

Percent Mode: Percent

To view these properties, double-click the block on the **Wire Sheet** or the block name in the Nav tree.

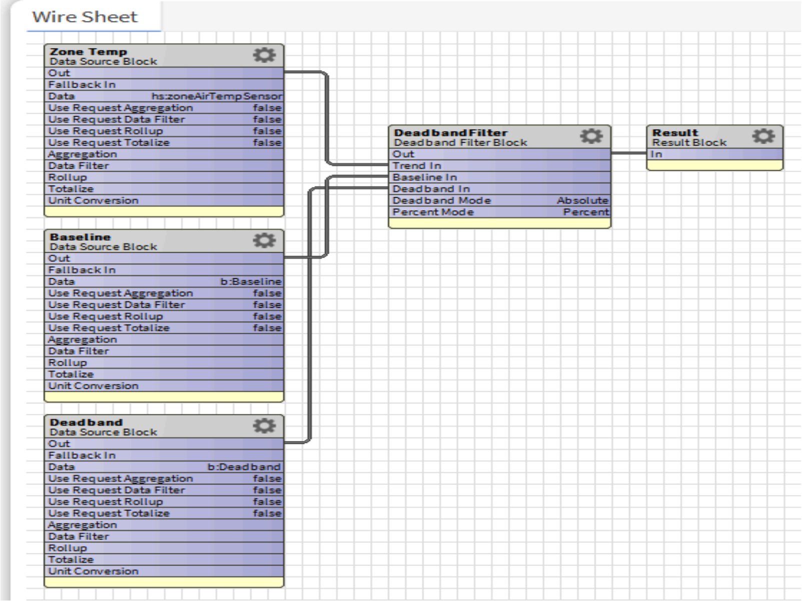
Property	Value	Description
Out	read-only value slot	Outputs either the True In or False In value based on evaluating the Boolean condition.
Test In	required value slot	Links from the output of other logic blocks or data sources to supply one or more input values to be tested for validity in the logic block.
Baseline In	required numeric value slot	Links from the output of other logic blocks

Property	Value	Description
		or data sources to supply a trend. This trend provides a baseline value upon which to apply a deadband.
Deadband In	required numeric value slot	Links from the output of other logic blocks or data sources to supply one or more input values that represent a dead or neutral zone around a baseline.
True In	optional value slot (defaults to `True` if unlinked)	Links from the output of other logic blocks or data sources to supply one or more valid values to the logic block.
False In	optional value slot (defaults to `False` if unlinked)	Links from the output of other logic blocks or data sources to supply one or more invalid values to a logic block.
Deadband Mode	drop-down list (defaults to Absolute)	Determines what the engine uses to establish the deadband. Absolute, configures a specific value to use as the deadband. Percent, configures the deadband as a percentage.
Percent Mode	drop-down list (defaults to Percent)	Determines the form in which the constant appears if **Deadband Mode** is a percentage. Percent treats the value as a percent (0-100). For example, value of 30% would be represented as 30. Decimal treats the value as a decimal number (0-1). For example, a value of 30% would be represented as .3.

Example

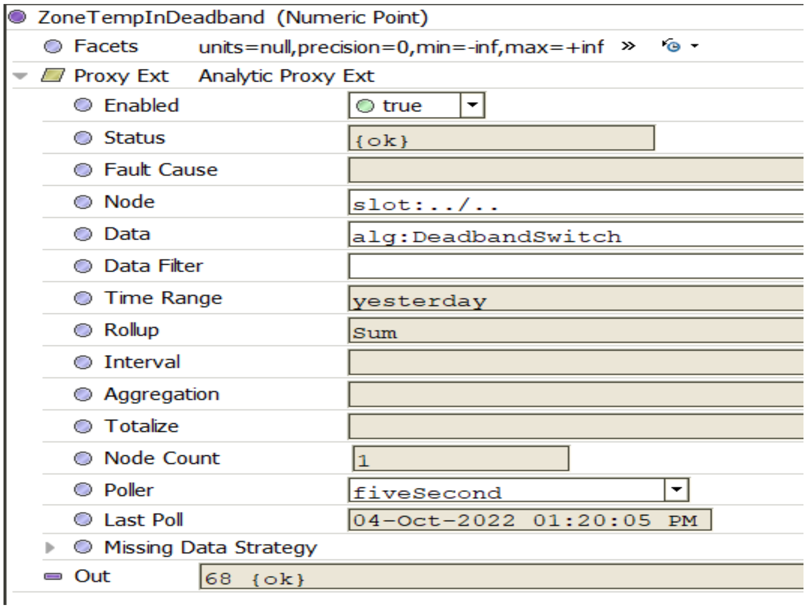
Consider three numeric points with numeric interval histories for Zone Temp tagged with hs:zoneAirTempSensor and a:a marker tags, a baseline value of 72 (b:Baseline and a:a marker tags) and a deadband value of 5 (b:Deadband and a:a marker tags). The algorithm looks like this:

Figure 1. Deadband algorithm



A numeric point with an Analytic Proxy Ext is configured with Data = alg:DeadbandSwitch, Time Range = yesterdayand Rollup = Sum. The algorithm result is the sum of the true (1) and false (0) values during the time range, which indicates how many 15-minute intervals that the Zone Temp value was within the calculated range.

Figure 2. Deadband Numeric Point



ZoneTempInDeadband (Numeric Point)

Facets	units=null,precision=0,min=-inf,max=+inf
Proxy Ext	Analytic Proxy Ext
Enabled	true
Status	{ok}
Fault Cause	
Node	slot:../..
Data	alg:DeadbandSwitch
Data Filter	
Time Range	yesterday
Rollup	Sum
Interval	
Aggregation	
Totalize	
Node Count	1
Poller	fiveSecond
Last Poll	04-Oct-2022 01:20:05 PM
Missing Data Strategy	
Out	68 {ok}

Parent topic: Switch blocks

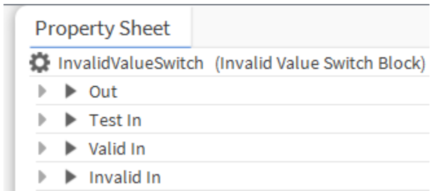
Invalid Value Switch (BInvalidValueSwitchBlock)

This block evaluates a numeric value of Test In to determine whether its value is valid. A value is considered to be invalid if it is NaN (not a number), infinity, or if it is a BStatusValue with any of the following statuses: disabled, down, fault, stale or null. This block supports both value and trend requests.

If the value of Test In is valid, the block outputs the Test In value as the Valid In value, otherwise it outputs an invalid Test In value as the Invalid In value. If the Valid In slot is not linked, the block outputs a numeric value of one (1) for Valid In. If the Invalid In slot is not linked, the block outputs a numeric value of zero (0) Valid In.

A value is considered invalid if it is not a number (NaN) or infinity, or it is a BStatusValue with any of the following statuses: disabled, down, fault, stale or null.

Figure 1. Invalid Value Switch properties



Property Sheet

InvalidValueSwitch (Invalid Value Switch Block)

- Out
- Test In
- Valid In
- Invalid In

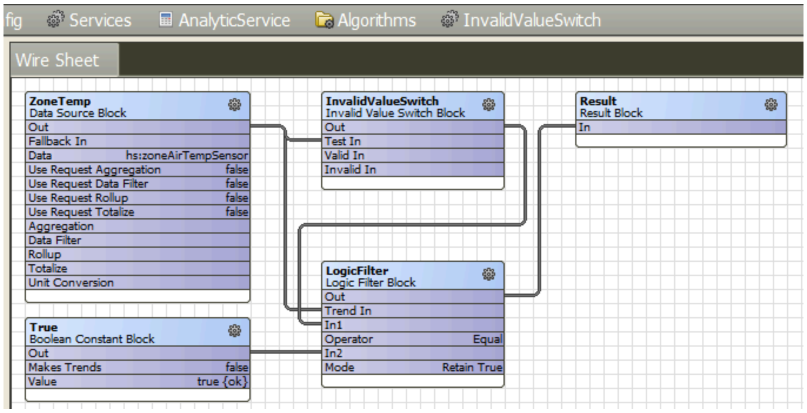
To view these properties, double-click the block on the **Wire Sheet** or the block name in the Nav tree.

Property	Value	Description
Out	read-only value slot	Outputs either the Valid In or Invalid In value based on evaluating the Boolean condition.
Test In	value slot	Links from the output of other logic blocks or data sources to supply one or more input values to be tested for validity in the logic block.
Valid In	optional value slot (defaults to True if not linked)	Links from the output of other logic blocks or data sources to supply one or more valid values to a logic block.
Invalid In	optional value slot (defaults to False if not linked)	Links from the output of other logic blocks or data sources to supply one or more invalid values to a logic block.

Example

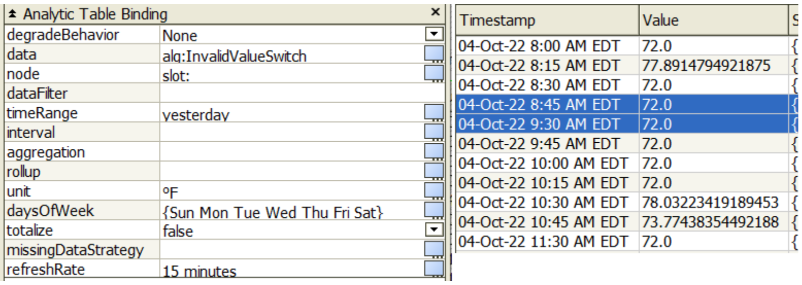
An algorithm with an **Invalid Value Switch Block** can detect invalid values and either substitute another value using the Invalid In slot, or it could be used in conjunction with a **Logic Filter Block** to only return the valid values.

Figure 2. Invalid Value Switch algorithm



In this example the 9:00 AM and 9:15 AM records are not displayed because the algorithm filtered out those invalid values.

Figure 3. Example data



Analytic Table Binding

degradeBehavior	None
data	alg:InvalidValueSwitch
node	slot:
dataFilter	
timeRange	yesterday
interval	
aggregation	
rollup	
unit	°F
daysOfWeek	{Sun Mon Tue Wed Thu Fri Sat}
totalize	false
missingDataStrategy	
refreshRate	15 minutes

Timestamp	Value	S
04-Oct-22 8:00 AM EDT	72.0	{
04-Oct-22 8:15 AM EDT	77.8914794921875	{
04-Oct-22 8:30 AM EDT	72.0	{
04-Oct-22 8:45 AM EDT	72.0	{
04-Oct-22 9:30 AM EDT	72.0	{
04-Oct-22 9:45 AM EDT	72.0	{
04-Oct-22 10:00 AM EDT	72.0	{
04-Oct-22 10:15 AM EDT	72.0	{
04-Oct-22 10:30 AM EDT	78.03223419189453	{
04-Oct-22 10:45 AM EDT	73.77438354492188	{
04-Oct-22 11:30 AM EDT	72.0	{

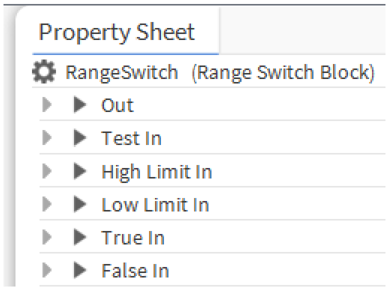
Parent topic: Switch blocks

Range switch (BRangeSwitchBlock)

This block evaluates whether the value of Test In is inside the configured range and outputs either a value for True In or a value for False In. This block supports both value and trend requests.

Values are considered in range if they are greater than or equal to the low limit and less than or equal to the high limit. If the value is within the range the block outputs a value for True In, otherwise it outputs a value for False In. If the True In slot is not linked the block outputs the default numeric value of one (1). If the False In slot is not linked the block outputs the default numeric value of zero (0). This block is similar to the **Deadband Switch Block**, except that you configure independent high and low limits instead of the limits calculated from a baseline and deadband.

Figure 1. Range Switch properties



Property Sheet

RangeSwitch (Range Switch Block)

- Out
- Test In
- High Limit In
- Low Limit In
- True In
- False In

To view these properties, double-click the block on the **Wire Sheet** or the block name in the Nav tree.

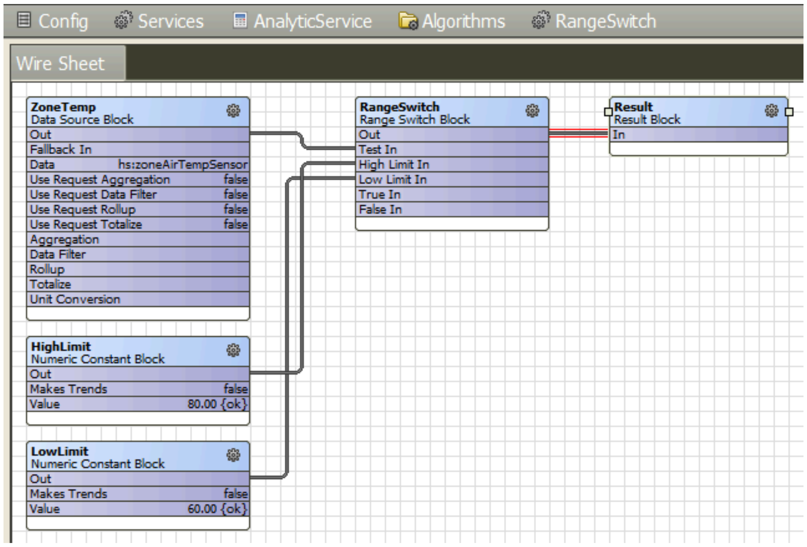
Property	Value	Description
Out	read-only value slot	Outputs either a True In or False In value based on evaluating the Boolean condition.
Test In	value slot	Links from the output of other logic blocks or data sources to supply one or more input values to be tested for validity in the logic block.
High Limit In	numeric value slot	Links from the output of other logic blocks or data sources to supply one or more values that define the maximum valid

Property	Value	Description
		value.
Low Limit In	numeric value slot	Links from the output of other logic blocks or data sources to supply one or more input values to define a minimum valid value.
True In	optional value slot (defaults to `True` if not linked)	Links from the output of other logic blocks or data sources to supply one or more valid values to the logic block.
False In	optional value slot (defaults to `False` if not linked)	Links from the output of other logic blocks or data sources to supply one or more invalid values to a logic block.

Example

Consider a numeric point with an interval history for Zone Temp that is tagged with hs:zoneAirTempSensor and a:a marker tags. An algorithm with a **Range Switch Block** determines whether the Zone Temp value is greater than or equal to 60 degrees and less than or equal to 80 degrees.

Figure 2. Range Switch algorithm



ZoneTemp
Data Source Block
Out
Fallback In
Data hs:zoneAirTempSensor
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Use Request Totalize false
Aggregation
Data Filter
Rollup
Totalize
Unit Conversion

HighLimit
Numeric Constant Block
Out
Makes Trends false
Value 80.00 {ok}

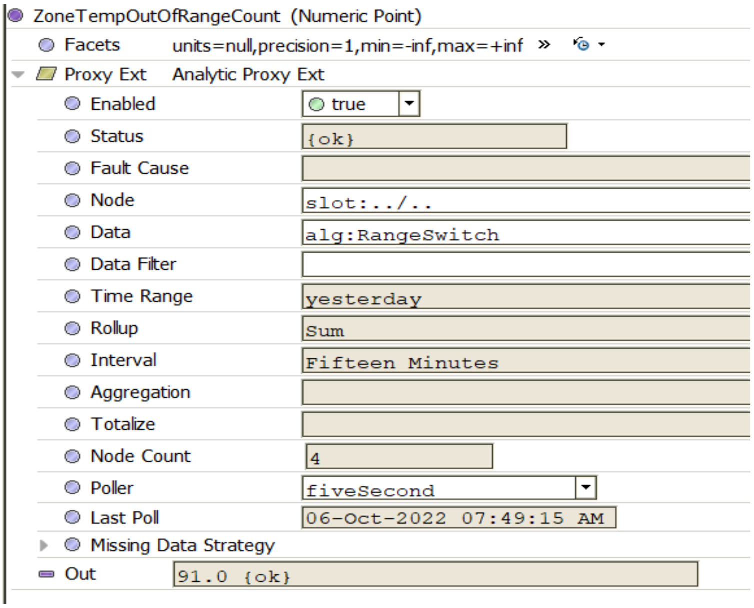
LowLimit
Numeric Constant Block
Out
Makes Trends false
Value 60.00 {ok}

RangeSwitch
Range Switch Block
Out
Test In
High Limit In
Low Limit In
True In
False In

Result
Result Block
In

A numeric point with an Analytic Proxy Ext is configured with Data = alg:RangeSwitch, Time Range = yesterday, Rollup = Sum. The algorithm result is the sum of the true (1) and false (0) values during the time range, which indicates how many 15-minute intervals that the Zone Temp value was within the calculated range.

Figure 3. Configuration



ZoneTempOutOfRangeCount (Numeric Point)

Facets units=null,precision=1,min=-inf,max=+inf

Proxy Ext Analytic Proxy Ext

Enabled: true

Status: {ok}

Fault Cause:

Node: slot:../..

Data: alg:RangeSwitch

Data Filter:

Time Range: yesterday

Rollup: Sum

Interval: Fifteen Minutes

Aggregation:

Totalize:

Node Count: 4

Poller: fiveSecond

Last Poll: 06-Oct-2022 07:49:15 AM

Missing Data Strategy

Out: 91.0 {ok}

Parent topic: Switch blocks

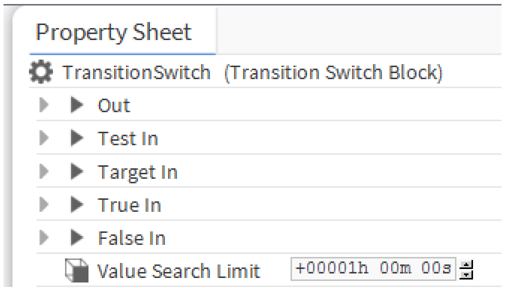
TransitionSwitch (BTransitionSwitchBlock)

This block determines whether the Test In value (both value and status) equals the Target In value and whether the Test In value has changed value or status from its prior value. This block supports both value and trend requests.

This block is similar to the **Cov Switch Block** with the added criterion that the value of Test In equals the value of Target In. The **Cov Switch Block** detects any change in value (both value and status), whereas, you may configure this block to only detect a change in value from `false` to `true`.

For value requests, the block requests a trend from Test In using the Value Search Limit property as the time range prior to the current station time, and then compares the last record from the trend to Test In value. If the value of Test In equals the value of Target In and the value of Target In changed from the prior value, the block outputs the True In value, otherwise it outputs the False In value. If the True In slot is not linked, the block outputs the default numeric value of true, one (1). If the False In slot is not linked, the block outputs the default numeric value of false, zero (0).

Figure 1. Transition Switch properties



Property Sheet

TransitionSwitch (Transition Switch Block)

Out

Test In

Target In

True In

False In

Value Search Limit: +00001h 00m 00s

To view these properties, double-click the block on the **Wire Sheet** or the block name in the Nav tree.

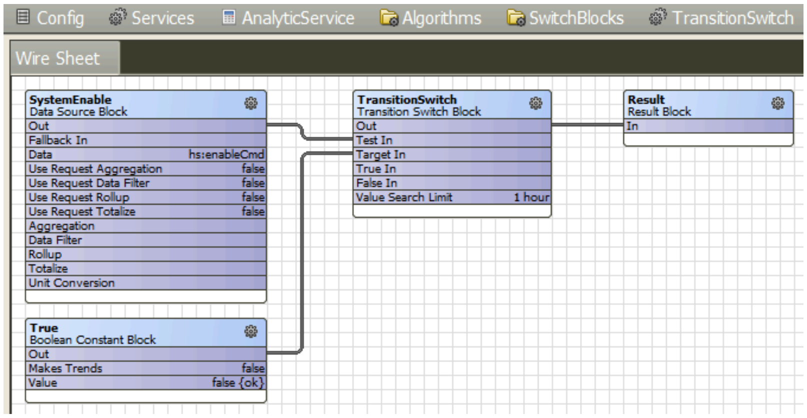
Property	Value	Description
Out	read-only value slot	Outputs either the True In or False In value

Property	Value	Description
		based on evaluating the Boolean condition.
Test In	value slot	Links from the output of other logic blocks or data sources to supply one or more input values to be tested for validity in the logic block.
Target in	value slot	Links from the output of other logic blocks or data sources to supply one or more input values to be tested for validity in the logic block.
True In	optional value slot (defaults to `True` if not linked)	Links from the output of other logic blocks or data sources to supply one or more valid values to the logic block.
False In	optional value slot (defaults to `False` if not linked)	Links from the output of other logic blocks or data sources to supply one or more invalid values to a logic block.
Value Search Limit	hours minutes seconds (defaults to 1 hour)	Defines how far back in time to request a trend from the Test In (input) for value requests. The system compares the last record of this trend to the result of a value request on the Test In to determine if there was a change of value or status.

Example

Consider a Boolean point with either a COV or interval history for enabling a system, which is tagged with hs:enableCmd and a:a marker tags. An algorithm with a **Transition Switch Block** can determine when the system enable command changes state from false to true.

Figure 2. Transition Switch algorithm



SystemEnable
Data Source Block
Out
Fallback In
Data hs:enableCmd
Use Request Aggregation false
Use Request Data Filter false
Use Request Rollup false
Use Request Totalize false
Aggregation
Data Filter
Rollup
Totalize
Unit Conversion

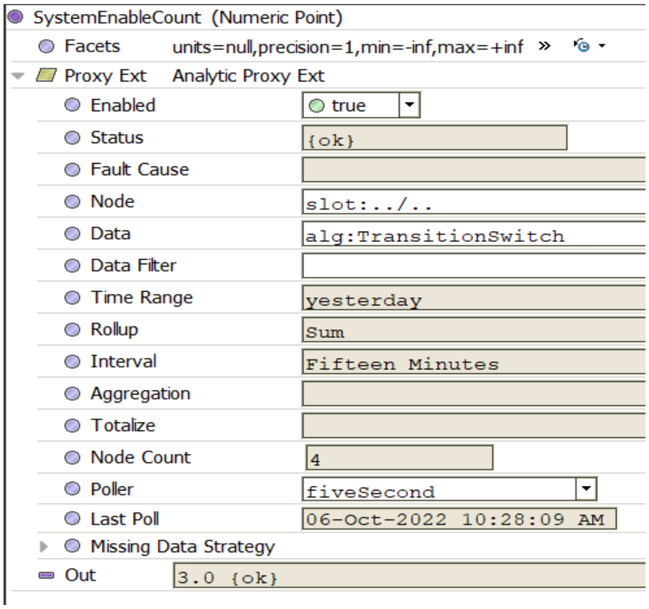
True
Boolean Constant Block
Out
Makes Trends false
Value false {ok}

TransitionSwitch
Transition Switch Block
Out
Test In
Target In
True In
False In
Value Search Limit 1 hour

Result
Result Block
In

A numeric point with an Analytic Proxy Ext is configured with Data = alg:TransitionSwitch, Time Range = yesterday, Rollup = Sum and Interval = Fifteen Minutes. The algorithm result is the sum of the true (1) and false (0) values during the time range, which indicates how many times the system enable command changed state from false to true.

Figure 3. Configuration



SystemEnableCount (Numeric Point)	
Facets	units=null,precision=1,min=-inf,max=+inf
Proxy Ext	Analytic Proxy Ext
Enabled	true
Status	{ok}
Fault Cause	
Node	slot:../..
Data	alg:TransitionSwitch
Data Filter	
Time Range	yesterday
Rollup	Sum
Interval	Fifteen Minutes
Aggregation	
Totalize	
Node Count	4
Poller	fiveSecond
Last Poll	06-Oct-2022 10:28:09 AM
Missing Data Strategy	
Out	3.0 {ok}

Parent topic: Switch blocks

UniSwitch (BUniSwitchBlock)

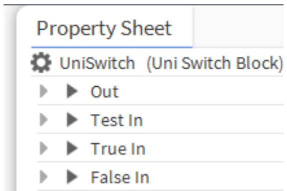
This block evaluates a Boolean input, Test In, and outputs either the True In or False In value. This block supports both value and trend requests.

If the True In slot is not linked, the block outputs the default numeric value of true, one (1). If the False In slot is not linked, the block outputs the default numeric value of false, zero (0).

For trend requests, at least one of the inputs, Test In, True In or False In, must map to a trend.

This block functions the same as the **Bi Switch Block** except the Boolean logic is tested externally using another logic block or Boolean data source in the algorithm, which is linked to the Test In slot.

Figure 1. UniSwitch properties



Property Sheet

UniSwitch (Uni Switch Block)
- Out
- Test In
- True In
- False In

To view these properties, double-click the block on the **Wire Sheet** or the block name in the Nav tree.

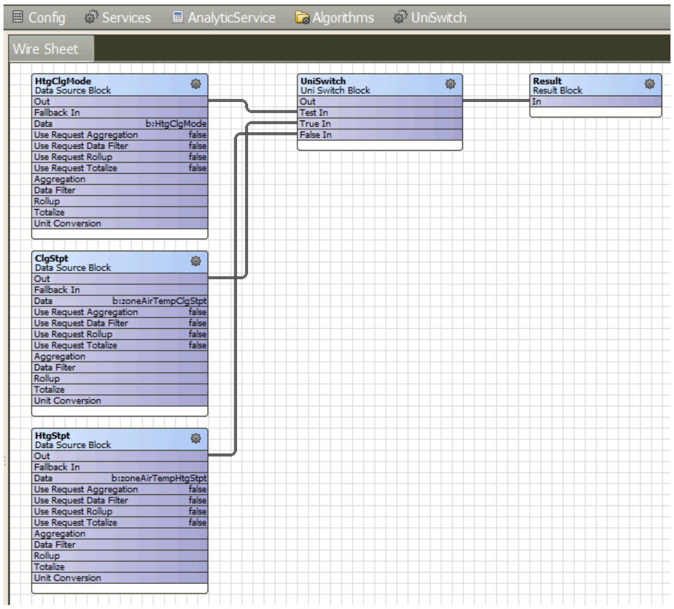
Property	Value	Description
Out	read-only value slot	Outputs either the True In or False In value based on evaluating the Boolean condition.
Test In	value slot	Links from the output of other logic blocks or data sources to supply one or more

Property	Value	Description
		input values to be tested for validity in the logic block.
True In	optional value slot (defaults to `True` if not linked)	Links from the output of other logic blocks or data sources to supply one or more valid values to the logic block.
False In	optional value slot (defaults to `False` if not linked)	Links from the output of other logic blocks or data sources to supply one or more invalid values to a logic block.

Example

Consider a Boolean point with an interval history for HVAC Heating and Cooling Mode that is tagged with `b:HtgClgMode` and `a:a` marker tags, and numeric points with interval histories for Heating and Cooling Setpoints that are tagged with `b:zoneAirTempClgStpt` or `b:zoneAirTempHtgStpt` marker tag and `a:a` marker tags. An algorithm with a **Uni Switch Block** can use the HVAC Heating and Cooling Mode to determine the effective setpoint.

Figure 2. Uni Switch algorithm



HtgClgMode
Data Source Block

Out	
Fallback In	
Data	b:HtgClgMode
Use Request Aggregation	false
Use Request Data Filter	false
Use Request Rollup	false
Use Request Totalize	false
Aggregation	
Data Filter	
Rollup	
Totalize	
Unit Conversion	

ClgStpt
Data Source Block

Out	
Fallback In	
Data	b:zoneAirTempClgStpt
Use Request Aggregation	false
Use Request Data Filter	false
Use Request Rollup	false
Use Request Totalize	false
Aggregation	
Data Filter	
Rollup	
Totalize	
Unit Conversion	

HtgStpt
Data Source Block

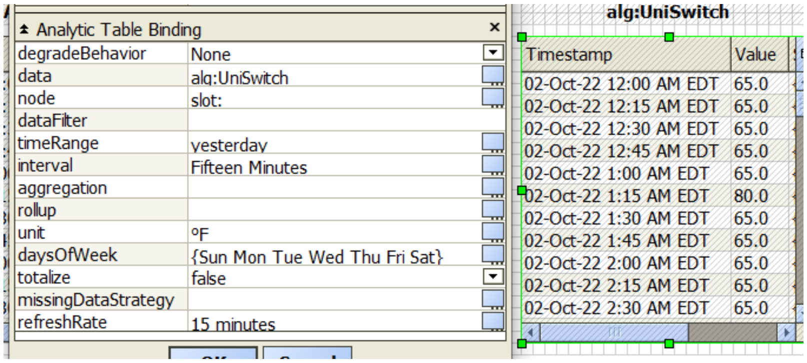
Out	
Fallback In	
Data	b:zoneAirTempHtgStpt
Use Request Aggregation	false
Use Request Data Filter	false
Use Request Rollup	false
Use Request Totalize	false
Aggregation	
Data Filter	
Rollup	
Totalize	
Unit Conversion	

UniSwitch
Uni Switch Block
Out
Test In
True In
False In

Result
Result Block
In

The Analytic Table Binding configuration and example output could be:

Figure 3. Configuration and example



Analytic Table Binding

degradeBehavior	None
data	alg:UniSwitch
node	slot:
dataFilter	
timeRange	yesterday
interval	Fifteen Minutes
aggregation	
rollup	
unit	°F
daysOfWeek	{Sun Mon Tue Wed Thu Fri Sat}
totalize	false
missingDataStrategy	
refreshRate	15 minutes

alg:UniSwitch

Timestamp	Value
02-Oct-22 12:00 AM EDT	65.0
02-Oct-22 12:15 AM EDT	65.0
02-Oct-22 12:30 AM EDT	65.0
02-Oct-22 12:45 AM EDT	65.0
02-Oct-22 1:00 AM EDT	65.0
02-Oct-22 1:15 AM EDT	80.0
02-Oct-22 1:30 AM EDT	65.0
02-Oct-22 1:45 AM EDT	65.0
02-Oct-22 2:00 AM EDT	65.0
02-Oct-22 2:15 AM EDT	65.0
02-Oct-22 2:30 AM EDT	65.0

Parent topic: Switch blocks

Graphics bindings

The bindings process the analytic value or trend request based on the configured properties (Data, Aggregation, Rollup, Interval, etc.) and return either a single value result or trend data. The binding passes the result to its parent widget that handles displaying the data.

The bindings include:

- **Analytics Value Binding** This binding configures a value request, which returns a single value. If the value request finds multiple data sources under the configured node, it uses the aggregation function to combine the data sources. This binding does not use a time range.

- **Analytics Rollup Binding** This binding configures a trend request using a time range and returns a single value by applying the rollup function (min, max, avg, sum, etc.) with an interval matching the time range.

- **Analytics Web Chart Binding** This binding configures a trend request using a time range and returns trend data designed to be visualized in a Web Chart or Web Table widget.

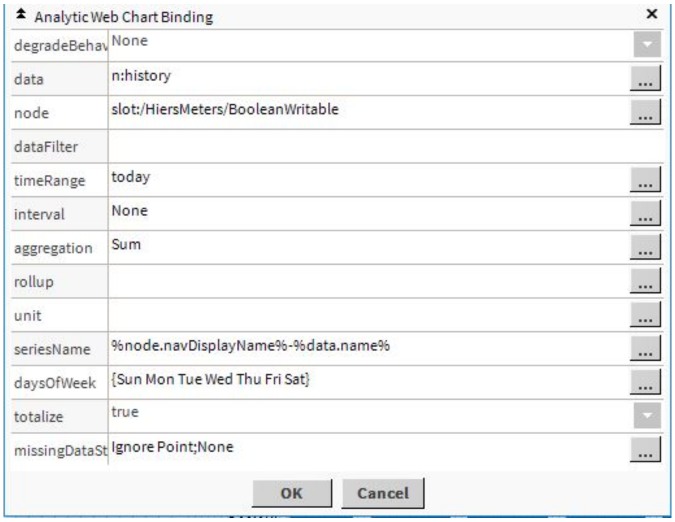
- **Analytics Table Binding** This binding configures a trend request using a time range and returns trend data designed to be visualized in a Bound Table widget.

- **Analytics Web Rollup Binding** This binding configures a trend request using a time range and rollup and returns a single trend record for the requested time range. This binding is used with the Ranking and Relative Contribution Charts, and the Ranking and Relative Contribution Reports, but could be used with a Web Chart or Web Table as well.

Analytic web chart binding

This binding configures a trend request and returns trend data. You can use it wherever a traditional table chart binding can be used.

Figure 1. AnalyticWebChartBinding properties



Analytic Web Chart Binding

degradeBehav	None
data	n:history
node	slot:/HiersMeters/BooleanWritable
dataFilter	
timeRange	today
interval	None
aggregation	Sum
rollup	
unit	
seriesName	%node.navDisplayName%-%data.name%
daysOfWeek	{Sun Mon Tue Wed Thu Fri Sat}
totalize	true
missingDataSt	Ignore Point;None

You access this view of the Analytic Web Chart Binding properties when you double-click the web widget in the Px view.

Property	Value	Description
degradeBehavior	Disable or None	Specifies the framework's behavior when the binding analytic ORD cannot be resolved or used due to security permissions.
data	tag or algorithm name	Specifies the an algorithm (or tag) used to retrieve data. The purpose of an alert is to detect a fault condition (Boolean value result). In most cases, you would pick an algorithm that has a Boolean result when configuring this property for an alert. Algorithms with a numeric or enum value greater than zero are considered true and less than zero are considered false. This is how output from one algorithm becomes the input data source to another algorithm. The prefix for algorithms is `alg:`.
node	ORD (defaults to a relative ORD slot, which is resolved against the config space to which the Px view is assigned)	Defines the ORD to the desired slot.
dataFilter	ORD parameter	Identifies data sources in the subtree of the request node. When a predicate accepts a node that contains the desired data, the system does not search the node's subtree for additional values (including the root node).
timeRange	ORD parameter (defaults to Today)	Defines the time period used to query historical data. Defines the beginning and end of a period of time over which to count the number of

Property	Value	Description
		occurrences or weight. If the alert depends on a certain number of occurrences or weight over time, this property is required, otherwise, it is optional.
interval	ORD parameter	Refers to the BInterval component, which the framework uses to identify the time between values in a trend (time series). When specified, a rollup is required, which causes the system to combine all values that fall into a single interval. Options range from None to a Year. Above the drop-down list, the Use This Value check box turns on and off the check box next to Interval in the **Settings** window (you access this window by clicking the Edit button (✎), followed by clicking the Settings button (⚙) on the chart). The availability of this check box provides an easy way for a user to enable and disable the use of intervals in chart calculations.
aggregation	ORD parameter	Configures the default function to apply when the analytic request needs to combine values from multiple data sources into a single value. This applies to both value and trend requests. If aggregation is not enabled in the binding/settings window, the aggregation value defined in the Data Definition applies to all chart bindings, reports and tables. And returns the logical “and” of Boolean values. Avg returns the statistical mean, which is determined by calculating the sum of all values and dividing by the number of values. Count returns the total number or quantity of values in a combination. If you request this value on a binding in a PX view, the system counts the number of values based on the properties defined by the data source block and the algorithm’s **Property Sheet**. First returns the first value in the combination. Last returns the last value in the combination. Max returns the highest value in the combination. Median returns the value in the middle of a sorted combination—the number that separates the higher half from the lower half.

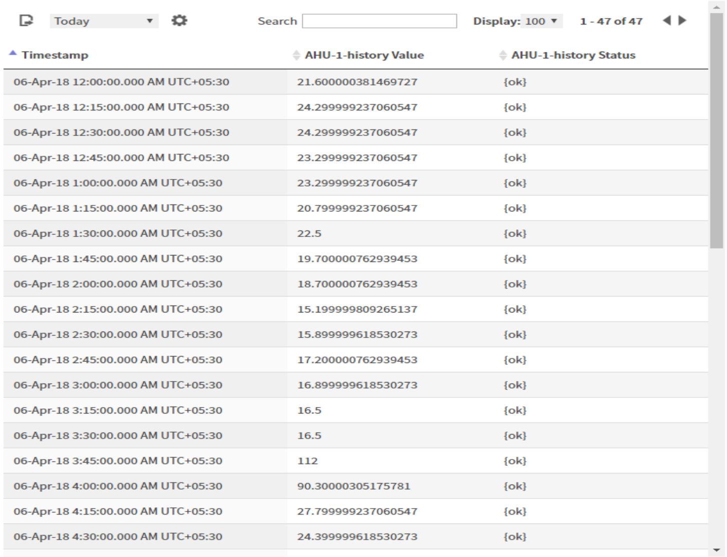
Property	Value	Description
		Min returns the lowest value in the combination. Mode returns the statistically most frequently occurring number in the combination. Or returns the logical “or” of Boolean values. Range returns the statistical difference between the largest and smallest values in the combination. Sum adds together all values in the combination resulting in a single value. Load Factor returns the average value divided by peak value. Std Dev returns the standard deviation of the values in the combination.
rollup	ORD parameter (rollup=option)	Configures the default function to apply when the analytic request needs to rollup records from a single data source into less granular records. This typically only applies to trend request, but may also apply to value requests where the Id is an algorithm that contains a block like Runtime or Sliding Window, which processes a trend request. If rollup is not enabled in the binding/ settings window, the rollup value configured in the Data Definition applies to all chart bindings, reports and tables. And returns the logical “and” of Boolean values. Avg returns the statistical mean, which is determined by calculating the sum of all values and dividing by the number of values. Count returns the total number or quantity of values in a combination. If you request this value on a binding in a PX view, the system counts the number of values based on the properties defined by the data source block and the algorithm’s **Property Sheet**. First returns the first value in the combination. Last returns the last value in the combination. Max returns the highest value in the combination. Median returns the value in the middle of a sorted combination—the number that separates the higher half from the lower half. Min returns the lowest value in the

Property	Value	Description
		combination. Mode returns the statistically most frequently occurring number in the combination. Or returns the logical "or" of Boolean values. Range returns the statistical difference between the largest and smallest values in the combination. Sum adds together all values in the combination resulting in a single value. Std Dev calculates the standard deviation of the values in the combination. Load Factor calculates the average divided by peak (Max) value.
unit	unit of measure	Selects the unit name and display symbol to associate with a data source.
seriesName	BFormat (defaults to %node.navDisplayName-data.name%	Defines the node and tag name configured by the Data property. The BFormat value may contain static text or scripts that use the percentage symbol delimiter, such as %node.navDisplayName%. The default value is %node.navDisplayName%-%data.name%, which resolves to the navDisplayName of the node and the name of the tag configured by the Data property, such as "Building1-Power." The %data.name% script drops the tags namespace, such as "hs:" or the algorithm reference, such as "alg:." If you enter an invalid value, the system displays, "Problem with seriesName BFormat," and logs a message in the station console. The name you enter here displays as the legend and tool tip in the Web chart. If left blank, no legend displays for the tool tip.
daysOfWeek	ORD scheme	Specifies the days of the week to include.
totalize	true or false	Turns on (true) and off (false) value accumulation. By default, the framework totalizes (accumulates) all consumption history values in charts, tables and reports. To prevent cumulative values, disable this property (set it to false).
missingDataStrategy	chooser	Configures how the framework handles missing data in a series when processing analytic requests and one or more records are missing for an interval. It applies when

Property	Value	Description
		even a single record for an interval is missing. It does not apply to value requests.

Settings window

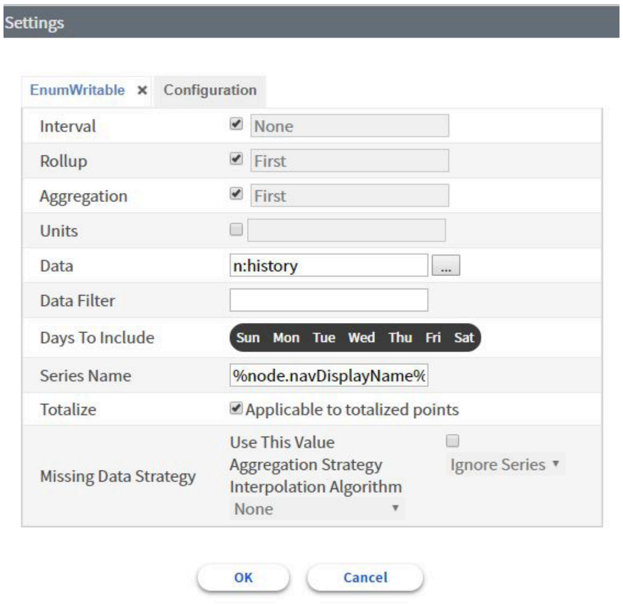
Figure 2. Sample table configured with an analytic web chart binding



Timestamp	AHU-1-history Value	AHU-1-history Status
06-Apr-18 12:00:00.000 AM UTC+05:30	21.600000381469727	{ok}
06-Apr-18 12:15:00.000 AM UTC+05:30	24.299999237060547	{ok}
06-Apr-18 12:30:00.000 AM UTC+05:30	24.299999237060547	{ok}
06-Apr-18 12:45:00.000 AM UTC+05:30	23.299999237060547	{ok}
06-Apr-18 1:00:00.000 AM UTC+05:30	23.299999237060547	{ok}
06-Apr-18 1:15:00.000 AM UTC+05:30	20.799999237060547	{ok}
06-Apr-18 1:30:00.000 AM UTC+05:30	22.5	{ok}
06-Apr-18 1:45:00.000 AM UTC+05:30	19.700000762939453	{ok}
06-Apr-18 2:00:00.000 AM UTC+05:30	18.700000762939453	{ok}
06-Apr-18 2:15:00.000 AM UTC+05:30	15.199999809265137	{ok}
06-Apr-18 2:30:00.000 AM UTC+05:30	15.899999618530273	{ok}
06-Apr-18 2:45:00.000 AM UTC+05:30	17.200000762939453	{ok}
06-Apr-18 3:00:00.000 AM UTC+05:30	16.899999618530273	{ok}
06-Apr-18 3:15:00.000 AM UTC+05:30	16.5	{ok}
06-Apr-18 3:30:00.000 AM UTC+05:30	16.5	{ok}
06-Apr-18 3:45:00.000 AM UTC+05:30	112	{ok}
06-Apr-18 4:00:00.000 AM UTC+05:30	90.30000305175781	{ok}
06-Apr-18 4:15:00.000 AM UTC+05:30	27.799999237060547	{ok}
06-Apr-18 4:30:00.000 AM UTC+05:30	24.399999618530273	{ok}

Clicking the wrench icon opens the **Settings** window with which to configure table properties.

Figure 3. Settings window



Interval	☑ None
Rollup	☑ First
Aggregation	☑ First
Units	□
Data	n:history ...
Data Filter	
Days To Include	Sun Mon Tue Wed Thu Fri Sat
Series Name	%node.navDisplayName%
Totalize	☑ Applicable to totalized points
Missing Data Strategy	Use This Value □ Aggregation Strategy Ignore Series Interpolation Algorithm None

For history data, the table shows the timestamp history value and status. Pagination and search features are available for the table.

Property	Value	Description
Interval	optional drop-down list (defaults to the optimal number of records on reports)	Defines the frequency at which the framework displays a history record.
Rollup	value (when configured in the Data Definition, defaults to First, when configured elsewhere, defaults to the value as defined in the Data Definition)	Configures the default function to apply when the analytic request needs to rollup records from a single data source into less granular records. This typically only applies to trend request, but may also apply to value requests where the Id is an algorithm that contains a block like Runtime or Sliding Window, which processes a trend request. If rollup is not enabled in the binding/settings window, the rollup value configured in the Data Definition applies to all chart bindings, reports and tables. And returns the logical “and” of Boolean values. Avg returns the statistical mean, which is determined by calculating the sum of all values and dividing by the number of values. Count returns the total number or quantity of values in a combination. If you request this value on a binding in a PX view, the system counts the number of values based on the properties defined by the data source block and the algorithm’s **Property Sheet**. First returns the first value in the combination. Last returns the last value in the combination. Max returns the highest value in the combination. Median returns the value in the middle of a sorted combination—the number that separates the higher half from the lower half. Min returns the lowest value in the combination. Mode returns the statistically most frequently occurring number in the combination. Or returns the logical “or” of Boolean values. Range returns the statistical difference between the largest and smallest values in the combination. Sum adds together all values in the combination resulting in a single value.

Property	Value	Description
		Std Dev calculates the standard deviation of the values in the combination. Load Factor calculates the average divided by peak (Max) value.
Aggregation	check box (if optional, and) drop-down list (property, defaults to First) or ORD parameter (aggregation=option)	Configures the default function to apply when the analytic request needs to combine values from multiple data sources into a single value. This applies to both value and trend requests. If aggregation is not enabled in the binding/settings window, the aggregation value defined in the Data Definition applies to all chart bindings, reports and tables. And returns the logical “and” of Boolean values. Avg returns the statistical mean, which is determined by calculating the sum of all values and dividing by the number of values. Count returns the total number or quantity of values in a combination. If you request this value on a binding in a PX view, the system counts the number of values based on the properties defined by the data source block and the algorithm’s **Property Sheet**. First returns the first value in the combination. Last returns the last value in the combination. Max returns the highest value in the combination. Median returns the value in the middle of a sorted combination—the number that separates the higher half from the lower half. Min returns the lowest value in the combination. Mode returns the statistically most frequently occurring number in the combination. Or returns the logical “or” of Boolean values. Range returns the statistical difference between the largest and smallest values in the combination. Sum adds together all values in the combination resulting in a single value. Load Factor returns the average value divided by peak value. Std Dev returns the standard deviation of the values in the combination.

Property	Value	Description
Units	drop-down list of units of measure	Defines the unit of measure for the data gathered from each point.
Data	tag or algorithm name	Specifies the an algorithm (or tag) used to retrieve data. The purpose of an alert is to detect a fault condition (Boolean value result). In most cases, you would pick an algorithm that has a Boolean result when configuring this property for an alert. Algorithms with a numeric or enum value greater than zero are considered true and less than zero are considered false. This is how output from one algorithm becomes the input data source to another algorithm. The prefix for algorithms is `alg:`.
Data Filter	optional NEQL predicate (property) or ORD parameter (`dataFilter=query`)	Identifies data sources in the subtree of the request node. When a predicate accepts a node that contains the desired data, the system does not search the node's subtree for additional values (including the root node).
Days to Include	days-of-the-week selector	Specifies the days of the week to include.
Series Name	BFormat	Defines the node and tag name configured by the Data property. The BFormat value may contain static text or scripts that use the percentage symbol delimiter, such as %node.navDisplayName%. The default value is %node.navDisplayName%-%data.name%, which resolves to the navDisplayName of the node and the name of the tag configured by the Data property, such as ”Building1-Power.” The %data.name% script drops the tags namespace, such as “hs:” or the algorithm reference, such as “alg:.” If you enter an invalid value, the system displays, “Problem with seriesName BFormat,” and logs a message in the station console. The name you enter here displays as the legend and tool tip in the Web chart. If left blank, no legend displays for the tool tip.
Totalize	true or false	Turns on (true) and off (false) value accumulation. By default, the framework totalizes (accumulates) all consumption history values in charts, tables and reports. To prevent cumulative values, disable this

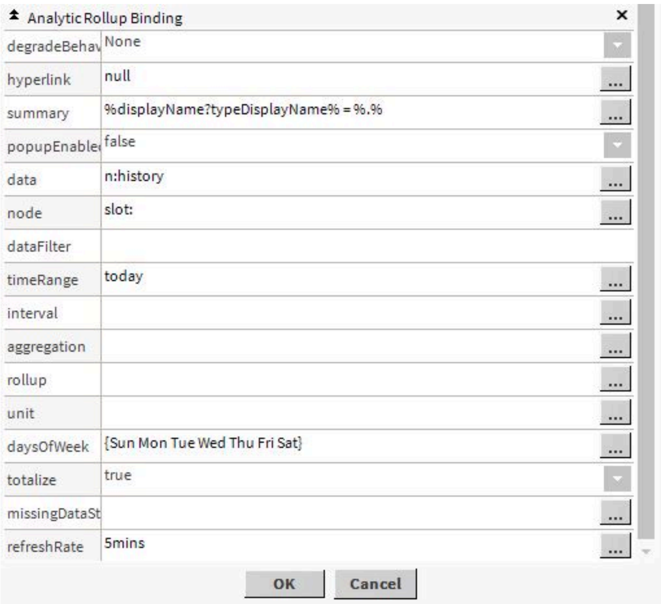
Property	Value	Description
		property (set it to false).
Missing Data Strategy, Use This Value	check box	Enables and disables missing data interpolation for the current value. When enabled, the framework applies this strategy to all requests. When enabled, the framework applies this strategy to all requests.
Missing Data Strategy, Aggregation Strategy	drop-down list	Configures how the framework handles missing trend data (data in a series) when processing analytic requests and one or more records are missing for an interval. It applies when even a single record for an interval is missing. It does not apply to value requests. Note: If the analytic trend request specifies Interval = none, the framework ignores the Missing Data Strategy. Ignore Series ignores the entire series if any record even one interval is missing. Ignore Point ignores any missing records and aggregates the values in the existing records.
Missing Data Strategy, Interpolation Algorithm	drop-down list	Defines the algorithm used to interpolate values for missing values (missing records). None does not interpolate values for missing records. Linear Interpolation replaces a missing record by linearly interpolating the missing value using the prior and next records on either side of the missing record. K-Nearest Neighbor is for numeric, enum and Boolean records. This strategy replaces a missing value by calculating the majority value recorded for the item’s nearest K number of neighbors.
Missing Data Strategy, K Value	number	Defines the number of records used by the configured interpolation algorithm when a record is missing to calculate the interpolated value.

Parent topic: Graphics bindings

Analytic rollup binding

You can use this binding wherever a traditional value ORD binding can be used.

Figure 1. Analytic Rollup Binding properties



degradeBehav	None
hyperlink	null
summary	%displayName?typeDisplayName% = %.%
popupEnable	false
data	n:history
node	slot:
dataFilter	
timeRange	today
interval	
aggregation	
rollup	
unit	
daysOfWeek	{Sun Mon Tue Wed Thu Fri Sat}
totalize	true
missingDataSt	
refreshRate	5mins

You access this view of the Analytic Rollup Binding properties when you double-click the web widget in the Px view.

Under the hood, this binding creates an analyticRollup: ord. However, since analytic values cannot be subscribed to like points, this binding automatically refreshes the value.

Property	Value	Description
degradeBehavior	Disable or None	Specifies the framework's behavior when the binding analytic ORD cannot be resolved or used due to security permissions.
hyperlink	ORD	Provides a link to another object. When this property contains an ORD, the hyperlink is active in the browser or in the Px Viewer.
summary	text or BFormat	Formats the summary string that is displayed in the status bar on mouse over.
popupEnabled	true or false (default)	Turns the use of a second popup on and off. true allows a second popup (right-click menu) to open when a user clicks on this label from a browser or from the Px Viewer. false disables the popup action. You would use this setting to prevent actions from being executed at a control point from a specific Px graphic while allowing access to the action from a different Px graphic or **Property Sheet**.
data	tag or algorithm name	Specifies the an algorithm (or tag) used to retrieve data.

Property	Value	Description
		The purpose of an alert is to detect a fault condition (Boolean value result). In most cases, you would pick an algorithm that has a Boolean result when configuring this property for an alert. Algorithms with a numeric or enum value greater than zero are considered true and less than zero are considered false. This is how output from one algorithm becomes the input data source to another algorithm. The prefix for algorithms is `alg:`.
node	ORD (defaults to a relative ORD slot, which is resolved against the config space to which the Px view is assigned)	Defines the ORD to the desired slot.
dataFilter	ORD parameter	Identifies data sources in the subtree of the request node. When a predicate accepts a node that contains the desired data, the system does not search the node’s subtree for additional values (including the root node).
timeRange	ORD parameter (defaults to Today)	Defines the time period used to query historical data. Defines the beginning and end of a period of time over which to count the number of occurrences or weight. If the alert depends on a certain number of occurrences or weight over time, this property is required, otherwise, it is optional.
interval	optional drop-down list (defaults to the optimal number of records on reports)	Determines if the associated interval should be used. to combine output data.
aggregation	value (defaults to First)	Configures the default function to apply when the analytic request needs to combine values from multiple data sources into a single value. This applies to both value and trend requests. If aggregation is not enabled in the binding/settings window, the aggregation value defined in the Data Definition applies to all chart bindings, reports and tables. And returns the logical “and” of Boolean values. Avg returns the statistical mean, which is determined by calculating the sum of all values and dividing by the number of values. Count returns the total number or quantity of values in a combination. If you request this value on a binding in a PX view, the system counts the number of values based on the properties defined by the data

Property	Value	Description
		source block and the algorithm's **Property Sheet**. First returns the first value in the combination. Last returns the last value in the combination. Max returns the highest value in the combination. Median returns the value in the middle of a sorted combination—the number that separates the higher half from the lower half. Min returns the lowest value in the combination. Mode returns the statistically most frequently occurring number in the combination. Or returns the logical "or" of Boolean values. Range returns the statistical difference between the largest and smallest values in the combination. Sum adds together all values in the combination resulting in a single value. Load Factor returns the average value divided by peak value. Std Dev returns the standard deviation of the values in the combination.
rollup	value (when configured in the Data Definition, defaults to First, when configured elsewhere, defaults to the value as defined in the Data Definition)	Configures the default function to apply when the analytic request needs to rollup records from a single data source into less granular records. This typically only applies to trend request, but may also apply to value requests where the Id is an algorithm that contains a block like **Runtime** or **Sliding Window**, which processes a trend request. If rollup is not enabled in the binding/ settings window, the rollup value configured in the Data Definition applies to all chart bindings, reports and tables. And returns the logical "and" of Boolean values. Avg returns the statistical mean, which is determined by calculating the sum of all values and dividing by the number of values. Count returns the total number or quantity of values in a combination. If you request this value on a binding in a PX view, the system counts the number of values based on the properties defined by the data source block and the algorithm's **Property**

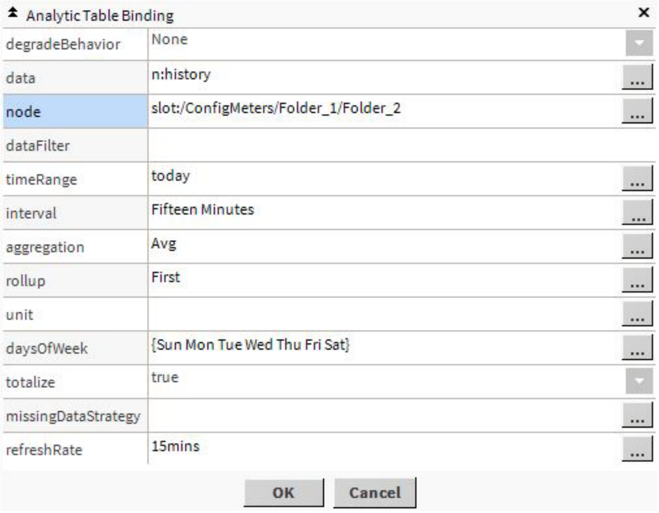
Property	Value	Description
		Sheet. First returns the first value in the combination. Last returns the last value in the combination. Max returns the highest value in the combination. Median returns the value in the middle of a sorted combination—the number that separates the higher half from the lower half. Min returns the lowest value in the combination. Mode returns the statistically most frequently occurring number in the combination. Or returns the logical “or” of Boolean values. Range returns the statistical difference between the largest and smallest values in the combination. Sum adds together all values in the combination resulting in a single value. Std Dev calculates the standard deviation of the values in the combination. Load Factor calculates the average divided by peak (Max) value.
unit	unit of measure	Selects the unit name and display symbol to associate with a data source.
daysOfWeek	ORD scheme	Specifies the days of the week to include.
totalize	true or false	Turns on (true) and off (false) value accumulation. By default, the framework totalizes (accumulates) all consumption history values in charts, tables and reports. To prevent cumulative values, disable this property (set it to false).
missingDataStrategy	chooser	Configures how the framework handles missing data in a series when processing analytic requests and one or more records are missing for an interval. It applies when even a single record for an interval is missing. It does not apply to value requests.
refreshRate	hours minutes seconds (defaults to 5 minutes)	Determines how frequently the framework resolves the analytic request to provide updated data to the widget.

Parent topic: Graphics bindings

Analytic table binding

You can use this binding wherever a traditional value ORD binding can be used.

Figure 1. Analytic table binding properties



Analytic Table Binding

degradeBehavior	None
data	n:history
node	slot:/ConfigMeters/Folder_1/Folder_2
dataFilter	
timeRange	today
interval	Fifteen Minutes
aggregation	Avg
rollup	First
unit	
daysOfWeek	{Sun Mon Tue Wed Thu Fri Sat}
totalize	true
missingDataStrategy	
refreshRate	15mins

You access this view of the Analytic Table Binding properties when you double-click the widget in the Px view.

Property	Value	Description
degradeBehavior	Disable or None	Specifies the framework’s behavior when the binding analytic ORD cannot be resolved or used due to security permissions.
data	algorithm (or tag) name; data=algorithm or tag name	Specifies the an algorithm (or tag) used to retrieve data. The purpose of an alert is to detect a fault condition (Boolean value result). In most cases, you would pick an algorithm that has a Boolean result when configuring this property for an alert. Algorithms with a numeric or enum value greater than zero are considered true and less than zero are considered false. This is how output from one algorithm becomes the input data source to another algorithm. The prefix for algorithms is alg:.
node	ORD (defaults to a relative ORD slot, which is resolved against the config space to which the Px view is assigned)	Defines the ORD to the desired slot.
dataFilter	optional NEQL query (property) or ORD parameter (dataFilter=query)	Identifies data sources in the subtree of the request node. When a predicate accepts a node that contains the desired data, the system does not search the node’s subtree for additional values

Property	Value	Description
		(including the root node).
timeRange	ORD parameter (defaults to Today)	Defines the time period used to query historical data. This property is required for rollup requests (analyticRollup), trends (analyticTrend), and rollup bindings. It is optional elsewhere. It is not used if the block’s Use Request Time Range property is true and the request specifies a time range. Options range from From to All.
interval	optional drop-down list (defaults to the optimal number of records on reports); `interval=option`	Refers to the BInterval component, which the framework uses to identify the time between values in a trend (time series). When specified, a rollup is required, which causes the system to combine all values that fall into a single interval. Options range from None to a Year. Above the drop-down list, the Use This Value check box turns on and off the check box next to Interval in the **Settings** window (you access this window by clicking the Edit button (✎), followed by clicking the Settings button (⚙) on the chart). The availability of this check box provides an easy way for a user to enable and disable the use of intervals in chart calculations.
aggregation	ORD parameter	Configures the default function to apply when the analytic request needs to combine values from multiple data sources into a single value. This applies to both value and trend requests. If aggregation is not enabled in the binding/settings window, the aggregation value defined in the Data Definition applies to all chart bindings, reports and tables. And returns the logical “and” of Boolean values. Avg returns the statistical mean, which is determined by calculating the sum of all values and dividing by the number of values. Count returns the total number or quantity of values in a combination. If you request this value on a binding in a PX view, the system counts the number of values based on the properties defined by the data source block and the algorithm’s **Property Sheet**. First returns the first value in the combination.

Property	Value	Description
		Last returns the last value in the combination. Max returns the highest value in the combination. Median returns the value in the middle of a sorted combination—the number that separates the higher half from the lower half. Min returns the lowest value in the combination. Mode returns the statistically most frequently occurring number in the combination. Or returns the logical “or” of Boolean values. Range returns the statistical difference between the largest and smallest values in the combination. Sum adds together all values in the combination resulting in a single value. Load Factor returns the average value divided by peak value. Std Dev returns the standard deviation of the values in the combination.
rollup	check box (if optional, and) drop-down list or ORD parameter (when configured in the data definition, defaults to First, when configured elsewhere, defaults to the value as defined in the Data Definition); rollup=option	Configures the default function to apply when the analytic request needs to rollup records from a single data source into less granular records. This typically only applies to trend request, but may also apply to value requests where the Id is an algorithm that contains a block like Runtime or Sliding Window, which processes a trend request. If rollup is not enabled in the binding/ settings window, the rollup value configured in the Data Definition applies to all chart bindings, reports and tables. And returns the logical “and” of Boolean values. Avg returns the statistical mean, which is determined by calculating the sum of all values and dividing by the number of values. Count returns the total number or quantity of values in a combination. If you request this value on a binding in a PX view, the system counts the number of values based on the properties defined by the data source block and the algorithm’s **Property Sheet**. First returns the first value in the combination. Last returns the last value in the

Property	Value	Description
		combination. Max returns the highest value in the combination. Median returns the value in the middle of a sorted combination—the number that separates the higher half from the lower half. Min returns the lowest value in the combination. Mode returns the statistically most frequently occurring number in the combination. Or returns the logical “or” of Boolean values. Range returns the statistical difference between the largest and smallest values in the combination. Sum adds together all values in the combination resulting in a single value. Std Dev calculates the standard deviation of the values in the combination. Load Factor calculates the average divided by peak (Max) value.
unit	unit=option (unit of measure)	Selects the unit name and display symbol to associate with a data source.
daysOfWeek	ORD scheme	Specifies the days of the week to include.
totalize	true (true) or false or hisTotEnabled=option (ORD scheme parameter)	Turns on (true) and off (false) value accumulation. By default, the framework totalizes (accumulates) all consumption history values in charts, tables and reports. To prevent cumulative values, disable this property (set it to false).
missingDataStrategy	additional properties	Configures how the framework handles missing data in a series when processing analytic requests and one or more records are missing for an interval. It applies when even a single record for an interval is missing. It does not apply to value requests.
refreshRate	hours minutes seconds	Determines how frequently the framework resolves the analytic request to provide updated data to the widget.

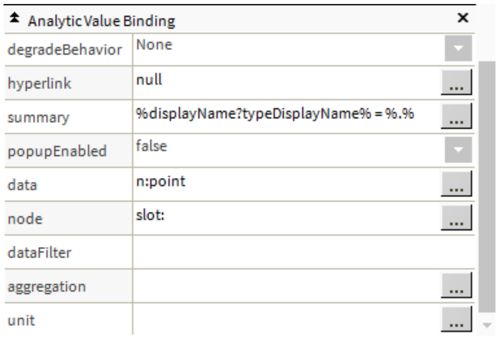
Parent topic: Graphics bindings

Analytic value binding

This binding associates a traditional value ORD with the Px View.

Under the hood, this binding creates an analyticValue: ord. However, since analytic values cannot be subscribed to like points, this binding automatically refreshes the value.

Figure 1. Analytic Value Binding properties



degradeBehavior	None
hyperlink	null
summary	%displayName?typeDisplayName% = %.%
popupEnabled	false
data	n:point
node	slot:
dataFilter	
aggregation	
unit	

Property	Value	Description
degradeBehavior	Disable or None	Specifies the framework's behavior when the binding analytic ORD cannot be resolved or used due to security permissions.
hyperlink	ORD	Provides a link to another object. When this property contains an ORD, the hyperlink is active in the browser or in the Px Viewer.
summary	text or BFormat	Formats the summary string that is displayed in the status bar on mouse over.
popupEnabled	true or false (default)	Turns the use of a second popup on and off. true allows a second popup (right-click menu) to open when a user clicks on this label from a browser or from the Px Viewer. false disables the popup action. You would use this setting to prevent actions from being executed at a control point from a specific Px graphic while allowing access to the action from a different Px graphic or **Property Sheet**.
data	tag or algorithm name	Specifies the an algorithm (or tag) used to retrieve data. The purpose of an alert is to detect a fault condition (Boolean value result). In most cases, you would pick an algorithm that has a Boolean result when configuring this property for an alert. Algorithms with a numeric or enum value greater than zero are considered true and less than zero are considered false. This is how output from one algorithm becomes the input data source to another algorithm. The prefix for algorithms is

Property	Value	Description
		alg:.
node	ORD(defaults to a relative ORD slot, which is resolved against the config space to which the Px view is assigned)	Defines the ORD to the desired slot.
dataFilter	ORD parameter	Identifies data sources in the subtree of the request node. When a predicate accepts a node that contains the desired data, the system does not search the node’s subtree for additional values (including the root node).
aggregation	check box (if optional, and) drop-down list (property, defaults to First) or ORD parameter (aggregation=option)	Configures the default function to apply when the analytic request needs to combine values from multiple data sources into a single value. This applies to both value and trend requests. If aggregation is not enabled in the binding/settings window, the aggregation value defined in the Data Definition applies to all chart bindings, reports and tables. And returns the logical “and” of Boolean values. Avg returns the statistical mean, which is determined by calculating the sum of all values and dividing by the number of values. Count returns the total number or quantity of values in a combination. If you request this value on a binding in a PX view, the system counts the number of values based on the properties defined by the data source block and the algorithm’s **Property Sheet**. First returns the first value in the combination. Last returns the last value in the combination. Max returns the highest value in the combination. Median returns the value in the middle of a sorted combination—the number that separates the higher half from the lower half. Min returns the lowest value in the combination. Mode returns the statistically most frequently occurring number in the combination. Or returns the logical “or” of Boolean values. Range returns the statistical difference between the largest and smallest values in

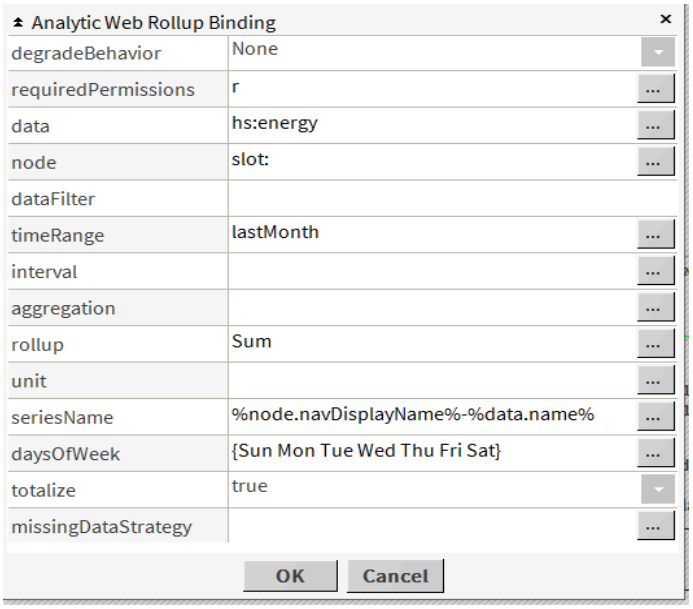
Property	Value	Description
		the combination. Sum adds together all values in the combination resulting in a single value. Load Factor returns the average value divided by peak value. Std Dev returns the standard deviation of the values in the combination.
unit	unit=option (unit of measure)	Selects the unit name and display symbol to associate with a data source.

Parent topic: Graphics bindings

Analytic web rollup binding

This binding works with a Web Chart or Web Table widget and is the default binding used for the Ranking and Relative Contribution charts. It configures a trend request using a time range and rollup function and returns a single trend record for the requested time range.

Figure 1. Analytic Web Rollup Binding properties



Analytic Web Rollup Binding

degradeBehavior	None
requiredPermissions	r
data	hs:energy
node	slot:
dataFilter	
timeRange	lastMonth
interval	
aggregation	
rollup	Sum
unit	
seriesName	%node.navDisplayName%-%data.name%
daysOfWeek	{Sun Mon Tue Wed Thu Fri Sat}
totalize	true
missingDataStrategy	

Property	Value	Description
degradeBehavior	Disable or None	Specifies the framework's behavior when the binding analytic ORD cannot be resolved or used due to security permissions.
requiredPermissions		
data	algorithm (or tag) name; data=algorithm or tag name	Specifies the an algorithm (or tag) used to retrieve data. The purpose of an alert is to detect a fault condition (Boolean value result). In most

Property	Value	Description
		cases, you would pick an algorithm that has a Boolean result when configuring this property for an alert. Algorithms with a numeric or enum value greater than zero are considered true and less than zero are considered false. This is how output from one algorithm becomes the input data source to another algorithm. The prefix for algorithms is `alg:`.
node	ORD (defaults to a relative ORD slot, which is resolved against the config space to which the Px view is assigned)	Defines the ORD to the desired slot.
dataFilter	optional NEQL query (property) or ORD parameter (`dataFilter=query`)	Identifies data sources in the subtree of the request node. When a predicate accepts a node that contains the desired data, the system does not search the node's subtree for additional values (including the root node).
timeRange	drop-down list or ORD parameter (defaults to Today (current value);`timerange=option`	Defines the time period used to query historical data. This property is required for rollup requests (analyticRollup), trends (analyticTrend), and rollup bindings. It is optional elsewhere. It is not used if the block's Use Request Time Range property is true and the request specifies a time range. Options range from From to All.
interval	optional drop-down list (defaults to the optimal number of records on reports); `interval=option`	Refers to the BInterval component, which the framework uses to identify the time between values in a trend (time series). When specified, a rollup is required, which causes the system to combine all values that fall into a single interval. Options range from None to a Year. Above the drop-down list, the Use This Value check box turns on and off the check box next to Interval in the **Settings** window (you access this window by clicking the Edit button (✎), followed by clicking the Settings button (⚙) on the chart). The availability of this check box provides an easy way for a user to enable and disable the use of intervals in chart calculations.
aggregation	check box (if optional, and) drop-down list (property, defaults to First) or ORD parameter (`aggregation=option`)	Configures the default function to apply when the analytic request needs to combine values from multiple data sources into a single value. This applies to both value and trend requests.

Property	Value	Description
		If aggregation is not enabled in the binding/settings window, the aggregation value defined in the Data Definition applies to all chart bindings, reports and tables. And returns the logical “and” of Boolean values. Avg returns the statistical mean, which is determined by calculating the sum of all values and dividing by the number of values. Count returns the total number or quantity of values in a combination. If you request this value on a binding in a PX view, the system counts the number of values based on the properties defined by the data source block and the algorithm’s **Property Sheet**. First returns the first value in the combination. Last returns the last value in the combination. Max returns the highest value in the combination. Median returns the value in the middle of a sorted combination—the number that separates the higher half from the lower half. Min returns the lowest value in the combination. Mode returns the statistically most frequently occurring number in the combination. Or returns the logical “or” of Boolean values. Range returns the statistical difference between the largest and smallest values in the combination. Sum adds together all values in the combination resulting in a single value. Load Factor returns the average value divided by peak value. Std Dev returns the standard deviation of the values in the combination.
rollup	Check box (if optional, and) drop-down list or ORD parameter (when configured in the data definition, defaults to First, when configured elsewhere, defaults to the value as defined in the Data Definition); `rollup=option`	Configures the default function to apply when the analytic request needs to rollup records from a single data source into less granular records. This typically only applies to trend request, but may also apply to value requests where the Id is an algorithm that contains a block like Runtime or Sliding Window, which processes a trend request. If rollup is not enabled in the binding/

Property	Value	Description
		settings window, the rollup value configured in the Data Definition applies to all chart bindings, reports and tables. And returns the logical “and” of Boolean values. Avg returns the statistical mean, which is determined by calculating the sum of all values and dividing by the number of values. Count returns the total number or quantity of values in a combination. If you request this value on a binding in a PX view, the system counts the number of values based on the properties defined by the data source block and the algorithm’s **Property Sheet**. First returns the first value in the combination. Last returns the last value in the combination. Max returns the highest value in the combination. Median returns the value in the middle of a sorted combination—the number that separates the higher half from the lower half. Min returns the lowest value in the combination. Mode returns the statistically most frequently occurring number in the combination. Or returns the logical “or” of Boolean values. Range returns the statistical difference between the largest and smallest values in the combination. Sum adds together all values in the combination resulting in a single value. Std Dev calculates the standard deviation of the values in the combination. Load Factor calculates the average divided by peak (Max) value.
unit	`unit=option` (unit of measure)	Selects the unit name and display symbol to associate with a data source.
seriesName	BFormat (defaults to `%node.navDisplayName-data.name%`)	Defines the node and tag name configured by the Data property. The BFormat value may contain static text or scripts that use the percentage symbol delimiter, such as %node.navDisplayName%. The default value is %node.navDisplayName%-%data.name%, which resolves to the

Property	Value	Description
		navDisplayName of the node and the name of the tag configured by the Data property, such as ”Building1-Power.” The %data.name% script drops the tags namespace, such as “hs:” or the algorithm reference, such as “alg:.” If you enter an invalid value, the system displays, “Problem with seriesName BFormat,” and logs a message in the station console. The name you enter here displays as the legend and tool tip in the Web chart. If left blank, no legend displays for the tool tip.
daysOfWeek	ORD scheme	Specifies the days of the week to include.
totalize	true (true) or false or hisTotEnabled=option (ORD scheme parameter)	Turns on (true) and off (false) value accumulation. By default, the framework totalizes (accumulates) all consumption history values in charts, tables and reports. To prevent cumulative values, disable this property (set it to false).
missingDataStrategy	additional properties	Configures how the framework handles missing data in a series when processing analytic requests and one or more records are missing for an interval. It applies when even a single record for an interval is missing. It does not apply to value requests.

Parent topic: Graphics bindings

Charts

Charts support standard Px views and views designed to display in a browser. The information in this topic is generally applicable to all analytics charts.

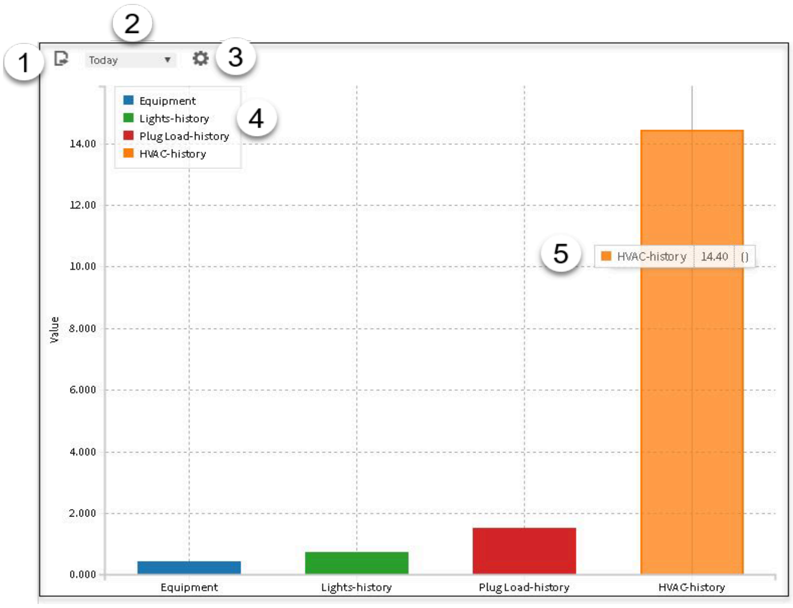
General information about charts

Each of the following topics describes the individual chart types. For detailed information about working with charts, and, specifically, about exporting charts, refer to the specific chart topic and also refer to the *Niagara Web Charts Guide*.

Chart controls

You use the chart controls to configure all the charts without going back to the **Property Sheet**.

Figure 1. Example of a ranking chart



1. The document icon opens the Export Wizard.
2. The time range drop-down list selects the x-axis.
3. The settings icon opens the configuration window.
4. The value legend identifies the meaning of each color.
5. Passing the cursor over a chart element displays the graphed value.

- **Aggregation chart**
  This chart displays aggregated data, that is, data from multiple data sources, that have been combined for viewing.
- **AnalyticWebChart**
  This multipurpose chart is capable of displaying data using a bar, line, area, step, area step, spline, area spline or scatter chart. In addition you can enable a second y-axis. This chart requires a Px view. You can configure it on a dashboard in a Px view.

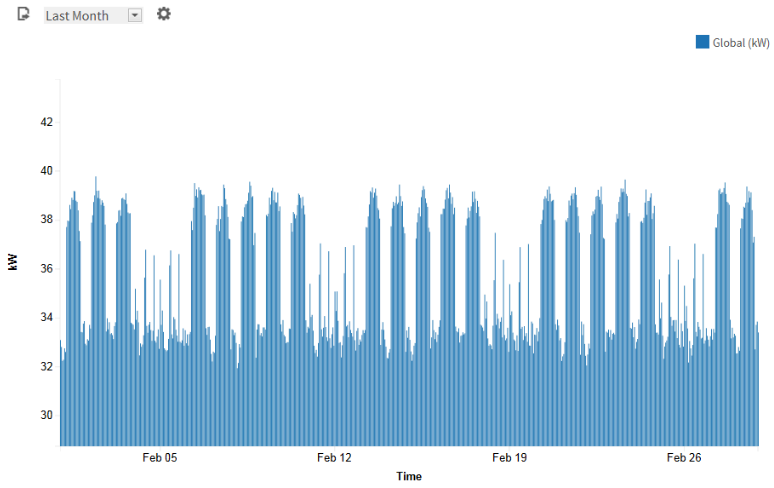
- **Web chart Ux settings window**
  This window configures AnalyticWebChart properties in the live Px view.
- **Average Profile chart**
  This chart shows a pattern that represents the average of a data value over a specified time period. It provides flexibility when identifying high values at various times within a selected time period. This chart supports multiple bindings.
- **Equipment Operation chart**
  This chart indicates when a piece of equipment is on and off. It supports multiple bindings, each binding representing a piece of equipment.
- **Load duration chart**
  This chart summarizes how long a value was at a certain value. It can help you observe the duration of peak demand levels for the purpose of identifying correct demand limiting strategies.
- **Ranking chart**
  This chart compares values from selected nodes using vertical bars. It supports multiple bindings. Using it you can quickly compare similar buildings or systems to find items with unexpectedly high or low values, such as one building using more energy per month than other similar buildings in the portfolio. If you identify an item with abnormal values, use other analytic web charts and reports to further analyze the data and determine the root cause.
- **Relative contribution chart**
  This pie chart plots the contributions from individual pieces of equipment (or any data model node) to the total value of a group or a site. For example, if meters are grouped appropriately, you can identify that HVAC consumption is 45%, lighting consumption is 35%, and other loads contribute 20% to the total consumption of a particular building or campus.
- **Spectrum chart**
  Using color coding, the Spectrum Chart provides insight into the reasonableness of the value obtained from a data or aggregated point. A quick glance at a consistent color pattern, which indicates valid data, can visually confirm an expected condition, allowing you to move on to other functions. A quick glance at an inconsistent color pattern can draw immediate attention to a situation requiring further analysis.

Aggregation chart

This chart displays aggregated data, that is, data from multiple data sources, that have been combined for viewing.

You add this chart to a Px view by dragging the AggregationChart component from the analytics palette’s Charts folder to the desired Px view in the Px editor mode.

Figure 1. Aggregation chart



This chart supports a single binding and is useful for combining a value for the root of a hierarchy.

Parent topic: Charts

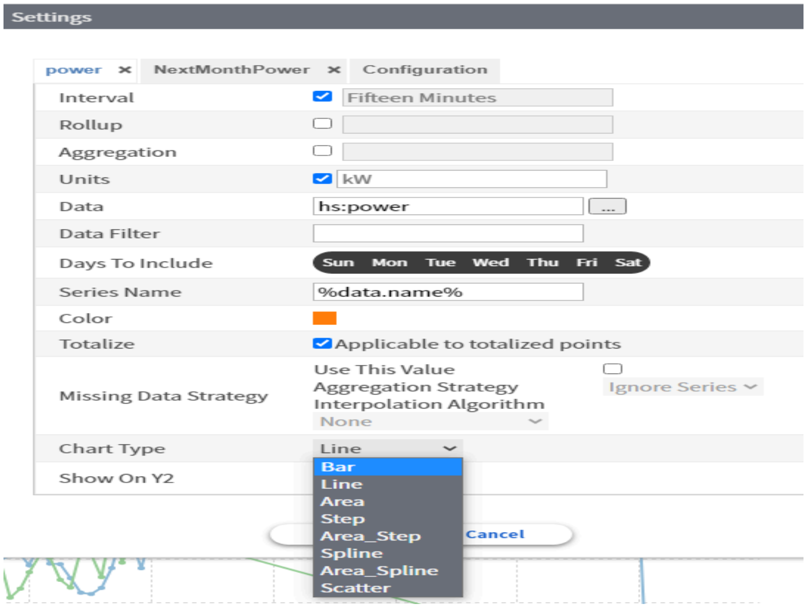
AnalyticWebChart

This multipurpose chart is capable of displaying data using a bar, line, area, step, area step, spline, area spline or scatter chart. In addition you can enable a second y-axis. This chart requires a Px view. You can configure it on a dashboard in a Px view.

You add one of these charts to a Px view by dragging the AnalyticWebChart component from the analytics palette's Charts folder to the desired Px view in the Px editor mode.

There is more than one way to select the chart type in the Px view. You may click the Settings icon (⚙) or click the **Configuration** tab. The resulting properties view may look like this in a browser:

Figure 1. Configuration tab being used to select Chart Type



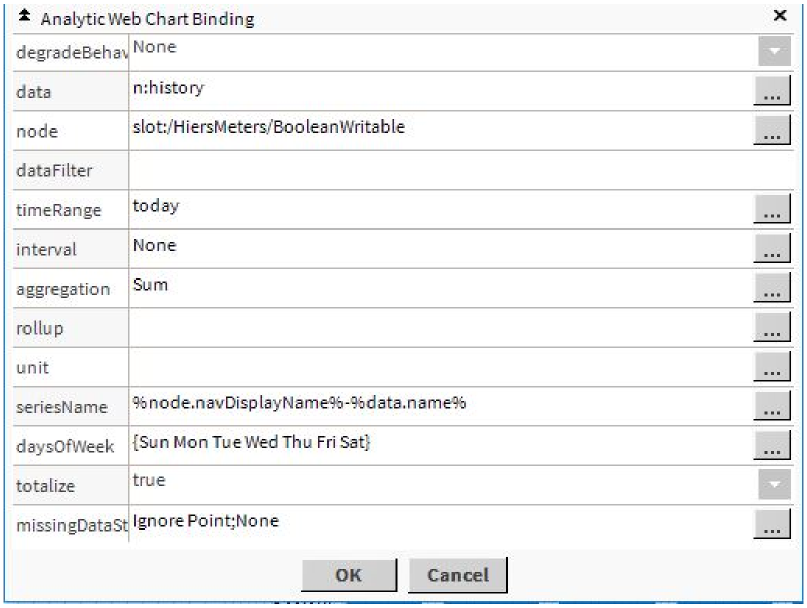
Settings

power ✕ NextMonthPower ✕ Configuration

Interval	☑ Fifteen Minutes
Rollup	□
Aggregation	□
Units	☑ kW
Data	hs:power ...
Data Filter	
Days To Include	Sun Mon Tue Wed Thu Fri Sat
Series Name	%data.name%
Color	🟧
Totalize	☑ Applicable to totalized points
Missing Data Strategy	Use This Value □ Aggregation Strategy Ignore Series Interpolation Algorithm None
Chart Type	Line
Show On Y2	

To configure the Chart Type in the Px view, click the Settings icon (⚙) and select the type of chart from the drop-down list.

Figure 2. Analytic Web Chart Binding properties



Analytic Web Chart Binding

degradeBehav	None
data	n:history
node	slot:/HiersMeters/BooleanWritable
dataFilter	
timeRange	today
interval	None
aggregation	Sum
rollup	
unit	
seriesName	%node.navDisplayName%-%data.name%
daysOfWeek	{Sun Mon Tue Wed Thu Fri Sat}
totalize	true
missingDataSt	Ignore Point;None

To view these properties, double-click the web widget in the Px editor window or Px editor Widget Tree side bar.

Property	Value	Description
degradeBehavior	Disable or None	Specifies the framework’s behavior when the binding analytic ORD cannot be resolved or used due to security permissions.
data	tag or algorithm name	Specifies the an algorithm (or tag) used to retrieve data. The purpose of an alert is to detect a fault condition (Boolean value result). In most cases, you would pick an algorithm that has a Boolean result when configuring this property for an alert. Algorithms with a numeric or enum value greater than zero are considered true and less than zero are considered false. This is how output from one algorithm becomes the input data source to another algorithm. The prefix for algorithms is alg:.
node	ORD (defaults to a relative ORD slot, which is resolved against the config space to which the Px view is assigned)	Defines the ORD to the desired slot.
dataFilter	optional NEQL query (property) or ORD parameter (dataFilter=query)	Identifies data sources in the subtree of the request node. When a predicate accepts a node that contains the desired data, the system does not search the node’s subtree for additional values (including the root node).
timeRange	drop-down list or ORD parameter (defaults to Today (current	Defines the time period used to query historical data.

Property	Value	Description
	value);timerange=option	This property is required for rollup requests (analyticRollup), trends (analyticTrend), and rollup bindings. It is optional elsewhere. It is not used if the block’s Use Request Time Range property is true and the request specifies a time range. Options range from From to All.
interval	optional drop-down list (defaults to the optimal number of records on reports); interval=option	Refers to the BInterval component, which the framework uses to identify the time between values in a trend (time series). When specified, a rollup is required, which causes the system to combine all values that fall into a single interval. Options range from None to a Year. Above the drop-down list, the Use This Value check box turns on and off the check box next to Interval in the **Settings** window (you access this window by clicking the Edit button (✎), followed by clicking the Settings button (⚙) on the chart). The availability of this check box provides an easy way for a user to enable and disable the use of intervals in chart calculations.
aggregation	check box (if optional, and) drop-down list (property, defaults to First) or ORD parameter (aggregation=option)	Configures the default function to apply when the analytic request needs to combine values from multiple data sources into a single value. This applies to both value and trend requests. If aggregation is not enabled in the binding/settings window, the aggregation value defined in the Data Definition applies to all chart bindings, reports and tables. And returns the logical “and” of Boolean values. Avg returns the statistical mean, which is determined by calculating the sum of all values and dividing by the number of values. Count returns the total number or quantity of values in a combination. If you request this value on a binding in a PX view, the system counts the number of values based on the properties defined by the data source block and the algorithm’s **Property Sheet**. First returns the first value in the combination. Last returns the last value in the combination. Max returns the highest value in the combination.

Property	Value	Description
		Median returns the value in the middle of a sorted combination—the number that separates the higher half from the lower half. Min returns the lowest value in the combination. Mode returns the statistically most frequently occurring number in the combination. Or returns the logical “or” of Boolean values. Range returns the statistical difference between the largest and smallest values in the combination. Sum adds together all values in the combination resulting in a single value. Load Factor returns the average value divided by peak value. Std Dev returns the standard deviation of the values in the combination.
rollup	Check box (if optional, and) drop-down list or ORD parameter (when configured in the data definition, defaults to First, when configured elsewhere, defaults to the value as defined in the Data Definition); rollup=option	Configures the default function to apply when the analytic request needs to rollup records from a single data source into less granular records. This typically only applies to trend request, but may also apply to value requests where the Id is an algorithm that contains a block like Runtime or Sliding Window, which processes a trend request. If rollup is not enabled in the binding/settings window, the rollup value configured in the Data Definition applies to all chart bindings, reports and tables. And returns the logical “and” of Boolean values. Avg returns the statistical mean, which is determined by calculating the sum of all values and dividing by the number of values. Count returns the total number or quantity of values in a combination. If you request this value on a binding in a PX view, the system counts the number of values based on the properties defined by the data source block and the algorithm’s **Property Sheet**. First returns the first value in the combination. Last returns the last value in the combination. Max returns the highest value in the combination. Median returns the value in the middle of

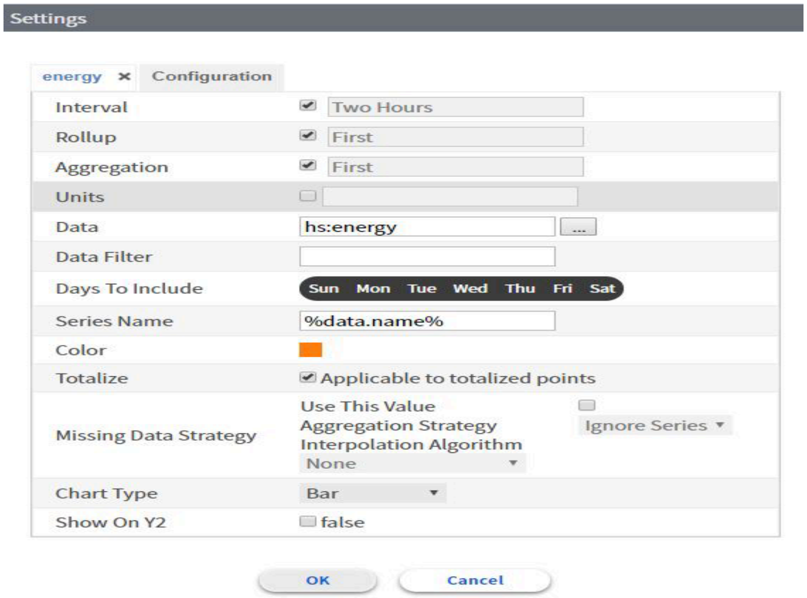
Property	Value	Description
		a sorted combination—the number that separates the higher half from the lower half. Min returns the lowest value in the combination. Mode returns the statistically most frequently occurring number in the combination. Or returns the logical “or” of Boolean values. Range returns the statistical difference between the largest and smallest values in the combination. Sum adds together all values in the combination resulting in a single value. Std Dev calculates the standard deviation of the values in the combination. Load Factor calculates the average divided by peak (Max) value.
unit	unit=option (unit of measure)	Selects the unit name and display symbol to associate with a data source.
seriesName	BFormat (defaults to %node.navDisplayName-data.name%)	Defines the node and tag name configured by the Data property. The BFormat value may contain static text or scripts that use the percentage symbol delimiter, such as %node.navDisplayName%. The default value is %node.navDisplayName%-%data.name%, which resolves to the navDisplayName of the node and the name of the tag configured by the Data property, such as ”Building1-Power.” The %data.name% script drops the tags namespace, such as “hs:” or the algorithm reference, such as “alg:.” If you enter an invalid value, the system displays, “Problem with seriesName BFormat,” and logs a message in the station console. The name you enter here displays as the legend and tool tip in the Web chart. If left blank, no legend displays for the tool tip.
daysOfWeek	ORD scheme	Specifies the days of the week to include.
totalize	true (true) or false or hisTotEnabled=option (ORD scheme parameter)	Turns on (true) and off (false) value accumulation. By default, the framework totalizes (accumulates) all consumption history values in charts, tables and reports. To prevent cumulative values, disable this property (set it to false).

Property	Value	Description
missingDataStrategy	additional properties	Configures how the framework handles missing data in a series when processing analytic requests and one or more records are missing for an interval. It applies when even a single record for an interval is missing. It does not apply to value requests. Refer to Missing Data Strategy

Parent topic: Charts

Web chart Ux settings window

This window configures AnalyticWebChart properties in the live Px view.



Settings

energy ✕ | Configuration

Interval ☑ Two Hours

Rollup ☑ First

Aggregation ☑ First

Units □

Data hs:energy ...

Data Filter

Days To Include Sun Mon Tue Wed Thu Fri Sat

Series Name %data.name%

Color 🟧

Totalize ☑ Applicable to totalized points

Missing Data Strategy Use This Value □ Aggregation Strategy Ignore Series ▾ Interpolation Algorithm None ▾

Chart Type Bar ▾

Show On Y2 □ false

You access this window by clicking the Settings icon (⚙) on the web widget.

Property	Value	Description
interval	optional drop-down list (defaults to the optimal number of records on reports); interval=option	Refers to the BInterval component, which the framework uses to identify the time between values in a trend (time series). When specified, a rollup is required, which causes the system to combine all values that fall into a single interval. Options range from None to a Year. Above the drop-down list, the Use This Value check box turns on and off the check box next to Interval in the **Settings** window (you access this window by clicking the Edit button (✎), followed by clicking the Settings button (⚙) on the chart). The

Property	Value	Description
		availability of this check box provides an easy way for a user to enable and disable the use of intervals in chart calculations.
rollup	Check box (if optional, and) drop-down list or ORD parameter (when configured in the data definition, defaults to First, when configured elsewhere, defaults to the value as defined in the Data Definition); rollup=option	Configures the default function to apply when the analytic request needs to rollup records from a single data source into less granular records. This typically only applies to trend request, but may also apply to value requests where the Id is an algorithm that contains a block like Runtime or Sliding Window, which processes a trend request. If rollup is not enabled in the binding/settings window, the rollup value configured in the Data Definition applies to all chart bindings, reports and tables. And returns the logical “and” of Boolean values. Avg returns the statistical mean, which is determined by calculating the sum of all values and dividing by the number of values. Count returns the total number or quantity of values in a combination. If you request this value on a binding in a PX view, the system counts the number of values based on the properties defined by the data source block and the algorithm’s **Property Sheet**. First returns the first value in the combination. Last returns the last value in the combination. Max returns the highest value in the combination. Median returns the value in the middle of a sorted combination—the number that separates the higher half from the lower half. Min returns the lowest value in the combination. Mode returns the statistically most frequently occurring number in the combination. Or returns the logical “or” of Boolean values. Range returns the statistical difference between the largest and smallest values in the combination. Sum adds together all values in the combination resulting in a single value. Std Dev calculates the standard deviation of the values in the combination.

Property	Value	Description
		Load Factor calculates the average divided by peak (Max) value.
aggregation	check box (if optional, and) drop-down list (property, defaults to First) or ORD parameter (aggregation=option)	Configures the default function to apply when the analytic request needs to combine values from multiple data sources into a single value. This applies to both value and trend requests. If aggregation is not enabled in the binding/settings window, the aggregation value defined in the Data Definition applies to all chart bindings, reports and tables. And returns the logical “and” of Boolean values. Avg returns the statistical mean, which is determined by calculating the sum of all values and dividing by the number of values. Count returns the total number or quantity of values in a combination. If you request this value on a binding in a PX view, the system counts the number of values based on the properties defined by the data source block and the algorithm’s **Property Sheet**. First returns the first value in the combination. Last returns the last value in the combination. Max returns the highest value in the combination. Median returns the value in the middle of a sorted combination—the number that separates the higher half from the lower half. Min returns the lowest value in the combination. Mode returns the statistically most frequently occurring number in the combination. Or returns the logical “or” of Boolean values. Range returns the statistical difference between the largest and smallest values in the combination. Sum adds together all values in the combination resulting in a single value. Load Factor returns the average value divided by peak value. Std Dev returns the standard deviation of the values in the combination.
unit	unit=option (unit of measure)	Selects the unit name and display symbol to associate with a data source.

Property	Value	Description
data	algorithm (or tag) name; data=algorithm or tag name	Specifies the an algorithm (or tag) used to retrieve data. The purpose of an alert is to detect a fault condition (Boolean value result). In most cases, you would pick an algorithm that has a Boolean result when configuring this property for an alert. Algorithms with a numeric or enum value greater than zero are considered true and less than zero are considered false. This is how output from one algorithm becomes the input data source to another algorithm. The prefix for algorithms is alg:.
dataFilter	optional NEQL query (property) or ORD parameter (dataFilter=query)	Identifies data sources in the subtree of the request node. When a predicate accepts a node that contains the desired data, the system does not search the node's subtree for additional values (including the root node).
Days to Include	graphic with the days	Selects which days of the week to include.
seriesName	BFormat (defaults to %node.navDisplayName-data.name%)	Defines the node and tag name configured by the Data property. The BFormat value may contain static text or scripts that use the percentage symbol delimiter, such as %node.navDisplayName%. The default value is %node.navDisplayName%-%data.name%, which resolves to the navDisplayName of the node and the name of the tag configured by the Data property, such as ”Building1-Power.” The %data.name% script drops the tags namespace, such as “hs:” or the algorithm reference, such as “alg:.” If you enter an invalid value, the system displays, “Problem with seriesName BFormat,” and logs a message in the station console. The name you enter here displays as the legend and tool tip in the Web chart. If left blank, no legend displays for the tool tip.
Color	color chooser	Configures the color to use on the chart.
totalize	true (true) or false or hisTotEnabled=option (ORD scheme parameter)	Turns on (true) and off (false) value accumulation. By default, the framework totalizes (accumulates) all consumption history values in charts, tables and reports. To prevent cumulative values, disable this property (set it to false).

Property	Value	Description
Missing Data Strategy, Use This Value	check box	Enables and disables missing data interpolation for the current value. When enabled, the framework applies this strategy to all requests. When enabled, the framework applies this strategy to all requests.
missingDataStrategy, Aggregation Strategy	drop-down list; `aggStrategy=option` (ORD scheme parameter)	Configures how the framework handles missing trend data (data in a series) when processing analytic requests and one or more records are missing for an interval. It applies when even a single record for an interval is missing. It does not apply to value requests. Note: If the analytic trend request specifies Interval = none, the framework ignores the Missing Data Strategy. Ignore Series ignores the entire series if any record even one interval is missing. Ignore Point ignores any missing records and aggregates the values in the existing records.
Missing Data Strategy, Interpolation Algorithm	drop-down list	Defines the algorithm used to interpolate values for missing values (missing records). None does not interpolate values for missing records. Linear Interpolation replaces a missing record by linearly interpolating the missing value using the prior and next records on either side of the missing record. K-Nearest Neighbor is for numeric, enum and Boolean records. This strategy replaces a missing value by calculating the majority value recorded for the item’s nearest K number of neighbors.
Missing Data Strategy, K Value	numeric field editor (defaults to =1, min = 1, max = 30	Defines the number of records used by the configured interpolation algorithm when a record is missing to calculate the interpolated value.
Chart Type	drop-down list	Selects the type of AnalyticWebChart: bar, line, spline, area, area step, scatter, etc. Choosing a type changes the chart rendering.
Show on Y2	true or false (default)	Displays the data for the selected series using the y2–axis (right hand side of chart).

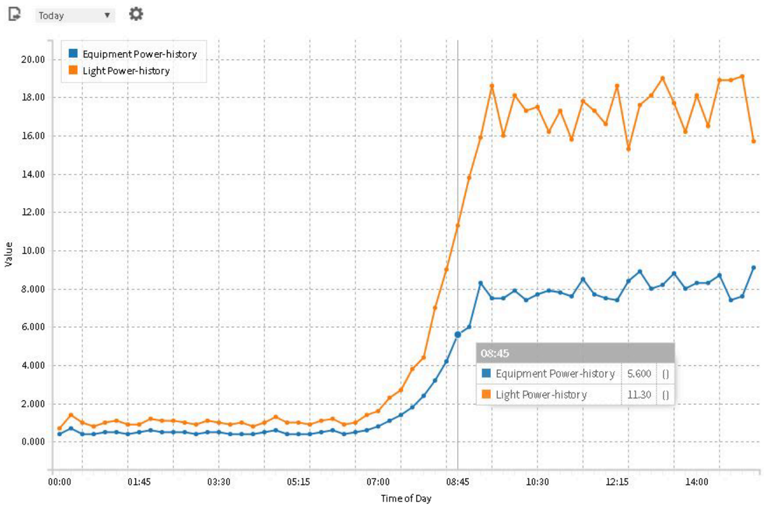
Parent topic: Charts

Average Profile chart

This chart shows a pattern that represents the average of a data value over a specified time period. It provides flexibility when identifying high values at various times within a selected time period. This chart supports multiple bindings.

You add this chart to a Px view by dragging the AverageProfileChart component from the analytics palette’s Charts folder to the desired Px view in the Px editor mode.

Figure 1. Average Profile chart



This chart plots time of day (x-axis) against the average of the data values (y-axis). If the requested time range spans more than one day, the block rolls up the records with matching times of day using the average function. Each binding displays using a color identified by the color key in the upper right corner of the chart. The tool tip shows the value for each binding at the interval selected.

The drop-down list in the upper left defines the Time Range, which requires a value greater than or equal to one day.

Clicking the Settings icon (⚙) next to the drop-down list opens the **Settings** window:

- Configure Interval to be less than one day.
- Configure Time Range to be at least one day.

Parent topic: Charts

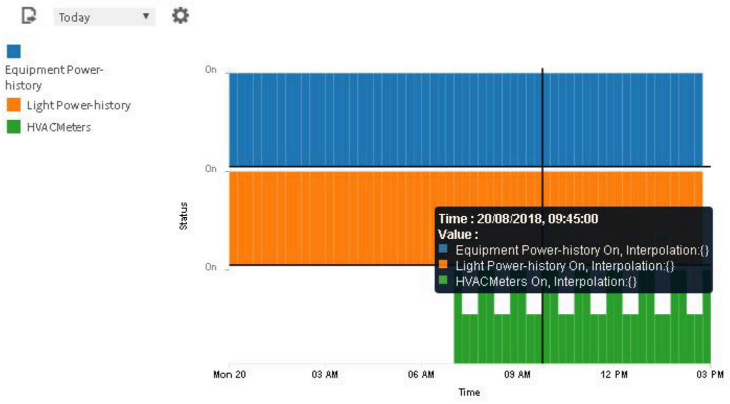
Equipment Operation chart

This chart indicates when a piece of equipment is on and off. It supports multiple bindings, each binding representing a piece of equipment.

You add this chart to a Px view by dragging the EquipmentOperationChart component from the analytics palette’s Charts folder to a the desired Px view in Px editor mode.

When observing at what time the power to a piece of equipment went off and on, this chart can expose a trend. Comparing the same time period, for example, with a temperature spike that appears on a chart may indicate a problem. This chart is useful in a dashboard.

Figure 1. Equipment Operation chart


Today
Equipment Power-history
Light Power-history
HVACMeters
On
On
On
Status
Time : 20/08/2018, 09:45:00
Value :
Equipment Power-history On, Interpolation:{}
Light Power-history On, Interpolation:{}
HVACMeters On, Interpolation:{}
Mon 20
03 AM
06 AM
09 AM
12 PM
03 PM
Time

The tool tip identifies the node and indicates the time. When you can drag onto the view an additional equipment point the system automatically assigns a color.

Click the settings icon (⚙) to modify chart properties, such as Interval.

You can export and save this chart.

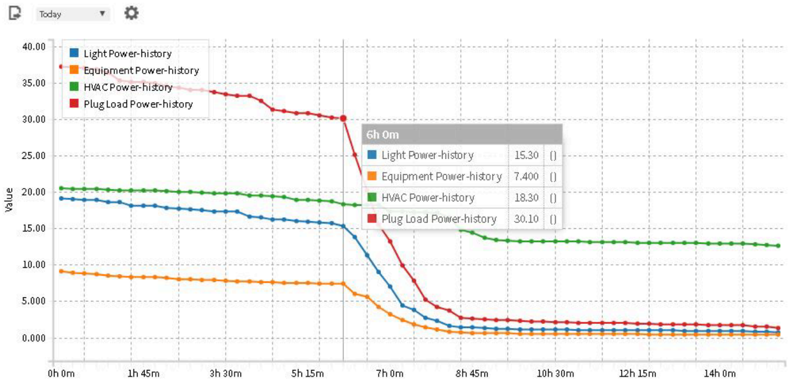
Parent topic: Charts

Load duration chart

This chart summarizes how long a value was at a certain value. It can help you observe the duration of peak demand levels for the purpose of identifying correct demand limiting strategies.

You add this chart to a Px view by dragging the **LoadDurationChart** component from the analytics palette's Charts folder to the desired Px view in Px editor mode.

Figure 1. Load Duration chart



Parent topic: Charts

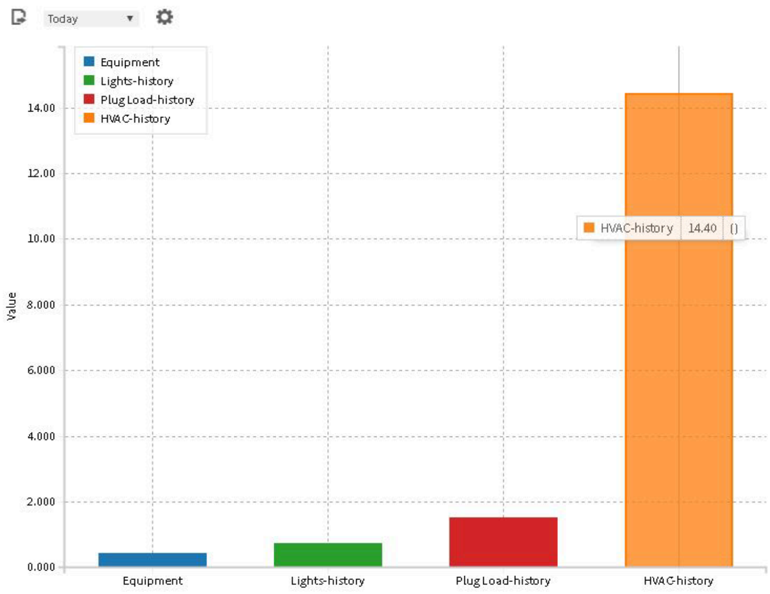
Ranking chart

This chart compares values from selected nodes using vertical bars. It supports multiple bindings. Using it you can quickly compare similar buildings or systems to find items with unexpectedly high or low values, such as one building using more energy per month than other similar buildings in the portfolio. If you identify an item with abnormal values, use other analytic web charts and reports to further analyze the data and determine the root cause.

You add this chart to a Px view by dragging the RankingChart component from the analytics palette's Charts folder to a the desired Px view in the Px editor mode.

The important properties to set are Rollup, Aggregation and Time Range. The chart uses the Analytic Web Rollup Binding, which automatically adjusts the interval to match the requested time range, and returns a single value for each binding.

Figure 1. Ranking chart



This chart supports multiple bindings. It displays each binding as a bar from left to right in ascending, increasing-values order, regardless of the actual order of the bindings on the web chart widget. The series name for each binding is displayed below the bar, in the legend. Depending on the number of bars and the length of the series names, text may be truncated below each bar. As a best practice, configure short series names, if possible.

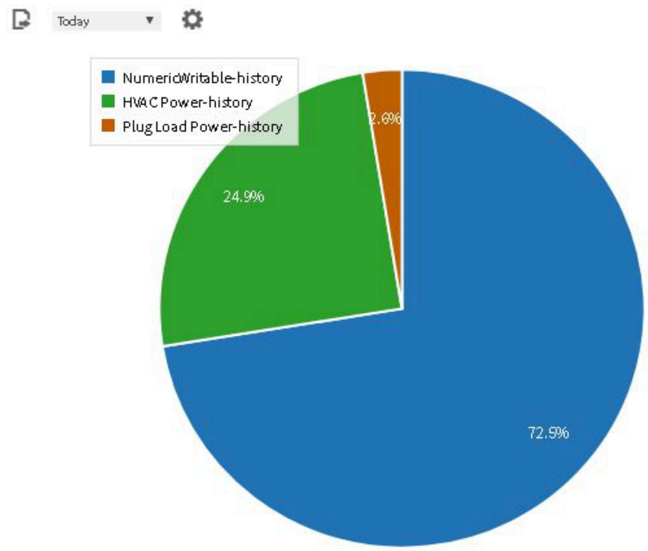
Parent topic: Charts

Relative contribution chart

This pie chart plots the contributions from individual pieces of equipment (or any data model node) to the total value of a group or a site. For example, if meters are grouped appropriately, you can identify that HVAC consumption is 45%, lighting consumption is 35%, and other loads contribute 20% to the total consumption of a particular building or campus.

You add this chart to a Px view by dragging the RelativeContributionChart component from the analytics palette's Charts folder to the desired Px view in the Px editor mode.

Figure 1. Relative Contribution chart



This chart can display 20 dynamic colors. It is sorted in descending order beginning the largest segment at 12 o'clock position in the clockwise direction. You may configure this chart may to display bars instead of a pie, in which case the chart sorts the bars from left to right in descending order. The color legend along the top, right side of the chart also presents in order from largest to smallest.

Parent topic: Charts

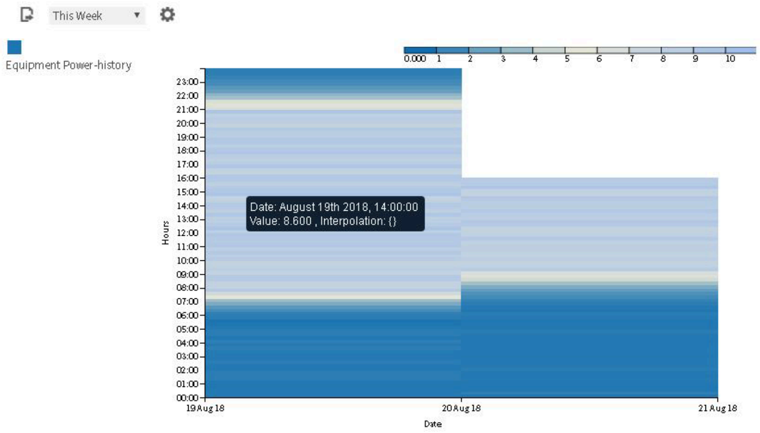
Spectrum chart

Using color coding, the Spectrum Chart provides insight into the reasonableness of the value obtained from a data or aggregated point. A quick glance at a consistent color pattern, which indicates valid data, can visually confirm an expected condition, allowing you to move on to other functions. A quick glance at an inconsistent color pattern can draw immediate attention to a situation requiring further analysis.

You add this chart by dragging the SpectrumChart component from the analytics palette's Charts folder to the desired Px view in Px editor mode.

When the chart loads a specific pattern emerges based on the arrangement of colors.

Figure 1. Spectrum Summary chart



This chart plots days (x-axis) against the hours in a day (y-axis). This chart shows values for the entire time range where colored swatches indicate the actual values for a specific time and day during the period. Vertical columns represen each day in the period, and the size of the swatches in a vertical column are based on the configured interval. If the interval is 15 minutes there should be 96 swatches in each vertical column.

Outliers in the data appear as an inconsistent pattern. Passing the cursor over the chart rectangles opens a tool tip that identifies the date the system collected the data. Clicking the wrench icon in the top left opens the settings window shown superimposed on the chart.

This chart is limited to a single binding with a minimum Time Range of 1 Day, a maximum Interval of 1 Day, and a minimum of three data points. To calculate the data points, divide Time Range by Interval. If the quotient is less than three (3) adjust the Time Range or Interval so that the quotient is three or more.

Note: For best results, do not reduce the height of this chart below 480 Abs.

Parent topic: Charts

Ord schemes

You use ORD schemes to extract information from the Niagara Analytics Framework.

- **Value requests (analyticValue:)**
  Value requests generate BStatusValues.
- **Rollup requests (analyticRollup:)**
  Rollup requests generate a StatusValue by combining all values of an underlying trend.
- **Trend requests (analyticTrend:)**
  Trend requests generate BITables representing a time series.
- **Analytic multi-trend requests (analyticMultiTrend:)**
  Analytic multi-trends are similar to trend requests, but they also consolidate trend data from multiple ORDs into a single BITable.
- **Analytic multi-rollup requests (analyticMultiRollup:)**
  These requests are similar to trend requests, but they consolidate trend data from multiple ORDs into a single value.
- **Analytic alerts requests (alerts:)**
  Alert requests generate BITables representing all active alerts.

Value requests (analyticValue:)

Value requests generate BStatusValues.

Properties

The scheme is similar to the query string of a URL, where properties follow a colon, name and values are joined with an equal sign, and pairs are separated with the ampersand. For example:

```
slot:/Drivers/foo|analyticValue:data=n:realPower&aggregation=sum
```

ORD scheme parameter	Syntax	Description
aggregation	check box (if optional, and) drop-down list (property, defaults to First) or ORD parameter (aggregation=option)	Configures the default function to apply when the analytic request needs to combine values from multiple data sources into a single value. This applies to both value and trend requests. If aggregation is not enabled in the binding/settings window, the aggregation value defined in the Data Definition applies to all chart bindings, reports and tables. And returns the logical “and” of Boolean values. Avg returns the statistical mean, which is determined by calculating the sum of all values and dividing by the number of values. Count returns the total number or quantity of values in a combination. If you request this value on a binding in a PX view, the system counts the number of values based on the properties defined by the data source block and the algorithm’s **Property Sheet**.

ORD scheme parameter	Syntax	Description
		First returns the first value in the combination. Last returns the last value in the combination. Max returns the highest value in the combination. Median returns the value in the middle of a sorted combination—the number that separates the higher half from the lower half. Min returns the lowest value in the combination. Mode returns the statistically most frequently occurring number in the combination. Or returns the logical "or" of Boolean values. Range returns the statistical difference between the largest and smallest values in the combination. Sum adds together all values in the combination resulting in a single value. Load Factor returns the average value divided by peak value. Std Dev returns the standard deviation of the values in the combination.
dataFilter	optional NEQL query (property) or ORD parameter (`dataFilter=query`)	Identifies data sources in the subtree of the request node. When a predicate accepts a node that contains the desired data, the system does not search the node's subtree for additional values (including the root node).

data	algorithm (or tag) name;
	`data=algorithm` or `tag name`
		Specifies the an algorithm (or tag) used to retrieve data.
		The purpose of an alert is to detect a fault condition (Boolean value result). In most cases, you would pick an algorithm that has a Boolean result when configuring this property for an alert. Algorithms with a numeric or enum value greater than zero are considered true and less than zero are considered false.
		This is how output from one algorithm becomes the input data source to another algorithm. The prefix for algorithms is `alg:`.

useCache	optional ORD parameter true or false; `useCache=option`	Turns on and off the use of cache memory. true causes the framework to request cache memory for value, trend and rollup requests that take longer than 200ms to process.

ORD scheme parameter	Syntax	Description
		false uses no cache. This property is not related to the data availability cache.
unit	unit=option (unit of measure)	Selects the unit name and display symbol to associate with a data source.

Parent topic: Ord schemes

Rollup requests (analyticRollup:)

Rollup requests generate a StatusValue by combining all values of an underlying trend.

This ORD scheme is similar to the query string of a URL, where properties follow a colon, name and values are joined with an equal character, and pairs are separated with an ampersand character. For example:

```
slot:/Drivers/foo|analyticRollup:data=n:realPower&timeRange=lastYear:rollup=median:ti
```

ORD scheme parameter	Syntax	Description
aggregation	aggregation=option (defaults to First)	Configures the default function to apply when the analytic request needs to combine values from multiple data sources into a single value. This applies to both value and trend requests. If aggregation is not enabled in the binding/settings window, the aggregation value defined in the Data Definition applies to all chart bindings, reports and tables. And returns the logical “and” of Boolean values. Avg returns the statistical mean, which is determined by calculating the sum of all values and dividing by the number of values. Count returns the total number or quantity of values in a combination. If you request this value on a binding in a PX view, the system counts the number of values based on the properties defined by the data source block and the algorithm’s **Property Sheet**. First returns the first value in the combination. Last returns the last value in the combination. Max returns the highest value in the combination. Median returns the value in the middle of a sorted combination—the number that separates the higher half from the lower half. Min returns the lowest value in the combination.

ORD scheme parameter	Syntax	Description
		Mode returns the statistically most frequently occurring number in the combination. Or returns the logical “or” of Boolean values. Range returns the statistical difference between the largest and smallest values in the combination. Sum adds together all values in the combination resulting in a single value. Load Factor returns the average value divided by peak value. Std Dev returns the standard deviation of the values in the combination.
Data	algorithm (or tag) name; `data=algorithm` or `tag name`	Specifies the an algorithm (or tag) used to retrieve data. The purpose of an alert is to detect a fault condition (Boolean value result). In most cases, you would pick an algorithm that has a Boolean result when configuring this property for an alert. Algorithms with a numeric or enum value greater than zero are considered true and less than zero are considered false. This is how output from one algorithm becomes the input data source to another algorithm. The prefix for algorithms is `alg:`.
Data Filter, dataFlter	optional NEQL query (property) or ORD parameter (`dataFilter=query`)	Identifies data sources in the subtree of the request node. When a predicate accepts a node that contains the desired data, the system does not search the node’s subtree for additional values (including the root node).
Interval, interval	optional drop-down list (defaults to the optimal number of records on reports); `interval=option`	Refers to the BInterval component, which the framework uses to identify the time between values in a trend (time series). When specified, a rollup is required, which causes the system to combine all values that fall into a single interval. Options range from None to a Year. Above the drop-down list, the Use This Value check box turns on and off the check box next to Interval in the **Settings** window (you access this window by clicking the Edit button (✎), followed by clicking the Settings button (⚙) on the chart). The availability of this check box provides an easy way for a user to enable and disable the use of intervals in chart calculations.
normalizeTime	true (default) or false;	Manages how time is represented.

ORD scheme parameter	Syntax	Description
	normalizeTime=option	true converts time to the time zone of the request. This preserves time properties, such as the hour of day, while changing the time zone. false does not perform the conversion.
rollup	Check box (if optional, and) drop-down list or ORD parameter (when configured in the data definition, defaults to First, when configured elsewhere, defaults to the value as defined in the Data Definition); rollup=option	Configures the default function to apply when the analytic request needs to rollup records from a single data source into less granular records. This typically only applies to trend request, but may also apply to value requests where the Id is an algorithm that contains a block like Runtime or Sliding Window, which processes a trend request. If rollup is not enabled in the binding/ settings window, the rollup value configured in the Data Definition applies to all chart bindings, reports and tables. And returns the logical “and” of Boolean values. Avg returns the statistical mean, which is determined by calculating the sum of all values and dividing by the number of values. Count returns the total number or quantity of values in a combination. If you request this value on a binding in a PX view, the system counts the number of values based on the properties defined by the data source block and the algorithm’s **Property Sheet**. First returns the first value in the combination. Last returns the last value in the combination. Max returns the highest value in the combination. Median returns the value in the middle of a sorted combination—the number that separates the higher half from the lower half. Min returns the lowest value in the combination. Mode returns the statistically most frequently occurring number in the combination. Or returns the logical “or” of Boolean values. Range returns the statistical difference between the largest and smallest values in the combination. Sum adds together all values in the combination resulting in a single value.

ORD scheme parameter	Syntax	Description
		Std Dev calculates the standard deviation of the values in the combination. Load Factor calculates the average divided by peak (Max) value.
timeRange	drop-down list or ORD parameter (defaults to Today (current value);timerange=option	Defines the tie period used to query historical data. This property is required for rollup requests (analyticRollup), trends (analyticTrend), and rollup bindings. It is optional elsewhere. It is not used if the block's Use Request Time Range property is true and the request specifies a time range. Options range from From to All.
useCache	optional ORD parameter true or false; useCache=option	Turns on and off the use of cache memory. true causes the framework to request cache memory for value, trend and rollup requests that take longer than 200ms to process. false uses no cache. This property is not related to the data availability cache.
Series Name, seriesName	BFormat (defaults to %node.navDisplayName-data.name%)	Defines the node and tag name configured by the Data property. The BFormat value may contain static text or scripts that use the percentage symbol delimiter, such as %node.navDisplayName%. The default value is %node.navDisplayName%-%data.name%, which resolves to the navDisplayName of the node and the name of the tag configured by the Data property, such as ”Building1-Power.” The %data.name% script drops the tags namespace, such as “hs:” or the algorithm reference, such as “alg:.” If you enter an invalid value, the system displays, “Problem with seriesName BFormat,” and logs a message in the station console. The name you enter here displays as the legend and tool tip in the Web chart. If left blank, no legend displays for the tool tip.
Totalize, totalize	true (true) or false or hisTotEnabled=option (ORD scheme parameter)	Turns on (true) and off (false) value accumulation. By default, the framework totalizes (accumulates) all consumption history values in charts, tables and reports. To prevent cumulative values, disable this property (set it to false).

ORD scheme parameter	Syntax	Description
daysOfWeek	dow=option	Specifies the days of the week to include.
aggStrategy	aggStrategy=option	Selects the missing data aggregation strategy, which defines how to handle data in a series when even a single record for an interval is missing. Ignore Point tells the system to ignore any missing records and aggregate the values in the existing records. Ignore Series tells the system to ignore the entire series if the record for even one interval in the series is missing.
intpAlgorithm	intpAlgorithm=option (ORD scheme parameter)	Selects the missing data interpolation algorithm, which defines the value to replace a missing value. Linear Interpolation replaces a missing value by linearly interpolating the missing value. K-Nearest Neighbor is for numeric, enum and Boolean records. This strategy replaces a missing value by calculating the majority value recorded for the item's nearest neighbors.
knnValue	knnvalue=n (ORD scheme parameter	Indicates the number of neighbors to a missing data item that the interpolation algorithm should include in its calculation.
unit	unit=option (unit of measure)	Selects the unit name and display symbol to associate with a data source.

Parent topic: Ord schemes

Trend requests (analyticTrend:)

Trend requests generate BITables representing a time series.

Columns

This ORD scheme is similar to the query string of a URL, where properties follow a colon, name and values are joined with an equal character, and pairs are separated with an ampersand character. For example:

```
slot:/Drivers/foo|analyticTrend:data=n:realPower&timeRange=lastYear
```

Column	Description
Timestamp	Reports the date and time the event occurred.
value	BBoolean, BDouble, BEnum, or BString
status	Reports the BStatus for the value.

ORD scheme parameter	Syntax	Description
Aggregation	check box (if optional, and) drop-down list (property, defaults to First) or ORD parameter (aggregation=option)	If aggregation is not enabled in the binding/settings window, the aggregation value defined in the Data Definition applies to all chart bindings, reports and tables. And returns the logical “and” of Boolean values. Avg returns the statistical mean, which is determined by calculating the sum of all values and dividing by the number of values. Count returns the total number or quantity of values in a combination. If you request this value on a binding in a PX view, the system counts the number of values based on the properties defined by the data source block and the algorithm’s **Property Sheet**. First returns the first value in the combination. Last returns the last value in the combination. Max returns the highest value in the combination. Median returns the value in the middle of a sorted combination—the number that separates the higher half from the lower half. Min returns the lowest value in the combination. Mode returns the statistically most frequently occurring number in the combination. Or returns the logical “or” of Boolean values. Range returns the statistical difference between the largest and smallest values in the combination. Sum adds together all values in the combination resulting in a single value. Load Factor returns the average value divided by peak value. Std Dev returns the standard deviation of the values in the combination.
Data	algorithm (or tag) name; data=algorithm or tag name	Specifies the an algorithm (or tag) used to retrieve data. The purpose of an alert is to detect a fault condition (Boolean value result). In most cases, you would pick an algorithm that has a Boolean result when configuring this property for an alert. Algorithms with a numeric or enum value greater than zero are considered true and less than zero are considered false.

ORD scheme parameter	Syntax	Description
		This is how output from one algorithm becomes the input data source to another algorithm. The prefix for algorithms is `alg:`.
Data Filter, dataFlter	optional NEQL query (property) or ORD parameter (`dataFilter=query`)	Identifies data sources in the subtree of the request node. When a predicate accepts a node that contains the desired data, the system does not search the node's subtree for additional values (including the root node).
Interval, interval	optional drop-down list (defaults to the optimal number of records on reports); `interval=option`	Refers to the BInterval component, which the framework uses to identify the time between values in a trend (time series). When specified, a rollup is required, which causes the system to combine all values that fall into a single interval. Options range from None to a Year. Above the drop-down list, the Use This Value check box turns on and off the check box next to Interval in the **Settings** window (you access this window by clicking the Edit button (✎), followed by clicking the Settings button (⚙) on the chart). The availability of this check box provides an easy way for a user to enable and disable the use of intervals in chart calculations.
normalizeTime	true (default) or false; `normalizeTime=option`	Manages how time is represented. true converts time to the time zone of the request. This preserves time properties, such as the hour of day, while changing the time zone. false does not perform the conversion.
Rollup	check box (if optional, and) drop-down list or ORD parameter (when configured in the data definition, defaults to First, when configured elsewhere, defaults to the value as defined in the Data Definition); `rollup=option`	If rollup is not enabled in the binding/ settings window, the rollup value configured in the Data Definition applies to all chart bindings, reports and tables. And returns the logical "and" of Boolean values. Avg returns the statistical mean, which is determined by calculating the sum of all values and dividing by the number of values. Count returns the total number or quantity of values in a combination. If you request this value on a binding in a PX view, the system counts the number of values based on the properties defined by the data source block and the algorithm's **Property Sheet**. First returns the first value in the

ORD scheme parameter	Syntax	Description
		combination. Last returns the last value in the combination. Max returns the highest value in the combination. Median returns the value in the middle of a sorted combination—the number that separates the higher half from the lower half. Min returns the lowest value in the combination. Mode returns the statistically most frequently occurring number in the combination. Or returns the logical “or” of Boolean values. Range returns the statistical difference between the largest and smallest values in the combination. Sum adds together all values in the combination resulting in a single value. Std Dev calculates the standard deviation of the values in the combination. Load Factor calculates the average divided by peak (Max) value.
timeRange	`timerange=option`	This property is required for rollup requests (analyticRollup), trends (analyticTrend), and rollup bindings. It is optional elsewhere. It is not used if the block’s Use Request Time Range property is true and the request specifies a time range. Options range from From to All.
useCache	optional ORD parameter true or false; `useCache=option`	Turns on and off the use of cache memory. true causes the framework to request cache memory for value, trend and rollup requests that take longer than 200ms to process. false uses no cache. This property is not related to the data availability cache.
Series Name, seriesName	BFormat (defaults to `%node.navDisplayName-data.name%`)	Defines the node and tag name configured by the Data property. The BFormat value may contain static text or scripts that use the percentage symbol delimiter, such as %node.navDisplayName%. The default value is %node.navDisplayName%-%data.name%, which resolves to the navDisplayName of the node and the name of the tag configured by the Data

ORD scheme parameter	Syntax	Description
		property, such as ”Building1-Power.” The %data.name% script drops the tags namespace, such as “hs:” or the algorithm reference, such as “alg:.” If you enter an invalid value, the system displays, “Problem with seriesName BFormat,” and logs a message in the station console. The name you enter here displays as the legend and tool tip in the Web chart. If left blank, no legend displays for the tool tip.
Totalize, totalize	true (true) or false or hisTotEnabled=option (ORD scheme parameter)	Turns on (true) and off (false) value accumulation. By default, the framework totalizes (accumulates) all consumption history values in charts, tables and reports. To prevent cumulative values, disable this property (set it to false).
daysOfWeek	dow=option	Specifies the days of the week to include.
aggStrategy	aggStrategy=option	Selects the missing data aggregation strategy, which defines how to handle data in a series when even a single record for an interval is missing. Ignore Point tells the system to ignore any missing records and aggregate the values in the existing records. Ignore Series tells the system to ignore the entire series if the record for even one interval in the series is missing.
intpAlgorithm	intpAlgorithm=option (ORD scheme parameter)	Selects the missing data interpolation algorithm, which defines the value to replace a missing value. Linear Interpolation replaces a missing value by linearly interpolating the missing value. K-Nearest Neighbor is for numeric, enum and Boolean records. This strategy replaces a missing value by calculating the majority value recorded for the item’s nearest neighbors.
knnValue	knnvalue=n (ORD scheme parameter	Indicates the number of neighbors to a missing data item that the interpolation algorithm should include in its calculation.
unit	unit=option (unit of measure)	Selects the unit name and display symbol to associate with a data source.

Parent topic: Ord schemes

Analytic multi-trend requests (analyticMultiTrend:)

Analytic multi-trends are similar to trend requests, but they also consolidate trend data from multiple ORDs into a single BITable.

The list of ORDs formed depends on the multiOrd parameter passed as part of ORD query. The consolidation depends on the aggMode and agg properties. If aggMode is true, the system consolidates the data from multiple ORDs using the combination defined by the agg parameter. Else, it creates a multi-column table, which has a column for the output of each ORD query.

For example:

```
node=slot:/Drivers/BacnetNetwork/Building1/ElectricMeter,node=slot:
/Drivers/ModbusNetwork/Building2/ElectricMeter
```

ORD scheme parameter	Syntax	Description
aggregation	aggregation=option (defaults to First)	Configures the default function to apply when the analytic request needs to combine values from multiple data sources into a single value. This applies to both value and trend requests. If aggregation is not enabled in the binding/settings window, the aggregation value defined in the Data Definition applies to all chart bindings, reports and tables. And returns the logical “and” of Boolean values. Avg returns the statistical mean, which is determined by calculating the sum of all values and dividing by the number of values. Count returns the total number or quantity of values in a combination. If you request this value on a binding in a PX view, the system counts the number of values based on the properties defined by the data source block and the algorithm’s **Property Sheet**. First returns the first value in the combination. Last returns the last value in the combination. Max returns the highest value in the combination. Median returns the value in the middle of a sorted combination—the number that separates the higher half from the lower half. Min returns the lowest value in the combination. Mode returns the statistically most frequently occurring number in the combination. Or returns the logical “or” of Boolean values.

ORD scheme parameter	Syntax	Description
		Range returns the statistical difference between the largest and smallest values in the combination. Sum adds together all values in the combination resulting in a single value. Load Factor returns the average value divided by peak value. Std Dev returns the standard deviation of the values in the combination.
Data	algorithm (or tag) name; data=algorithm or tag name	Specifies the an algorithm (or tag) used to retrieve data. The purpose of an alert is to detect a fault condition (Boolean value result). In most cases, you would pick an algorithm that has a Boolean result when configuring this property for an alert. Algorithms with a numeric or enum value greater than zero are considered true and less than zero are considered false. This is how output from one algorithm becomes the input data source to another algorithm. The prefix for algorithms is alg:.
Data Filter, dataFlter	optional NEQL query (property) or ORD parameter (dataFilter=query)	Identifies data sources in the subtree of the request node. When a predicate accepts a node that contains the desired data, the system does not search the node’s subtree for additional values (including the root node).
Interval, interval	optional drop-down list (defaults to the optimal number of records on reports); interval=option	Refers to the BInterval component, which the framework uses to identify the time between values in a trend (time series). When specified, a rollup is required, which causes the system to combine all values that fall into a single interval. Options range from None to a Year. Above the drop-down list, the Use This Value check box turns on and off the check box next to Interval in the **Settings** window (you access this window by clicking the Edit button (✎), followed by clicking the Settings button (⚙) on the chart). The availability of this check box provides an easy way for a user to enable and disable the use of intervals in chart calculations.
normalizeTime	true (default) or false; normalizeTime=option	Manages how time is represented. true converts time to the time zone of the request. This preserves time properties, such as the hour of day, while changing the time zone.

ORD scheme parameter	Syntax	Description
		false does not perform the conversion.
rollup	rollup=option	Configures the default function to apply when the analytic request needs to rollup records from a single data source into less granular records. This typically only applies to trend request, but may also apply to value requests where the Id is an algorithm that contains a block like Runtime or Sliding Window, which processes a trend request. If rollup is not enabled in the binding/ settings window, the rollup value configured in the Data Definition applies to all chart bindings, reports and tables. And returns the logical “and” of Boolean values. Avg returns the statistical mean, which is determined by calculating the sum of all values and dividing by the number of values. Count returns the total number or quantity of values in a combination. If you request this value on a binding in a PX view, the system counts the number of values based on the properties defined by the data source block and the algorithm’s **Property Sheet**. First returns the first value in the combination. Last returns the last value in the combination. Max returns the highest value in the combination. Median returns the value in the middle of a sorted combination—the number that separates the higher half from the lower half. Min returns the lowest value in the combination. Mode returns the statistically most frequently occurring number in the combination. Or returns the logical “or” of Boolean values. Range returns the statistical difference between the largest and smallest values in the combination. Sum adds together all values in the combination resulting in a single value. Std Dev calculates the standard deviation of the values in the combination. Load Factor calculates the average divided by peak (Max) value.

ORD scheme parameter	Syntax	Description
timeRange	timerange=option	Defines the time period used to query historical data. This property is required for rollup requests (analyticRollup), trends (analyticTrend), and rollup bindings. It is optional elsewhere. It is not used if the block's Use Request Time Range property is true and the request specifies a time range. Options range from From to All.
useCache	optional ORD parameter true or false; useCache=option	Turns on and off the use of cache memory. true causes the framework to request cache memory for value, trend and rollup requests that take longer than 200ms to process. false uses no cache. This property is not related to the data availability cache.
Series Name, seriesName	BFormat (defaults to %node.navDisplayName-data.name%)	Defines the node and tag name configured by the Data property. The BFormat value may contain static text or scripts that use the percentage symbol delimiter, such as %node.navDisplayName%. The default value is %node.navDisplayName%-%data.name%, which resolves to the navDisplayName of the node and the name of the tag configured by the Data property, such as ”Building1-Power.” The %data.name% script drops the tags namespace, such as “hs:” or the algorithm reference, such as “alg:.” If you enter an invalid value, the system displays, “Problem with seriesName BFormat,” and logs a message in the station console. The name you enter here displays as the legend and tool tip in the Web chart. If left blank, no legend displays for the tool tip.
Totalize, totalize	true (true) or false or hisTotEnabled=option (ORD scheme parameter)	Turns on (true) and off (false) value accumulation. By default, the framework totalizes (accumulates) all consumption history values in charts, tables and reports. To prevent cumulative values, disable this property (set it to false).
daysOfWeek	dow=option	Specifies the days of the week to include.
aggStrategy	aggStrategy=option	Selects the missing data aggregation strategy, which defines how to handle data in a series when even a single record for an interval is missing.

ORD scheme parameter	Syntax	Description
		Ignore Point tells the system to ignore any missing records and aggregate the values in the existing records. Ignore Series tells the system to ignore the entire series if the record for even one interval in the series is missing.
intpAlgorithm	intpAlgorithm=option (ORD scheme parameter)	Selects the missing data interpolation algorithm, which defines the value to replace a missing value. Linear Interpolation replaces a missing value by linearly interpolating the missing value. K-Nearest Neighbor is for numeric, enum and Boolean records. This strategy replaces a missing value by calculating the majority value recorded for the item’s nearest neighbors.
knnValue	knnvalue=n (ORD scheme parameter	Indicates the number of neighbors to a missing data item that the interpolation algorithm should include in its calculation.
unit	unit=option (unit of measure)	Selects the unit name and display symbol to associate with a data source.
multiOrd	comma separated list of ORD parameters	Identifies the nodes to add to the multi trend ord scheme.
aggMode	true or false	true enables agg. false returns a multi column response, which results in a union of the timestamps
agg	agg=option	Aggregates the result of all the trend requests that correspond to each of the nodes. And returns the logical “and” of Boolean values. Avg returns the statistical mean, which is determined by calculating the sum of all values and dividing by the number of values. Count returns the total number or quantity of values in a combination. If you request this value on a binding in a PX view, the system counts the number of values based on the properties defined by the data source block and the algorithm’s **Property Sheet**. First returns the first value in the combination. Last returns the last value in the combination. Max returns the highest value in the combination.

ORD scheme parameter	Syntax	Description
		Median returns the value in the middle of a sorted combination—the number that separates the higher half from the lower half. Min returns the lowest value in the combination. Mode returns the statistically most frequently occurring number in the combination. Or returns the logical "or" of Boolean values. Range returns the statistical difference between the largest and smallest values in the combination. Sum adds together all values in the combination resulting in a single value. Load Factor returns the average value divided by peak value. Std Dev returns the standard deviation of the values in the combination.

Parent topic: Ord schemes

Analytic multi-rollup requests (analyticMultiRollup:)

These requests are similar to trend requests, but they consolidate trend data from multiple ORDs into a single value.

The list of ORDs formed depends on the multiOrd property passed as part of ORD query. The consolidation depends on the aggMode and agg properties. If aggMode is true, the system consolidates the data from the multiple ORDs using a combination defined by the agg property. Else, it creates a multi-column table, which has a column for the output of each ORD query.

For example:

```
station:|slot:|analyticMultiRollup:data=hs:power&dow=
7f&timeRange=today&unit=celcius&aggMode=true&rollup=avg&aggregation=
sum&agg=sum&interval=threeHours&hisTotEnabled=false&aggStrategy=
ignorePoint&intpAlgorithm=kNearestNeighbour&knnValue=3&multiOrd=
[node=slot:/HiersMeters/Folder_2,node=slot:/Services/RoleService]
```

ORD scheme parameter	Syntax	Description
aggregation	aggregation=option (defaults to First)	Configures the default function to apply when the analytic request needs to combine values from multiple data sources into a single value. This applies to both value and trend requests. If aggregation is not enabled in the binding/settings window, the aggregation value defined in the Data Definition applies to all chart bindings, reports and tables. And returns the logical "and" of Boolean

ORD scheme parameter	Syntax	Description
		values. Avg returns the statistical mean, which is determined by calculating the sum of all values and dividing by the number of values. Count returns the total number or quantity of values in a combination. If you request this value on a binding in a PX view, the system counts the number of values based on the properties defined by the data source block and the algorithm's **Property Sheet**. First returns the first value in the combination. Last returns the last value in the combination. Max returns the highest value in the combination. Median returns the value in the middle of a sorted combination—the number that separates the higher half from the lower half. Min returns the lowest value in the combination. Mode returns the statistically most frequently occurring number in the combination. Or returns the logical "or" of Boolean values. Range returns the statistical difference between the largest and smallest values in the combination. Sum adds together all values in the combination resulting in a single value. Load Factor returns the average value divided by peak value. Std Dev returns the standard deviation of the values in the combination.

Data	algorithm (or tag) name;
	`data=algorithm` or `tag name`
		Specifies the an algorithm (or tag) used to retrieve data.
		The purpose of an alert is to detect a fault condition (Boolean value result). In most cases, you would pick an algorithm that has a Boolean result when configuring this property for an alert. Algorithms with a numeric or enum value greater than zero are considered true and less than zero are considered false.
		This is how output from one algorithm becomes the input data source to another algorithm. The prefix for algorithms is `alg:`.

ORD scheme parameter	Syntax	Description
Data Filter, dataFlter	optional NEQL query (property) or ORD parameter (dataFilter=query)	Identifies data sources in the subtree of the request node. When a predicate accepts a node that contains the desired data, the system does not search the node’s subtree for additional values (including the root node).
Interval, interval	optional drop-down list (defaults to the optimal number of records on reports); interval=option	Refers to the BInterval component, which the framework uses to identify the time between values in a trend (time series). When specified, a rollup is required, which causes the system to combine all values that fall into a single interval. Options range from None to a Year. Above the drop-down list, the Use This Value check box turns on and off the check box next to Interval in the **Settings** window (you access this window by clicking the Edit button (✎), followed by clicking the Settings button (⚙) on the chart). The availability of this check box provides an easy way for a user to enable and disable the use of intervals in chart calculations.
normalizeTime	true (default) or false; normalizeTime=option	Manages how time is represented. true converts time to the time zone of the request. This preserves time properties, such as the hour of day, while changing the time zone. false does not perform the conversion.
rollup	rollup=option	Configures the default function to apply when the analytic request needs to rollup records from a single data source into less granular records. This typically only applies to trend request, but may also apply to value requests where the Id is an algorithm that contains a block like Runtime or Sliding Window, which processes a trend request. If rollup is not enabled in the binding/ settings window, the rollup value configured in the Data Definition applies to all chart bindings, reports and tables. And returns the logical “and” of Boolean values. Avg returns the statistical mean, which is determined by calculating the sum of all values and dividing by the number of values. Count returns the total number or quantity of values in a combination. If you request this value on a binding in a PX view, the system counts the number of values based on the properties defined by the data

ORD scheme parameter	Syntax	Description
		source block and the algorithm's **Property Sheet**. First returns the first value in the combination. Last returns the last value in the combination. Max returns the highest value in the combination. Median returns the value in the middle of a sorted combination—the number that separates the higher half from the lower half. Min returns the lowest value in the combination. Mode returns the statistically most frequently occurring number in the combination. Or returns the logical "or" of Boolean values. Range returns the statistical difference between the largest and smallest values in the combination. Sum adds together all values in the combination resulting in a single value. Std Dev calculates the standard deviation of the values in the combination. Load Factor calculates the average divided by peak (Max) value.
timeRange	`timerange=option`	Defines the time period used to query historical data. This property is required for rollup requests (analyticRollup), trends (analyticTrend), and rollup bindings. It is optional elsewhere. It is not used if the block's Use Request Time Range property is true and the request specifies a time range. Options range from From to All.
useCache	optional ORD parameter true or false; `useCache=option`	Turns on and off the use of cache memory. true causes the framework to request cache memory for value, trend and rollup requests that take longer than 200ms to process. false uses no cache. This property is not related to the data availability cache.
Series Name, seriesName	BFormat (defaults to `%node.navDisplayName-data.name%`)	Defines the node and tag name configured by the Data property. The BFormat value may contain static text or scripts that use the percentage symbol delimiter, such as

ORD scheme parameter	Syntax	Description
		%node.navDisplayName%. The default value is %node.navDisplayName%-%data.name%, which resolves to the navDisplayName of the node and the name of the tag configured by the Data property, such as "Building1-Power." The %data.name% script drops the tags namespace, such as "hs:" or the algorithm reference, such as "alg:." If you enter an invalid value, the system displays, "Problem with seriesName BFormat," and logs a message in the station console. The name you enter here displays as the legend and tool tip in the Web chart. If left blank, no legend displays for the tool tip.
Totalize, totalize	true (true) or false or hisTotEnabled=option (ORD scheme parameter)	Turns on (true) and off (false) value accumulation. By default, the framework totalizes (accumulates) all consumption history values in charts, tables and reports. To prevent cumulative values, disable this property (set it to false).
daysOfWeek	dow=option	Specifies the days of the week to include.
aggStrategy	aggStrategy=option	Selects the missing data aggregation strategy, which defines how to handle data in a series when even a single record for an interval is missing. Ignore Point tells the system to ignore any missing records and aggregate the values in the existing records. Ignore Series tells the system to ignore the entire series if the record for even one interval in the series is missing.
intpAlgorithm	intpAlgorithm=option (ORD scheme parameter)	Selects the missing data interpolation algorithm, which defines the value to replace a missing value. Linear Interpolation replaces a missing value by linearly interpolating the missing value. K-Nearest Neighbor is for numeric, enum and Boolean records. This strategy replaces a missing value by calculating the majority value recorded for the item's nearest neighbors.
knnValue	knnvalue=n (ORD scheme parameter	Indicates the number of neighbors to a missing data item that the interpolation algorithm should include in its calculation.
unit	unit=option (unit of measure)	Selects the unit name and display symbol

ORD scheme parameter	Syntax	Description
		to associate with a data source.
multiOrd	comma separated list of ORD parameters	Identifies the nodes to add to the multi trend ord scheme.
aggMode	true or false	true enables agg. false returns a multi column response, which results in a union of the time stamps
agg	`agg=option`	Aggregates the result of all the trend requests that correspond to each of the nodes. And returns the logical “and” of Boolean values. Avg returns the statistical mean, which is determined by calculating the sum of all values and dividing by the number of values. Count returns the total number or quantity of values in a combination. If you request this value on a binding in a PX view, the system counts the number of values based on the properties defined by the data source block and the algorithm’s **Property Sheet**. First returns the first value in the combination. Last returns the last value in the combination. Max returns the highest value in the combination. Median returns the value in the middle of a sorted combination—the number that separates the higher half from the lower half. Min returns the lowest value in the combination. Mode returns the statistically most frequently occurring number in the combination. Or returns the logical “or” of Boolean values. Range returns the statistical difference between the largest and smallest values in the combination. Sum adds together all values in the combination resulting in a single value. Load Factor returns the average value divided by peak value. Std Dev returns the standard deviation of the values in the combination.

Parent topic: Ord schemes

Analytic alerts requests (alerts:)

Alert requests generate BITables representing all active alerts.

Columns

The ORD scheme is similar to the query string of a URL, where parameters follow a colon, name and values are joined with an equal character, and pairs are separated with an ampersand character. For example:

```
slot:/|alerts:
```

The ORD you use in the bound table needs to be in the same scope as your alert is configured. If you have alerts configured to run on a hierarchy, configure something like this for the ORD in the table binding:

```
hierarchy:/AnalyticDataModel|alerts:
```

You would not use both examples simultaneously.

Column	Description
alert	The ORD to the active alert.
node	The ORD to the node on which there is an alert.
state	`Alert` or `Normal`.
totalCost	The cost of the alert multiplied by the corresponding alert mode.

Parent topic: Ord schemes

Reports

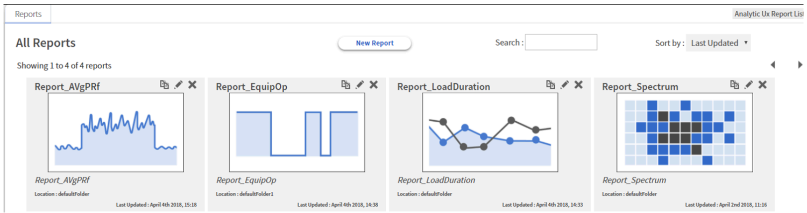
In general, a report is a document that contains information, which is organized as a narrative, graphic, or table. Reports are prepared on an ad hoc, periodic, recurring, regular, or as-required basis. They may refer to specific periods, events, occurrences or subjects. In the framework reports are located in the analytics palette. They mirror the charts by the same names. Each report has the flexibility to analyze an unlimited number of values, turning raw data into useful information for easy interpretation.

- **All Reports view**
  The **Analytic Ux Report List View** is accessible on the Reports sub folder under the **AnalyticService** and provides a graphical representation of each Ux (User Experience) report created using the view.
- **Report editor**
  Each report includes a Report Editor pane on the left hand side of the report that allows users to configure which nodes to use in the report, the Data (tag, tag group or algorithm), and the Time Range, Baseline and Normalization properties. Some reports may not include all of the configuration properties.
- **Aggregation report**
  This report plots aggregation (consumption) against demand using total, peak, minimum, and load factor values at various sites, meters, time periods and commodities. With this information you can increase total energy procured and determine complementary loads to improve your load factor. When negotiating a contract, this information can provide accurate consumption patterns versus the arbitrary classifications used by commercial and industrial customers.
- **Average Profile report**
  This report analyzes energy consumption by identifying unfavorable peaks and patterns. Using it you can adjust device behavior as part of your energy procurement strategy. This information helps reduce consumption volatility, and makes the load more attractive for energy providers, which can reduce your energy costs.
- **Equipment Operation report**
  This report runs exceptions on a piece of equipment to determine run times compared to similar equipment in the enterprise. The report can identify the run times for the points associated with various pieces of equipment, such as HVAC, lighting, fans, refrigeration, chillers, and more. The report expresses results in both time and as a percentage using both tabular and graphical formats. With this information, you can determine if the run time for piece of equipment is in line with the manufacturer’s specifications and schedule maintenance accordingly.
- **Load Duration report**
  This report identifies the duration, or length of time, that demand (or consumption) for a point, aggregate point, or group of points exceeds certain levels. The load duration report provides input when considering demand-limiting strategies and possible capital investments. For example, this report can identify the peak for a demand meter and then indicate how much time (duration) that kW is above certain levels near the peak.
- **Ranking report**
  This report identifies the highest and lowest sites or points with a common characteristic. Using this report you can identify the most efficient facilities in your enterprise and benchmark against other facilities, or determine the least efficient facility and perform further analysis. Energy managers also use this report to rank lighting, HVAC, and refrigeration strategies within an enterprise. With this information, you can identify best-in-class equipment for energy consuming loads and reduce energy consumption across the enterprise.
- **Relative Contribution report**
  This report calculates the total energy consumption and displays the individual contribution of each underlying component. You use it to indicate how appliances within a building contribute to the total energy load at a facility or how different buildings contribute to an aggregated load. This becomes especially powerful when normalized for square footage and weather. Armed with this insight, you can identify the most logical place to allocate capital expenditures.
- **Spectrum Summary report**
  This report provides a quick view of any point or aggregated point with color coding, which identifies the reasonableness of the data value. The colors make evaluation quick and easy.

All Reports view

The **Analytic Ux Report List View** is accessible on the Reports sub folder under the **AnalyticService** and provides a graphical representation of each Ux (User Experience) report created using the view.

Figure 1. Ux reports view



Showing 1 to 4 of 4 reports

Report_AVgPRf
Location : defaultFolder
Last Updated : April 4th 2018, 15:18

Report_EquipOp
Location : defaultFolder1
Last Updated : April 4th 2018, 14:38

Report_LoadDuration
Location : defaultFolder
Last Updated : April 4th 2018, 14:33

Report_Spectrum
Location : defaultFolder
Last Updated : April 2nd 2018, 11:16

Every thumbnail corresponds to a single report. You click on the thumbnail icon to open the report. Each thumbnail includes the same details.

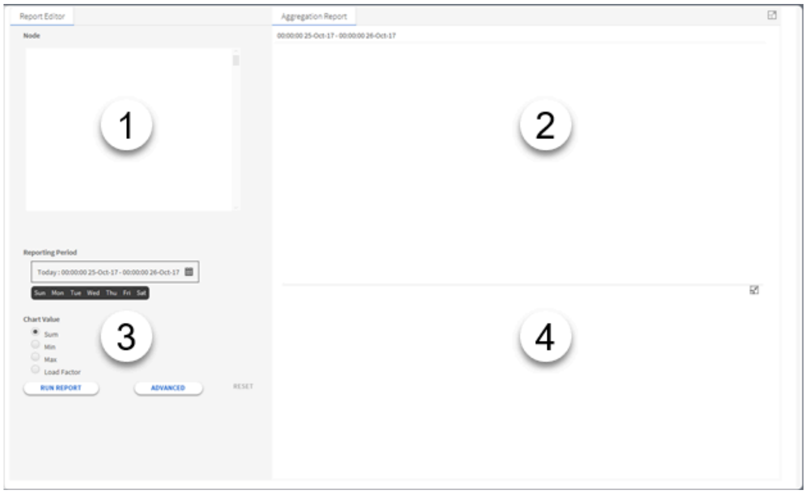
Detail	Description
Name	This is the name given to the report by its creator.
Command options	To the right of the report name are three icons: ⧉ Clone creates a new report based on the selected report. ✎ Edit opens the report properties window so you can make changes. ✖ Delete removes the report from the station so that no longer appears in the **All Reports** view.
Description	Summarizes the purpose of the report and what the report shows.
Location	Indicates the ORD for the BFolder component under reports where the report is saved.
Last Updated	Indicates the last time the data in the report was refreshed.

Parent topic: Reports

Report editor

Each report includes a Report Editor pane on the left hand side of the report that allows users to configure which nodes to use in the report, the Data (tag, tag group or algorithm), and the Time Range, Baseline and Normalization properties. Some reports may not include all of the configuration properties.

Figure 1. Report Editor pane


Report Editor
Aggregation Report
Node
00:00:00 25-Oct-17 - 00:00:00 26-Oct-17
1
2
3
4
Sun Mon Tue Wed Thu Fri Sat
RUN REPORT
ADVANCED
RESET

Reporting Period: Today : 00:00:00 25-Oct-17 - 00:00:00 26-Oct-17

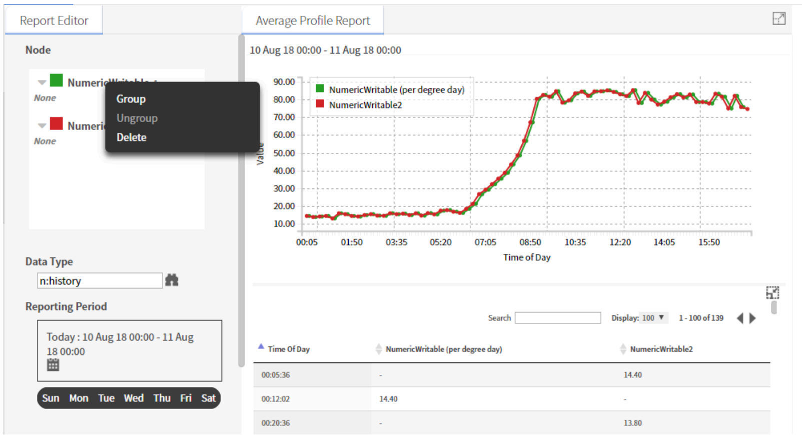
Chart Value: ● Sum ○ Min ○ Max ○ Load Factor

1. The node selection editor identifies and groups the data record(s) of interest.
2. The graph area contains the graphical representation of the data.
3. The minimal settings contains basic configuration properties.
4. The table area provides a summary of the raw data for each group.

Node selection editor

This area identifies the record types included in the report.

Figure 2. Node selection area



Data Type: n:history

Reporting Period: Today : 10 Aug 18 00:00 - 11 Aug 18 00:00

Search Display: 100 ▼ 1 - 100 of 139

Time Of Day	NumericWritable (per degree day)	NumericWritable2
00:05:36	-	14.40
00:12:02	14.40	-
00:20:36	-	13.80

To get started, drag one or more points into the Node pane.

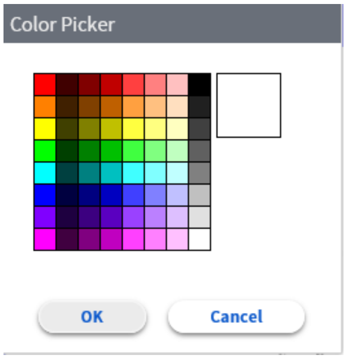
Note: If you modify any node using the configuration editor (nav tree of the station) and that node is part of a report, the node selection editor shows red text for that particular node. This indicates that the node was modified and needs to be re-configured on the report.

When you right-click a node in the node selection editor, four options for managing nodes and groups open in a drop-down list.

Option	Node	Description
Group	Not applicable.	Groups two selected groups.
Ungroup	Not applicable.	Converts all nodes inside the group to multiple nodes.
Rename	Not applicable.	Renames the group.
Delete	Deletes the node from the group.	Deletes the group and the nodes it contains.

Clicking the colored square next to the record type in the Node selection editor opens the color picker.

Figure 3. Color picker window

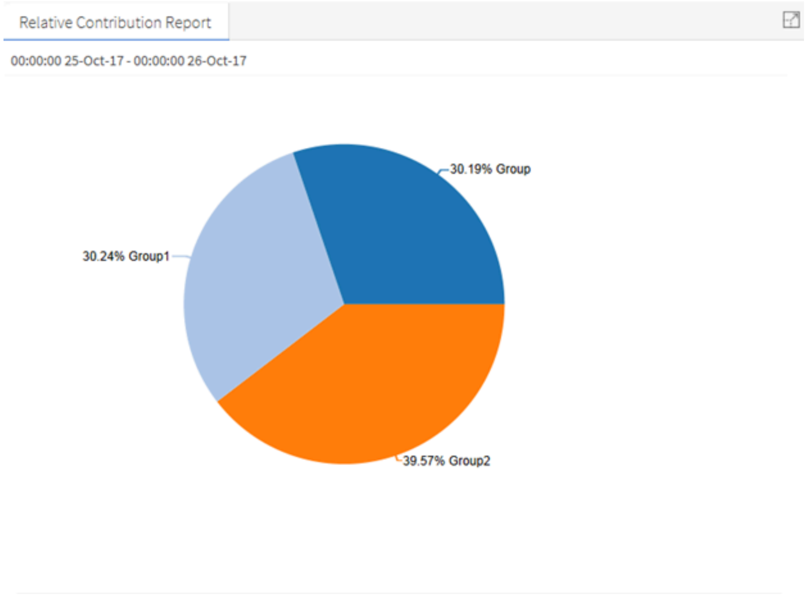


You click a color to select it. The points configured with color are groups. The nodes inside the groups are nodes.

Chart area

This area represents the selected data in the form of a pie chart, graph or time line.

Figure 4. Chart area


Relative Contribution Report
00:00:00 25-Oct-17 - 00:00:00 26-Oct-17
30.19% Group
30.24% Group1
39.57% Group2

The time and dates included in the report appear at the top of the chart area. A legend identifies the points on the chart. Passing the cursor over the chart activates a tool tip that displays the date, time and value of the selected record.

Minimal settings area

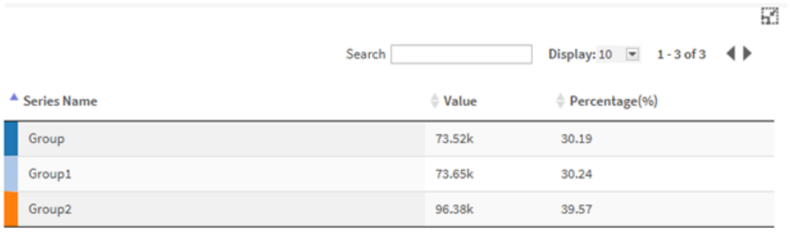
These properties provide basic configuration options. Individual reports may or may not include these properties.

Property	Value	Description
Data Type	Chooser	Selects the source point.
Reporting Period	drop-down list	Provides a calendar with which to select records from between specific dates. The days of the week limit the included data within the selected time range based on the day of the week. For more information, refer to the Time Range topic.
Chart Value	radio buttons	Sum adds together all values in the combination resulting in a single value. Min returns the lowest value in the combination. Max returns the highest value in the combination. Load Factor calculates the average divided by peak value.
Baseline	additional properties	Refer to *Baseline configuration*.
Normalization	additional properties	Refer to *Value Normalization*.
Run/Update Report	button	**Run** executes the report for the first time. **Update** refreshes the selected data.
Advanced	button	Opens another configuration window. This window provides a standard set of properties. Only the Aggregation report provides a slightly different set. For the standard properties refer to *Advanced Settings window*.

Table area

This area of the report displays a table of the raw data used to draw the chart. You can search for a specific record and configure the number of records that display on each page.

Figure 5. Table area



Series Name	Value	Percentage(%)
Group	73.52k	30.19
Group1	73.65k	30.24
Group2	96.38k	39.57

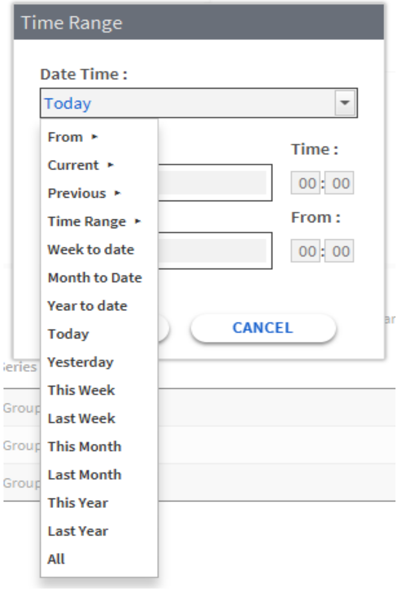
Property	Value	Description
Search	general search function	Locates a specific record.
Display	drop-down list	Controls how many records appear in the table at a time.
Previous and next buttons	buttons	Display additional records.

- **Time Range window**
  This window selects a general time range (Today, Year to date, etc.) or defines a specific time and date range.
- **Baseline configuration**
  Both the **Average Profile** and **Load Duration Reports** may use data values within the trend. This baseline makes it possible to compare a current data value against itself at a previous period of time. For example you set a baseline to compare Main kWh for last week with Main kWh for the same week in the previous year.
- **Value normalization**
  All reports, except the Spectrum and Equipment Operation reports, can benefit from normalizing data values to improve the usefulness of comparisons.
- **Advanced window**
  This window opens when you click the **Advanced** button in for all reports except the Aggregation report, which requires different advanced settings.

Parent topic: Reports

Time Range window

This window selects a general time range (Today, Year to date, etc.) or defines a specific time and date range.



Time Range

Date Time : Today

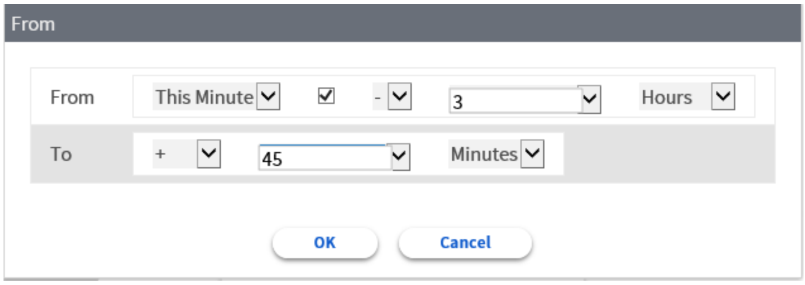
Time : 00 : 00

From : 00 : 00

This window opens when you click the calendar icon next to the default Reporting Period in the settings area.

Property	Value	Description
Date Time	drop-down list	Selects the period to include using a single option. Refer to the sections that follow this table.
From	drop-down list	Configures when to start reporting data.
To	drop-down list	Configures when to end reporting data. This value is not inclusive.

From window



From

From This Minute ☑ - 3 Hours

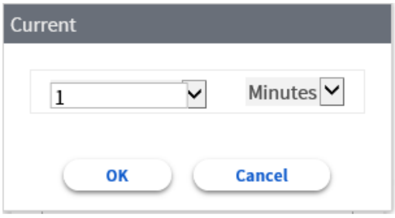
To + 45 Minutes

This window fine tunes the from and to dates and times without requiring you to specify calendar dates and clock times. For Example, the screen capture begins data reporting three hours ago and ends 45 minutes later.

Property	Value	Description
From: This Minute	drop-down list (defaults to This Minute)	Specifies a period of time from which to add or subtract minutes through years. It establishes the starting point for data reporting.

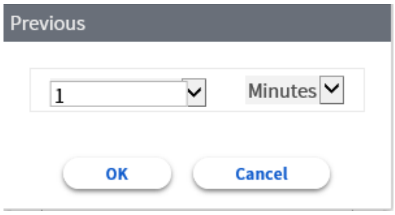
Property	Value	Description
From check box	defaults to checked	Enables and disables the addition or subtraction of minutes through years.
From +	drop-down list (defaults to +)	Indicates addition (+) or subtraction (-).
From 1–365	drop-down list (defaults to 1)	Specifies the number of minutes through years to add or subtract from the specified start time (This Minute, This Hour, Today, This Week, This Month, This Year). For example, From Today - 3 Hours subtracts 3 hours from the start (12:00 AM) of Today to calculate a From time of 9:00 PM the previous day.
From Minutes	drop-down list (defaults to Minutes)	Defines what the number to the calculated From time the left means (minutes through years).
To +	drop-down list (defaults to +)	Indicates addition (+), subtraction (-) or Now (the current time)
To 1–365	drop-down list (defaults to 1)	Specifies the number of minutes through years to add to or subtract from the specified start time (This Minute, This Hour, Today, This Week, This Month, This Year).
To Minutes	drop-down list (defaults to Minutes)	Defines what the number to the left means (minutes through years).

Current window



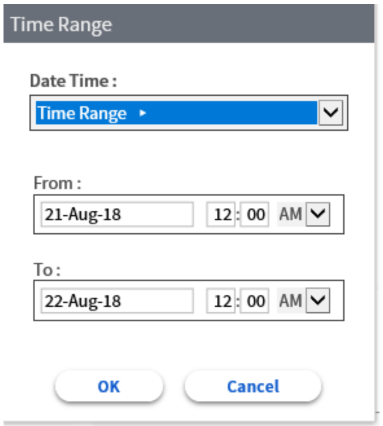
This option displays data from the configured window prior to the current time. For example, if the current time is 9:35 AM and the Current Window is 2 Hours the report displays data from 8:00 AM - 9:35 AM.

Previous window



This option displays data from the configured window prior to the start of selected time period. For example, if the current time is 9:35 AM and the Previous Window is 2 Hours, the report displays data from 7:00 AM - 9:00 AM. Since the time period is Hours and the current time is 9:35 AM, the end time is 9:00 AM, which is the start of the current hour. The From time 7:00 AM is calculated as 2 hours prior to the calculated end time 9:00 AM.

Time Range window



Time Range

Date Time : Time Range ▸

From : 21-Aug-18 12 : 00 AM

To : 22-Aug-18 12 : 00 AM

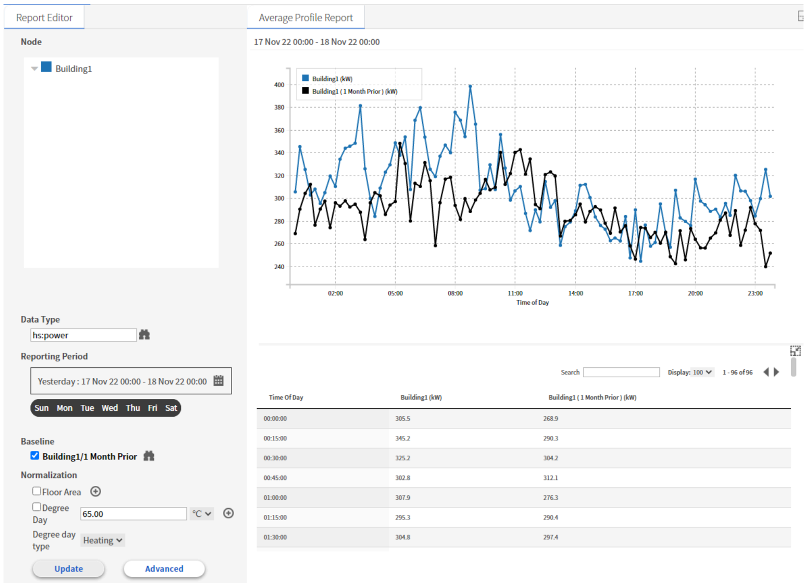
Selecting Time Range activates the From and To properties, which configure a calendar date and clock time.

Parent topic: Report editor

Baseline configuration

Both the **Average Profile** and **Load Duration Reports** may use data values within the trend. This baseline makes it possible to compare a current data value against itself at a previous period of time. For example you set a baseline to compare Main kWh for last week with Main kWh for the same week in the previous year.

Figure 1. Baseline properties on an Average Profile Report



Report Editor

Node

Building1

Data Type: hs:power

Reporting Period: Yesterday : 17 Nov 22 00:00 - 18 Nov 22 00:00

Baseline
☑ Building1/1 Month Prior

Normalization
□ Floor Area
□ Degree Day 65.00 °C
Degree day type: Heating

Average Profile Report

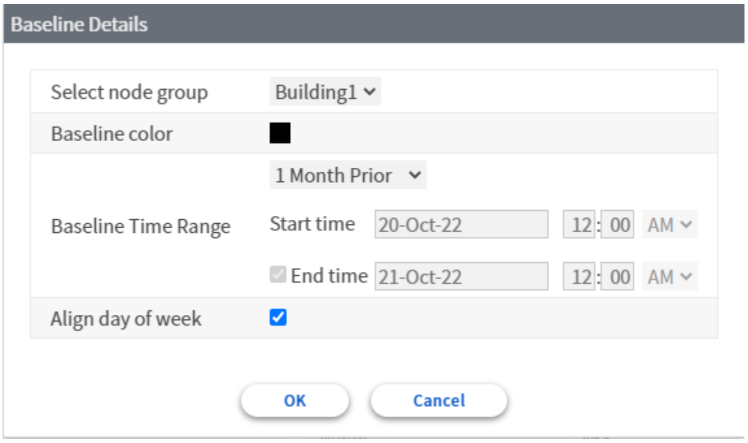
17 Nov 22 00:00 - 18 Nov 22 00:00

Search Display: 100 1 - 96 of 96

Time Of Day	Building1 (kW)	Building1 (1 Month Prior) (kW)
00:00:00	305.5	268.9
00:15:00	345.2	290.3
00:30:00	325.2	304.2
00:45:00	302.8	312.1
01:00:00	307.9	276.3
01:15:00	295.3	290.4
01:30:00	304.8	297.4

You establish a baseline by enabling the check box under Baseline on the **Report Editor**. To configure baseline details, you click the **Show Details** button ().

Figure 2. Baseline Details window



Baseline Details

Select node group: Building1

Baseline color: ■

Baseline Time Range: 1 Month Prior

Start time 20-Oct-22 12 : 00 AM

☑ End time 21-Oct-22 12 : 00 AM

Align day of week ☑

To open this window, click the down-arrow icon next to Custom Period. Once you select a node group, the text indicates the node group name, such as “Building 1/1 Month Prior” or “Building 1/Custom Period.” The screen capture shows the configuration used to produce the Baseline properties on an Average Profile Report above.

Property	Value	Description
Select Node Group	drop-down list (defaults to the first node group)	Selects the node group to use for baseline data. You select one node group at a time to establish the baseline.
Baseline Color	Color chooser	Selects a series color for the baseline data.
Baseline Time Range, Custom Period	drop-down list	Calculates the baseline in relationship to the other Baseline Time Range properties. Refer to The baseline calculations.
Baseline Time Range, Start time	date and 12-hr time selector	Selects what time of day to establish the baseline. These values are only valid if the Custom Period option is in use. Otherwise, the framework automatically calculates the start date and time.
Baseline Time Range, End time check box	check box (defaults to disabled, that is, no check mark)	Enables and disables the ability to edit the End time property. The default automatically calculates the end date. These values are only valid if the Custom Period option is in use. Otherwise, the framework automatically calculates the end date and time.
Baseline Time Range, End time	date and 12-hr time selector	Selects the time of day to stop the collecting of baseline data.
Align day of week	check box (defaults to disabled, that is, no check mark if Custom Period is in use, otherwise, it defaults to enabled)	Plays a critical role in automatically calculating the start and end dates and times for the selected baseline. Refer to The baseline calculations.

The baseline calculations

All baseline calculations reference the Reporting Period as defined in the **Report Editor**.

If Custom Period is selected for Baseline Time Range, the framework activates the Start time properties and provides optional End time properties. If no End time is defined, the framework uses the selected time range (the main time range) to determine when the baseline period ends. If End time is enabled and a time selected, the framework uses this date and time to end the baseline period.

When you select a Baseline Time Range, such as 1 Week prior or 1 Month prior, the framework calculates the baseline start and end dates and times based on the selected time range and the Align day of week property. If Align day of week is disabled, the framework subtracts the time range from both start and end date times of the selected range.

If the Align day of week is enabled, the framework takes the start date of the selected time range as a reference and calculates the same day of the week in the previous week or month. For example, if the start date is the second Wednesday of this month and the baseline period is 1 Month Prior, the baseline starts the second Wednesday of previous month. The framework calculates the end date and time by adding the duration of the time range to the calculated baseline start date and time.

When 1 Year prior is selected for Baseline Time Range, the framework calculates the baseline start and end dates based on the selected time range and the Align day of week property. If the Align day of week is disabled, the framework subtracts one year from both start and end date times of the selected time range.

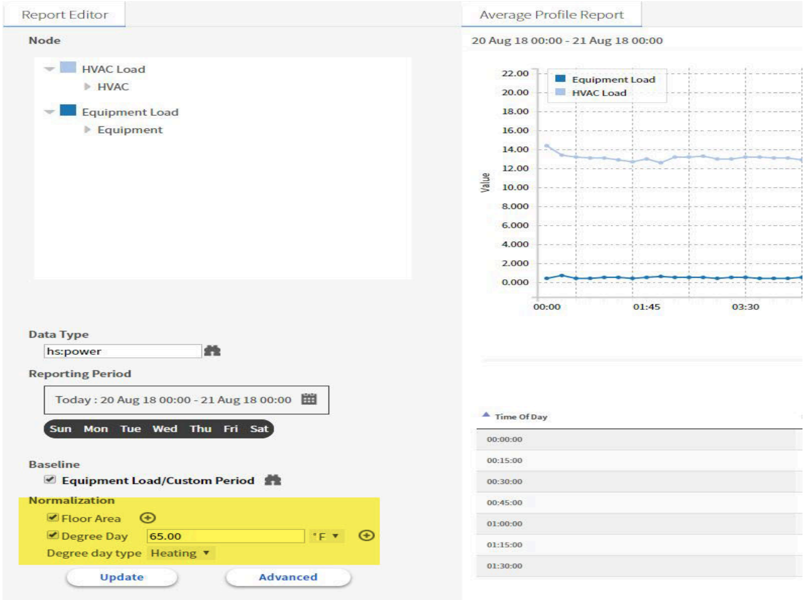
If the Align day of week option is enabled, the framework takes the start date of the selected time range as a reference and calculates the same day of the week in the previous year. For example, if the start date is the Wednesday in the eighth week of the year, the baseline starts the Wednesday in the eighth week of the previous year. The framework calculates the end date and time by adding the duration of the time range to the calculated baseline start date and time.

Parent topic: Report editor

Value normalization

All reports, except the Spectrum and Equipment Operation reports, can benefit from normalizing data values to improve the usefulness of comparisons.

Figure 1. Normalization properties



Node

- HVAC Load
  - HVAC
- Equipment Load
  - Equipment

Data Type: hs:power

Reporting Period: Today : 20 Aug 18 00:00 - 21 Aug 18 00:00

Sun Mon Tue Wed Thu Fri Sat

Baseline

☑ Equipment Load/Custom Period

Normalization

☑ Floor Area

☑ Degree Day 65.00 °F

Degree day type: Heating

20 Aug 18 00:00 - 21 Aug 18 00:00

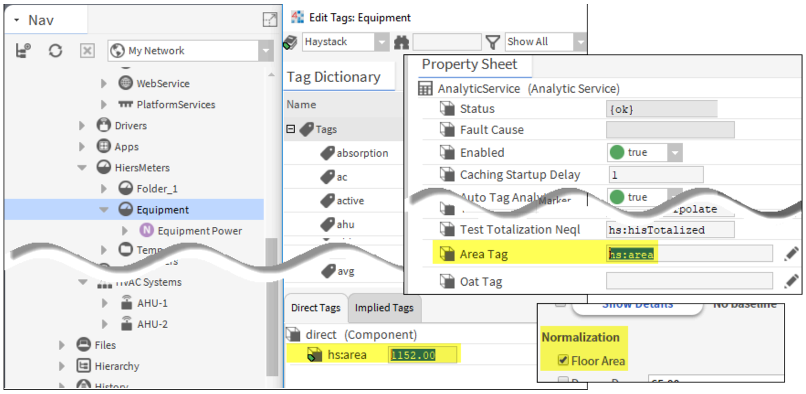
Time Of Day
00:00:00
00:15:00
00:30:00
00:45:00
01:00:00
01:15:00
01:30:00

Property	Value	Description
Normalization, Floor Area	check box	Enables the use of the floor area of the facility to normalize the values used in the resulting chart. Refer to Floor area normalization.
Plus icon to the right of the Floor Area row (⊕)	Node editor	Opens a window to configure the node from which to find the floor area tag for each node group.
Normalization, Degree Day	check box	Enables normalization based on the degree-day calculation. Refer to Degree-day temperature normalization.
Normalization, base outside air temperature	number with two decimal places (defaults to 65.00, which is a Fahrenheit temperature measurement)	Selects a base outside air temperature, which is used to calculate degree-day normalization. This base temperature is the outside air temperature at which neither heat nor air conditioning is required inside the building.
Normalization, scale	drop-down list	Selects the temperature scale to use: Fahrenheit, Centigrade or Kelvin in the degree-day normalization calculations.
Plus icon to the right of the Degree Day row (⊕)	Node editor	Opens a window to configure the node from which to find the outside air temperature tag for each node group.
Degree day type	drop-down list	Selects the temperature supplementation being used: Heating or Cooling.

Floor area normalization

The average energy consumption of a commercial building varies depending on the building usage, such as commercial office space, warehouse storage, data center or industrial manufacturing. When comparing energy consumption from multiple buildings of the same type (commercial office space), those buildings may be different sizes (square footage), so simply comparing the total energy consumption of the buildings may not provide useful analysis. Normalizing each building's energy consumption by its floor area yields energy consumption per square foot, which allows more accurate comparison of the energy efficiency among multiple buildings.

Figure 2. Floor area configuration



AnalyticService (Analytic Service)

Status	{ok}
Fault Cause	
Enabled	true
Caching Startup Delay	1
Auto Tag Analytics	true
Test Totalization Neql	hs:hisTotalized
Area Tag	hs:area
Oat Tag	

direct (Component)

hs:area 1152.00

Normalization

☑ Floor Area

To set up your system for area normalization, you enter the area (square units) in a tag, such as the hs:area tag of the Haystack tag dictionary, associate that tag with each node (typically building or floor to be compared), configure the Area Tag property on the **AnalyticService** to use the tag for this purpose, and enable the Floor Area property under Normalization in the **Report Editor**.

For multiple groups, the framework aggregates (sums) the area of each group into a single total figure.

Degree-day temperature normalization

Degree-day normalization answers the question, "What would the value of energy consumption be if the temperature during the report period was equal to some base temperature?" Degree-day normalization applies only to temperature-dependent points. This type of normalization works for building areas and equipment whose energy consumption is subject to changes in outside air temperature. For example, you would not use this calculation with lighting energy consumption.

The normalization calculation requires actual air temperature values taken at different times of day and a base outside air temperature value with which to compare the actual values. The point used to collect OAT (Outside Air Temperature) values must be tagged with a unique OAT Tag. You configure the OAT Tag used by the framework by editing the Area Tag property on the **AnalyticService Property Sheet**.

The base outside air temperature is usually set to 65 degrees Fahrenheit, which is the balance at which no cooling or heating is required to maintain a comfortable indoor air temperature. Starting with the days of the Reporting Period, the base outside air temperature, and multiple OAT readings taken at intervals during the day, the calculation normalizes temperature using a simple ratio as follows:

1. For each day, it calculates the average of the differences between the base outside air temperature and each actual OAT reading. This calculation excludes days that were not included in the Reporting Period days of the week. If the interval is greater than a day, the calculation sums (rolls up) the values to match the interval.

   The result is a value for each day (degree-day value), which indicates when cooling and heating was required.

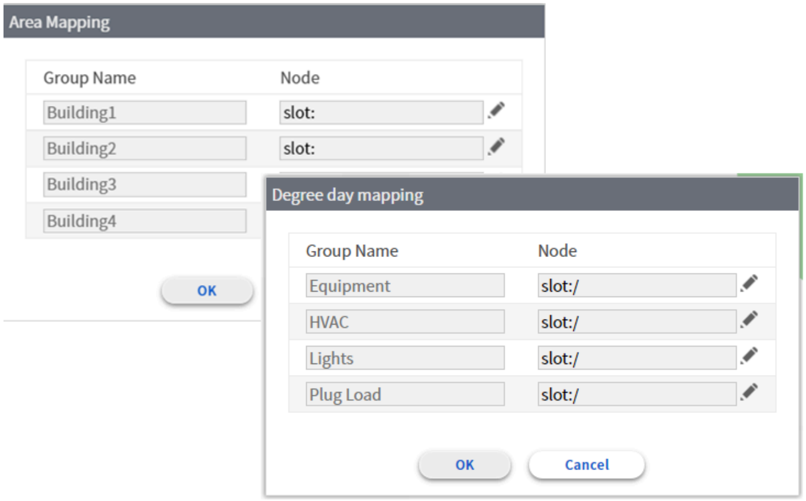
2. Then, it divides the energy consumption for each day by the degree-day value. This gives the kWh per degree-day. In theory, dividing by the degree-day value factors out the effect of outside air temperature so you can compare the resulting kWh fairly.

Node editor for floor area and outside temperature mapping

This editor maps tag groups to the root nodes used to either normalize floor area or search for outside air temperature.

(**Degree day mapping** window)

Figure 3. Area and Degree day mapping windows



Area Mapping

Group Name	Node
Building1	slot:
Building2	slot:
Building3	
Building4	

Degree day mapping

Group Name	Node
Equipment	slot:/
HVAC	slot:/
Lights	slot:/
Plug Load	slot:/

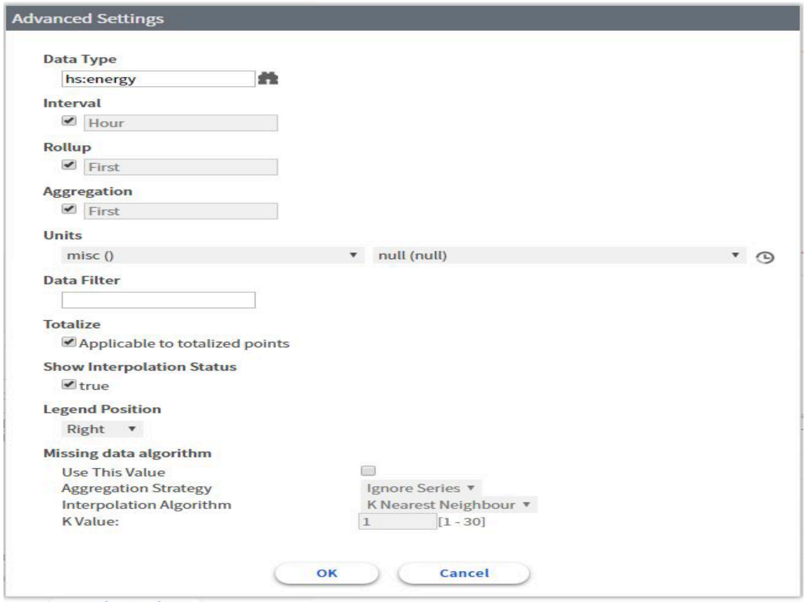
Property	Value	Description
Group	name	Identifies the node groups in the report.
Node	ORD	Identifies the location in the station database that contains the data (square footage or outdoor air temperature).

Parent topic: Report editor

Advanced window

This window opens when you click the **Advanced** button in for all reports except the Aggregation report, which requires different advanced settings.

Figure 1. Advanced Settings window



Advanced Settings

Data Type

hs:energy

Interval

☑ Hour

Rollup

☑ First

Aggregation

☑ First

Units

misc () ▾ null (null) ▾

Data Filter

Totalize

☑ Applicable to totalized points

Show Interpolation Status

☑ true

Legend Position

Right ▾

Missing data algorithm

Use This Value	□
Aggregation Strategy	Ignore Series ▾
Interpolation Algorithm	K Nearest Neighbour ▾
K Value:	1 [1 - 30]

Property	Value	Description
Data Type	Chooser	Selects the source point.
Interval	optional drop-down list (defaults to the optimal number of records on reports); interval=option	Refers to the BInterval component, which the framework uses to identify the time between values in a trend (time series). When specified, a rollup is required, which causes the system to combine all values that fall into a single interval. Options range from None to a Year. Above the drop-down list, the Use This Value check box turns on and off the check box next to Interval in the **Settings** window (you access this window by clicking the Edit button (✎), followed by clicking the Settings button (⚙) on the chart). The availability of this check box provides an easy way for a user to enable and disable the use of intervals in chart calculations.
Rollup	check box (if optional, and) drop-down list or ORD parameter (when configured in the data definition, defaults to First, when configured elsewhere, defaults to the value as defined in the Data Definition); rollup=option	Configures the default function to apply when the analytic request needs to rollup records from a single data source into less granular records. This typically only applies to trend request, but may also apply to value requests where the Id is an algorithm that contains a block like Runtime or Sliding Window, which processes a trend request. If rollup is not enabled in the binding/ settings window, the rollup value configured in the Data Definition applies to all chart bindings, reports and tables. And returns the logical “and” of Boolean values. Avg returns the statistical mean, which is

Property	Value	Description
		determined by calculating the sum of all values and dividing by the number of values. Count returns the total number or quantity of values in a combination. If you request this value on a binding in a PX view, the system counts the number of values based on the properties defined by the data source block and the algorithm’s **Property Sheet**. First returns the first value in the combination. Last returns the last value in the combination. Max returns the highest value in the combination. Median returns the value in the middle of a sorted combination—the number that separates the higher half from the lower half. Min returns the lowest value in the combination. Mode returns the statistically most frequently occurring number in the combination. Or returns the logical “or” of Boolean values. Range returns the statistical difference between the largest and smallest values in the combination. Sum adds together all values in the combination resulting in a single value. Std Dev calculates the standard deviation of the values in the combination. Load Factor calculates the average divided by peak (Max) value.
Aggregation	check box (if optional, and) drop-down list (property, defaults to First) or ORD parameter (`aggregation=option`)	Configures the default function to apply when the analytic request needs to combine values from multiple data sources into a single value. This applies to both value and trend requests. If aggregation is not enabled in the binding/settings window, the aggregation value defined in the Data Definition applies to all chart bindings, reports and tables. And returns the logical “and” of Boolean values. Avg returns the statistical mean, which is determined by calculating the sum of all values and dividing by the number of values. Count returns the total number or quantity

Property	Value	Description
		of values in a combination. If you request this value on a binding in a PX view, the system counts the number of values based on the properties defined by the data source block and the algorithm's **Property Sheet**. First returns the first value in the combination. Last returns the last value in the combination. Max returns the highest value in the combination. Median returns the value in the middle of a sorted combination—the number that separates the higher half from the lower half. Min returns the lowest value in the combination. Mode returns the statistically most frequently occurring number in the combination. Or returns the logical "or" of Boolean values. Range returns the statistical difference between the largest and smallest values in the combination. Sum adds together all values in the combination resulting in a single value. Load Factor returns the average value divided by peak value. Std Dev returns the standard deviation of the values in the combination.
Units	drop-down list of units of measure	Defines the unit of measure for the data gathered from each point.
Data Filter	optional NEQL query (property) or ORD parameter (dataFilter=query)	Identifies data sources in the subtree of the request node. When a predicate accepts a node that contains the desired data, the system does not search the node's subtree for additional values (including the root node).
Totalize	true (true) or false or hisTotEnabled=option (ORD scheme parameter)	Turns on (true) and off (false) value accumulation. By default, the framework totalizes (accumulates) all consumption history values in charts, tables and reports. To prevent cumulative values, disable this property (set it to false).
Show Interpolation Status	false check box	Enables and disables the display of the interpolation status columns in report tables. Clicking this check box removes the status columns leaving only the record

Property	Value	Description
		value columns.
Legend Position	drop-down list (defaults to Inset)	Changes the position of the legends that label graphical elements in charts. Three positions are possible: None removes the legend from the chart. Inset places the legend inside the chart area. Bottom places the legend below the chart area. Right places the legend to the right of the chart area.
Missing Data Strategy, Use This Value	check box	Enables and disables missing data interpolation for the current value. When enabled, the framework applies this strategy to all requests. When enabled, the framework applies this strategy to all requests.
Missing Data Strategy, Aggregation Strategy	drop-down list; `aggStrategy=option` (ORD scheme parameter)	Configures how the framework handles missing trend data (data in a series) when processing analytic requests and one or more records are missing for an interval. It applies when even a single record for an interval is missing. It does not apply to value requests. Note: If the analytic trend request specifies Interval = none, the framework ignores the Missing Data Strategy. Ignore Series ignores the entire series if any record even one interval is missing. Ignore Point ignores any missing records and aggregates the values in the existing records.
Missing Data Strategy, Interpolation Algorithm	drop-down list	Defines the algorithm used to interpolate values for missing values (missing records). None does not interpolate values for missing records. Linear Interpolation replaces a missing record by linearly interpolating the missing value using the prior and next records on either side of the missing record. K-Nearest Neighbor is for numeric, enum and Boolean records. This strategy replaces a missing value by calculating the majority value recorded for the item’s nearest K number of neighbors.
Missing Data Strategy, K Value	Numeric field editor (default = 1, min =1, max = 30)	Defines the number of records used by the configured interpolation algorithm when a record is missing to calculate the

Property	Value	Description
		interpolated value.

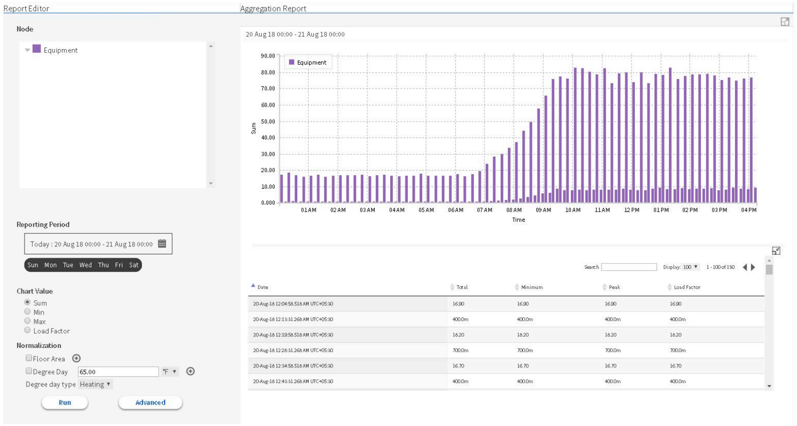
Parent topic: Report editor

Aggregation report

This report plots aggregation (consumption) against demand using total, peak, minimum, and load factor values at various sites, meters, time periods and commodities. With this information you can increase total energy procured and determine complementary loads to improve your load factor. When negotiating a contract, this information can provide accurate consumption patterns versus the arbitrary classifications used by commercial and industrial customers.

Load factor values are between 0 and 1. The closer the load factor value is to 1 the more consistent is the building's load profile, meaning that demand does not vary much.

Figure 1. Aggregation report



Today : 20 Aug 18 00:00 - 21 Aug 18 00:00

Chart Value
● Sum
○ Min
○ Max
○ Load Factor

Normalization
□ Floor Area
□ Degree Day 65.00 °F
Degree day type Heating

Display: 100

Date	Total	Minimum	Peak	Load Factor
20-Aug-18 12:04:58.518 AM UTC+05:30	16.90	16.90	16.90	16.90
20-Aug-18 12:13:31.268 AM UTC+05:30	400.0m	400.0m	400.0m	400.0m
20-Aug-18 12:19:58.518 AM UTC+05:30	18.20	18.20	18.20	18.20
20-Aug-18 12:28:31.268 AM UTC+05:30	700.0m	700.0m	700.0m	700.0m
20-Aug-18 12:34:58.518 AM UTC+05:30	16.70	16.70	16.70	16.70
20-Aug-18 12:43:31.268 AM UTC+05:30	400.0m	400.0m	400.0m	400.0m

This report supports only one group. You add multiple nodes under this group.

Note: In this multi-node group, the system aggregates the data for all records based on the **Aggregation** property defined in the **Advanced Settings** window.

Chart Value

In addition to the standard properties, this report allows you to select a single Chart Value to apply to the graph:

- Sum — sum of the energy consumption values for each interval.
- Min — minimum power value for each interval.
- Max — maximum power value for each interval.
- Load Factor — calculated load factor value for each interval.

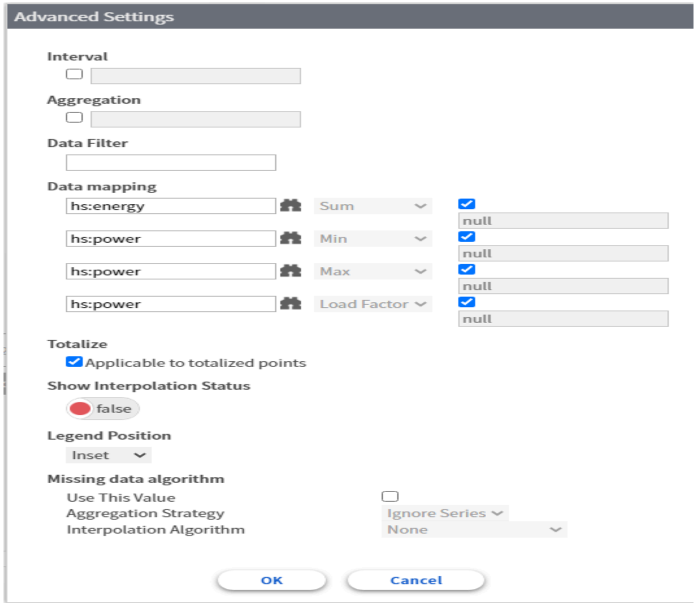
Table area columns

The table area shows information for each node.

Column	Description
Date	Identifies the start timestamp for each interval period.
Minimum	Reports the minimum power value for each interval.
Total	Reports the sum of the energy consumption values.
Peak	Reports the maximum power value for each interval.
Load Factor	Reports the load factor calculated for each interval.

Advanced Settings window

The rollup function for this report is fixed based on the data being displayed.



Advanced Settings

Interval
☐

Aggregation
☐

Data Filter

Data mapping

hs:energy	Sum	☑	null
hs:power	Min	☑	null
hs:power	Max	☑	null
hs:power	Load Factor	☑	null

Totalize
☑ Applicable to totalized points

Show Interpolation Status
● false

Legend Position
Inset

Missing data algorithm
Use This Value ☐
Aggregation Strategy Ignore Series
Interpolation Algorithm None

Property	Value	Description
Interval	optional drop-down list (defaults to the optimal number of records on reports); interval=option	Refers to the BInterval component, which the framework uses to identify the time between values in a trend (time series). When specified, a rollup is required, which causes the system to combine all values that fall into a single interval. Options range from None to a Year. Above the drop-down list, the Use This Value check box turns on and off the check box next to Interval in the **Settings** window (you access this window by clicking the Edit button (✎), followed by clicking the Settings button (⚙) on the chart). The availability of this check box provides an easy way for a user to enable and disable the use of intervals in chart calculations.

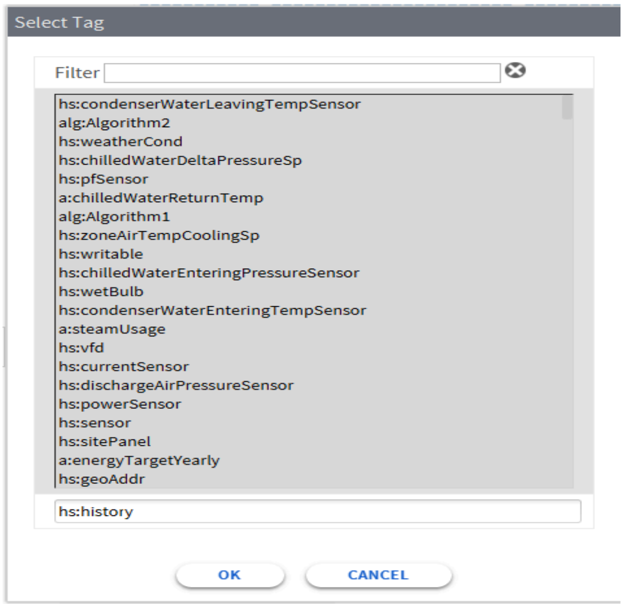
Property	Value	Description
Aggregation	check box (if optional, and) drop-down list (property, defaults to First) or ORD parameter (aggregation=option)	Configures the default function to apply when the analytic request needs to combine values from multiple data sources into a single value. This applies to both value and trend requests. If aggregation is not enabled in the binding/settings window, the aggregation value defined in the Data Definition applies to all chart bindings, reports and tables. And returns the logical "and" of Boolean values. Avg returns the statistical mean, which is determined by calculating the sum of all values and dividing by the number of values. Count returns the total number or quantity of values in a combination. If you request this value on a binding in a PX view, the system counts the number of values based on the properties defined by the data source block and the algorithm's **Property Sheet**. First returns the first value in the combination. Last returns the last value in the combination. Max returns the highest value in the combination. Median returns the value in the middle of a sorted combination—the number that separates the higher half from the lower half. Min returns the lowest value in the combination. Mode returns the statistically most frequently occurring number in the combination. Or returns the logical "or" of Boolean values. Range returns the statistical difference between the largest and smallest values in the combination. Sum adds together all values in the combination resulting in a single value. Load Factor returns the average value divided by peak value. Std Dev returns the standard deviation of the values in the combination.
Data Filter	optional NEQL query (property) or ORD parameter (dataFilter=query)	Identifies data sources in the subtree of the request node. When a predicate accepts a node that contains the desired data, the system does not search the node's subtree for additional values

Property	Value	Description
		(including the root node).
Data Mapping	Tag Choosers (defaults to Sum = hs:energy, Min = hs:power, Max = hs:power, Load Factor = hs:power)	Assigns tags to four options. Clicking the **Tag Chooser** opens a standard **Select Tag** window.
Totalize	true (true) or false or hisTotEnabled=option (ORD scheme parameter)	Turns on (true) and off (false) value accumulation. By default, the framework totalizes (accumulates) all consumption history values in charts, tables and reports. To prevent cumulative values, disable this property (set it to false).
Show Interpolation Status	false check box	Enables and disables the display of the interpolation status columns in report tables. Clicking this check box removes the status columns leaving only the record value columns.
Legend Position	drop-down list (defaults to Inset)	Changes the position of the legends that label graphical elements in charts. Three positions are possible: None removes the legend from the chart. Inset places the legend inside the chart area. Bottom places the legend below the chart area. Right places the legend to the right of the chart area.
Missing Data Strategy, Use This Value	check box	Enables and disables missing data interpolation for the current value. When enabled, the framework applies this strategy to all requests. When enabled, the framework applies this strategy to all requests.
Missing Data Strategy, Aggregation Strategy	drop-down list; aggStrategy=option (ORD scheme parameter)	Configures how the framework handles missing trend data (data in a series) when processing analytic requests and one or more records are missing for an interval. It applies when even a single record for an interval is missing. It does not apply to value requests. Note: If the analytic trend request specifies Interval = none, the framework ignores the Missing Data Strategy. Ignore Series ignores the entire series if any record even one interval is missing. Ignore Point ignores any missing records and aggregates the values in the existing records.

Property	Value	Description
Missing Data Strategy, Interpolation Algorithm	drop-down list	Defines the algorithm used to interpolate values for missing values (missing records). None does not interpolate values for missing records. Linear Interpolation replaces a missing record by linearly interpolating the missing value using the prior and next records on either side of the missing record. K-Nearest Neighbor is for numeric, enum and Boolean records. This strategy replaces a missing value by calculating the majority value recorded for the item’s nearest K number of neighbors.
Missing Data Strategy, K Value	Numeric field editor (default = 1, min =1, max = 30)	Defines the number of records used by the configured interpolation algorithm when a record is missing to calculate the interpolated value.

Select Tag window

Figure 2. Select Tag window



hs:condenserWaterLeavingTempSensor
alg:Algorithm2
hs:weatherCond
hs:chilledWaterDeltaPressureSp
hs:pfSensor
a:chilledWaterReturnTemp
alg:Algorithm1
hs:zoneAirTempCoolingSp
hs:writable
hs:chilledWaterEnteringPressureSensor
hs:wetBulb
hs:condenserWaterEnteringTempSensor
a:steamUsage
hs:vfd
hs:currentSensor
hs:dischargeAirPressureSensor
hs:powerSensor
hs:sensor
hs:sitePanel
a:energyTargetYearly
hs:geoAddr

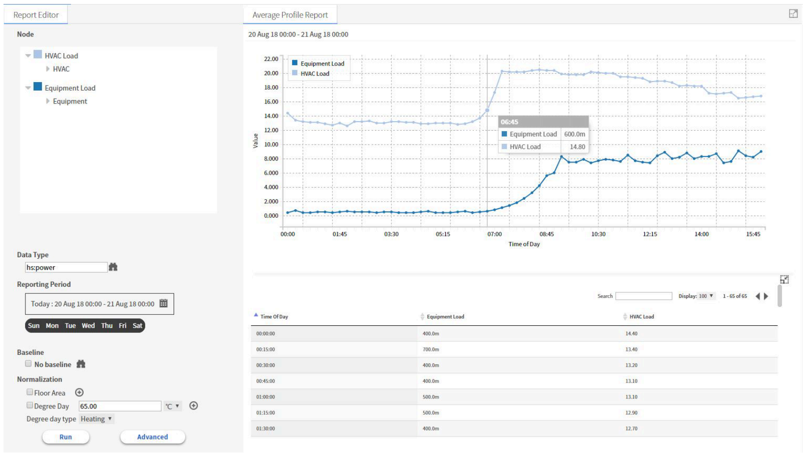
hs:history

Parent topic: Reports

Average Profile report

This report analyzes energy consumption by identifying unfavorable peaks and patterns. Using it you can adjust device behavior as part of your energy procurement strategy. This information helps reduce consumption volatility, and makes the load more attractive for energy providers, which can reduce your energy costs.

Figure 1. Average Profile report



20 Aug 18 00:00 - 21 Aug 18 00:00

06:45	
Equipment Load	600.0m
HVAC Load	14.80

Data Type: hs:power

Reporting Period: Today : 20 Aug 18 00:00 - 21 Aug 18 00:00

Baseline: □ No baseline

Normalization: □ Floor Area; □ Degree Day 65.00 °C; Degree day type Heating

Time Of Day	Equipment Load	HVAC Load
00:00:00	400.0m	14.40
00:15:00	700.0m	13.40
00:30:00	400.0m	13.20
00:45:00	400.0m	13.10
01:00:00	500.0m	13.10
01:15:00	500.0m	12.90
01:30:00	400.0m	12.70

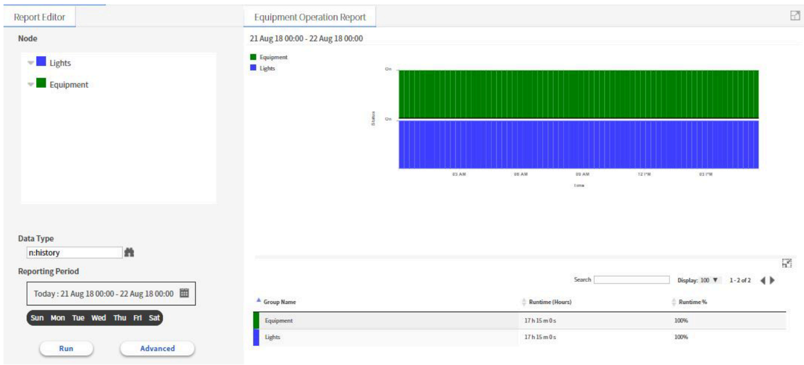
The report plots the average load (y-axis) for one or more pieces of equipment against the time of day (x-axis). The data included in the report depends on the points you are monitoring. This report's Advanced Settings window provides the standard set of properties. Refer to *Advanced Settings window* in this chapter.

Parent topic: Reports

Equipment Operation report

This report runs exceptions on a piece of equipment to determine run times compared to similar equipment in the enterprise. The report can identify the run times for the points associated with various pieces of equipment, such as HVAC, lighting, fans, refrigeration, chillers, and more. The report expresses results in both time and as a percentage using both tabular and graphical formats. With this information, you can determine if the run time for piece of equipment is in line with the manufacturer's specifications and schedule maintenance accordingly.

Figure 1. Equipment Operation report



21 Aug 18 00:00 - 22 Aug 18 00:00

Data Type: n:history

Reporting Period: Today : 21 Aug 18 00:00 - 22 Aug 18 00:00

Group Name	Runtime (Hours)	Runtime %
Equipment	17 h 15 m 0 s	100%
Lights	17 h 15 m 0 s	100%

The chart in the report shows the On Off status of the selected pieces of equipment stacked over each other vertically. The table below shows the runtime in hours and the percentage of time the equipment was running within the given time range.

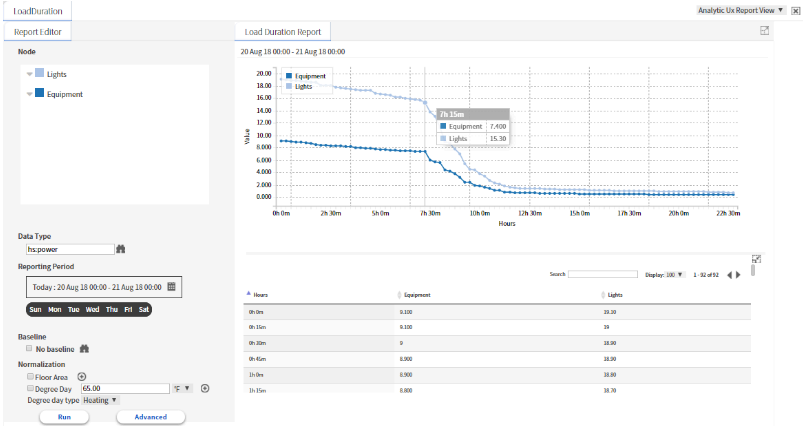
This report supports multiple node groups from which to select a different series color for each group.

Parent topic: Reports

Load Duration report

This report identifies the duration, or length of time, that demand (or consumption) for a point, aggregate point, or group of points exceeds certain levels. The load duration report provides input when considering demand-limiting strategies and possible capital investments. For example, this report can identify the peak for a demand meter and then indicate how much time (duration) that kW is above certain levels near the peak.

Figure 1. Example of a load duration report



20 Aug 18 00:00 - 21 Aug 18 00:00

Data Type: hs:power

Reporting Period: Today : 20 Aug 18 00:00 - 21 Aug 18 00:00

Baseline: □ No baseline

Normalization: □ Floor Area; □ Degree Day 65.00 °F ▼; Degree day type Heating ▼

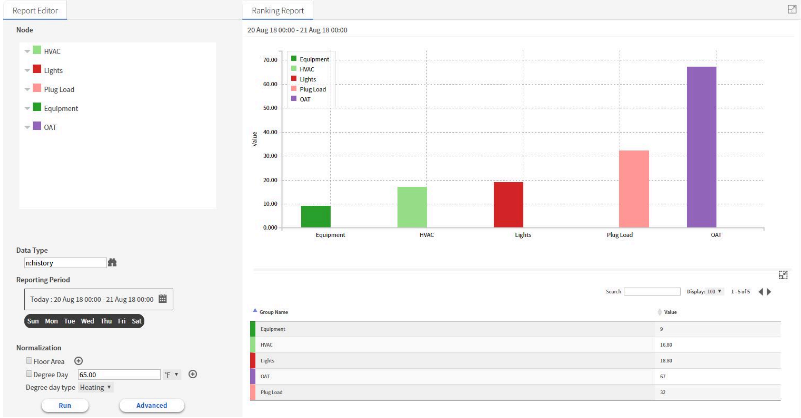
Hours	Equipment	Lights
0h 0m	9.100	19.10
0h 15m	9.100	19
0h 30m	9	18.90
0h 45m	8.900	18.90
1h 0m	8.900	18.80
1h 15m	8.800	18.70

Parent topic: Reports

Ranking report

This report identifies the highest and lowest sites or points with a common characteristic. Using this report you can identify the most efficient facilities in your enterprise and benchmark against other facilities, or determine the least efficient facility and perform further analysis. Energy managers also use this report to rank lighting, HVAC, and refrigeration strategies within an enterprise. With this information, you can identify best-in-class equipment for energy consuming loads and reduce energy consumption across the enterprise.

Figure 1. Example of a ranking report



20 Aug 18 00:00 - 21 Aug 18 00:00

Data Type: n:history

Reporting Period: Today : 20 Aug 18 00:00 - 21 Aug 18 00:00

Normalization: □ Floor Area; □ Degree Day 65.00 °F ▼; Degree day type Heating ▼

Group Name	Value
Equipment	9
HVAC	16.80
Lights	18.80
OAT	67
Plug Load	32

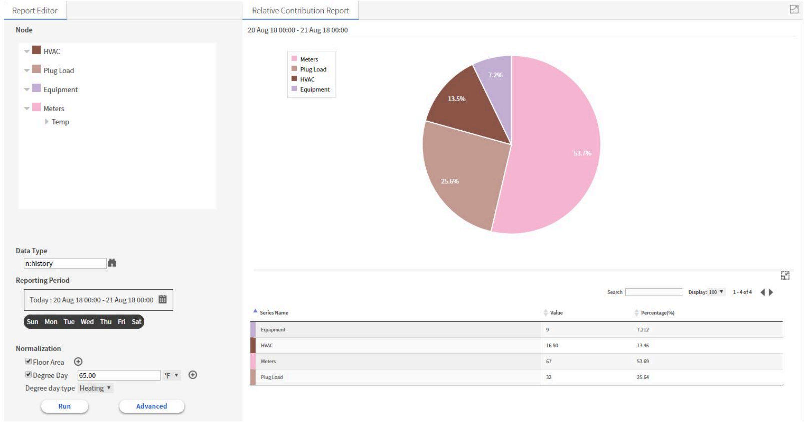
This report supports multiple groups (nodes). You can select colors and rename groups.

Parent topic: Reports

Relative Contribution report

This report calculates the total energy consumption and displays the individual contribution of each underlying component. You use it to indicate how appliances within a building contribute to the total energy load at a facility or how different buildings contribute to an aggregated load. This becomes especially powerful when normalized for square footage and weather. Armed with this insight, you can identify the most logical place to allocate capital expenditures.

Figure 1. Example of a relative contribution report



20 Aug 18 00:00 - 21 Aug 18 00:00

Data Type: n:history

Reporting Period: Today : 20 Aug 18 00:00 - 21 Aug 18 00:00

Normalization

☑ Floor Area

☑ Degree Day 65.00 °F

Degree day type: Heating

Search: Display: 100 1 - 4 of 4

Series Name	Value	Percentage(%)
Equipment	9	7.212
HVAC	16.80	13.46
Meters	67	53.69
Plug Load	32	25.64

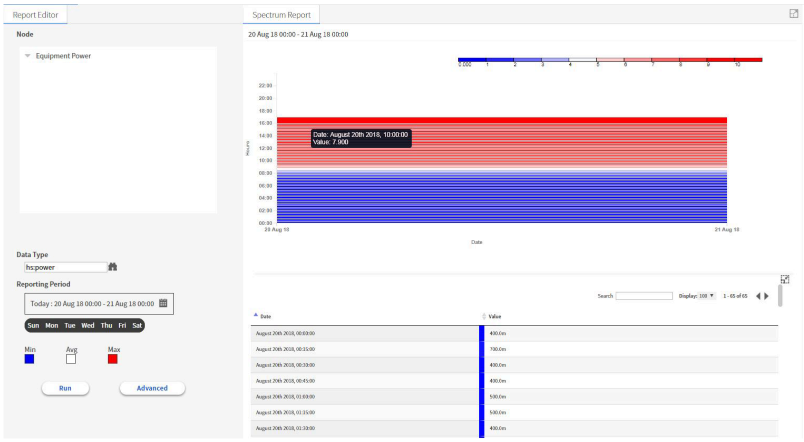
This report supports multiple groups and data points.

Parent topic: Reports

Spectrum Summary report

This report provides a quick view of any point or aggregated point with color coding, which identifies the reasonableness of the data value. The colors make evaluation quick and easy.

Figure 1. Example of a spectrum report



20 Aug 18 00:00 - 21 Aug 18 00:00

Data Type: hs:power

Reporting Period: Today : 20 Aug 18 00:00 - 21 Aug 18 00:00

Search: Display: 100 1 - 65 of 65

Date	Value
August 20th 2018, 00:00:00	400.0m
August 20th 2018, 00:15:00	700.0m
August 20th 2018, 00:30:00	400.0m
August 20th 2018, 00:45:00	400.0m
August 20th 2018, 01:00:00	500.0m
August 20th 2018, 01:15:00	500.0m
August 20th 2018, 01:30:00	400.0m

If all data values are within historical ranges, the report colors display a consistent pattern. You can move on to other functions. If data values are unusual or inconsistent, the pattern captures the consistency. A quick glance then demands further analysis.

Parent topic: Reports

Glossary

The following glossary entries relate specifically to the topics that are included as part of this document.

To find more glossary terms and definitions refer to glossaries in other individual documents.

Alphabetical listing

- **alert**
- **algorithm**
- **direct tag/relation**
- **implied tag/relation**
- **request**
- **scope**
- **tag**

alert

A warning regarding a condition identified by a Niagara Analytics Framework that can be routed to an alarm or used to visualize real-time and historical data.
Parent topic: Glossary

algorithm

A formula that uses real-time values, historical trend data, and the results of calculations made by other algorithms to analyze data collected by the system.
Parent topic: Glossary

direct tag/relation

A tag or relationship that has been manually associated with an entity.
Parent topic: Glossary

implied tag/relation

A tag or relationship tag automatically assigned by the system to an entity.
Parent topic: Glossary

request

The query for input data that seeks either a point’s current or historical value.
Parent topic: Glossary

scope

In programming, the range within a program’s source code within which an element name is recognized without qualification. Variable definitions are not limited to the beginning of a block of code, however, they must be declared before they can be used. In Niagara, the scope of an action applies to the selected components. The component tree is hierarchical. If you delete or move a component that contains other components, you are deleting or moving all items that are contained in that container component (its scope).
Parent topic: Glossary

tag

A piece of semantic information (metadata) associated with a device or point (entity) for the purpose of filtering or grouping entities. Tags identify the purpose of the component or point and its relationship to other entities. For example, you may wish to view only data collected from meters located in maintenance buildings as opposed to those located in office buildings or schools. For this grouping to work, the metering device in each maintenance building includes a tag that associates the meter with all the other maintenance buildings in your system.

Controllers are associated with Supervisors based on tags; searching is done based on tags.

Tags are contained in tag dictionaries. Each tag dictionary is referenced by a unique namespace.

Parent topic: Glossary