

- Once finished, click the **OK** button.

Step 2 - Update the Message Handler

Now that we're including a payload with the message, we need to modify our handler so that it will extract the time from the payload.

- Right-click on the Label component, and select **Configure scripts...**
- Replace the original code with the following:

```
# Access the time by referencing the 'time' key  
self.props.text = payload['time']
```

- Click **OK**.
- To test it, enable **Preview mode**, and click the Button component. You will see the current time populate in the Label.

Related Topics ...

- [Scripting](#)
- [Dictionaries](#)

Perspective Property Change Scripts

With Perspective, individual component properties can have a property change script. When a change script is set up on a property, it will run when the property changes its value. Multiple different properties on the same component can each have different scripts configured. In Perspective, you can put a property change script on any component property.

A very common example of a property change script would be to take the dataset from a binding and modify it into a new dataset using other information on screen. This can be accomplished with a Script Transform instead.

On this page ...

- [Add a Property Change Script](#)
- [Property Change Arguments](#)
- [Change Script Example](#)

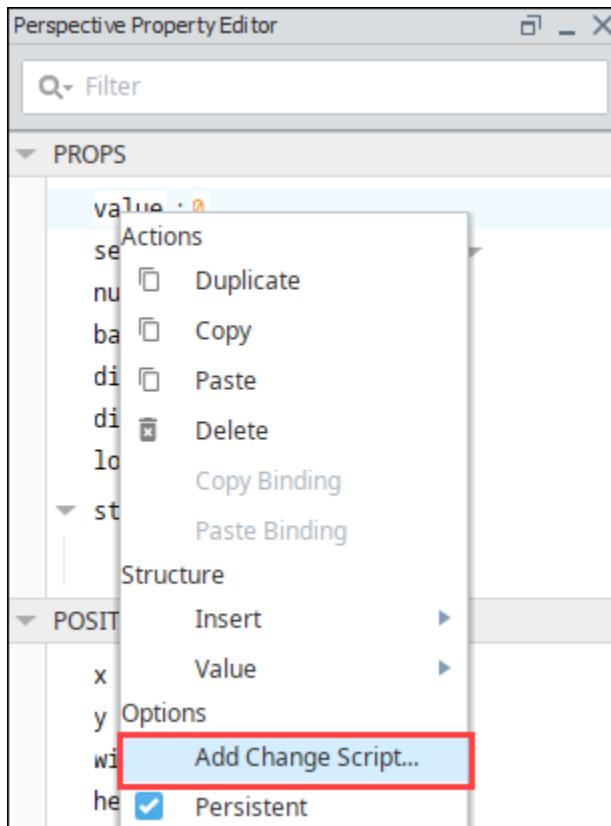


Property Change Scripts

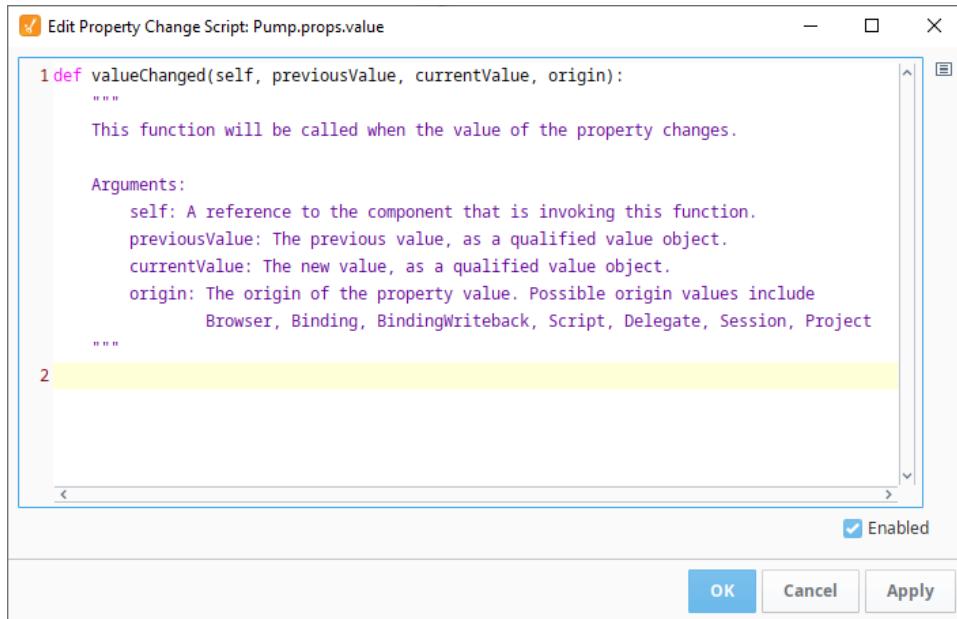
[Watch the Video](#)

Add a Property Change Script

1. To add a property change script to a property, right click on the property Property Editor and select **Add Change Script...**



2. The Edit Property Change Script screen is displayed.



3. Type in the script that you want to run and click **OK**.

Property Change Arguments

Argument	Description		
self	A reference to the component that has the property in question. If the property change script is on a session property , the session object will be passed.		
previousValue	The previous value, as a QualifiedValue object. QualifiedValue objects have a value, quality, and timestamp. See also Scripting Object Reference .		
currentValue	The new value, as a QualifiedValue object. QualifiedValue objects have a value, quality, and timestamp. See also Scripting Object Reference .		
origin	The origin of the property value, as a unicode string. The origin parameter will take on one of six types depending on how the property value is being updated:		
Name	Description	Example	
Browser	Used when the change comes from the Browser interface.	The user changes the text property on a text field by typing a word into the field.	
Binding	Used when the change comes from a binding (or transform) generating a new value.	A Tag changes value, and a property with a binding to that tag is updated.	
Binding Writeback	Used when the change comes from a bidirectional binding writing back to its source.	ComponentB's value property has a bidirectional binding to the value property on ComponentA. If ComponentB's value changes, then a property change script on ComponentA will have an origin of BindingWriteback .	
Script	Used when the change comes from a script.	A user presses a button, and a script on the button assigns a new value to a custom property.	
Delegate	Used when a change to a property comes from something intrinsic to the component's design.	A complex component that automatically fills itself with data, like the alarm status table component.	
Session	Used when the session itself causes the property change.	A change in user privileges causes access to be revoked, resulting in a change in the auth session property.	
Project	Used when the default property value is changed in the designer and saved.	A session property was set to a value of "A." The default value of that property was then changed in the designer to "B" and saved. The value for that property changes from "A" to "B" in the running sessions.	
missedEvent			

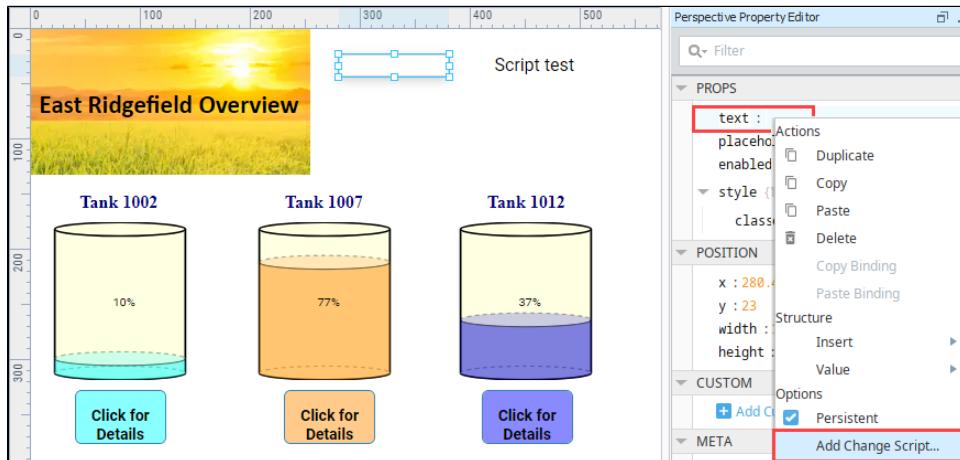
ts

This feature is new in Ignition version **8.1.4**
[Click here](#) to check out the other new features

A flag indicating that some events have been skipped due to event overflow.

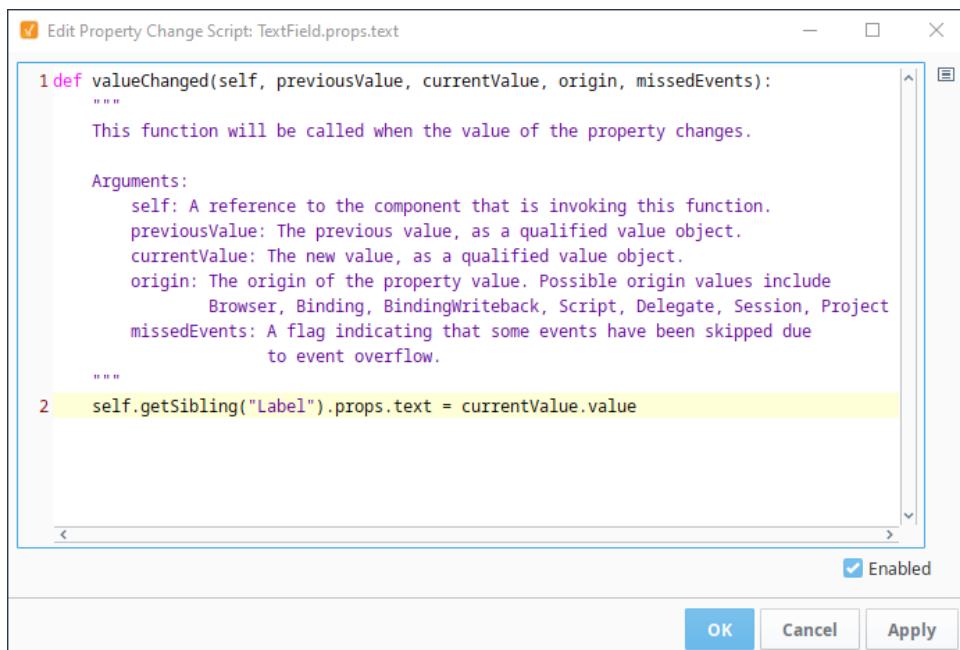
Change Script Example

1. Place a **Text Field** component and **Label** component on a Perspective view.
2. Select the Text Field component then right click on the **text** property.
3. Click on **Add Change Script**.

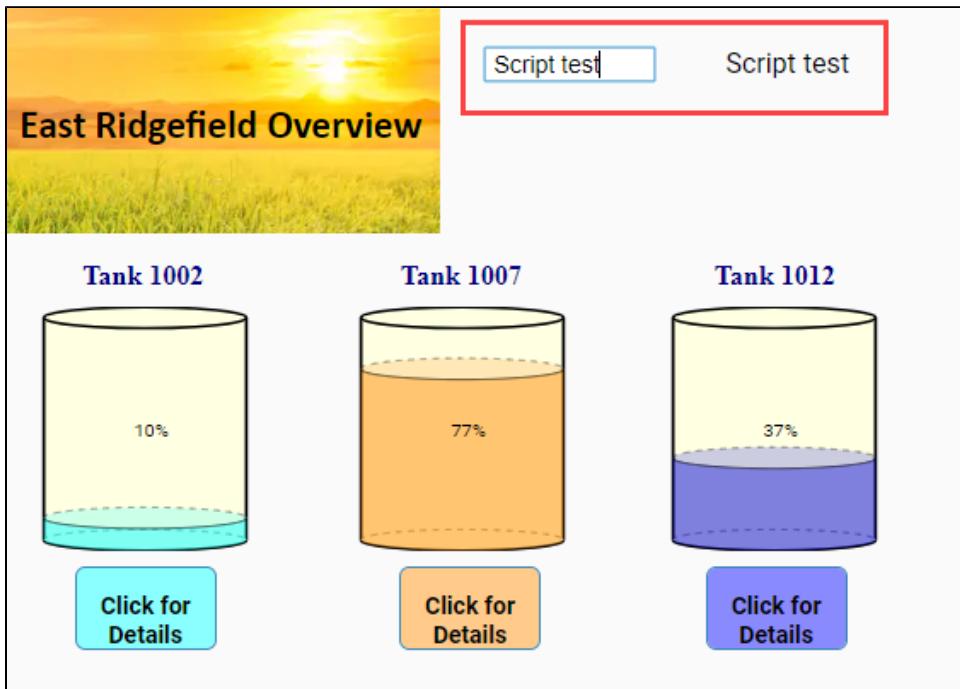


4. The Edit Property Change Script screen is displayed. Enter the following script, which will write the current value of the Text Field component to the Label component.

```
self.getSibling("Label").props.text = currentValue.value
```



5. Click **OK** to commit the script. You'll see that the Property Editor now shows a **Change Script** icon next to the text property.
6. **Save** your project.
7. In a Perspective Session, enter some text into the Text Field component and hit Return. You'll see that the contents are repeated to the Label component.



Perspective Session Event Scripts

Perspective offers a collection of *Session Events* designed to allow the Gateway to track and interact with the Session at critical moments. Specifically, they are scripts that run in the Gateway when a Session starts up, shuts down, or runs a [Native App Action](#).

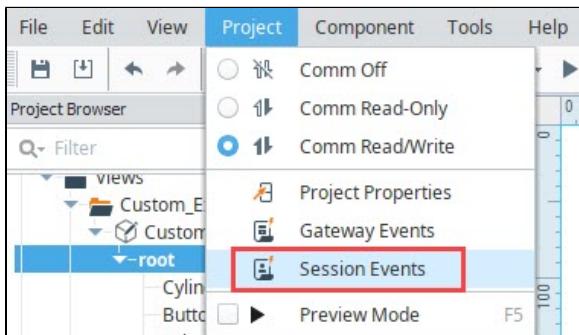
- There are five configurable Session events in Perspective:
- Startup
- Shutdown
- Page Startup (new in 8.1.0)
- Barcode Scanned
- Bluetooth Device Data Received
- NFC NDEF Scanned
- Accelerometer Data Received
- Message

Note: Although they are designed to handle Session Events, the scripts that you write will be run in a *Gateway* scope, not a *Session* scope.

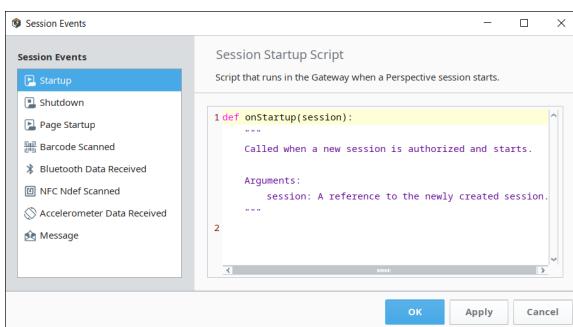
Configuring Session Events

To start working on a Session event script:

1. In the Project Browser, double-click on the **Session Events** section:



2. The **Session Events** dialog will appear:



3. Configure a script by selecting one of the events on the left-hand side.

On this page ...

- Configuring Session Events
- Startup and Shutdown Event Scripts
 - Startup and Shutdown Example
 - Test the Example
- Page Startup
- Native App Event Scripts
- Barcode Scanned
 - Barcode Scanned Example
 - Test the Example
- Bluetooth Data Received
- Accelerometer Data Received
 - Accelerometer Data Received Example
 - Test the Example
- NFC Ndef Scanned
 - NFC Ndef Scanned Example
 - Test the Example
- Message

Startup and Shutdown Event Scripts

Startup and **Shutdown** events run, naturally, whenever a Session starts or ends. In each case, the Gateway will have access to the **session** object associated with the section, complete with all Session properties.

When designing a startup or shutdown event script:

Custom Session properties can be used to pass any additional information to the Gateway, or, in the case of a startup script, to pass information to the newly opening Session. You can configure custom Session properties from the Page Configuration dialog, by clicking on the **Settings** icon in the Designer.

A shutdown event script will run *specifically* when a Session is ending. This happens specifically when any of the following events occur:

- The Session closes due to a timeout. Session timeout is configurable in the **Perspective > General** section of **Project Properties** in the Designer
- The user is no longer authorized to run the Session.
- The redundancy system determines that the Gateway is inactive.
- The licensing system no longer permits the user to run the Session.
- The project is no longer runnable.
- The project is deleted.

Note: Closing the browser tab with the Session will *not* immediately close the Session; the Session must first time out.

Startup and Shutdown Example

This example will record the Session start and end time to the database.

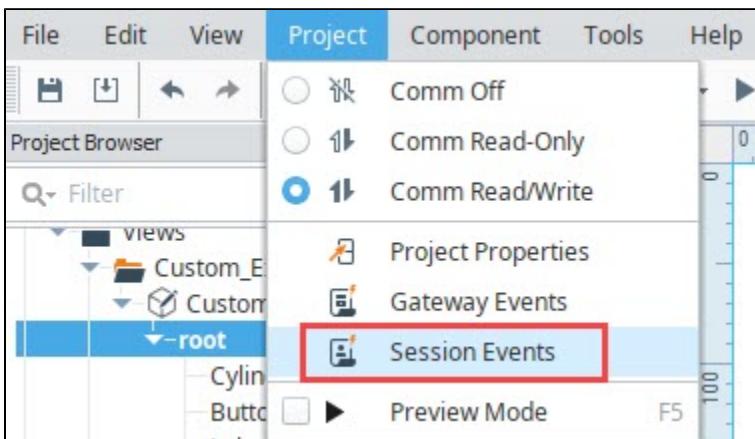
1. For this example, we need to set up a few queries that we can use to write our data to the database.
2. Make a new named query called **Startup Query**.
 - a. Set the Query Type to **Update Query**.
 - b. Set up a single Value type parameter with a name of **SessionID** and a datatype of string.
 - c. Add the query:

```
INSERT INTO sessions (session_id, start_time)
VALUES (:SessionID, CURRENT_TIMESTAMP)
```

3. Make a second new named query called **Shutdown Query**.
 - a. Set the Query Type to **Update Query**.
 - b. Set up a single Value type parameter with a name of **SessionID** and a datatype of string.
 - c. Add the query:

```
UPDATE sessions
SET end_time = CURRENT_TIMESTAMP
WHERE session_id = :SessionID
```

4. Next, we need to add Session Events so that Perspective knows to run those queries on startup and shutdown.
5. Under the **Project** tab, select **Session Events**.



6. On the **Session Events** screen, click the **Startup** icon.
7. Add the following script to the page:

```
# This script will record the time when the Session is opened.

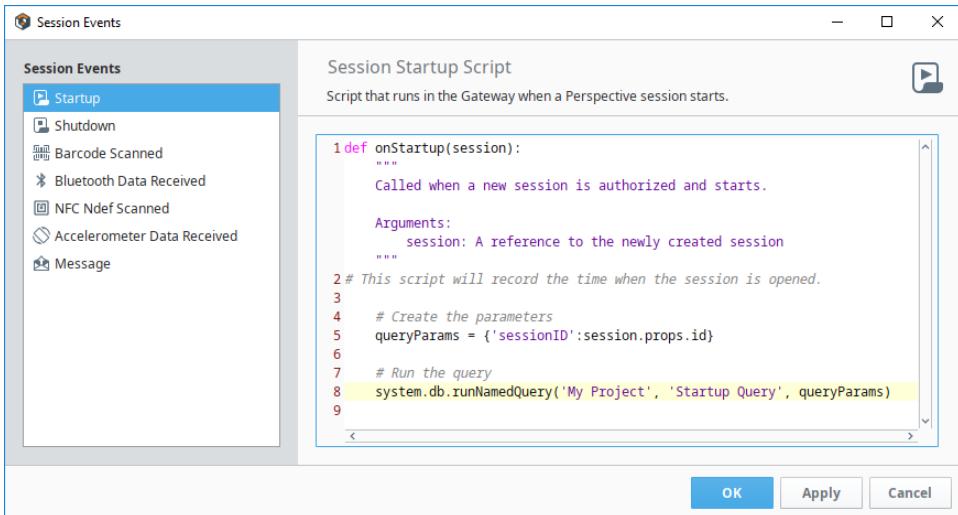
# Create the parameters
```

```

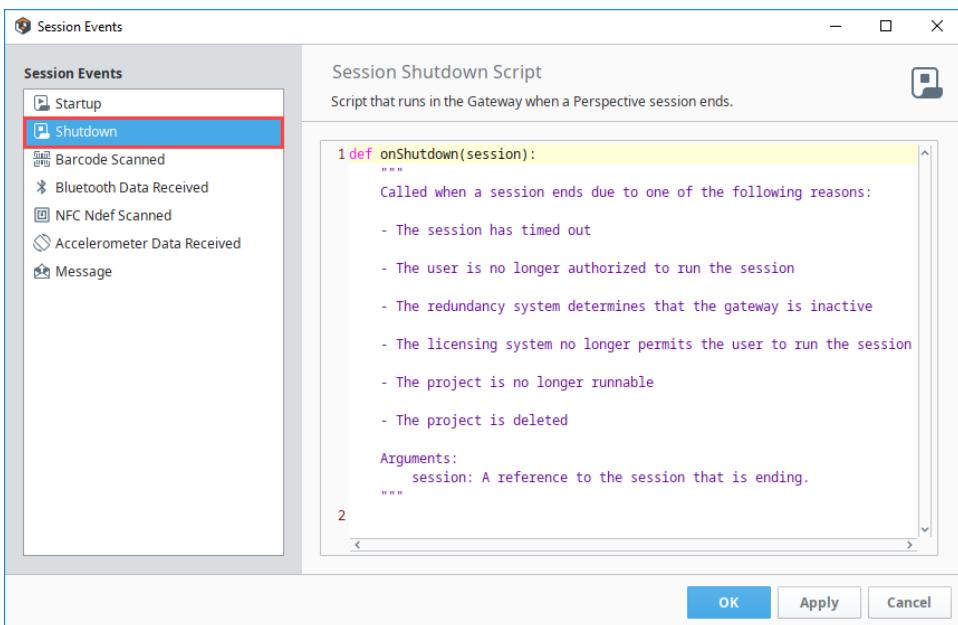
queryParams = {'sessionID':session.props.id}

# Run the query
system.db.runNamedQuery('My Project', 'Startup Query', queryParams)

```



8. Click **Apply** and then click the **Shutdown**  icon.



9. Add the following script to the page:

```

# This script will record the time when the Session times out. Note that after the Session is
closed,
# the Session won't time out until the time out period is reached.

# Create the parameters.
queryParams = {'sessionID':session.props.id}

# Run the query.
system.db.runNamedQuery('My Project', 'Shutdown Query', queryParams)

```

10. Click **OK**.
11. Save your project.

Test the Example

To test the example, open the Perspective App on your mobile device and load the project.

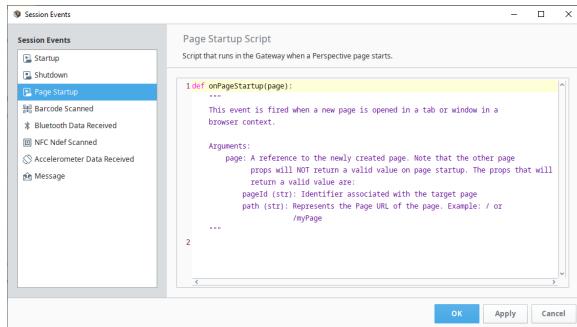
1. You should see a new entry in the database table with the time the Session was started as well as the session ID.
2. After the Session times out (such as after it is closed) you should see the original entry get updated to include the new shutdown time.

Page Startup

This feature is new in Ignition version **8.1.0**
[Click here](#) to check out the other new features

Script that runs in the Gateway when a Perspective page *starts* in a new tab or window.

Note: Navigating to a page configuration in a tab that's already opened (such as using `system.navigate`) will not trigger this event.



Arguments	Description
page	A reference to the newly created page. Note that other page props will not return a valid value on page startup. Accessing properties under this argument requires that you include ".props." along the path. Demonstrations are included below:

Property	Description	Example
pageId	Identifier associated with the target page.	# Provides the page id. pageId = page.props.pageId
path	Represents the Page URL of the page, for example: / or /myPage.	# Provides a path to the page. pagePath = page.props.path

In addition to the arguments above, the Session is available from this event:

```
# For example, we can access the session id with the following:  
sessionId = page.session.props.id
```



Page Startup

[Watch the Video](#)

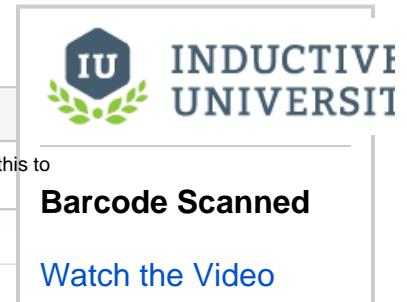
Native App Event Scripts

When the Perspective App is running on a mobile device, it enables users to use tools available on the device, such as GPS location data, the camera, or the accelerometer. The remaining Session events are designed specifically to handle the three [Native App Actions](#).

Barcode Scanned

The scanned barcode action can make use of a mobile device's built in camera.

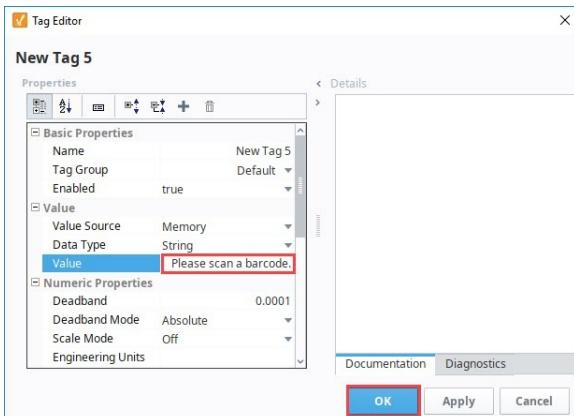
Arguments	Description
session	An object that references the project Session that called the Barcode Scanned event. Use this to identify the specific Session that scanned the barcode.
data	The data returned from the barcode scan. Access the underlying barcode data using: <pre># The value that was scanned data.text # An integer representing a unix timestamp of when the barcode was scanned data.timestamp</pre> <div style="background-color: #f0e68c; padding: 5px; border: 1px solid #d4af37;"><p>This feature is new in Perspective Mobile App version 0.98 Click here to check out the other new features</p></div> <p>Version 0.98 of the Perspective Mobile App added the following property:</p> <pre># The type of barcode that was scanned. data.barcodeType</pre>
context	The user defined context object that can be defined on the action.



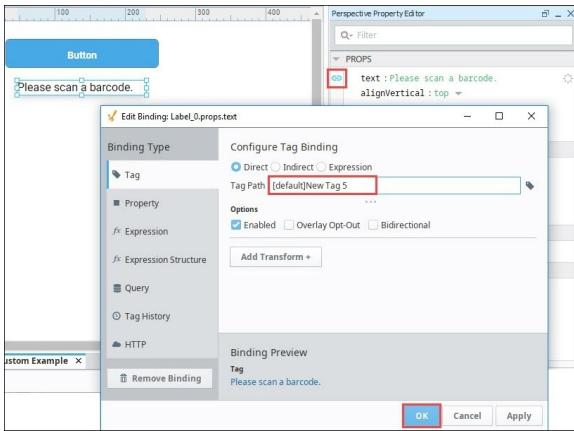
Barcode Scanned Example

This example will scan a barcode, and write its value to a Tag.

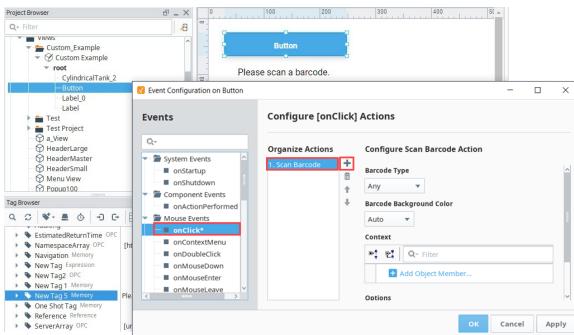
1. For this example, drag a Button component and a Label component onto a view.
2. Create a new memory Tag with a data type of String. Set the value to "Please scan a barcode."



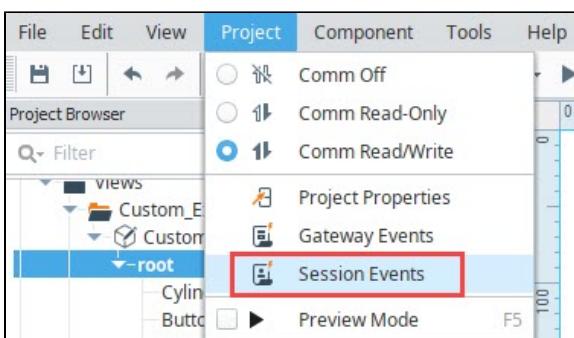
3. Bind the text of the label to the new Tag.



4. Next right-click on the Button component and choose **Configure Events**.
5. On the Event Configuration screen, select **Mouse Events > onClick**.
6. Click the Add icon and select **Scan Barcode** action.



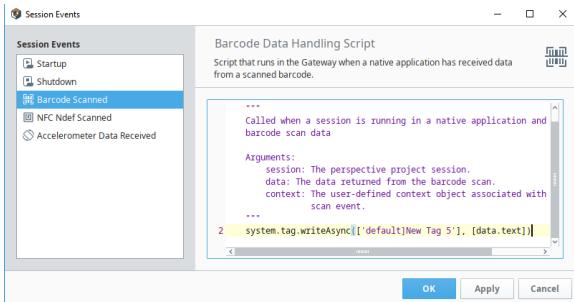
7. Click **OK**. Next we need to set up a session event so that Perspective knows how to interpret the scanned barcode data.
8. Under the **Project** tab, select **Session Events**.



9. On the Session Events screen, select the **Barcode Scanned** icon.
10. Add the following script to the page:

```
system.tag.writeAsync(['[default]New Tag 5'], [data.text])
```

Note: We used the Tag created for this example, New Tag 5. You can enter your own Tag name if different.



11. Click **OK** and save your project.

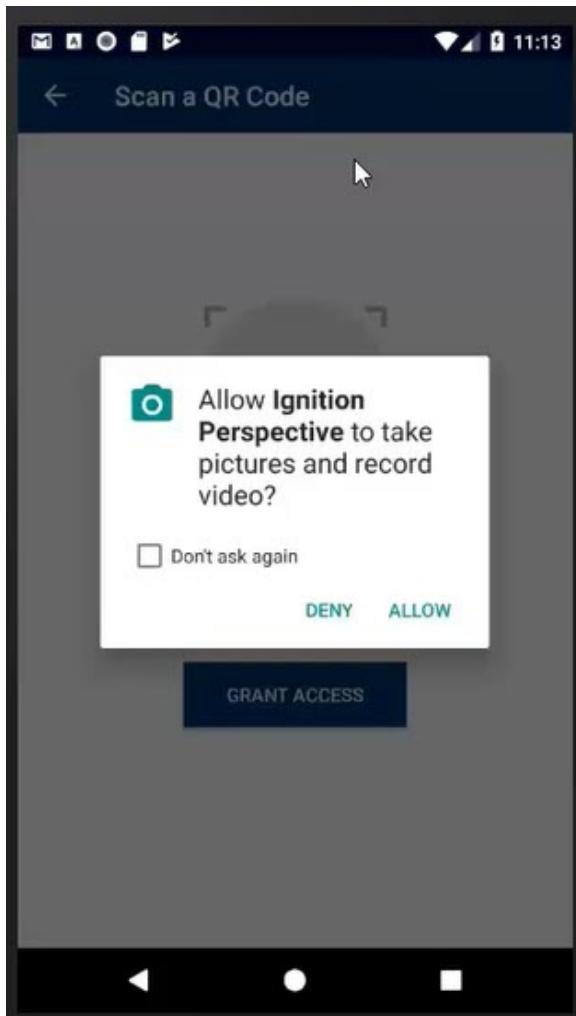
Test the Example

To test the example, open the Perspective App on your mobile device and load the project.

1. Click the **Scan Barcode** button.



2. If this is the first time scanning a barcode, you'll get a message requesting permission for Perspective to take pictures and record.



3. Click **Allow**.
4. You can now use the camera on the mobile device to scan the barcode.



5. Ignition scans the barcode. Once it recognizes the bar code, the script will run and the text is written to the Tag. The label then shows the new Tag value.

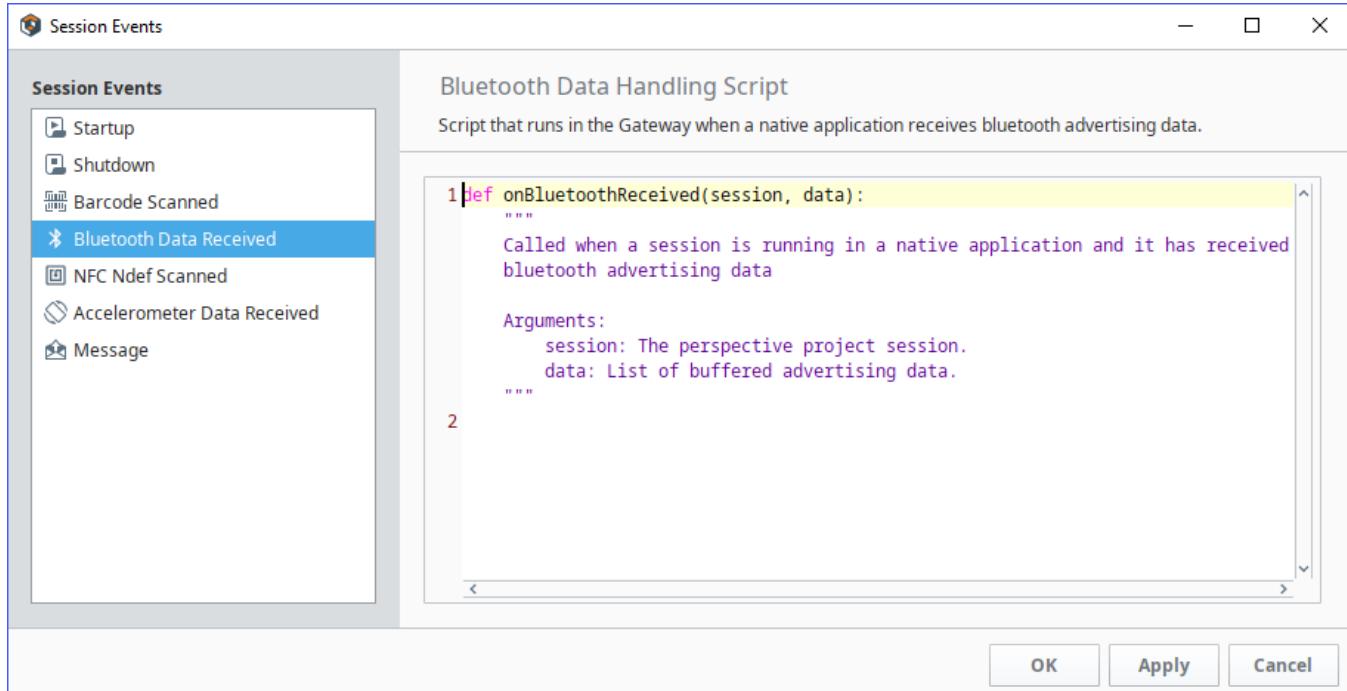


Bluetooth Data Received

The Bluetooth Data Received event is only used when a Session is running in a native application and it has received bluetooth advertising data. This Session event script sends Bluetooth advertising data to Perspective. It supports iBeacon and Eddystone formats.

- Eddystone will work on iOS and Android
- iBeacon on iOS requires user to specify the specific region (iBeacon UUID) located on Session props bluetooth.config.iBeaconRegion.

Note: Bluetooth "Advertising Data" is the name of the communication data according to the Bluetooth spec. There is no connection to advertising as an industry.



Arguments	Description
session	An object that references the Project Session
data	<p>List of buffered advertising data. The data comes in as a data object, which has various parts. The following is example output.</p> <pre> { "values": [{ "rssi": -48, "timestamp": 1570147485167, "manufacturerData": { "companyId": 6, "dataBase64Encoded": "AQkgAl_edccdnHClMvecMCiM--aAkLHAPibd" } }, { "rssi": -48, "timestamp": 1570147486012, "serviceUUIDs": ["FEAA"], "serviceData": { "uuid": "FEAA", "dataBase64Encoded": "AOyqqqqqqqqqqqqqAAAAAAAAA" }, "eddystoneUID": { "txPower": -20, "namespaceID": "AAAAAAAAAAAAAAAAAAAA", "instanceID": "000000000000" } }, { "rssi": -54, "timestamp": 1570147485919, "manufacturerData": { "companyId": 76, "dataBase64Encoded": "AhV88isfQjVLY4VnXXfYqpTyAAAAAL8=" }, "iBeacon": { "uuid": "7CF22B1F-4235-4B63-8567-5D77D8AA94F2", </pre>

```

        "major":0,
        "minor":0,
        "txPower":-65
    }
},
{
    "rssi":-54,
    "timestamp":1570147485987,
    "manufacturerData":{
        "companyId":65535,
        "dataBase64Encoded":"vqwSNFZ4EjQSNBIOEjRWeJASAAAAAOwA"
    },
    "AltBeacon":{
        "manufacturerId":65535,
        "uuid":"12345678-1234-1234-1234-123456789012",
        "instance":"00000000",
        "txPower":-20,
        "manufacturerReserved":"00"
    }
}
]
}

```

Accelerometer Data Received

The Accelerometer Data Received event is only used when data is coming in from a batched Accelerometer Action.

Arguments	Description
session	An object that references the Project Session that called the Accelerometer Data Received event. Use this to identify the specific Session that triggered the batching of accelerometer data.
data	The data returned from the batched accelerometer. The data comes in as a data object, which has various parts. Access the various parts like:
	<pre> logger = system.util.logger('accelerometer') for row in data.values.data: logger.info('X:' + str(row['x'])) logger.info('Y:' + str(row['y'])) logger.info('Z:' + str(row['z'])) </pre>
context	The user defined context object that can be defined on the action.

Accelerometer Data Received Example

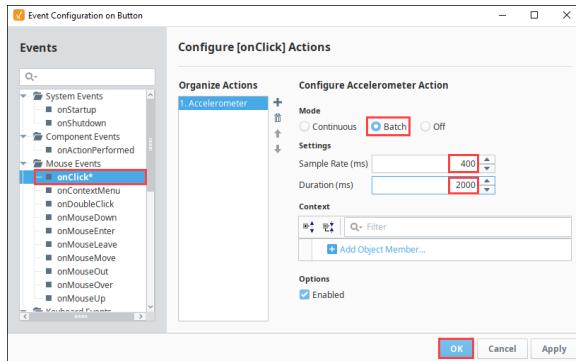
This example will write accelerometer data to the Gateway logs.

1. For this example, put a Button component onto a view.
2. Next right-click on the Button component and choose **Configure Events**.
3. On the Event Configuration screen, select **Mouse Events > onClick**.
4. Click the Add  icon and select **Accelerometer** action.
 - a. Select **Batch** mode.
 - b. Set a **Sample Rate** of 400.
 - c. Set a **Duration** of 2000.

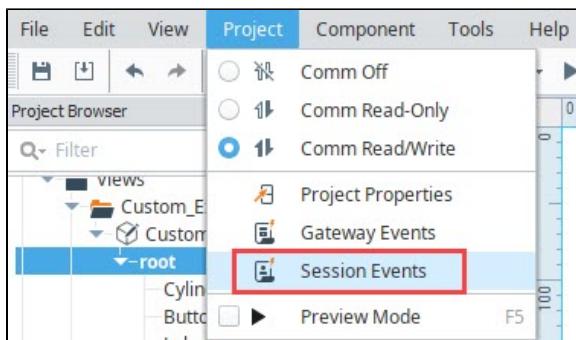


Accelerometer Data Received

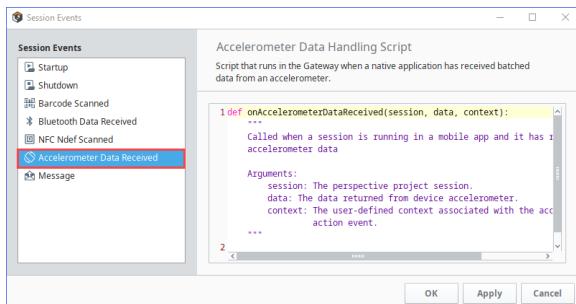
[Watch the Video](#)



5. Click **OK**. Next we need to set up a Session Event so that Perspective knows how to interpret the accelerometer data.
6. Under the **Project** tab, select **Session Events**.



7. On the **Session Events** screen, click the **Accelerometer Data Received** icon.



8. Add the following script to the page:

```
# This script will take the accelerometer data and print it to
the Gateway logs.

# Create the logger.
logger = system.util.logger('accelerometer')

# Loop through the list of batched events and pull out the x, y,
and z values to print to the Gateway logs.
for row in data.values.data:
    logger.info('X:' + str(row['x']) + ', Y:' + str(row
['y']) + ', Z:' + str(row['z']))
```

9. Click **OK**.
10. Save your project.

Test the Example

To test the example, open the Perspective App on your mobile device and load the project.

1. Click the Accelerometer button.

- After clicking the button, the script will record accelerometer data for the next two seconds, so try moving the phone around.
- After the two seconds, you should see the logged information appear in the Gateway logs.

NFC Ndef Scanned

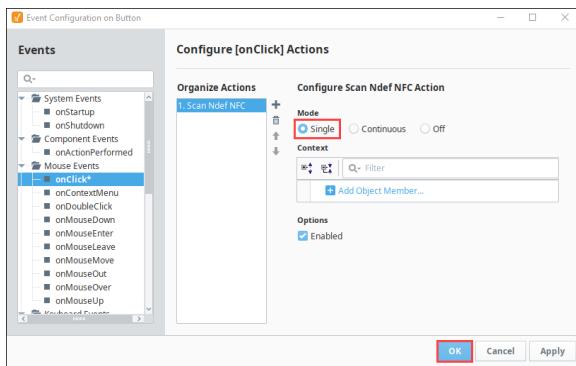
The NFC Ndef Scanned event is used when the NFC Action is used and the mobile device scans an NFC Tag.

Arguments	Description
session	An object that references the Project Session that called the NFC Ndef Scanned event. Use this to identify the specific Session that scanned the NFC Tag.
data	The data returned from the NFC Tag scan. The data object is a list which can contain multiple records from a single NFC Tag. Access the underlying NFC data using: <pre>logger = system.util.logger('NFC') for row in data: logger.info('Type:' + str(row['type'])) logger.info('Type Name Format:' + str(row['typeNameFormat'])) logger.info('Payload:' + str(row['payload'])) logger.info('String:' + str(row['string'])) logger.info('Bytes:' + str(row['bytes']))</pre>
context	The user defined context object associated with the NFC scan event.

NFC Ndef Scanned Example

This example will write the NFC Tag data to the Gateway logs.

- For this example, put a Button component onto a view.
- Next right-click on the Button component and choose **Configure Events**.
- On the Event Configuration screen, select **Mouse Events > onClick**.
- Click the Add **+** icon and select **Scan Ndef NFC** action.
- Select **Single** mode, then click **OK**.

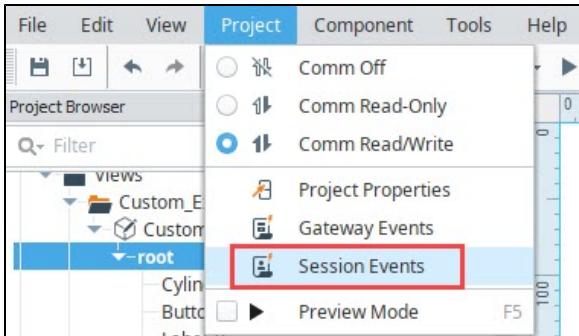


- Next we need to set up a Session Event so that Perspective knows how to interpret the NFC data. Under the Project tab, select **Session Events**.

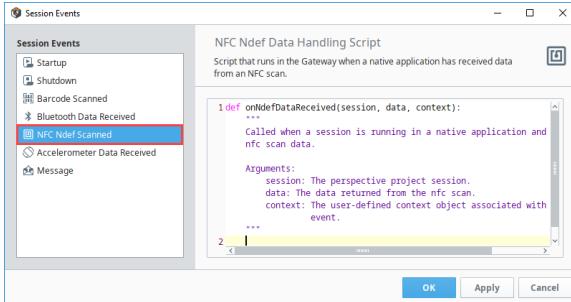


NFC Ndef Scanned

[Watch the Video](#)



7. On the Session Events screen, click the **NFC Ndef Scanned** icon.



8. Add the following script to the page:

```

# This script will take the NFC data and print it to the Gateway
logs.

# Create the logger.
logger = system.util.logger('NFC')

# Loop through the list of records stored in the NFC Tag and
pull out the type, type name format, payload, string data, and
raw byte data from each record and print it to the Gateway logs.
for row in data:
    logger.info('Type:' + str(row['type']) + ', Type Name
Format:' + str(row['typeNameFormat']) + ', Payload:' + str(row
['payload']) + ', String:' + str(row['string']) + ', Bytes:' +
str(row['bytes']))

```

9. Click **OK**.
10. Save your project.

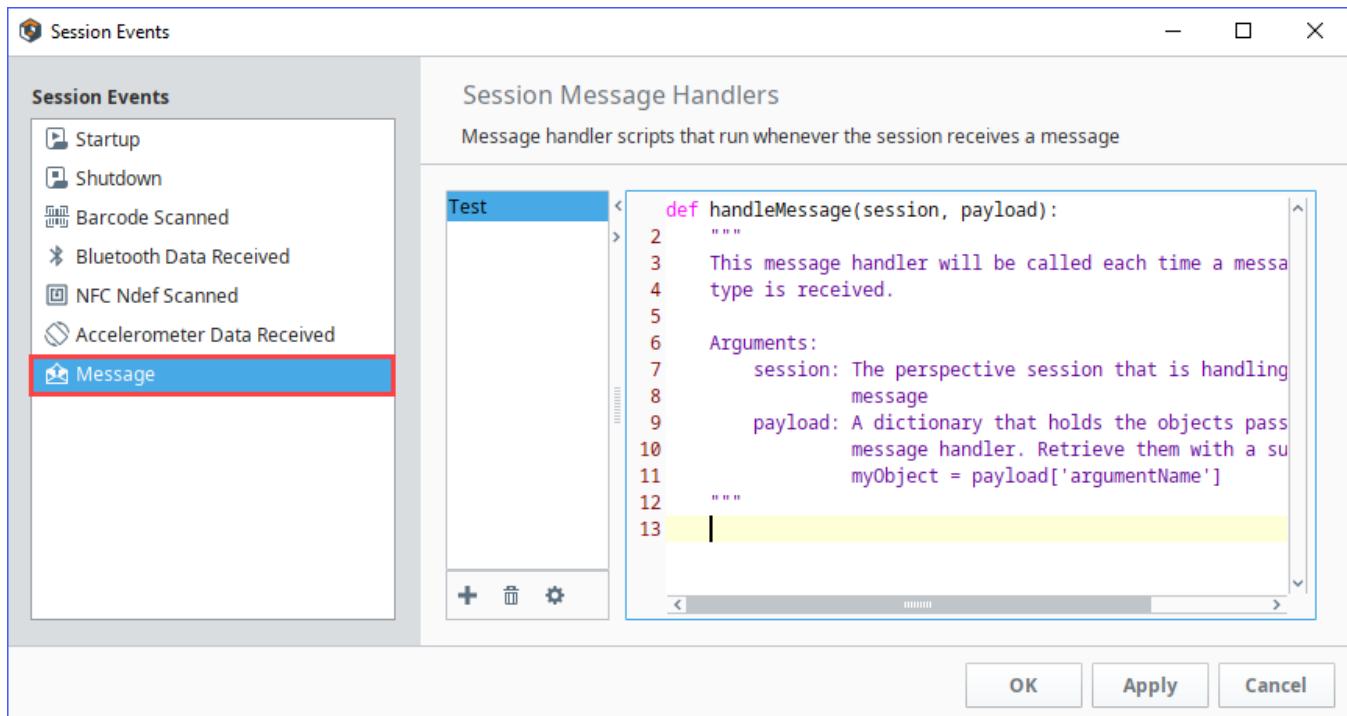
Test the Example

To test the example, open the Perspective App on your mobile device and load the project.

1. Click the **NFC button**.
2. After clicking the button, the script will pass the next NFC Tag scanned to the script to be handled.
3. After scanning an NFC Tag, you should see the logged information appear in the Gateway logs.

Message

The Message Handler scripts will run whenever the Session receives a `system.util.sendMessage` or `system.util.sendRequest`. Note that these types of message handlers are different than component-based message handlers, which are accessed with `system.perspective.sendMessage`. Session Message Handlers can not be called by `system.perspective.sendMessage`.



Arguments	Description
session	The Perspective Session that is handling this message.
payload	A dictionary that holds the objects passed to this message handler. Retrieve them with a subscript, e.g., <code>myObject = payload['argumentName']</code>

Security in Perspective

Security in Perspective is managed through [Identity Providers](#) (IdP). IdPs offers a way for users to log in to Ignition using credentials stored outside of Ignition. This level of security is set up through the Gateway. Setting up Security is covered in the [Security section](#) of the User Manual.

Permissions can also be set at the [Project level](#) in the Designer. This restricts actions such as publishing, viewing, saving, deleting, and editing of project resources to users who have sufficient security levels to do so.

Once you have an IdP setup as well as [Security Levels](#), [Security Level Rules](#), and [User Grants](#) there are additional ways to control security for the following:

- Perspective Sessions
- Perspective Views
- Event actions on Perspective components

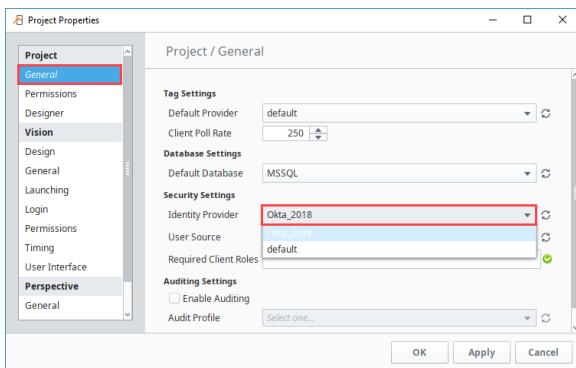
On this page ...

- [Perspective Sessions Security](#)
- [Perspective Views Security](#)
- [Event Actions on Perspective Components](#)

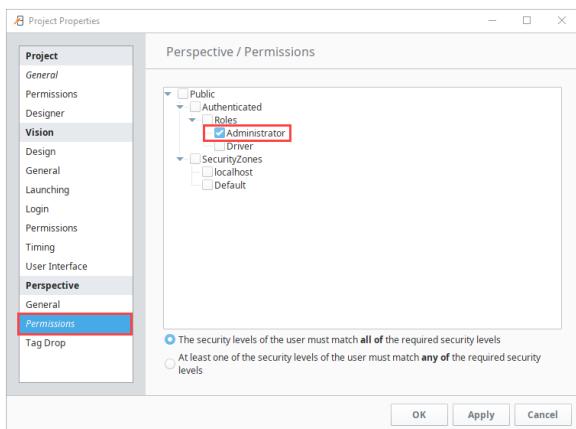
Perspective Sessions Security

For each Project, you can set the security for an associated Perspective Session. When you select the security levels, you are granting any user with that security level access to the Perspective Session for that Project.

1. In the Designer, select the **Project Properties** on the Project menu. Select Project > General.
2. In the Identity Provider field, use the dropdown to select the IdP you want to use or to select the default user source.



3. Scroll down to select **Perspective > Permissions**.
4. Expand the tree to view the security levels you want to be able to access this project in a Perspective Session.
5. Click the check box next to each of the security level you want to grant access.



6. Click **OK** to save all of the Project Properties changes.



Requiring Authentication

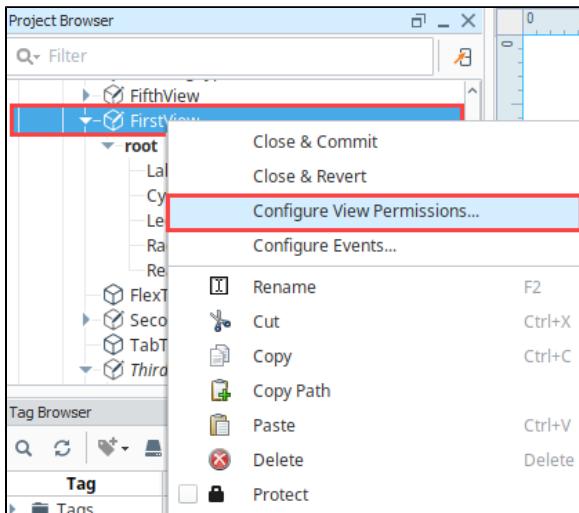
[Watch the Video](#)

Perspective Views Security

You can set the security for an individual View in Perspective. When you select the security levels, you are granting any user with that access to the Perspective Session for that Project.

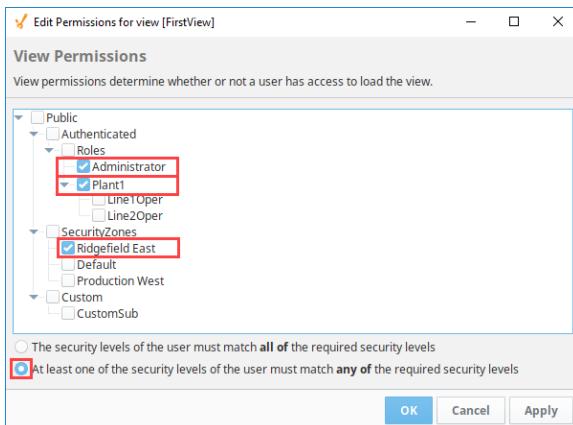
Note: Note that you must have the IdP selected in Project Properties > Project General.

1. In the Project Browser, right click on the view and select **Configure View Permissions...**



2. On the Edit Permissions screen, click the check box next to the security levels that will be able to access this View.
3. Next, click the check box next to the Security Zones that will be able to access this View.
4. Finally, choose one of the radio buttons at the bottom of the screen to indicate whether the user must match **all** of the required security levels you've checked or **any** of them.

In the example below, a user must have either the Administrator security level or Plant1 security level, or be in Ridgefield East to access this View.



5. Click **OK** to save the permissions for this View.

Event Actions on Perspective Components

All Perspective components can have event scripts. These are scripts that run on an action, such as when the user clicks with the mouse on a component. For more information about event scripts see, [Perspective scripting](#). Security can be configured on events. In the following example, set security for the action of clicking on a Button component in the Perspective View.

Note: Note that you must have the IdP selected in Project Properties > Project General.



View Security

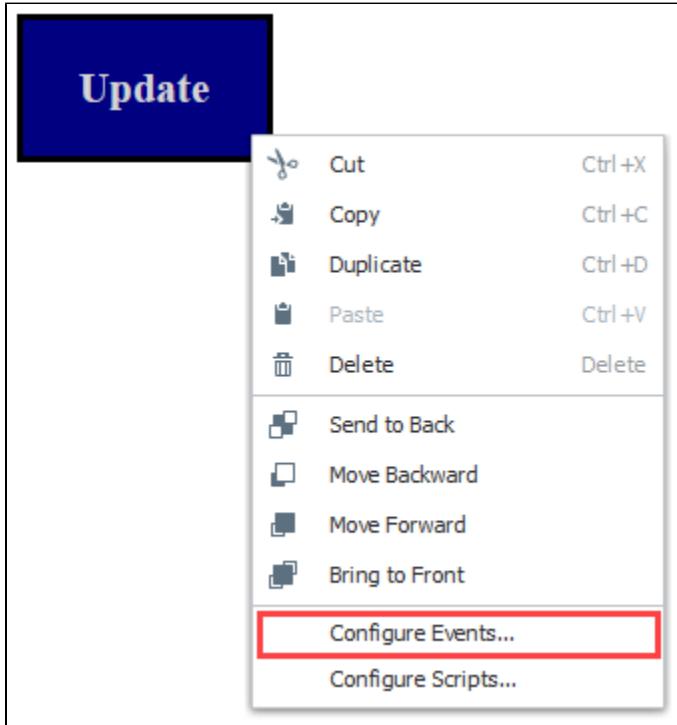
[Watch the Video](#)



Script Action Security

[Watch the Video](#)

1. To add security to an event on a component, right click on the component then choose **Configure Events...**



2. The Events Configuration screen is displayed. Many different types of events can be set for a component. For this example, choose **Mouse Events > onClick**.
3. Under **Organize Actions**, click the Add icon, then select **Script** from the list.

Event Configuration on Icon

Events

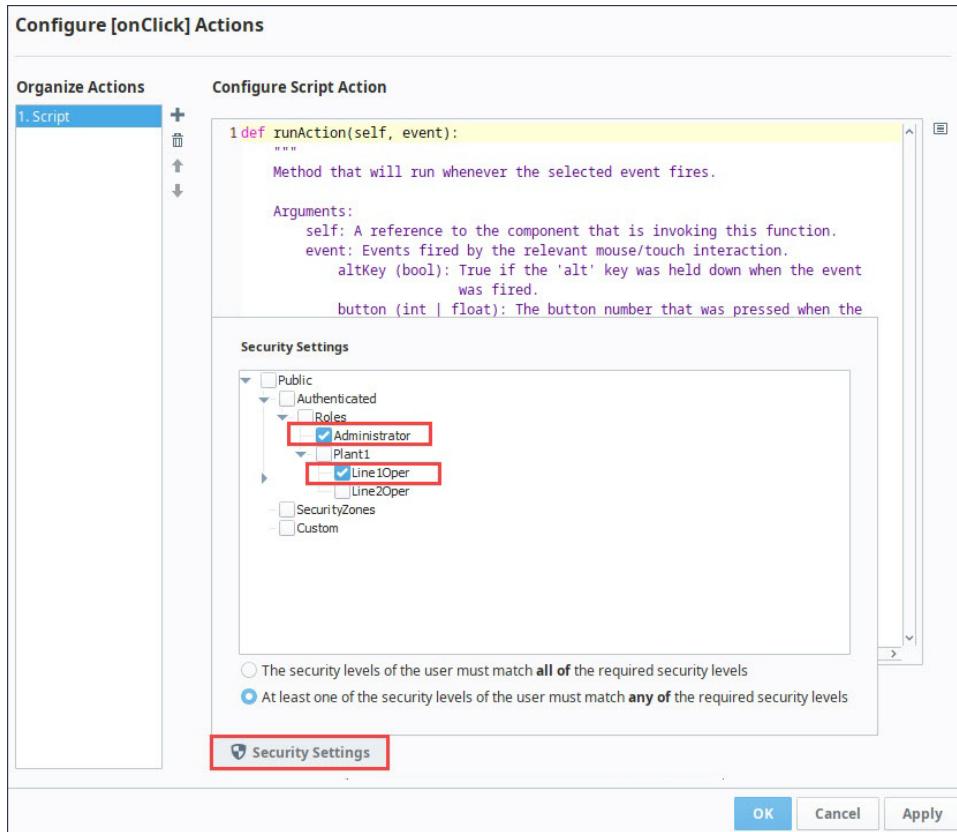
- System Events
 - onStartup
 - onShutdown
- Mouse Events
 - onClick
 - onContextMenu
 - onDoubleClick
 - onMouseDown
 - onMouseEnter
 - onMouseLeave
 - onMouseMove
 - onMouseOut
 - onMouseOver
 - onMouseUp
- Keyboard Events
 - onKeyDown
 - onKeyPress
 - onKeyUp

Configure [onClick] Actions

Organize Actions

- + Accelerometer
- Alter Logging
- Dock
- Login
- Logout
- Navigation
- Popup
- Refresh
- Scan Barcode
- Scan Ndef NFC
- Script**
- Theme

4. Click the **Security Settings** icon near the bottom of the screen.
5. Click the check box next to the security levels you want to grant access. In the example, anyone with Administrator or Line1Oper security levels will have permission to run the script associated with the **onClick** event on this button.



6. Click the **Security Settings**  icon to close the window, then click **OK**.

Alarming in Perspective

Alarming in Perspective is made simple with all the features and functions built right into both the [Alarm Status Table](#) and [Alarm Journal Table](#). If you used the Alarm Status or the Alarm Journal tables in a Vision Client, you'll be pleased to find that most of the functionality on both tables is built-in to the Perspective Alarm Status and Alarm Journal tables and ready to use in a [Perspective Session](#).

Users can filter on a variety of filter options, display alarm event data on the table by selecting different configuration settings, view Realtime and Historical data within a specified time period, view an alarm's details, sort alarm data to meet their individual needs, and use the search function to refine search results. There are a number of properties enabled by default and some other properties have some default options already selected for you. This allows users to hit the ground running right from the start! In addition, the properties in both tables can be configured in the Designer specifically for your project by your designer or Ignition administrator.

You can interface with Alarm Status and Alarm Journal tables in the Designer, Preview Mode and in a Perspective Session.

On this page ...

- [Alarm Status Table](#)
- [Alarm Journal Table](#)

Alarm Status Table

The [Alarm Status Table](#) allows you to view currently active [alarm](#) events in the system, providing an easy way to inspect the [alarm](#) details, shelve [alarms](#), and acknowledge them. At a glance, you will be able to see the current number of active and shelved alarms. The configuration settings, filtering, and search buttons are right on the table for easy access to modifying the alarm data. In a Perspective Session, the table is easily customizable and operators can make on demand changes to configuration settings, filter settings, searching, and sorting of alarm events.

Note: You must have alarms configured in your project first, before any alarms will appear in the Alarm Status Table.

When you drag an Alarm Status Table component into the Designer workspace for the first time, the table will automatically populate if you have alarms created and a [Alarm Journal Profile](#) created. Your Alarm Status Table will look something like the one below. By default, the Alarm Status Table is configured to show '**'Active,Unacked'**', '**'Active,Acked'**', and '**'Cleared,Unack'**' alarm events with priority levels of **Low**, **Medium**, **High** and **Critical**.

Active Time	Display Path	Priority	State	Source	Name
01/23/2020 15:47:51	Sine/Sine2/Low Level	Critical	Active, Unack...	prov:default:/tag:Sine/Sine2:/alm:Low Level	Low Level
01/23/2020 15:43:41	Sine/Sine1/High Level	High	Active, Unack...	prov:default:/tag:Sine/Sine1:/alm:High Level	High Level
01/23/2020 15:40:31	Speed/High Speed	Critical	Active, Unack...	prov:default:/tag:Speed:/alm:High Speed	High Speed
01/23/2020 15:39:31	Sine/Sine2/Low Level	Critical	Cleared, Unac...	prov:default:/tag:Sine/Sine2:/alm:Low Level	Low Level
01/23/2020 15:36:32	Sine/Sine1/High Level	High	Cleared, Unac...	prov:default:/tag:Sine/Sine1:/alm:High Level	High Level
01/23/2020 15:31:11	Sine/Sine2/Low Level	Critical	Cleared, Unac...	prov:default:/tag:Sine/Sine2:/alm:Low Level	Low Level
01/23/2020 15:29:21	Sine/Sine1/High Level	High	Cleared, Unac...	prov:default:/tag:Sine/Sine1:/alm:High Level	High Level

For more detailed information using the Alarm Status Table, refer to the [Common Tasks](#) section.

Alarm Journal Table

The [Alarm Journal](#) stores historical information about alarms in a database. It stores basic data about alarms that have occurred such as timestamp and values of the alarm's properties at the time the alarm event occurred. The Alarm Journal is used by the Alarm Status Table. You can create a single [Alarm Journal Profile](#) to store all of your alarms, or create multiple journals to store alarms across multiple databases. To learn more about alarm journals, refer to the [Alarm Journal](#) page.

Note: You must have an [Alarm Journal Profile](#) created and have a valid [database connection](#) to use the [Alarm Journal Table](#).

The Alarm Journal Table looks and behaves very much like the Alarm Status Table. When you drag an Alarm Journal Table component into the Designer workspace for the first time, the table will automatically populate if you have alarms created, an [Alarm Journal Profile](#), and a valid database connection. By default, the Alarm Journal Table is configured to show 'Active', 'Acked', and 'Cleared' alarm events with priority levels **Low**, **Medium**, **High** and **Critical**. Your Alarm Journal Table will look something like the one below.

The Perspective [Alarm Journal Table](#) has a number of configuration options that can be used to do things like filter on realtime and historical [alarm](#) data within a specified time period, filter on alarm event states and priorities, and the search feature will let you refine your alarm event search results. An operator can filter on demand based on their individual needs. You can also change how the component displays those alarm events by simply changing the configuration settings.

The screenshot shows a perspective component titled "5181 alarm events Last 15 days". At the top, there are several filter buttons: Active, Acknowledged, Cleared, Priority: Low, Priority: Medium, Priority: High, and Priorit. Below the filters, it says "5174 results within filters". The main area is a table with columns: Event Time, Name, Source, Event State, Priority, and Event Value. The table lists 8 rows of alarm data. At the bottom, there are navigation buttons for First, Last, and page numbers (1, 2, 3, 4, 5), and a "Jump to:" input field set to 1.

Event Time	Name	Source	Event State	Priority	Event Value
01/24/2020 11:59:46	High Level	prov:default:/tag:Sine/Sine1:/alm:High Level	Clear	High	79.69064159360343
01/24/2020 11:56:12	Low Level	prov:default:/tag:Sine/Sine2:/alm:Low Level	Active	Critical	23.961130142211914
01/24/2020 11:56:12	Low Level	prov:default:/tag:Sine/Sine2:/alm:Low Level	Ack	Critical	
01/24/2020 11:54:52	High Level	prov:default:/tag:Sine/Sine1:/alm:High Level	Active	High	80.93571064706028
01/24/2020 11:54:52	High Level	prov:default:/tag:Sine/Sine1:/alm:High Level	Ack	High	
01/24/2020 11:52:42	Low Level	prov:default:/tag:Sine/Sine2:/alm:Low Level	Clear	Critical	25.472211837768555
01/24/2020 11:52:37	High Level	prov:default:/tag:Sine/Sine1:/alm:High Level	Clear	High	78.77755076686506
01/24/2020 11:47:52	Low Level	prov:default:/tag:Sine/Sine2:/alm:Low Level	Active	Critical	24.713159561157227

For more detailed information using the Alarm Journal Table, refer to the [Common Tasks](#) section.

Related Topics ...

- [Perspective - Alarm Journal Table](#)
- [Perspective Alarm Journal Table - Common Tasks](#)

In This Section ...

Perspective Alarm Status Table - Common Tasks

The [Perspective Alarm Status Table](#) has a ton of configuration options that can be used to do things like filter the list of alarms being displayed, acknowledge and shelve alarms, and configure how the Alarm Status Table component displays the alarm events. Each of the pages in this section goes over setting up various aspects of the Alarm Status Table.

Perspective Alarm Status - User Interaction

The [User Interaction](#) page introduces you to the Alarm Status Table, its interface and how to navigate around the table. It also briefly describes the Alarm Status Table's features and functions including how to configure the table to get realtime and historical alarm event data.

Perspective Alarm Status - Configuring Properties

The project designer and [Ignition](#) administrator have the option to enable or disable properties and setup filtering options in the Property Editor of the [Designer](#) based on the project requirements and needs of the client users. This page describes which properties are enabled and which properties have some default filtering options preset. Learn how to configure your own Alarm Status properties in the Designer on the [Configuring Properties in the Designer](#) page.

On this page ...

- [Perspective Alarm Status - User Interaction](#)
- [Perspective Alarm Status - Configuring Properties](#)
- [Perspective Alarm Status - General Filtering](#)
- [Perspective Alarm Status - Acknowledgment](#)
- [Perspective Alarm Status - Shelving](#)
- [Perspective Alarm Status - Row Styles](#)

Perspective Alarm Status - General Filtering

The Alarm Status Table has a number of built-in functions right on the table so with a simple click of a button you can filter on alarm states and see alarm details. The Alarm Status Table component has a lot of properties that allow you to filter on various parts of alarms. Learn about all of the different built-in ways that the [Alarm Status Table can filter alarms](#).

Perspective Alarm Status - Acknowledgment

The first step in fixing an alarm is acknowledging that the alarm is happening. [Acknowledgement](#) plays a very important part in any alarm system which is why the ability to acknowledge alarms is built right in to the Alarm Status Table component. Learn how to acknowledge alarms and configure acknowledgement options.

Perspective Alarm Status - Shelving

[Shelving](#) alarms allows you to temporarily silence an alarm for a fixed period of time while you are working on the issue. This can be useful when doing maintenance if Tags are constantly going in and out of alarm.

Perspective Alarm Status - Row Styles

The Alarm Status Table uses different styled rows to differentiate between alarms in different states. These [Row Styles](#) can be completely customized using whatever colors and fonts you want. You can even add an animated style drawing an operator's attention to critical alarms!

In This Section ...

Perspective Alarm Status - User Interaction

Getting Started with the Alarm Status Table

If you're setting up the Alarm Status Table for the first time, you probably will want to take a look at the status table's properties in the Property Editor. There a number of properties that are enabled by default such as the shelve, unsheelve, acknowledge, filter, and configuration buttons, as well as the alarm details popup to name a few. You should become familiar with them in the event you would like to change them. Other properties have default options already selected for you, like alarm states, alarm priorities, and row styles.

When you drag your first Alarm Status Table component into the Designer, the Alarm Status Table will automatically populate if you have any alarms configured and an [Alarm Journal Profile](#) created. It will look something like the image below.

You can interact with the table in the Designer, in Preview Mode of the Designer, and in a [Perspective Session](#). The Alarm Status Table properties are configured in the Property Editor of the Designer.

When setting up your Alarm Status Table, you will have to toggle between both the Designer and Preview Modes to configure properties and organize the display of your alarm event data and size the data columns in the table.

This page will introduce you to the Alarm Status Table and its features and functions. Other pages in the [Perspective Alarm Status Table - Common Tasks](#) section address the details of how to [acknowledge](#), [she](#) [lve](#), and [filter](#) alarms.

Active Time	Display Path	Priority	State	Source	Event Value
01/28/2020 10:35:42	Sine/Sine1/High Temp	High	Active, Unackn...	prov:default:/tag:Sine/Sine1:/alm:High Temp	81.10549
01/28/2020 10:33:52	Sine/Sine2/Low Level	Critical	Active, Unackn...	prov:default:/tag:Sine/Sine2:/alm:Low Level	24.106287
01/28/2020 10:31:53	Speed/High Speed	Critical	Active, Unackn...	prov:default:/tag:Speed:/alm:High Speed	125
01/28/2020 10:28:32	Sine/Sine1/High Temp	High	Cleared, Unack...	prov:default:/tag:Sine/Sine1:/alm:High Temp	78.594696
01/28/2020 10:25:32	Sine/Sine2/Low Level	Critical	Cleared, Unack...	prov:default:/tag:Sine/Sine2:/alm:Low Level	25.312988
01/28/2020 10:24:00	Speed/High Speed	Critical	Cleared, Unack...	prov:default:/tag:Speed:/alm:High Speed	99
01/28/2020 10:22:49	Sine/Sine1/High Temp	High	Cleared, Unack...	prov:default:/tag:Sine/Sine1:/alm:High Temp	79.55426
01/28/2020 10:17:12	Sine/Sine2/Low Level	Critical	Cleared, Unack...	prov:default:/tag:Sine/Sine2:/alm:Low Level	25.788023
01/28/2020 10:15:57	Speed/High Speed	Critical	Cleared, Unack...	prov:default:/tag:Speed:/alm:High Speed	98
01/28/2020 10:08:52	Sine/Sine2/Low Level	Critical	Cleared, Unack...	prov:default:/tag:Sine/Sine2:/alm:Low Level	26.110838
01/28/2020 10:08:51	Speed/High Speed	Critical	Cleared, Unack...	prov:default:/tag:Speed:/alm:High Speed	77

Anatomy of an Alarm Status Table

The following image shows the basic anatomy of an [Alarm Status Table](#) with alarm event data populated in the rows. At the top of the table, you can immediately see two tabs, one for **Active** alarms and the other for **Shelved** alarms. The tab headers provide a count of the number of Active alarms and Shelved alarms. Click on each of the tabs to view all the currently Active alarms and Shelved alarms.

There are a host of **Configuration Settings** and **Filtering options** that you can set to provide relevant information about your alarms events. The **Configuration Settings** contain the available header columns that can be set to display data about the alarm event such as Ack Notes, Ack Pipeline, Ack Time, Ack User, Active Time, Display Path, Source Path, and many more. Filtering options are actually the alarm states and priorities that can be selected to get the most current alarm status in your system. Alarm states are **ActiveUnacknowledged**, **ActiveAcknowledged**, **ClearUnacknowledged** and **ClearAcknowledged**. Alarm Priorities are **Diagnostic**, **Low**, **Medium**, **High** and **Critical**.

On this page ...

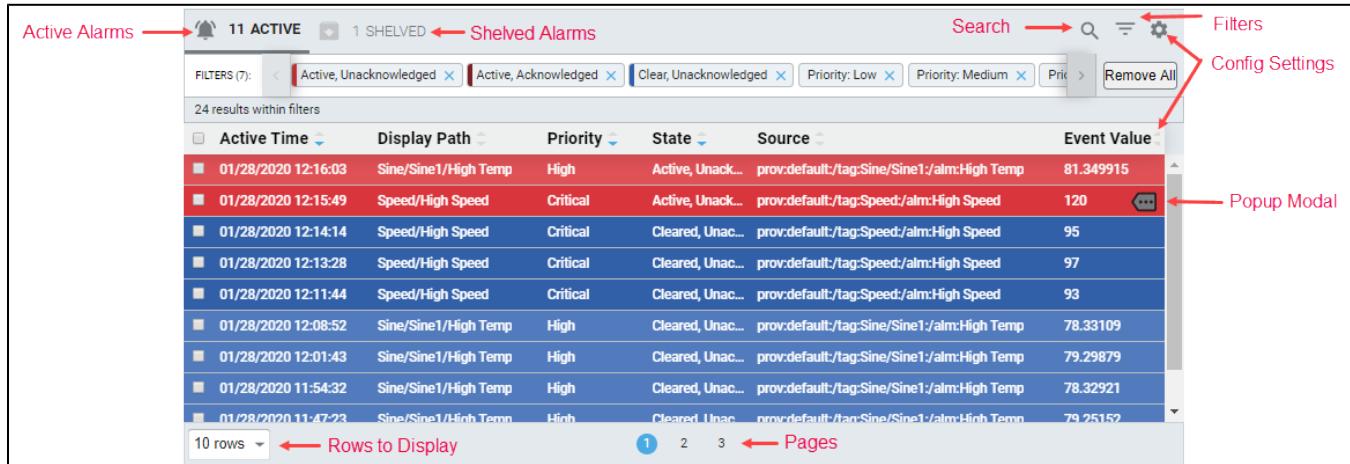
- [Getting Started with the Alarm Status Table](#)
 - [Anatomy of an Alarm Status Table](#)
 - [Alarm Details](#)
 - [Acknowledging and Shelving Alarms](#)

The **Filter Bar** shows you exactly what state and priority filter properties are set. The Filtering tab allows you select the alarm state and filter on specific alarm criteria. You can remove individual filter properties by clicking the 'X' next to each property or remove all the filter properties using the **Remove All** button. To add filter properties, click on the **Filter** icon  and select the properties you want to use.

You can also optimize your search results by using the **Search**  feature and entering keywords or a string to help further refine your results.

When you mouse over an alarm row, a popup modal will appear on the right side of the table. By clicking the **Popup Modal**  icon, you can see the alarm's details. You can also set the number of alarm rows to display and scroll through the list of pages at the bottom of the status table. Lastly, the Alarm Status Table comes with a default set of customizable row colors based on the alarm state and priority.

To learn more about using the **Alarm Status** functions and features such as alarm acknowledgement, shelving, filtering and row styles, refer to the [Perspective Alarm Status Table - Common Tasks](#) section. The [Alarm Status Table](#) component page describes all the alarm status properties and how to use them.



The screenshot shows a table titled "Active Alarms" with a count of "11 ACTIVE" and "1 SHELVED". The table has columns for Active Time, Display Path, Priority, State, Source, and Event Value. A "Filters" button is at the top right, along with a "Search" bar and a "Config Settings" button. Red arrows point to several UI elements: "Active Alarms" (top left), "Shelved Alarms" (top center), "Filters" (top right), "Config Settings" (top right), "Rows to Display" (bottom left), and "Pages" (bottom center). A "Popup Modal" icon is shown on the right side of the table, with a red arrow pointing to it.

Active Time	Display Path	Priority	State	Source	Event Value
01/28/2020 12:16:03	Sine/Sine1/High Temp	High	Active, Unack...	prov:default:/tag:Sine/Sine1:/alm:High Temp	81.349915
01/28/2020 12:15:49	Speed/High Speed	Critical	Active, Unack...	prov:default:/tag:Speed:/alm:High Speed	120
01/28/2020 12:14:14	Speed/High Speed	Critical	Cleared, Unac...	prov:default:/tag:Speed:/alm:High Speed	95
01/28/2020 12:13:28	Speed/High Speed	Critical	Cleared, Unac...	prov:default:/tag:Speed:/alm:High Speed	97
01/28/2020 12:11:44	Speed/High Speed	Critical	Cleared, Unac...	prov:default:/tag:Speed:/alm:High Speed	93
01/28/2020 12:08:52	Sine/Sine1/High Temp	High	Cleared, Unac...	prov:default:/tag:Sine/Sine1:/alm:High Temp	78.33109
01/28/2020 12:01:43	Sine/Sine1/High Temp	High	Cleared, Unac...	prov:default:/tag:Sine/Sine1:/alm:High Temp	79.29879
01/28/2020 11:54:32	Sine/Sine1/High Temp	High	Cleared, Unac...	prov:default:/tag:Sine/Sine1:/alm:High Temp	78.32921
01/28/2020 11:47:23	Sine/Sine1/High Temp	High	Cleared, Unac...	prov:default:/tag:Sine/Sine1:/alm:High Temp	79.25152

Alarm Details

When you click on the Popup Modal icon , this brings up the Alarms Detail window. Here you can scroll through a list of configuration properties and see the details about the alarm event.

Alarm Details

Config Properties	
On Active	
Group	Production
Ack Mode	Manual
Ack Notes Reqd	<input type="checkbox"/>
Active Pipeline	Production/Email Pipeline
Event Time	2020-01-29 14:25:37.057
Event Value	105
Mode	Above Setpoint
Name	High Speed
Notes	
Priority	Critical
Setpoint A	100.0

Acknowledging and Shelving Alarms

When an alarm is selected, a footer will slide into view below the table containing two buttons, one to **Shelf** alarms and the other to **Acknowledge** alarms. Check the box next to the alarm event to either shelf or acknowledge the alarm. With the shelved alarms visible, you'll have the ability to **Unshelf** selected alarms. This footer will also display messages as feedback when performing an action on the selected alarms (i.e., success, failure, etc.). To learn more, refer to the [Shelving](#) and [Acknowledgement](#) pages in this section.

The screenshot shows a table of active alarms with the following columns: Active Time, Display Path, Priority, State, Name, and Event Value. The first row is highlighted in red, indicating it is selected. At the bottom of the table, there is a footer bar with two buttons: 'Shelf' and 'Acknowledge'. The 'Acknowledge' button is highlighted with a red box.

FILTERS (3): Active, Unacknowledged, Priority: High, Priority: Critical

10 results within filters

Active Time	Display Path	Priority	State	Name	Event Value
01/29/2020 14:25:37	Speed/High Speed	Critical	Active, Unacknowledged	High Speed	105
01/29/2020 10:50:55	Writeable/WriteableInteger1/Low Ta...	Critical	Active, Unacknowledged	Low Tank Level	19
01/09/2020 11:39:18	Tank Level 2/Low SP2	Critical	Active, Unacknowledged	Low SP2	0.0
01/09/2020 11:38:43	High Temp/High Temp	Critical	Active, Unacknowledged	High Temp	98
01/30/2020 15:59:12	Sine/Sine1/High Temp	High	Active, Unacknowledged	High Temp	80.453865

Footer Buttons: Shelf, Acknowledge

Related Topics ...

- [Perspective Alarm Status Table - Common Tasks](#)
- [Perspective - Alarm Status Table](#)

Perspective Alarm Status - Configuring Properties in Designer

Note: Alarms must be set up on Tags for them to show up in the Alarm Status Table.

Getting Started

Filtering can be done in both the Designer and in a Perspective Session, but enabling and disabling Alarm Status Table functions and setting up specified filtering options for client users is configured in the Designer. The project designer or Ignition administrator have the permission to enable or disable properties and setup filtering options in the Property Editor of the Designer based on the project requirements and needs of the client users. If you are the project designer or Ignition administrator, you probably want to check out all the [Alarm Status Table](#) properties in the Perspective Property Editor. It's a good idea to scroll through all the alarm properties and expand them to see additional properties and see the default property settings and preselected filtering options.

The Alarm Status Table has some properties enabled by default, and some filter options already preset in the Designer to give client users a head start using the table. It's in the Property Editor of the Designer where table properties are enabled and disabled, and filtering options are configured such as the alarm states and priorities to display, setting the shelving times, defining row styles, choosing column headers and column sort options, or setting up a specific display path and source path for operators to view, and much more.

The first time an Alarm Status Table component is dragged on to a window in the Designer, by default, the table displays all the alarms that are currently 'Active and Unacknowledged,' 'Active and Acknowledged,' and 'Cleared and Unacknowledged,' with a sort priority ranging from Low to Critical.

The Alarm Status Table below is similar to what you'll see when you first drag in an Alarm Status Table to a Designer window.

The screenshot shows the Ignition Designer interface with a Perspective Alarm Status component. The component displays a table of alarms with the following data:

Active Time	Display Path	Priority	State	Source	Name
01/09/2020 11:39:18	Tank Level 2/Low SP2	Critical	Active, Unack...	prov:default:/tag:Tank Level 2:/alm:...	Low SP2
01/28/2020 17:05:31	Sine/Sine2/Low Level	Critical	Active, Unack...	prov:default:/tag:Sine/Sine2:/alm:Lo...	Low Level
01/28/2020 17:03:01	Speed/High Speed	Critical	Active, Unack...	prov:default:/tag:Speed:/alm:High S...	High Speed
01/09/2020 11:38:43	High Temp/High Temp	Critical	Active, Unack...	prov:default:/tag:High Temp:/alm:Hi...	High Temp
01/09/2020 11:39:18	Writeable/WriteableInteger1/Lo...	Critical	Active, Ackno...	prov:default:/tag:Writeable/Writeable...	Low Tank Level
01/28/2020 12:13:28	Speed/High Speed	Critical	Cleared, Unac...	prov:default:/tag:Speed:/alm:High S...	High Speed
01/28/2020 12:15:49	Speed/High Speed	Critical	Cleared, Unac...	prov:default:/tag:Speed:/alm:High S...	High Speed
01/28/2020 16:57:12	Sine/Sine2/Low Level	Critical	Cleared, Unac...	prov:default:/tag:Sine/Sine2:/alm:Lo...	Low Level

The Perspective Property Editor on the right shows the following configuration:

- PROPS:
 - enableHeader : true
 - enableDetails : true
 - enableAcknowledge : true
 - enableShelve : true
 - enableUnshelve : true
- toolbar (7):
 - enabled : true
 - enableActiveTab : true
 - enableShelvedTab : true
 - enableFilter : true
 - enableFilterResults : true
 - enablePreFilters : true
 - enableConfiguration : true
- shelvingTimes [6]
- responsive {2}
- filters {2}:
 - active {5}:
 - text :
 - states {4}
 - priorities {5}
 - conditions {3}
 - results {2}
 - shelved {2}

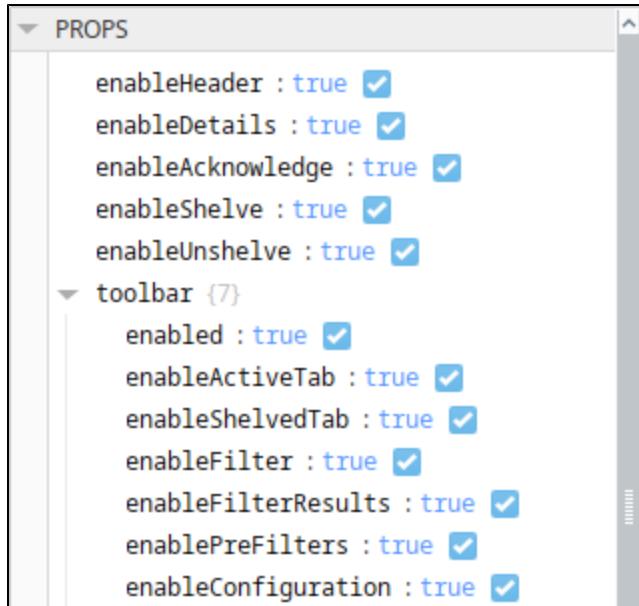
Configuring Properties in the Designer

When you drag a [Alarm Status Table](#) component into the Designer for the first time, it displays the alarm results with default properties already configured and some filtering options already preselected for you. Some of the default settings may work for your client users, but others may need to be modified in the Property Editor.

The first thing you might want to look at is the functionality that is enabled on the table component.

On this page ...

- [Getting Started](#)
- [Configuring Properties in the Designer](#)
 - [Preview Mode](#)
 - [Shelving Times](#)
 - [Filtering on Alarm States and Alarm Priorities](#)
 - [Filtering on Source Path and Display Path](#)



While in the Property Editor, expand the table's filters properties to see what filter properties are configured.

Preview Mode

When configuring the Alarm Status Table in the Designer, you have to go to [Preview Mode](#) to see your updates, filter, sort and organize the alarm data in the table. When making modifications to the properties in the Property Editor, you will find yourself toggling between the Designer and Preview Modes multiple times to make sure your modifications are displayed the way you want them. Each time you reset your filtering options and column headers in the Designer, the alarm table will refresh with new alarm data based on your configuration and filter settings

Here are a couple of examples of Alarm Status Table properties that you may want to modify.

Shelving Times

In this example, let's add a new shelve time.

The default shelvng times are as follows:

- 5 minutes
- 15 minutes
- 30 minutes
- 1 hour
- 2 hours
- 4 hours

Let's say you wanted to add another shelving time for 90 minutes. The project designer will need to update the '**shelvingTimes**' property to make 90 minutes available to operators. In the Property Editor, shelving times are entered as seconds, not minutes.

Index	Value (Seconds)
0	300
1	900
2	1,800
3	3,600
4	7,200
5	14,400

1. To add a shelving time for 90 minutes, mouse over the value 3,600 (1 hour) and right click, select 'Add after' and click 'Value.'

shelvingTimes [6]

- 0 : 300
- 1 : 900
- 2 : 1,800
- 3 : 3,600
- 4 : 7,200
- 5 : 14,400

Actions

- Duplicate
- Copy
- Paste
- Delete
- Copy Binding
- Paste Binding

Structure

- Add before ▶
- Add after ▶ **Value**
- Change to ▶ Object

Options

- Add Change Script...
- Persistent
- Access ▶

2. Enter **5400** (seconds).

shelvingTimes [7]

- 0 : 300
- 1 : 900
- 2 : 1,800
- 3 : 3,600
- 4 : 5,400
- 5 : 7,200
- 6 : 14,400

3. Now, open your window in **Preview Mode**, and you'll see your updated shelving times.

Time	Display Path	Priority	State	Source	Shelf Time
01/09/2020 11:39:18	Tank Level 2/Low SP2	Critical	Active, Unacknowledged	prov:default:/tag:Tank Level 2	1 hour, 30 minutes
01/31/2020 14:57:12	Sine/Sine2/Low Level	Critical	Active, Unacknowledged	prov:default:/tag:Sine/Sine2	15 minutes
01/29/2020 14:25:37	Speed/High Speed	Critical	Active, Unacknowledged	prov:default:/tag:Speed/High Speed	30 minutes
01/29/2020 10:50:55	Writeable/WriteableInteger1/Low Tank ...	Critical	Active, Unacknowledged	prov:default:/tag:Writeable/WriteableInteger1	1 hour
01/09/2020 11:38:43	High Temp/High Temp	Critical	Active, Unacknowledged	prov:default:/tag:High Temp/High Temp	1 hour, 30 minutes

Filtering on Alarm States and Alarm Priorities

By default, the '**clearAcked**' alarm state and the '**diagnostic**' priority are not enabled properties, but say for example the '**diagnostic**' property is required by operators. To make this property available, the project designer or Ignition administrator can go to the Property Editor to enable '**diagnostic**' for the operators to use in a Perspective Session.

1. Expand the **filters > active > states > priorities** properties. Here you see the default states and priorities.

The screenshot shows the 'filters > active > states > priorities' properties in the Property Editor. The 'states' section contains four items: 'activeUnacked' (checked), 'activeAcked' (checked), 'clearUnacked' (checked), and 'clearAcked' (unchecked). The 'priorities' section contains five items: 'diagnostic' (unchecked), 'low' (checked), 'medium' (checked), 'high' (checked), and 'critical' (checked). A yellow box highlights the 'clearAcked' and 'diagnostic' properties.

Property	Value	Status
activeUnacked	true	checked
activeAcked	true	checked
clearUnacked	true	checked
clearAcked	false	unchecked
diagnostic	false	unchecked
low	true	checked
medium	true	checked
high	true	checked
critical	true	checked

2. Click the checkbox for the '**diagnostic**' priority property to enable it making it available to the operators in a Perspective Session.

The screenshot shows the same properties after enabling the 'diagnostic' priority. The 'diagnostic' property in the 'priorities' section now has a checked checkbox, indicated by a red box around the entire row.

Property	Value	Status
activeUnacked	true	checked
activeAcked	true	checked
clearUnacked	true	checked
clearAcked	false	unchecked
diagnostic	true	checked
low	true	checked
medium	true	checked
high	true	checked
critical	true	checked

3. Go to **Preview Mode** to verify the '**diagnostic**' property was added. You will see it displayed in the Filter Bar.

12 ACTIVE						0 SHELVED
FILTERS (5): Active, Unacknowledged Active, Acknowledged Priority: Diagnostic Priority: High Priority: Critical Remove All						
11 results within filters						
Active Time	Display Path	Priority	State	Name	Event Value	
02/03/2020 13:47:12	Sine/Sine2/Low Level	Critical	Active, Unacknowledged	Low Level	24.308683	
02/03/2020 13:45:01	Sine/Sine1/High Temp	High	Active, Unacknowledged	High Temp	80.324554	
02/03/2020 13:44:12	Speed/High Speed	Diagnostic	Active, Unacknowledged	High Speed	115	
01/29/2020 11:03:24	Turbine Number 300 located at Fresno	High	Active, Unacknowledged	High Wind Speed		<input checked="" type="checkbox"/>
01/29/2020 11:03:24	Turbine Number 200 located at Livermore, CA	High	Active, Unacknowledged	High Wind Speed		<input checked="" type="checkbox"/>
01/29/2020 11:03:24	Turbine Number 150 located at Folsom, CA	High	Active, Unacknowledged	High Wind Speed		<input checked="" type="checkbox"/>
01/29/2020 11:03:24	Turbine Number 100 located at Sacramento, CA	High	Active, Unacknowledged	High Wind Speed		<input checked="" type="checkbox"/>

Filtering on Source Path and Display Path

You can filter the alarm list in the table to be a shorter list using the Filter properties instead of scrolling through every single alarm in your system. In the Property Editor of the Designer, there is a group called **Filters** that you can configure to focus on only those alarms you want to see. You can filter on **State**, **Source Path**, **Display Path**, and **Tag Provider**. The most helpful properties to an operator are Source Path and Display Path.

Source Path

The **Source** is the actual **Tag path** which means you can also use Tag folders in Ignition to filter for specific alarms. For example, you may want to filter for all Turbine alarms in the Turbines folder. You can enter ***Turbine*** (using * as a wildcard) to look for all alarms with the Turbine Tag Path. The **Source** and **Display** properties allow you to restrict the results of the query to one or more paths. Multiple paths may be specified with a comma. Additionally, these properties all use the asterisk (*) as wildcard character to denote any number of leading or trailing characters, depending on placement as shown in the image below.

You can see in this example, all the active alarms have 'Turbine' in the **Source Path**.

4 ACTIVE						0 SHELVED
FILTERS (7): Active, Unacknowledged Active, Acknowledged Clear, Unacknowledged Priority: Low Priority: Medium Remove All						
4 results within filters						
Active Time	Priority	State	Source	Name		
01/29/2020 11:03:24	High	Active, Unackn...	prov:default:/tag:Turbines/Turbine 200/Wind Speed:/alm:High Wind Speed	High Wind Speed		
01/29/2020 11:03:24	High	Active, Unackn...	prov:default:/tag:Turbines/Turbine 300/Wind Speed:/alm:High Wind Speed	High Wind Speed		
01/29/2020 11:03:24	High	Active, Unackn...	prov:default:/tag:Turbines/Turbine 150/Wind Speed:/alm:High Wind Speed	High Wind Speed		
01/29/2020 11:03:24	High	Active, Unackn...	prov:default:/tag:Turbines/Turbine 100/Wind Speed:/alm:High Wind Speed	High Wind Speed		

Perspective Property Editor

filters (2)

- active (5)**
 - text :
 - states (4)**
 - activeUnacked : true
 - activeAcked : true
 - clearUnacked : true
 - clearAcked : false
 - priorities (5)**
 - diagnostic : false
 - low : true
 - medium : true
 - high : true
 - critical : true
 - conditions (3)**
 - source : *Turbine***
 - displayPath :
 - path :

Source Path Examples

Example Filter	Result
prov:tagProvider:/tag:Inputs/PS_1:/alm:MyAlarm	Retrieve alarm information from the alarm at precisely the specified path: prov:tagProvider:/tag:Inputs/PS_1:/alm:MyAlarm
*PS_1:/alm:MyAlarm	Retrieves alarm information from any path that ends with PS_1:/alm:MyAlarm. Thus the following paths would be returned: prov:tagProvider:/tag:Inputs/PS_1:/alm:MyAlarm prov:tagProvider:/tag:anotherFolder/different_Path/PS_1:/alm:MyAlarm
prov:tagProvider:/tag:PS_*	Retrieves alarm information from any source path starting with prov:tagProvider:/tag:PS_ prov:tagProvider:/tag:PS_1/MyAlarm

	<pre>:/alm:MyAlarm prov:tagProvider:/tag:PS_2/MyAlarm:/alm:MyAlarm</pre>
MyAlarm	Retrieves any alarm information that has MyAlarm somewhere in the path.

Display Path

The **Display Path** is the Tag path that leads to the **Name** of the alarm which can be customized when you configure your alarm. In this example, ‘Low Level’ and ‘High Level’ were the names that were setup in the [Alarm Configuration](#), and the Display Path is set to the Name you want the operator to see. The Alarm Status Table below is filtered by **Display Path** with a value of ‘Low Level’ and ‘High Level.’ So in this example, you see results for both **Low Level** and **High Level** alarms in the **Active, Unacknowledged** state.

The screenshot shows the Perspective interface with the Alarm Status Table on the left and the Perspective Property Editor on the right.

Alarm Status Table:

- Header:** Active Time, Display Path, Priority, State, Name.
- Data:**

Active Time	Display Path	Priority	State	Name
01/29/2020 10:10:32	Sine/Sine2/Low Level	Critical	Active, Unacknowledged	Low Level
01/29/2020 10:07:31	Sine/Sine1/High Temp	High	Active, Unacknowledged	High Temp
01/09/2020 11:38:43	High Temp/High Temp	Critical	Active, Unacknowledged	High Temp
- Filters:** Active, Unacknowledged, Active, Acknowledged, Priority: Low, Priority: Medium.
- Perspective Property Editor:**
 - filters (2):**
 - active (5):** activeUnacked: true (checked), activeAcked: true (checked), clearUnacked: false, clearAcked: false.
 - states (4):** displayPath: *Low Level*, *High Temp
 - conditions (3):** source: displayPath: *Low Level*, *High Temp

To learn more about alarm table properties, refer to the [Perspective - Alarm Status Table](#) page.

Related Topics ...

- [Perspective - Alarm Status Table](#)
- [Configuring Alarms](#)

Perspective Alarm Status - Filtering



Configure Alarms

Alarms must be set up on Tags for them to show up in the Alarm Status Table.

The Alarm Status Table has many built-in properties that allow you to filter on various parts of an alarm. What's super nice about the Perspective Alarm Status Table is everything is right at your fingertips for filtering on alarm events. The table provides a host of built-in filtering options that are immediately available in a [Perspective Session](#) and easy to modify to help you get started.

You can choose to filter on alarm states and priorities by clicking on the **Filter** icon. In addition, there is a **Search** icon that lets you further optimize your search results. The **Configuration Settings** icon allows operators to view alarm data that is most important to them, as well as organize the alarm data any way they choose.

Filtering in a Perspective Session

In a Perspective Session, all the filtering tools the operators will need are built in to the Perspective Alarm Status Table. When first using the table in a session, an operator could be using the default alarm status properties, or your project designer or system administrator may have preconfigured some alarm status properties specific to your project. Either way, an operator can easily choose and filter on specific [alarm](#) data and how they want it displayed.

In a session, you can easily change the filtering options on the table by clicking on the **Filter** icon and adding or removing a filter option from the dropdown by checking or unchecking a filter option. Notice that there is a filter bar at the top of the table that also displays all the filter options that are currently set. You can remove any of these filter options by clicking the 'X' on the right side of the option, but to add a filter, use the **Filter** icon.

Notice how the Filter Bar displays the row color on the tab for Active, Unacknowledged , Active, Acknowledged , and Clear, Unacknowledged states.

The screenshot shows the Perspective Alarm Status Table interface. At the top, there are filters for 'ACTIVE' (12 ACTIVE) and 'SHELFED' (0 SHELFED). On the right, there is a 'Select Filters' panel with tabs for 'STATE' and 'PRIORITY'. The 'STATE' tab is active, showing filters for 'Active, Unacknowledged' (checked), 'Active, Acknowledged' (checked), and 'Clear, Unacknowledged' (checked). The 'PRIORITY' tab shows filters for 'Low', 'Medium', 'High', and 'Critical' (all checked). The main table lists 29 results with columns for 'Time', 'Priority', 'State', 'Source', and 'Name'. A red box highlights the filter bar at the top of the table.

Time	Priority	State	Source	Name
01/29/2020 12:52:23	High	Active, Unacknowledged	prov:default:/tag:Sine/Sine1:/alm:High Temp	High Temp
01/29/2020 12:48:53	Critical	Active, Unacknowledged	prov:default:/tag:Sine/Sine2:/alm:Low Level	Low Level
01/29/2020 12:45:12	High	Cleared, Unacknowledged	prov:default:/tag:Sine/Sine1:/alm:High Temp	High Temp
01/29/2020 12:40:32	Critical	Cleared, Unacknowledged	prov:default:/tag:Sine/Sine2:/alm:Low Level	Low Level
01/29/2020 12:38:03	High	Cleared, Unacknowledged	prov:default:/tag:Sine/Sine1:/alm:High Temp	High Temp
01/29/2020 12:32:12	Critical	Cleared, Unacknowledged	prov:default:/tag:Sine/Sine2:/alm:Low Level	Low Level
01/29/2020 12:30:52	High	Cleared, Unacknowledged	prov:default:/tag:Sine/Sine1:/alm:High Temp	High Temp
01/29/2020 12:23:53	Critical	Cleared, Unacknowledged	prov:default:/tag:Sine/Sine2:/alm:Low Level	Low Level
01/29/2020 12:23:43	High	Cleared, Unacknowledged	prov:default:/tag:Sine/Sine1:/alm:High Temp	High Temp
01/29/2020 12:15:32	Critical	Cleared, Unacknowledged	prov:default:/tag:Sine/Sine2:/alm:Low Level	Low Level

Using the Search Bar

On this page ...

- [Filtering in a Perspective Session](#)
 - [Using the Search Bar](#)
 - [Viewing and Sorting on Alarm Data](#)
 - [Sorting Alarm Data](#)

Some operators may want to do some more targeted filtering like only seeing the 'Active, Unacknowledged' alarms for a specific type of alarm. This is easily accomplished by deleting all the alarm state properties except for '**Active, Unacknowledged**' and entering a specific keyword or string in the search bar to find specific alarms, events, and conditions. In the image below, the search criteria was for '**High Speed.**' It was found under three column headers: Display Path, Source Path, and Name.

The screenshot shows a table titled '11 ACTIVE' with a search bar containing 'High Temp'. Below the search bar are three filters: 'Active, Unacknowledged', 'Priority: High', and 'Priority: Critical'. The table has columns: Active Time, Display Path, Priority, State, Source, Name, and Event Value. Two rows are listed:

Active Time	Display Path	Priority	State	Source	Name	Event Value
01/09/2020 11:38:43	High Temp/High Temp	Critical	Active, Unack...	prov:default/tag:High Temp/alm:High Temp	High Temp	98
01/31/2020 08:56:53	Sine/Sine1/High Temp	High	Active, Unack...	prov:default/tag:Sine/Sine1/alm:High Temp	High Temp	80.6042

Viewing and Sorting on Alarm Data

In a Perspective Session, operators can choose to display or hide alarm data in the table by checking or unchecking any of the **Configuration Settings**, or right clicking in the header row of the table. You'll find that it's super easy to filter and display alarm event data using the configuration options provided in the column header of the Alarm Status Table. You can do a multi-column sort by holding down **Ctrl + Shift** and clicking on the column headers you want to sort by.

If you need to align the columns, simply put your cursor to the left of the column header and drag left or right. There is a '**strictWidth**' property for each header column that can be configured in the Designer to strictly enforce the column width.

The screenshot shows a table titled '11 ACTIVE' with a search bar and a configuration sidebar. The configuration sidebar is open and shows various checkboxes for sorting and filtering options. The table has columns: Active Time, Display Path, Priority, State, Name, and Event Id. Ten rows are listed:

Active Time	Display Path	Priority	State	Name	Event Id
01/29/2020 17:10:22	Sine/Sine1/High Temp	High	Active, Unack...	High Temp	6aa3730c-dfb8-4532
01/29/2020 14:25:37	Speed/High Speed	Critical	Active, Unack...	High Speed	91325024-569b-49a
01/29/2020 11:03:24	Turbine Number 300 located at Fresno	High	Active, Unack...	High Wind Speed	225ccb57-3587-4af1
01/29/2020 11:03:24	Turbine Number 200 located at Livermore, CA	High	Active, Unack...	High Wind Speed	99d1cb7d-b4a7-432
01/29/2020 11:03:24	Turbine Number 150 located at Folsom, CA	High	Active, Unack...	High Wind Speed	7bed6991-d507-4bf7
01/29/2020 11:03:24	Turbine Number 100 located at Folsom, CA	High	Active, Unack...	High Wind Speed	d41206bc-4776-421
01/29/2020 10:50:55	WritableDatabase/WriteableInteger1/Low Tank Level	Critical	Active, Unack...	Low Tank Level	a78b7a41-ef54-4e65
01/27/2020 08:47:48	Tank 100	High	Active, Unack...	High SP	2ae79275-e4b3-465
01/09/2020 11:39:18	Tank Level 2/Low SP2	Critical	Active, Unack...	Low SP2	ea2bd2f6-acbe-4ba4

Sorting Alarm Data

Once your alarm is filtered, operators can also sort table columns in ascending or descending order by simply clicking the up or down arrows next to each column header.

Sorting on alarm State and Priority in the Alarm Status Table, by default, sorts in descending order. All the other columns the sort order is alphabetical. Sort order for **State** and **Priority** are as follows:

- **Alarm State** - ActiveUnacknowledged, ActiveAcknowledged, ClearUnacknowledged, and Clear Acknowledged.
- **Alarm Priority** - Critical, High, Medium, Low, and Diagnostic.

The image below was filtered by ActiveUnacknowledged state, with a priority Critical and High. Once your alarm data is filtered, you can also sort on the data in multiple columns by holding down the **Ctrl + Shift** and clicking on the column headers you want to multisort by.

The screenshot shows a table interface with the following details:

- Header:** 12 ACTIVE, 0 SHELVED.
- FILTERS:** Active, Unacknowledged, Priority: High, Priority: Critical. These filters are highlighted with a red box.
- Results:** 11 results within filters.
- Table Columns:**
 - Active Time
 - Display Path
 - Priority
 - State
 - Name
 - Event Value
- Data Rows:**

01/30/2020 14:47:12	Sine/Sine2/Low Level	Critical	Active, Unacknowledged	Low Level	23.920866
01/29/2020 14:25:37	Speed/High Speed	Critical	Active, Unacknowledged	High Speed	105
01/29/2020 10:50:55	Writeable/WriteableInteger1/Low Tank Level	Critical	Active, Unacknowledged	Low Tank Level	19
01/09/2020 11:39:18	Tank Level 2/Low SP2	Critical	Active, Unacknowledged	Low SP2	0.0
01/09/2020 11:38:43	High Temp/High Temp	Critical	Active, Unacknowledged	High Temp	98
01/30/2020 14:47:32	Sine/Sine1/High Temp	High	Active, Unacknowledged	High Temp	81.22193
01/29/2020 11:03:24	Turbine Number 300 located at Fresno	High	Active, Unacknowledged	High Wind Speed	<input checked="" type="checkbox"/>
- Pagination:** 25 rows, page 1 of 2.

Perspective's Alarm Status Table now supports multiple sort orders. The order is first determined by the sort order properties on the Alarm Status Table. There are two new sort order properties: 'activeSortOrder' and 'shelvedSortOrder.' Each column that you add to the sort order property will also need to have a sort defined under the columns property as either ascending or descending.

The screenshot shows the Perspective Property Editor with the following configuration:

- activeSortOrder [3]:**
 - 0 : state
 - 1 : priority
 - 2 : activeTime
- shelvedSortOrder [0]:** No items listed.
- columns {2}:**
 - active {21}:**
 - activeTime {4}:**
 - enabled : true
 - width : 150
 - strictWidth : false
 - sort : descending** (highlighted with a red box)

You also need to make sure the configuration columns you choose for your sort are displayed on your table. One of the most useful sort orders is by **State - Priority - Active Time**, as shown in the following image. Notice how each column is numbered according to your `activeSortOrder` property.

9 ACTIVE 3 SHELVED

FILTERS (7): Active, Unacknowledged Active, Acknowledged Clear, Unacknowledged Priority: Low Remove All

24 results within filters

<input type="checkbox"/> Active Time 3	Display Path	Priority 2	State 1	Name
■ 03/24/2020 10:47:56	Speed/High Speed	Critical	Active, Unacknowledged	High Speed
■ 03/02/2020 14:20:59	WritableDatabase/WriteableInteger1/Low T...	Critical	Active, Unacknowledged	Low Tank Level
■ 03/02/2020 14:20:59	Tank Level 2/Low SP2	Critical	Active, Unacknowledged	Low SP2
■ 03/24/2020 10:57:56	Tank 100	High	Active, Unacknowledged	Low SP
■ 03/16/2020 11:14:12	Tank 100	High	Active, Unacknowledged	High SP
■ 03/02/2020 14:20:46	Turbine Number 200 located at Liv...	High	Active, Unacknowledged	High Wind Speed
■ 03/24/2020 10:52:11	High Temp/High Temp	Medium	Active, Unacknowledged	High Temp

25 rows ▾ 1 2

Columns will first sort by anything defined in sortOrder and then fallback to the other sorts defined in the column configuration.

Related Topics ...

- [Perspective - Alarm Status Table](#)
- [Configuring Alarms](#)

Perspective Alarm Status - Acknowledgement

Acknowledging Alarms

The Alarm Status Table gives operators a ton of information at a glance about alarms that need attention. One of the most important things an operator is going to do in a [Perspective Session](#) is acknowledge alarms. Alarm acknowledgement is built in to the Alarm Status Table component. As soon as the operator selects and presses the Acknowledge button, the current state of the alarm will change, and the operator's credentials and the time the alarm was acknowledged will be recorded in the Alarm Status Table.

The Alarm Status Table component allows you to select an individual alarm, multiple alarms or use the 'Select All' checkbox in the header bar. To Acknowledge alarms, check all the alarms you want to acknowledge, and a footer will open at the bottom of the table to show the **Acknowledge** button. Press the **Acknowledge** button and the Alarm Status Table will record the time the alarm was acknowledged and the user that acknowledged the alarm in the database.

Note: To see this alarm information displayed in the table, the column headers may need to be configured as described later on this page.

On this page ...

- [Acknowledging Alarms](#)
 - [Configuring Table Headers](#)
 - [Acknowledgement Notes](#)
- [Security for Alarm Acknowledgement](#)

In the following example, one alarm was checked and Acknowledged.

Active Time	Display Path	Priority	State	Name	Event Value
02/03/2020 15:18:52	Sine/Sine2/Low Level	Critical	Active, Unacknowledged	Low Level	24.22091
02/03/2020 15:16:21	Speed/High Speed	Critical	Active, Unacknowledged	High Speed	115
01/29/2020 10:50:55	Writeable/WriteableInteger1/Lo...	Critical	Active, Unacknowledged	Low Tank Level	19
<input checked="" type="checkbox"/> 01/09/2020 11:39:18	Tank Level 2/Low SP2	Critical	Active, Unacknowledged	Low SP2	0.0

When you mouse-over an alarm row, you'll notice a popup modal () on the far right of the table that allows you to view the alarm details by simply clicking it. This brings up the **Alarms Details** window to view the alarm configuration properties.

12 ACTIVE 0 SHELVED

FILTERS (5): Active, Unacknowledged Active, Acknowledged Clear, Unacknowledged Priority: High Remove All

23 results within filters

Active Time	Display Path	Priority	State	Name	Event Value
02/03/2020 15:27:12	Sine/Sine2/Low Level	Critical	Active, Unack...	Low Level	24.72412
02/03/2020 15:16:21	Speed/High Speed	Critical	Active, Unack...	High Speed	115
01/29/2020 10:50:55	Writeable/WriteableInteger1/Low Tank Level	Critical	Active, Unack...	Low Tank Level	19
01/09/2020 11:39:18	Tank Level 2/Low SP2	Critical	Active, Unack...	Low SP2	0.0
01/09/2020 11:38:43	Hig				
02/03/2020 15:25:23	Sin				

25 rows ▾

Alarm Details X

Config Properties

Ack Mode	Manual
Ack Notes Reqd	<input type="checkbox"/>
Deadband	0.0
Deadband Mode	Absolute
Display Path	Writeable/WriteableInteger1/Low Tank Level
Enabled	<input checked="" type="checkbox"/>
Label	Low Tank Level
Mode	Below Setpoint
Name	Low Tank Level
Notes	
Priority	Critical
Shelving Allowed	<input checked="" type="checkbox"/>
Time Off Delay Secon...	0.0
Time On Delay Secon...	0.0

Configuring Table Headers

If you don't already have the '**Ack Time**' and '**Ack User**' displayed in your table header, select them from the **Configuration Settings**  dropdown. You can also add any other alarm data you want to display, or remove any alarm data that you don't need to display from the Alarm Status Table.

Notice that the time the alarm was acknowledged and who acknowledged the alarm is now shown in the table.

The screenshot shows a list of alarms with a context menu open over the second item. The menu includes options like Configuration, Hide This Column 'Display Path', Show All Hidden Columns, Ack Notes, Ack Pipeline, Ack Time (which is checked), Ack User (which is checked), Active Pipeline, Active Time, and Clear Pipeline.

Active Time	Display Path	Priority	State	Name	Ack Time	Ack User
02/04/2020 18:34:07	Writeable/WriteableInteger1/Low Ta...	Configuration		Low Tank Level	02/12/2020 11:56:47	admin
02/12/2020 11:33:11	Speed/High Speed	Hide This Column 'Display Path'		High Speed	02/12/2020 11:33:52	admin
02/06/2020 09:03:30	Speed/High Speed	COLUMNS		High Speed		
02/04/2020 18:33:53	Turbine Number 300 located at F...	Show All Hidden Columns		High Wind Sp...		
02/04/2020 18:33:53	Turbine Number 150 located at F...	Ack Notes		High Wind Sp...		
02/04/2020 18:33:53	Turbine Number 100 located at F...	Ack Pipeline		High Wind Sp...		
02/04/2020 18:33:52	Turbine Number 200 located at Live...	<input checked="" type="checkbox"/> Ack Time		High Wind Sp...		
		<input checked="" type="checkbox"/> Ack User				
		<input type="checkbox"/> Active Pipeline				
		<input checked="" type="checkbox"/> Active Time				
		<input type="checkbox"/> Clear Pipeline				

Acknowledgement Notes

If any of the selected alarms require **Acknowledgement Notes**, a small window will appear when the Acknowledgement button is pressed. The operator will be required to add notes, otherwise the alarm cannot be acknowledged. Enter your notes and press **Ack Alarm(s)**. If the operator wants to cancel the Acknowledgement Notes and close the window, click on the **Cancel** button to close the Acknowledgement notes window.

Note: Acknowledgement Notes are set up in the alarm's configuration settings. To set up Acknowledgement Notes, go to your alarm configuration settings and set '**Ack Notes Required**' to '**true**'. For more information, refer to [Configuring Alarms](#).

The screenshot shows a table component with the following data:

Active Time	Display Path	Priority	State	Name	Ack Time	Ack User
02/03/2020 16:54:48	Speed/High Speed	Critical	Active, Unack...	High Speed		
02/03/2020 16:51:22	Sine/Sine1/H...	Acknowledgement				
02/03/2020 16:50:32	Sine/Sine2/L...	Equipment was recalibrated and tested.				
02/03/2020 16:44:11	Sine/Sine1/H...					

Details for the selected row (02/03/2020 16:54:48, Speed/High Speed, Critical, Active, Unack..., High Speed):

Acknowledgement

Equipment was ~~recalibrated~~ and tested.

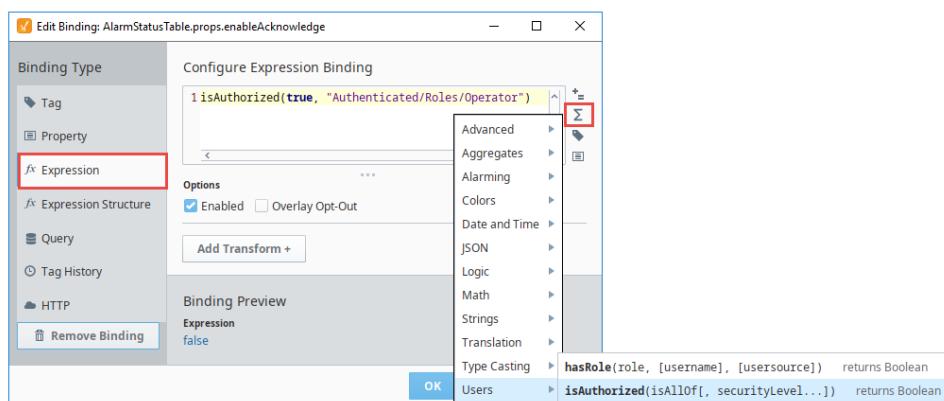
Buttons at the bottom right: **Ack Alarm(s)** (highlighted with a red box) and **Cancel**.

Security for Alarm Acknowledgement

You can restrict specific users or roles from Acknowledging alarms by setting the **enableAcknowledge** property in the Property Editor to 'false.' This hides the Acknowledge button on the Alarm Status Table for those users who do not have permission. You can setup permissions for any **role**, user and **user source** in your system.

For example, if you only want those users with the Operator role to acknowledge alarms, the correct permission must be assigned.

1. Select the Alarm Status Table component, and click the **enableAcknowledge** binding icon to open the Property Binding window.
2. Under Property Binding Type, select **Expression**.
3. Click the **Function** icon, scroll down to **Users**, and select 'isAuthorized.' This enters the function name.
4. Edit the expression to read: **isAuthorized(true, "Authenticated/Roles/Operator")**
5. Click **OK**.



If you currently have the 'Operator' role, you'll notice in the Property Editor of the Designer that the **enableAcknowledge** property is set to 'true,' and for other roles, it will be set to 'false.'

Related Topics ...

- [Configuring Alarms](#)

Perspective Alarm Status - Shelving

The capability to shelf alarms is another important feature of the [Perspective Alarm Status Table](#). Shelves alarms allows you to temporarily silence an alarm for a fixed period of time. This feature is extremely handy when an alarm is already active and you want to temporarily suppress the alarm while you're working on the issue. The [Perspective Alarm Status Table](#) component will not send any notifications while the alarm is shelved, and will be temporarily dropped from the Alarm Status list so operators don't get confused and think it's active. When the shelved time period is up and if the alarm is still active, it will return into the Alarm Status list.

The top of the Alarm Status Table displays the number of shelved alarms. To view all the shelved alarms, click on this **Shelf** button.

Note: Active alarms must be present in the Alarm Status Table before you can shelf an alarm.

How to Shelf an Alarm

To shelf an alarm, select one alarm or multiple alarms, and a footer will open at the bottom of the table to show the **Shelf** button. Click the **Shelf** button and a dropdown will automatically open so you can set a duration to silence the selected alarms. You can set a duration from 5 minutes to 4 hours to shelf selected alarms. Choose the duration and your alarm will immediately be shelved for your selected alarms.

The example below shows two alarms were shelved for one hour.

The screenshot shows the Perspective Alarm Status Table interface. At the top, there are buttons for '11 ACTIVE' and '0 SHELVED'. Below this is a search bar and filter options: 'FILTERS (5): Active, Unacknowledged', 'Active, Acknowledged', 'Clear, Unacknowledged', and 'Priority'. A dropdown menu titled 'Shelf the selected alarms for:' is open, listing durations: '5 minutes', '15 minutes', '30 minutes', '1 hour' (which is highlighted in red), '2 hours', and '4 hours'. At the bottom right of the table area are buttons for 'Shelf' (highlighted with a red box) and 'Acknowledge'.

Viewing Shelved Alarms

To see details for the shelved alarms click **Shelved** at the top of the Alarm Status Table. The Shelved tab will display the date/time for when the alarm expires. **Shelved By** and **Source Path** are also displayed for each alarm event. To return to active alarm, click on **Active**.

The screenshot shows the Perspective Alarm Status Table with the 'Shelved' tab selected. At the top, there are buttons for '9 ACTIVE' and '3 SHELVED'. The table has columns: 'Expires', 'Shelved By', and 'Source Path'. The data shows three entries:

Expires	Shelved By	Source Path
02/04/2020 16:23:02	admin	prov:default:/tag:Speed:/alm:High Speed
02/04/2020 14:37:29	admin	prov:default:/tag:Sine/Sine2:/alm:Low Level
02/04/2020 14:37:29	admin	prov:default:/tag:Sine/Sine1:/alm:High Temp

Unshelve Alarms

On this page ...

- How to Shelf an Alarm
 - Viewing Shelved Alarms
 - Unshelve Alarms
- Configuring Custom Shelving Duration

To unshelve alarms, click **Shelved** at the top of the Alarm Status Table. Select one or multiple alarms, and the footer at the bottom of the table will open with the Unshelved Selected Alarms button. Press the **Unshelved Selected Alarms** and the alarm will return back to the Alarm Status list.

9 ACTIVE 3 SHELVED		
Expires	Shelved By	Source Path
02/04/2020 16:23:02	admin	prov:default:/tag:Speed:/alm:High Speed
<input checked="" type="checkbox"/> 02/04/2020 14:37:29	admin	prov:default:/tag:Sine/Sine2:/alm:Low Level
<input checked="" type="checkbox"/> 02/04/2020 14:37:29	admin	prov:default:/tag:Sine/Sine1:/alm:High Temp

Unshelve Selected Alarms (2)

25 rows 1

Mouse over an alarm event and you'll notice a unshelve icon () on the right side of the alarm. You can delete a single alarm event by clicking the **Unshelve icon** ()

9 ACTIVE 3 SHELVED		
Expires	Shelved By	Source Path
02/04/2020 16:23:02	admin	prov:default:/tag:Speed:/alm:High Speed
<input checked="" type="checkbox"/> 02/04/2020 14:37:29	admin	prov:default:/tag:Sine/Sine2:/alm:Low Level 
02/04/2020 14:37:29	admin	prov:default:/tag:Sine/Sine1:/alm:High Temp

Unshelve Selected Alarms (1)

25 rows 1

After the amount of time expires on a shelved alarm it will be evaluated, and if it is still active, it will automatically return to the Alarm Status list. If the alarm transitions to a cleared state during the time shelved period, the alarm will show up as '**Cleared,Unacknowledged**' in the Alarm Status list instead of '**Active, Unacknowledged**' as shown in the blue rows in the example below.

The screenshot shows a table component with the following columns: Active Time, Display Path, Priority, State, Name, and Event Value. There are 22 results within filters. One row is highlighted in red, indicating it is selected.

Active Time	Display Path	Priority	State	Name	Event Value
02/04/2020 15:04:23	Sine/Sine1/High Temp	High	Active, Acknowledged	High Temp	81.74199
02/04/2020 15:03:52	Sine/Sine2/Low Level	Critical	Active, Unacknowledged	Low Level	24.821516
02/04/2020 14:57:12	Sine/Sine1/High Temp	High	Cleared, Unacknowledged	High Temp	79.85367
02/04/2020 14:55:32	Sine/Sine2/Low Level	Critical	Cleared, Unacknowledged	Low Level	25.845083
02/04/2020 14:50:03	Sine/Sine1/High Temp	High	Cleared, Unacknowledged	High Temp	78.85678
02/04/2020 14:47:11	Sine/Sine2/Low Level	Critical	Cleared, Unacknowledged	Low Level	25.188965
02/04/2020 14:42:52	Sine/Sine1/High Temp	High	Cleared, Unacknowledged	High Temp	79.74181

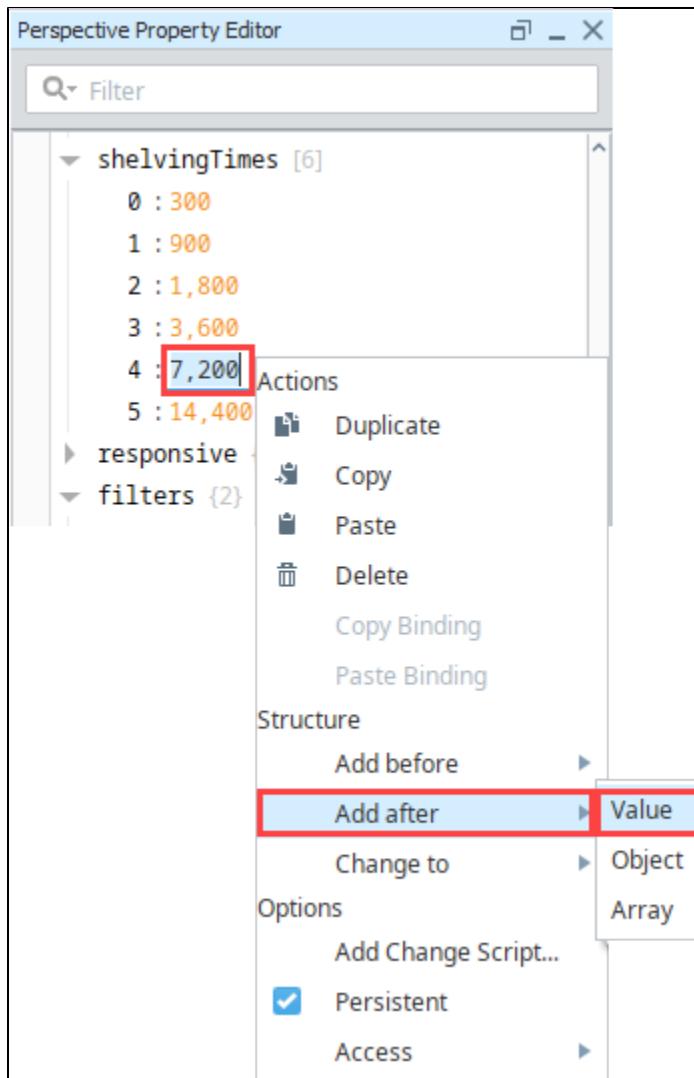
Configuring Custom Shelving Duration

The shelving duration values on the Alarm Status Table component can be customized in the Designer. By default, the Alarm Status Table has several preset shelving duration times, but you can add or remove times based on your requirements. One thing to notice when modifying the shelving times is that there is no option to set a shelving time indefinitely. Shelving times are meant to temporarily suppress the alarm while you're working on the issue. They are not meant to be long term.

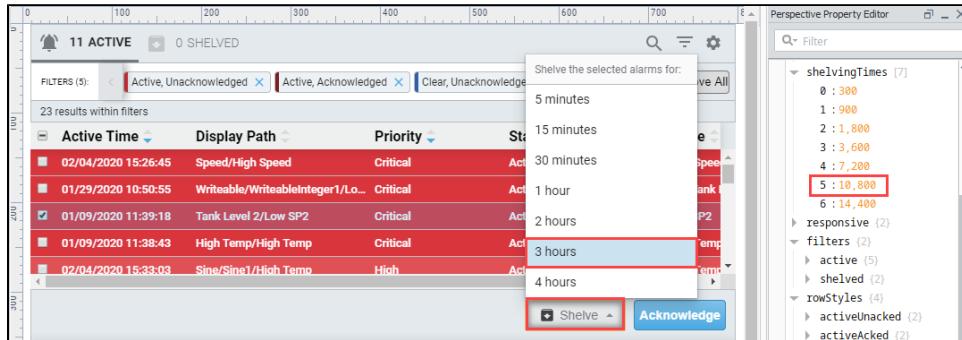
In the Property Editor, there is a property called **shelvingTimes**. This is where you can add, remove, or change the shelving times for alarms. Shelving time is calculated in seconds.

In the following example, let's add **3 hours** to the list of shelving times.

- Select the Alarm Status Table component. Under the **shelvingTimes** property, insert a **Value** of **10,800** seconds (3 hours).



- To verify your new time was entered correctly, go to **Preview Mode**, select the alarms you want to shelve, and press the **Shelf** button. A footer will open at the bottom of the table and you should see your new shelve time in the dropdown list (i.e., 3 hours).
- Click the shelving time to temporarily silence your alarms.



Related Topics ...

- Configuring Alarms

Perspective Alarm Status - Row Styles

The Perspective Alarm Status Table comes with a default set of row colors associated with each of the alarm states. Each alarm state has a preset row color, and each of its priorities has a variation of that same color. By default, when you drag a Alarm Status Table into the Designer for the first time, rows are red for Active Unacknowledged alarms events and blue for Cleared Unacknowledged alarm events, as shown in the image below.

On this page ...

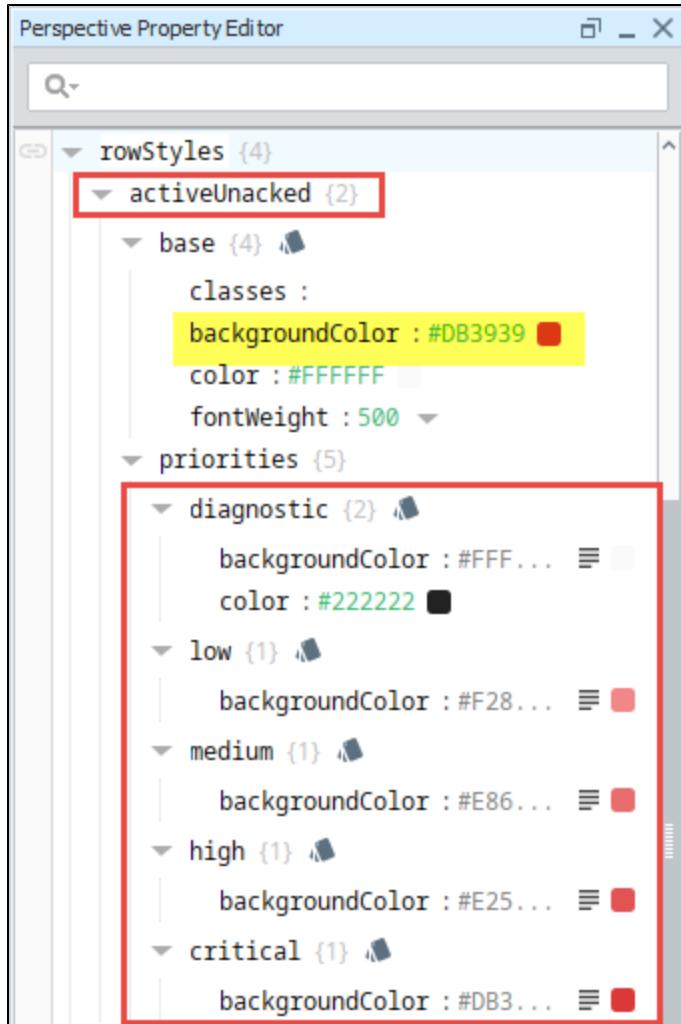
- Customizing Alarm Row Styles
 - Example 1 - Modifying the `rowStyle.backgroundColor` Property for an Alarm State
 - Example 2 - Adding a Style Class to a `rowStyle` to Make a Critical Priority Alarm Blink

Active Time	Display Path	Priority	State	Name
02/04/2020 18:33:53	F Temp/Alarm	Low	Active, Unacknowledged	Alarm
02/05/2020 09:57:57	High Temp/High Temp	Medium	Active, Unacknowledged	High Temp
02/04/2020 18:33:53	Turbine Number 300 located at Fresno	High	Active, Unacknowledged	High Wind Speed
02/04/2020 18:33:53	Turbine Number 150 located at Folsom, CA	High	Active, Unacknowledged	High Wind Speed
02/04/2020 18:34:07	Writeable/WriteableInteger1/Low Tank Level	Critical	Active, Unacknowledged	Low Tank Level
02/05/2020 10:44:09	Speed/High Speed	Critical	Active, Unacknowledged	High Speed
02/04/2020 18:34:07	Tank Level 2/Low SP2	Critical	Active, Acknowledged	Low SP2
02/05/2020 10:39:01	Speed/High Speed	Critical	Cleared, Unacknowledged	High Speed

Customizing Alarm Row Styles

In the Property Editor of the Designer you can modify an existing row style, add more styles, or delete a style. You can customize the row styles for any of the alarm states and their priorities. You can even create a [style class](#) to set a custom appearance to make a row for a particular state and priority blink.

In the Designer, right click on the [Alarm Status Table](#) component, then go to the Property Editor. Expand the `rowStyles` property and also expand each of the priorities for the `activeUnacked` state. You'll notice how each priority has a different shade of red for the `backgroundColor` property, except for diagnostic which is set to white with black text.



Each alarm state has the same default rowStyle properties, but their **backgroundColor** values are different. Here is a list of default properties for each rowStyle alarm state:

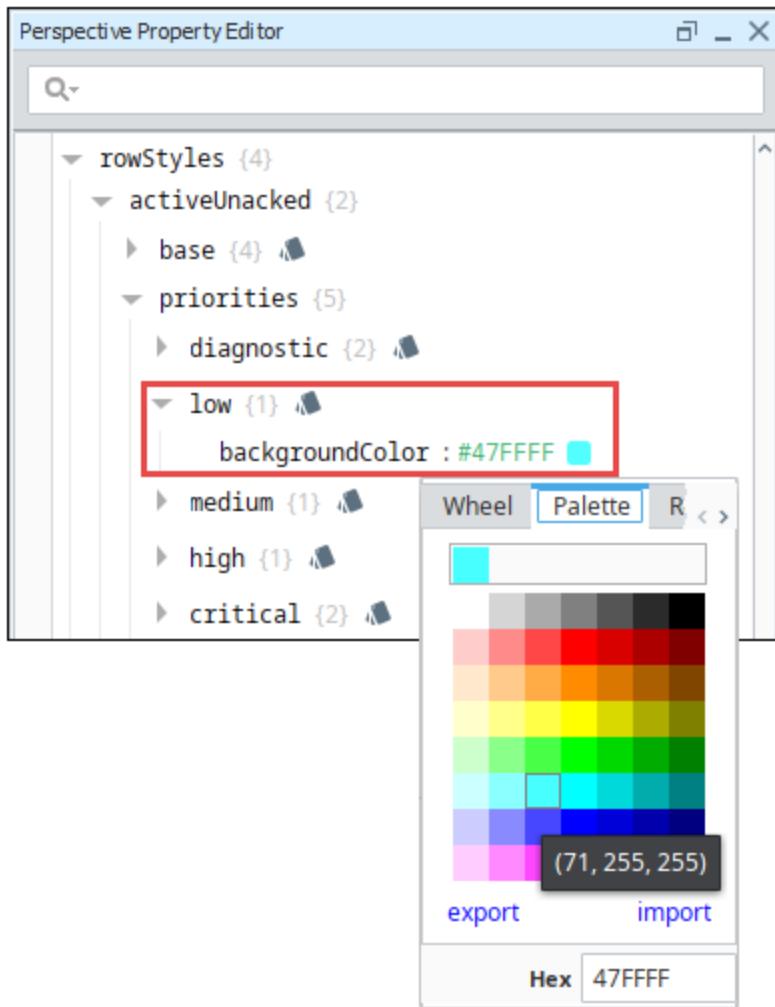
- **base** - Opens a [Style Options](#) menu to configure [style elements](#) for Text, Background, Margin and Padding, Border, Shape and Misc.
- **classes** - Allows to you set a pre-defined [style class](#).
- **backgroundColor** - Background color for each of the priorities (i.e., diagnostic, low, medium, high, critical) for an alarm state.
- **color** - Color of the Text.
- **fontWeight** - Font weight of the text.

You can modify rowStyles properties in multiple places: using the base (i.e., [Style Options](#) menu) or [property data types](#) in the Property Editor. When you modify, add, or remove a style property from either Style Options menu or the Property Editor, the change will immediately be visible in both locations, as well as in the table.

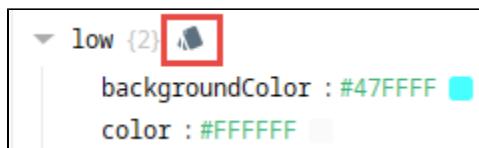
Example 1 - Modifying the rowStyle backgroundColor Property for an Alarm State

Let's change the backgroundColor property for the **Low** priority in the **activeUnacked** state. There are two different ways of changing style properties. Both are documented here.

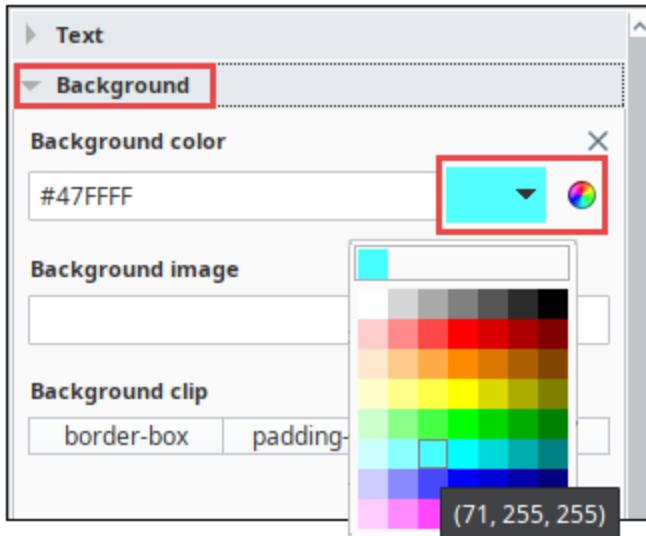
1. In the Designer, select the Alarm Status Table component.
2. Go to the Property Editor, expand **rowStyles** property and the **activeUnacked** state. Also, expand **priorities** and **low** property.
3. The easiest way to change the property is to click on the **backgroundColor** icon. This brings up the [Color Selector](#) window. Select one of the four tabs (Wheel, Palette, RGV and HSL) and choose a color.



4. Another way to change the backgroundColor in rowStyles is from the [Style Options](#) menu.
- Simply click on the Modify Style icon next to **low**. This opens the **Style Options** menu.



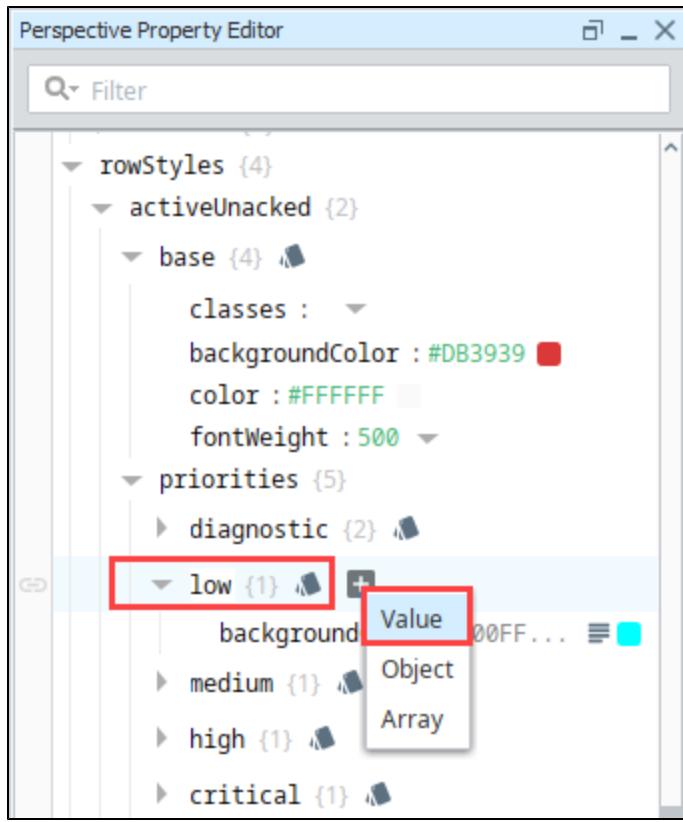
- Click the **Background** tab to open a list of background elements.
- Click on the dropdown color palette or color wheel. Choose a color.



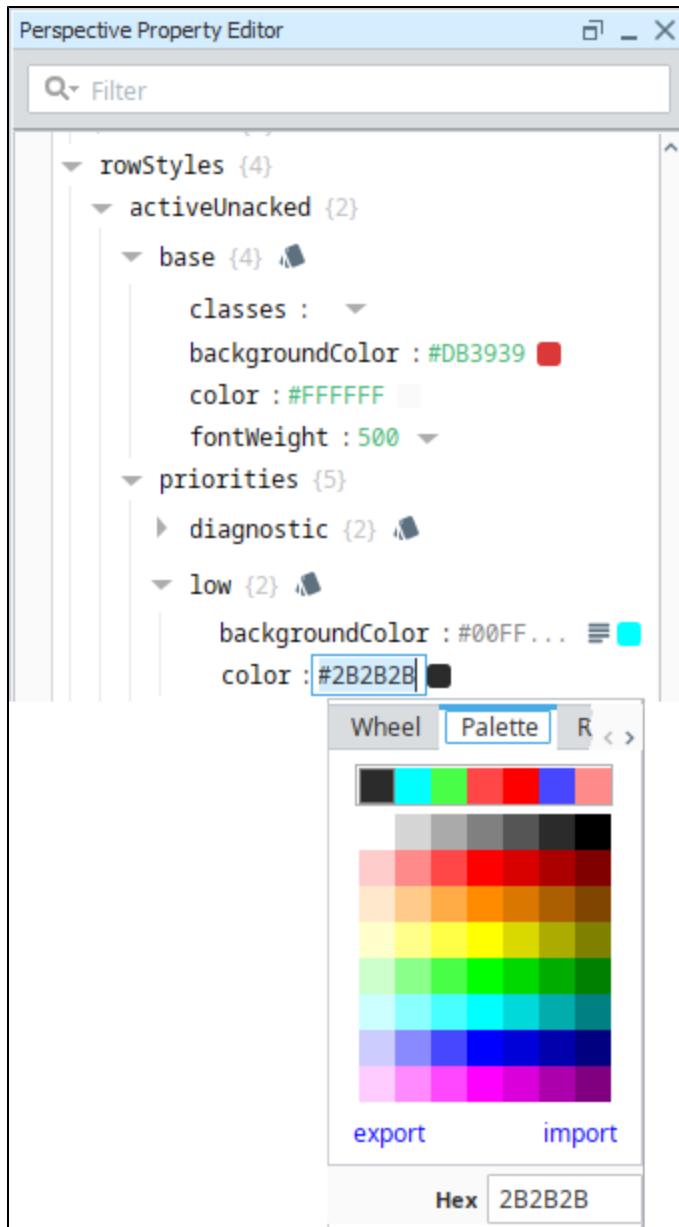
5. The **backgroundColor** for the 'Low' priority is now changed, but it's hard to read. Let's change to the color of the font to black.

12 ACTIVE 0 SHELFED					
FILTERS (7): Active, Unacknowledged Active, Acknowledged Clear, Unacknowledged Priority: Low > Remove All					
22 results within filters					
Active Time	Display Path	Priority	State	Name	
02/07/2020 16:19:19	Sine/Sine1/High Temp	Low	Active, Unacknowledged	High Temp	
02/07/2020 16:17:09	Sine/Sine2/Low Level	Critical	Active, Unacknowledged	Low Level	
02/07/2020 16:12:09	Sine/Sine1/High Temp	Low	Cleared, Unacknowledged	High Temp	
02/07/2020 16:08:49	Sine/Sine2/Low Level	Critical	Cleared, Unacknowledged	Low Level	
02/07/2020 16:04:59	Sine/Sine1/High Temp	Low	Cleared, Unacknowledged	High Temp	

6. Expand '**priorities**'. Mouse over the 'low' priority, and click the plus icon.



7. For the 'key', enter 'color' property.
8. Enter a value by clicking the gray square to open the popup color wheel or color palette, then chose the color 'black'.



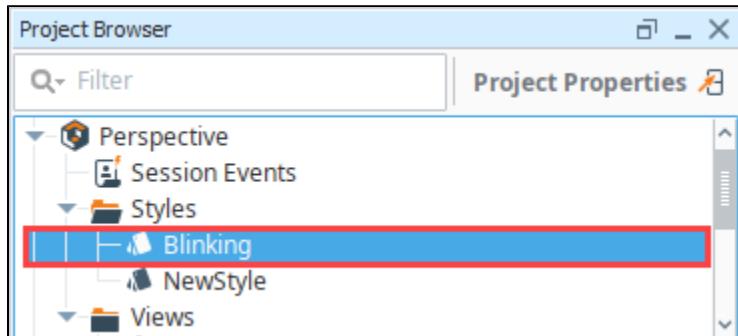
9. The text is now black and readable with the turquoise row style color.

	Active Time	Display Path	Priority	State	Name
■	02/07/2020 16:42:09	Sine/Sine2/Low Level	Critical	Active, Unacknowledged	Low Level
■	02/07/2020 16:40:49	Sine/Sine1/High Temp	Low	Active, Unacknowledged	High Temp
■	02/07/2020 16:33:49	Sine/Sine2/Low Level	Critical	Cleared, Unacknowledged	Low Level
■	02/07/2020 16:33:39	Sine/Sine1/High Temp	Low	Cleared, Unacknowledged	High Temp
■	02/07/2020 16:26:29	Sine/Sine1/High Temp	Low	Cleared, Unacknowledged	High Temp

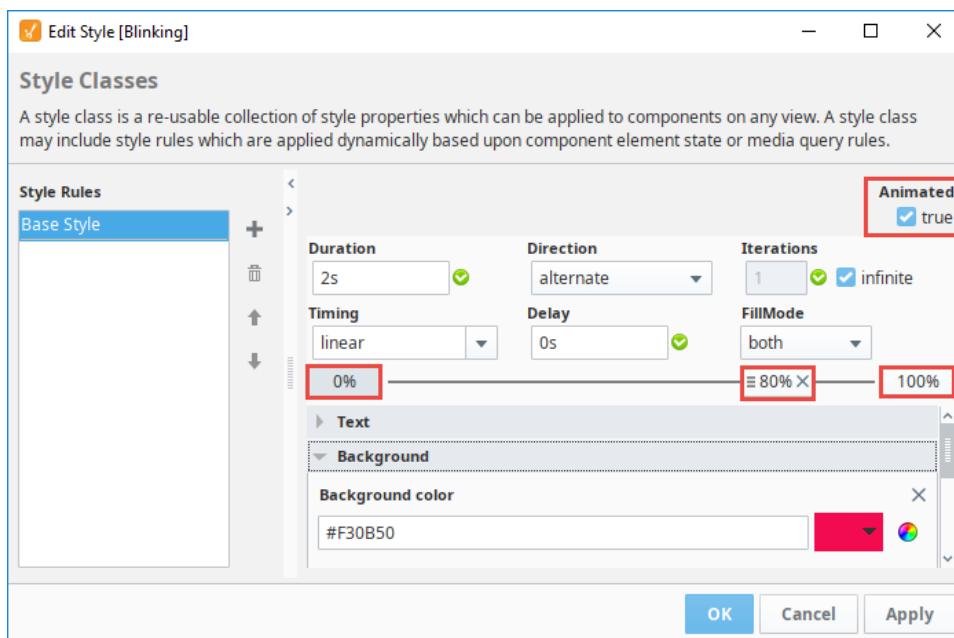
Example 2 - Adding a Style Class to a rowStyle to Make a Critical Priority Alarm Blink

In this example, let's make the activeUnacknowledged Critical property blink between two colors, red and yellow, so it captures the attention of the operator. To do this, we need to create a [Style Class](#).

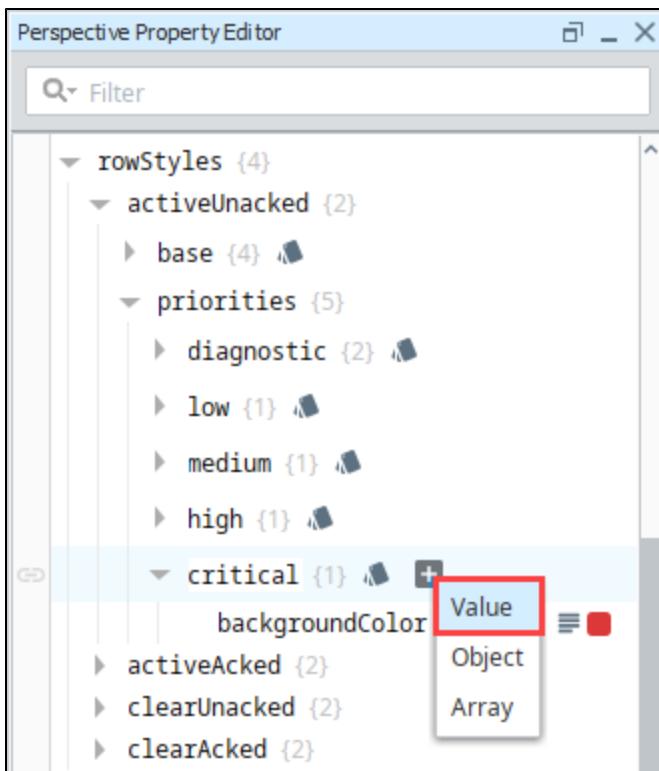
1. In the **Project Browser**, right click on the Styles folder, select **New Style**, and enter a name (i.e., Blinking). Click **OK**.



2. Set **Animated** to 'true' in the upper right corner.
3. Click on **0%** to set the style for the beginning of the animation.
4. Click on **Background** to see Background settings for Critical priority for the activeUnacked state and choose a color.
5. Next, click on **100%** to set the style for the end of the animation.
6. Click on **Background** to expand the Background settings, and choose a color from the either the color palette or color wheel (i.e., yellow). When choosing a second color, make sure it will be readable when it is blinking.
7. Lastly, let's add a stop so that the red rowStyle stays on for a longer period of time than yellow. Right click to add a stop somewhere along the linear line. Drag the stop to about **80%** because red is easier to read than the yellow.
8. Notice the other Animation properties on the Edit Style window. You can set Duration, Direction, number of Iterations, and more. For a detailed description of the Animated properties, refer to [Animated Style Classes](#).



9. Click **OK** to save your new style class.
10. Next, apply the style class to the property. In the Property Editor, expand **rowStyles** and the **activeUnacked** state. Then expand **priorities** and **critical**.
11. Add a '[value](#)' property data type under the Critical property:
 - a. Mouse over the '**critical**' property then click the gray plus sign to add a property.
 - b. Choose **value** from the property type popup menu.



- c. Click on **key** and change it to **classes** (case sensitive).
- d. Click on **value**. Select the **name of your new style class** from the dropdown (i.e., Blinking).

The image contains two side-by-side screenshots of the 'Perspective Property Editor'. The left screenshot shows the 'key : value' field highlighted with a red box. The right screenshot shows the 'Value' dropdown menu open, with 'Blinking' selected, also highlighted with a red box.

12. Now, all your **critical** alarms for **Active Unacknowledged** will blink red and yellow.

Active Time	Display Path	Priority	State	Name
02/04/2020 18:34:07	Writeable/WriteableInteger1/Low Tank Level	Critical	Active, Unacknowledged	Low Tank Level
02/05/2020 10:44:09	Speed/High Speed	Critical	Active, Unacknowledged	High Speed
02/05/2020 06:59:48	Tank 100	High	Active, Unacknowledged	High SP
02/05/2020 14:52:20	Sine/Sine1/High Temp	High	Active, Unacknowledged	High Temp
02/04/2020 18:33:53	Turbine Number 300 located at Fresno	High	Active, Unacknowledged	High Wind Speed

13. When you're finished creating your Style Class and have your rowStyles configured, don't forget to **Save** your project.

Related Topics ...

- [Color Selector Reference](#)
- [Style Classes](#)
- [Style Reference](#)
- [Alarm Event Properties Reference](#)

Perspective Alarm Journal Table - Common Tasks

The Perspective Alarm Journal Table has a number of configuration options that can be used to do things like filter on realtime and historical alarm data, and change how the component displays those alarms. This section describes some of the common tasks for the Alarm Journal Table.

Note: The [Alarm Journal](#) must first be set up with a valid database connection for the Alarm Journal Table to see alarm history from the database.

On this page ...

- [Alarm Journal Table - User Interaction](#)
- [Perspective Alarm Journal Table - Configuring Properties in the Designer](#)
- [Perspective Alarm Journal - Filtering](#)
- [Perspective Alarm Journal - Row Styles](#)

Alarm Journal Table - User Interaction

Before you get into the nuts and bolts of how to perform specific tasks outlined on this page, there are a couple things you should know about working with the [Alarm Journal Table](#). The [User Interaction](#) page introduces you to the look and feel of the Alarm Journal Table and how to use the basic functionality built in to the table.

Perspective Alarm Journal Table - Configuring Properties in the Designer

The project designer and [Ignition](#) administrator have the option to enable or disable properties and setup filtering options in the Property Editor of the [Designer](#) based on the project requirements and needs of the client users. This page describes which properties are enabled and which properties have some default filtering options preset. Learn how to configure your own [Alarm Journal](#) properties in the [Designer](#) on the [Configuring Properties in the Designer](#) page.

Perspective Alarm Journal - Filtering

The Perspective Alarm Journal Table component has many built in properties that allow you to filter on various parts of an alarm. Refer to the [Perspective Alarm Journal - Filtering](#) page to learn how to use the different built-in filtering options, the search bar to optimized your search results, and the date range feature to view alarm history in the journal table.

Perspective Alarm Journal - Row Styles

The Perspective Alarm Journal Table colors each row a specific color depending on what type of alarm event it is. It uses different styled rows to differentiate between alarms in different state and priorities. These [Row Styles](#) can be completely customized using whatever colors and fonts you want for different events. You can also setup new row styles based on other properties and even add a blinking style drawing an operator's attention to critical alarms!

[In This Section ...](#)

Perspective Alarm Journal - User Interaction

Getting Started with the Alarm Journal Table

If you're setting up the Alarm Journal Table for the first time, you probably will want to take a look at the table's properties in the [Property Editor](#). There a number of properties that are enabled by default such as the search toolbar, alarm details popup, and date range to name a few. You should become familiar with them in the event you would like to change them. Other properties have default options already selected for you, like alarm status, alarm priorities and row styles.

When you drag your first Alarm Journal Table component into the Designer workspace, the Alarm Journal Table will automatically populate if you have any alarms configured and an [Alarm Journal Profile](#) created. It will look something like the image below. You'll notice it resembles the same layout as the Alarm Status Table and shares the same Configuration settings and filter options. By default, the journal table will show you the last 8 hours of alarm journal data.

You can interface with the journal table in the Designer, in Preview Mode of the Designer, and in a [Perspective Session](#). The Alarm Journal Table properties are configured in the Property Editor of the Designer.

When setting up your Alarm Journal Table for the first time, you will have to toggle between both the Designer and Preview Modes to configure properties and organize how to display your alarm journal data, and how to size the data columns in the table. There is a column property you can set to strictly enforce the column width if you prefer.

The screenshot shows the Perspective Designer interface. On the left is the main workspace containing a single Alarm Journal Table component. The table displays 70 alarm events from the last 8 hours. The columns are Event Time, Event Id, Source, Event State, and Priority. The rows are color-coded by priority: Low (pink), Medium (light blue), High (yellow), and Critical (red). At the bottom of the table, there are filters for Active, Acknowledged, Cleared, Priority: Low, Priority: Medium, Priority: High, and Priority: Critical, along with a Remove All button. Below the table are buttons for 25 rows, page numbers 1, 2, 3, and a scroll bar. On the right is the Perspective Property Editor panel, which is expanded to show the 'PROPS' tab. This tab lists numerous properties for the Alarm Journal component, many of which are checked or set to true. A red box highlights the 'PROPS' tab and the entire list of properties. Some properties have dropdown menus or additional settings shown to the right of the main property value.

Anatomy of an Alarm Journal Table

The [Alarm Journal Table](#) may look overwhelming at first, but you will quickly get familiar with it. The following example shows the basic anatomy of an [Alarm Journal](#) table with event data populated in the rows. There are a host of [Configuration Settings](#) and [Filter Settings](#) that you can set to provide relevant information about your alarms events. Use can filter on Realtime or Historical alarms by clicking on the [Data Range](#) icon. You can also optimize your search results by clicking the [Search](#) icon and entering keywords to help further refine your results. When you hover over a row, a popup modal will appear. By clicking the [Popup Modal](#) icon you can see the [alarm](#)'s details. You can also set the number of [alarm](#) rows to display and scroll through the list of pages at the bottom of the journal table. Lastly, the [Alarm Journal](#) Table comes with a default set of row colors based on the [alarm](#) state and priority which are totally customizable.

To learn more about using the [Alarm Journal](#) functions and features, refer to the filtering and row styles pages in this section. The [Alarm Journal Table](#) component page describes all the journal properties and how to use them.

On this page ...

- [Getting Started with the Alarm Journal Table](#)
- [Anatomy of an Alarm Journal Table](#)

Date Range → 168 alarm events Last 8 hours

Search → Filter

Filters (7): Active, Acknowledged, Cleared, Priority: Low, Priority: Medium, Priority: High, Priority: Critical, Remove All

Config Settings → Settings

Popup Modal → More Options

Rows to Display → 25 rows

Pages → 1 2 3 4 5 6 7

Event Time	Name	Source	Event State	Priority
01/14/2020 12:16:32	High Speed	prov:default:/tag:Speed:/alm:High Speed	Active	Critical
01/14/2020 12:16:32	High Speed	prov:default:/tag:Speed:/alm:High Speed	Ack	Critical
01/14/2020 12:16:21	High Speed	prov:default:/tag:Speed:/alm:High Speed	Clear	Critical
01/14/2020 12:14:30	Low Level	prov:default:/tag:Sine/Sine2:/alm:Low Level	Active	Low
01/14/2020 12:14:30	Low Level	prov:default:/tag:Sine/Sine2:/alm:Low Level	Ack	Low
01/14/2020 12:13:32	Low Level	prov:default:/tag:Sine/Sine2:/alm:Low Level	Clear	Low
01/14/2020 12:11:49	Low Level	prov:default:/tag:Sine/Sine2:/alm:Low Level	Active	Low

Perspective Alarm Journal - Configuring Properties in Designer

Note: To view alarm history, an [Alarm Journal Profile](#) and a valid connection to a [database](#) must be created first before logging alarms.

Filtering can be done in both the Designer and in a Perspective Session, but enabling and disabling Alarm Journal Table functions and setting up specified filtering options for client users is configured in the Designer. The project designer and Ignition administrator has the option to enable or disable properties and setup filtering options in the Property Editor of the Designer based on the project requirements and needs of the client users. If you are the project designer or Ignition administrator, it's a good idea to check out all the [Alarm JournalTable](#) properties in the Perspective Property Editor. Scroll through all the alarm properties and expand them to see additional properties and see the default property settings and filtering options.

Getting Started
The Alarm Journal Table has some properties enabled by default, and some filter options have already been selected in the Designer to give client users a head start using the table. It's in the Property Editor of the Designer where table properties are enabled and disabled, and filtering options are configured such as the alarm states and priorities to display, setting the shelving times, defining row styles, choosing column headers and column sort options, or setting up a specific display path and source path for the operators to view, and much more.

The first time an Alarm Journal Table component is dragged on to a window in the Designer, by default, the table displays all alarm events for the last 8 hours that are in an **Active**, **Acknowledged** and **Cleared** state with a priority of **Low**, **Medium**, **High** and **Critical**.

The Alarm Journal Table below is similar to what you'll see when you first drag it into a Designer window.

The screenshot shows the Ignition Perspective Property Editor with a red box highlighting the 'PROPS' section. The 'PROPS' section contains various configuration properties for the 'Journal' component, including checkboxes for 'enableHeader' and 'enableDetails', and dropdown menus for 'dateFormat' (set to 'MM/DD/YYYY') and 'responsive'. The 'filter' section is expanded, showing options for 'text', 'events', 'priorities', 'conditions', 'results', 'rowStyles', 'columns', 'pager', and 'style'. On the left, there is a preview of the Alarm Journal Table component showing 91 alarm events from the last 8 hours, filtered by Active, Acknowledged, Cleared, Priority: Low, Medium, High, and Critical. The table includes columns for Event Time, Event Id, Source, Event State, and Priority.

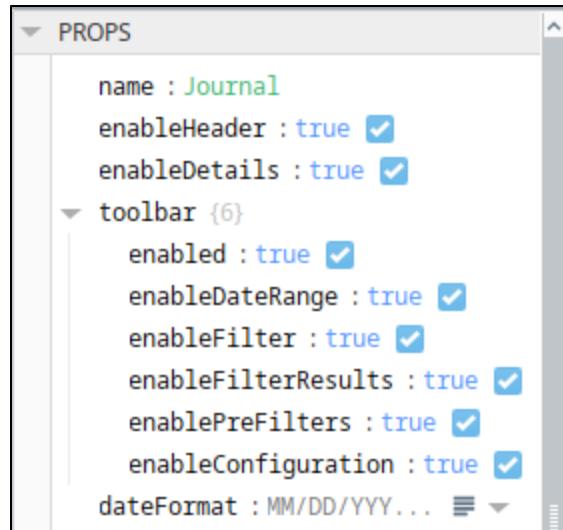
Configuring Properties in the Designer

When you drag a Alarm Journal Table component into the Designer for the first time, it displays the alarm history with default properties already configured and some filtering options already selected for you. Some of the default settings may work for your client users, but others may need to be modified in the Property Editor.

On this page ...

- [Configuring Properties in the Designer](#)
 - [Filter Properties](#)
 - [Condition Properties](#)
 - [Row Styles Properties](#)
 - [Column Properties](#)

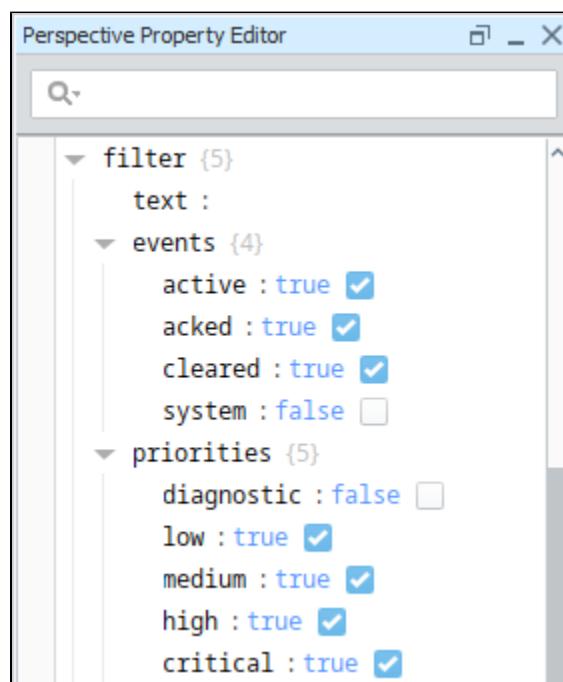
As the project designer, the first thing you might want to look at is the functionality that is enabled on the journal table component.



Note: When configuring the Alarm Journal Table in the Designer, you have to go to [Preview Mode](#) to see your updates, filter, sort and organize the alarm data in the table. When making modifications to the properties in the Property Editor, you will find yourself toggling between the Designer and Preview Modes multiple times to make sure your modifications are displayed the way you want them. Each time you reset your filter options and column headers in the Designer, the journal table will refresh with new alarm data based on your configuration and filter settings.

Filter Properties

While in the Property Editor, expand all the table's properties to see what **filter** properties are configured. The first thing you might want to look at is what **events** settings are set. **Active**, **Acked**, and **Cleared** are set by default, but you'll need to decide if these settings work for your operators. Next, look at the alarm priorities and verify if **Low**, **Medium**, **High**, and **Critical** priorities are the alarm events you want your operators to monitor.



Condition Properties

As the project designer, you might want to control what **source path**, **display path**, and/or **tag provider** you want your operators monitoring. Under the **conditions** property, you can set these values so operators are not navigating to other source paths, display paths or tag providers other than what you configure. In the following example, wildcards were used to search on 'High Speed' in the Display Path property.

The screenshot shows a 'Perspective Property Editor' window with a 'filter' dialog open. The filter criteria is set to 'displayPath : *High ...'. The main table view displays 5 alarm events from the last 8 hours, filtered by active, acknowledged, cleared, low priority, medium priority, and high priority. The 'Display Path' column is highlighted with a red box, and the corresponding rows in the table are also highlighted.

Event Time	Source	Event State	Priority	Display Path
02/06/2020 08:56:35	prov:default:/tag:Speed:/...	Active	Critical	Speed/High Speed
02/06/2020 09:03:31	prov:default:/tag:Speed:/...	Active	Critical	Speed/High Speed
02/06/2020 09:04:08	prov:default:/tag:Speed:/...	Active	Critical	Speed/High Speed
02/06/2020 08:56:28	prov:default:/tag:Speed:/...	Clear	Critical	Speed/High Speed
02/06/2020 09:04:02	prov:default:/tag:Speed:/...	Clear	Critical	Speed/High Speed

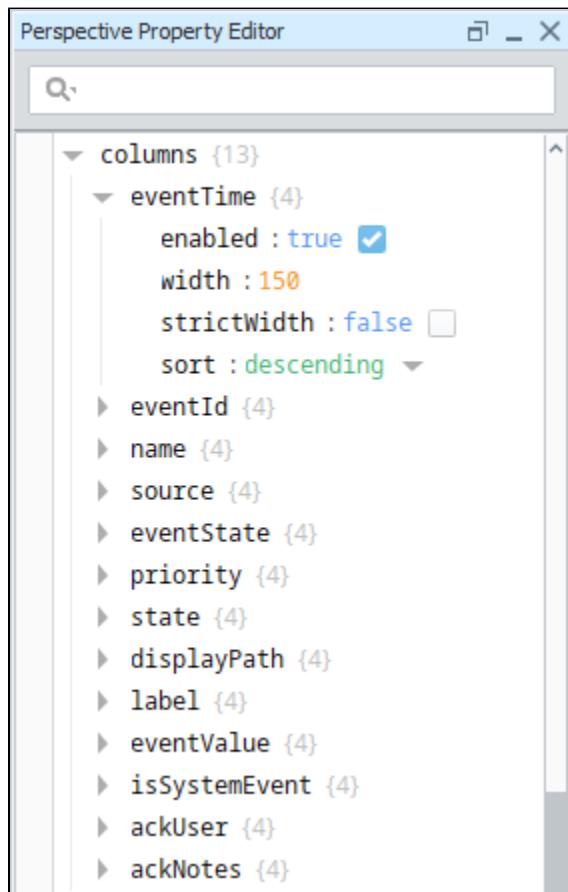
Row Styles Properties

There are some preset row styles background color configured for each alarm state. Open each alarm state property to see its default color. These properties are easy to modify using the color palette. You also have the option to create your own row style for each priority for each alarm state. It's best not too have too many colors and to have a standard set of colors so operators can quickly identify alarms they need to respond to quickly.

The screenshot shows the 'Perspective Property Editor' with the 'rowStyles' tree expanded. The tree includes categories for 'active', 'priorities', 'acked', 'cleared', and 'system'. Under 'active', there is a 'base' node with properties for 'classes', 'backgroundColor' (set to #F0F0F0), 'color' (#2B2B2B), and 'fontWeight' (500). Under 'priorities', there are nodes for 'diagnostic', 'low', 'medium', 'high', and 'critical'.

Column Properties

The column properties are the Configuration Settings  or column headings on the journal table. By default, all the column headings are enabled so client users can pick and choose which headings or type of alarm data they want to display. Each column heading property has four subproperties: enabled, column width, strict width, and sort order. The designer can configure the column settings for client users, but it's recommended to give client users flexibility to display alarm data in the manner that helps them perform their jobs.



To learn more about alarm journal properties, refer to the [Perspective - Alarm Journal Table](#) page.

Related Topics ...

- [Perspective - Alarm Status Table](#)
- [Configuring Alarms](#)

Perspective Alarm Journal - Filtering

Note: To view alarm history, an [Alarm Journal Profile](#) and a valid connection to a [database](#) must be created first before logging alarms.

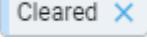
The Alarm Journal Table has many built-in properties that allow you to filter on various parts of an alarm in a Perspective Session. It provides a host of built-in filter options that are immediately available and easy to modify to help you get started. You can filter on Realtime and Historical alarm events using the

Data Range selector . The Search Bar  lets you further optimized your search results. You can even organize the alarm data to show or hide columns, reorganize and resize columns, and change the sort order to ascending or descending.

General Filtering

When you use the Alarm Journal Table for the first time in a Perspective Session, you could be using the default journal properties, or your project designer or system administrator may have preconfigured some journal properties specific to your project. Either way, an operator can easily choose and filter the alarm data and how they want it displayed.

By default, the Alarm Journal Table displays all alarm events for the last 8 hour and displays event alarms that are in an **Active**, **Acknowledged** and **Cleared** state with a priority of **Low**, **Medium**, **High** and

Critical. You can easily change these filter options by clicking on the **Filter icon**  and adding or removing a filter option from the dropdown by checking or unchecking a filter option. Notice that there is a filter bar at the top of the table that also displays all the filter options that are currently set. You can remove any of these filter options by clicking the 'X' on the right side of the option, but to add a filter, use the Select Filter dropdown . Notice how the 'state' filter options show the row color on the tab for Active , Acknowledged  and Cleared .

On this page ...

- General Filtering
 - Using the Search Bar
 - Filtering on Date Range
- Viewing and Sorting on Alarm Data
- Sorting Alarm Journal Data
- Viewing Alarm Details
 - Viewing All Instances of an Alarm
 - Viewing the Source Path of an Alarm



INDUCTIVE
UNIVERSITY

Alarm Journal - General Filtering

[Watch the Video](#)

My First Project

Login

386 alarm events Last 8 hours

FILTERS (7): Active Acknowledged Cleared Priority: Low Priority: Medium Priority: High Priority: Critical

384 results within filters

Event Time	Source	Event State	Priority	Label
01/17/2020 17:18:05	prov:default:/tag:Speed:/alm:High Speed	Active	Critical	High Speed
01/17/2020 17:18:05	prov:default:/tag:Speed:/alm:High Speed	Ack	Critical	High Speed
01/17/2020 17:18:00	prov:default:/tag:Speed:/alm:High Speed	Clear	Critical	High Speed
01/17/2020 17:17:51	prov:default:/tag:Sine/Sine2:/alm:Low Level	Active	Critical	Low Level
01/17/2020 17:17:51	prov:default:/tag:Sine/Sine2:/alm:Low Level	Ack	Critical	Low Level
01/17/2020 17:14:22	prov:default:/tag:Sine/Sine2:/alm:Low Level	Clear	Critical	Low Level
01/17/2020 17:13:51	prov:default:/tag:Sine/Sine1:/alm:High Level	Active	High	High Level
01/17/2020 17:13:51	prov:default:/tag:Sine/Sine1:/alm:High Level	Ack	High	High Level

25 rows ▾ First < 1 2 3 4 5 > Last

Select Filters X

EVENT

- Show All
- Active**
- Acknowledged**
- Cleared**
- System

PRIORITY

- Show All
- Diagnostic
- Low**
- Medium**
- High**
- Critical**

Using the Search Bar

You can optimize your filtered results using the Search Bar. Click on the **Search icon** and create your own search string to find specific alarms, events, and conditions.

3747 alarm events Last 15 days

SEARCH: High Speed X

FILTERS (7): Active Acknowledged Cleared Priority: Low Priority: Medium Priority: High Remove All

115 results within filters

Event Time	Name	Event State	Priority	Display Path	Event Value
01/21/2020 11:47:32	High Speed	Active	Critical	Speed/High Speed	132
01/21/2020 11:47:32	High Speed	Ack	Critical	Speed/High Speed	
01/21/2020 11:47:15	High Speed	Clear	Critical	Speed/High Speed	88
01/21/2020 10:42:53	High Speed	Active	Critical	Speed/High Speed	101
01/21/2020 10:42:53	High Speed	Ack	Critical	Speed/High Speed	
01/21/2020 10:42:44	High Speed	Clear	Critical	Speed/High Speed	88
01/21/2020 10:01:18	High Speed	Active	Critical	Speed/High Speed	109
01/21/2020 10:01:18	High Speed	Ack	Critical	Speed/High Speed	

25 rows ▾ 1 2 3 4 5

Filtering on Date Range

The Alarm Journal Table can filter for alarm events in either Realtime or Historical. To filter on Realtime or Historical alarm events, select the **Date Range icon**.

Realtime

You can filter on Realtime alarm events using the **Data Range** feature. Simply enter select Realtime, and enter the amount of time in hours, days, weeks, months, or years of the alarm events to display. Then click **Apply**.

The screenshot shows the Alarm Journal Table interface. At the top, there is a header bar with a date range icon (calendar with a clock), the text "3618 alarm events Last 15 days", and search/filter icons. Below the header is a row of filters: Active, Acknowledged, Cleared, Priority: Low, Priority: Medium, Priority: High, and Prior. A "Remove All" button is also present. The main area displays 3611 results within filters. The table has columns: Event Time, Name, Event State, Priority, Display Path, Label, and Event Value. One row is highlighted in red, indicating it is selected. A context menu is open over this row, with "Realtime" selected. A dropdown menu for setting the date range is open, showing options: Show last 12 hours, hours, days, weeks, months, and years. The "hours" option is currently selected. At the bottom of the table area, there are navigation buttons for First, Last, and page numbers (1, 2, 3, 4, 5). A "Jump to:" input field is also present.

Historical

The **Alarm Journal Table** displays the complete history of all alarms within a specific time period that you set. Typically, operators want to filter **alarm events** within a specific time period so they can narrow down the number of alarm events that need to be viewed. Select **Historical**, enter the date range by clicking on the **Start Date** and **End Date**, then select the **Start Time** and **End Time**, and click **Apply**.

The screenshot shows a software interface for managing alarm events. On the left, there is a table titled "106 alarm events Last 8 hours" with columns: Event Time, Event State, and Priority. The table lists various events from January 21, 2020, at 10:34:31 to 10:25:07. On the right, there is a list of event values with their corresponding levels. A modal dialog box titled "Set Date Range" is open in the center. The dialog has tabs for "Realtime" and "Historical", with "Historical" selected. It shows a date range from "2020/01/21 - 2020/01/22". Below the date range is a calendar for January 2020, with days 21 and 22 highlighted. At the bottom of the dialog, there are time selection boxes for "Start time on 2020/01/21" (set to 12 : 00 AM) and "End time on 2020/01/22" (set to 11 : 59 PM), and buttons for "Cancel" and "Apply".

Viewing and Sorting on Alarm Data

There are a host of configuration settings to choose from in which to display alarm event data. In a Perspective Session, operators can choose to display or hide **alarm** data in the table by checking or unchecking any of the **Configuration Settings** or by right clicking in the header row of the table. It's super easy to filter and display **alarm** event data using the configuration options provided in the column header of the **Alarm Status Table**. You can do a multi-column sort by holding down **Ctrl + Shift** and clicking on the column headers you want to sort by.

If you need to align the columns, simply put your cursor to the left of the column header and drag left or right. There is a '**strictWidth**' property for each header column that can be configured in the Designer to strictly enforce the column width.

The screenshot shows the 'My First Project' interface. At the top, there are three horizontal bars, the project name 'My First Project', and a 'Login' button with a user icon. Below the header is a search bar with a magnifying glass icon and a gear icon. To the right of the search bar is a red-bordered 'Configuration' button with an 'X' icon. The main area contains a table titled '386 alarm events Last 8 hours'. The table has columns: Event Time, Source, Event State, Priority, Label, Event Value, and Ack Notes. There are 384 results within filters. The table is sorted by Event Time. A configuration menu is open on the right, titled 'Configuration' with a close 'X' icon. It includes a 'COLUMNS' section with checkboxes for various fields: Ack Notes, Ack User, Display Path, Event Id, Event State (which is checked), Event Time (checked), Event Value (checked), Is System Event, Label (checked), Name, Priority (checked), Source (checked), and State. The 'Event State' checkbox is highlighted with a blue background.

Sorting Alarm Journal Data

Perspective's [Alarm Journal](#) Table now supports multiple sort orders. The order is first determined by the sort order properties on the [Alarm Journal](#) Table. The new sort order property is 'activeSortOrder.' Each column that you add to the sort order [property](#) will also need to have a sort defined under the columns [property](#) as either ascending or descending.

The screenshot shows two instances of the 'Perspective Property Editor' side-by-side. The left editor shows the 'activeSortOrder' property, which is an array containing three objects: {0 : eventState}, {1 : priority}, and {2 : eventTime}. The right editor shows the 'columns' property, which is an array of 13 objects. One object for 'eventState' is highlighted with a red border, showing its properties: enabled (true checked), width (110), strictWidth (false unchecked), and sort (descending). Both editors have a 'Filter' search bar at the top.

You also need to make sure the configuration columns you choose for your sort are displayed on your table. In the following image, you can see that the sort order is **Event State - Priority - Event Time**. Notice how each column is numbered according to the order you defined in your [activeSortOrder](#) [property](#).

387 alarm events
Last 8 hours

FILTERS (6): < Active X Cleared X Priority: Low X Priority: Medium X Priority: High > Remove All

256 results within filters

Event Time	Name	Event State	Priority	Display Path
03/24/2020 15:08:16	Low Level	Active	Critical	Sine/Sine2/Low Level
03/24/2020 15:08:16	Low Level	Active	Critical	Sine/Sine2/Low Level
03/24/2020 15:08:16	Low Level	Active	Critical	Sine/Sine2/Low Level
03/24/2020 15:08:16	Low Level	Active	Critical	Sine/Sine2/Low Level
03/24/2020 15:08:16	Low Level	Active	Critical	Sine/Sine2/Low Level
03/24/2020 15:08:16	Low Level	Active	Critical	Sine/Sine2/Low Level

Columns will first sort by anything defined in `activeSortOrder` property and then fallback to the other sorts defined in the columns configuration.

Viewing Alarm Details

If you want to find out more details about an alarm event, simply mouse over an alarm event and click on the popup modal that will appear on the right side of table. This opens an Alarm Detail window that displays the alarm properties and any notes that were entered by an operator.

3952 alarm events
Last 15 days

FILTERS (7): Active Acknowledged Cleared Priority: Low Priority: Medium Priority: High Remove All

3945 results within filters

Event Time	Name	Event State	Priority	Display Path	Event Value																														
01/22/2020 08:49:32	Low Level	Active	Critical	Sine/Sine2/Low Level	24.391563415527344																														
01/22/2020 08:49:32	Low Level	Ack	Critical	Sine/Sine2/Low Level																															
01/22/2020 08:48:58	High Speed	Active	Critical	Speed/High Speed	110																														
01/22/2020	Alarm Details																																		
01/22/2020	<table border="1"> <tr> <td>Properties</td> <td>Notes</td> </tr> </table>					Properties	Notes																												
Properties	Notes																																		
01/22/2020	<p>▼ Config Properties</p> <table border="1"> <tr><td>Ack Mode</td><td>Manual</td></tr> <tr><td>Ack Notes Rqrd</td><td><input type="checkbox"/></td></tr> <tr><td>Deadband</td><td>0.0</td></tr> <tr><td>Deadband Mode</td><td>Absolute</td></tr> <tr><td>Display Path</td><td>Speed/High Speed</td></tr> <tr><td>Enabled</td><td><input checked="" type="checkbox"/></td></tr> <tr><td>Label</td><td>High Speed</td></tr> <tr><td>Mode</td><td>Equal</td></tr> <tr><td>Name</td><td>High Speed</td></tr> <tr><td>Notes</td><td></td></tr> <tr><td>Priority</td><td>Critical</td></tr> <tr><td>Shelving Allowed</td><td><input checked="" type="checkbox"/></td></tr> <tr><td>Time Off Delay Secon...</td><td>0.0</td></tr> <tr><td>Time On Delay Secon...</td><td>0.0</td></tr> <tr><td>Timestamp Source</td><td>System</td></tr> </table>					Ack Mode	Manual	Ack Notes Rqrd	<input type="checkbox"/>	Deadband	0.0	Deadband Mode	Absolute	Display Path	Speed/High Speed	Enabled	<input checked="" type="checkbox"/>	Label	High Speed	Mode	Equal	Name	High Speed	Notes		Priority	Critical	Shelving Allowed	<input checked="" type="checkbox"/>	Time Off Delay Secon...	0.0	Time On Delay Secon...	0.0	Timestamp Source	System
Ack Mode	Manual																																		
Ack Notes Rqrd	<input type="checkbox"/>																																		
Deadband	0.0																																		
Deadband Mode	Absolute																																		
Display Path	Speed/High Speed																																		
Enabled	<input checked="" type="checkbox"/>																																		
Label	High Speed																																		
Mode	Equal																																		
Name	High Speed																																		
Notes																																			
Priority	Critical																																		
Shelving Allowed	<input checked="" type="checkbox"/>																																		
Time Off Delay Secon...	0.0																																		
Time On Delay Secon...	0.0																																		
Timestamp Source	System																																		

Viewing All Instances of an Alarm

If you select an alarm, a window will appear at the bottom of the journal table where you can choose to view instances of that specific alarm or its source path. When you click on the **Alarm** button, the journal table will refresh with instances of the selected alarm.

In this example, High Speed alarm was selected to view all instances of the alarm.

4023 alarm events
Last 15 days

FILTERS (7): Active Acknowledged Cleared Priority: Low Priority: Medium Priority: High > Remove All

4016 results within filters

Event Time	Name	Event State	Priority	Display Path	Event Value
01/22/2020 10:18:46	High Speed	Active	Critical	Speed/High Speed	125
01/22/2020 10:18:46	High Speed	Ack	Critical	Speed/High Speed	
01/22/2020 10:18:38	High Speed	Clear	Critical	Speed/High Speed	85
01/22/2020 10:18:27	High Level	Clear	High	Sine/Sine1/High Level	78.21446589319956
01/22/2020 10:17:42	Low Level	Clear	Critical	Sine/Sine2/Low Level	25.758882522583008
01/22/2020 10:13:32	High Level	Active	High	Sine/Sine1/High Level	80.45196419133792
View instances for this: Alarm Source Path					
25 rows	First	1	2	3	4
	5	>	Last	Jump to: <input type="text" value="1"/>	

Click the **View all events** link to return to all alarm events on the journal table.

4027 alarm events
Last 15 days

Event Time	Name	Event State	Priority	Display Path	Event Value
01/22/2020 10:18:46	High Speed	Active	Critical	Speed/High Speed	125
01/22/2020 10:18:46	High Speed	Ack	Critical	Speed/High Speed	
01/22/2020 10:18:38	High Speed	Clear	Critical	Speed/High Speed	85
01/22/2020 08:48:58	High Speed	Active	Critical	Speed/High Speed	110
01/22/2020 08:48:58	High Speed	Ack	Critical	Speed/High Speed	
01/22/2020 08:48:48	High Speed	Clear	Critical	Speed/High Speed	87
01/21/2020 11:47:32	High Speed	Active	Critical	Speed/High Speed	132
01/21/2020 11:47:32	High Speed	Ack	Critical	Speed/High Speed	
01/21/2020 11:47:15	High Speed	Clear	Critical	Speed/High Speed	88
Viewing instances of High Speed View all events					
25 rows	1	2	3	4	5

Viewing the Source Path of an Alarm

You can also click the **Source Path** button to display the source path of the selected alarm.

4037 alarm events
Last 15 days

FILTERS (7): Active Acknowledged Cleared Priority: Low Priority: Medium Priority: High > Remove All

4030 results within filters

Event Time	Name	Event State	Priority	Display Path	Event Value
01/22/2020 10:18:46	High Speed	Active	Critical	Speed/High Speed	125
01/22/2020 10:18:46	High Speed	Ack	Critical	Speed/High Speed	
01/22/2020 10:18:38	High Speed	Clear	Critical	Speed/High Speed	85
01/22/2020 10:18:27	High Level	Clear	High	Sine/Sine1/High Level	78.21446589319956
01/22/2020 10:17:42	Low Level	Clear	Critical	Sine/Sine2/Low Level	25.758882522583008
01/22/2020 10:13:32	High Level	Active	High	Sine/Sine1/High Level	80.45196419133792
01/22/2020 10:12:20	High Level	Ack	Critical	Sine/Sine1/High Level	
View instances for this:					
Alarm Source Path					
25 rows	First	1	2	3	Last
Jump to:	1				

Click the [View all events](#) link to return to the alarm events on the journal table.

4040 alarm events
Last 15 days

Event Time	Name	Event State	Priority	Display Path	Event Value
01/22/2020 10:18:46	High Speed	Active	Critical	Speed/High Speed	125
01/22/2020 10:18:46	High Speed	Ack	Critical	Speed/High Speed	
01/22/2020 10:18:38	High Speed	Clear	Critical	Speed/High Speed	85
01/22/2020 08:48:58	High Speed	Active	Critical	Speed/High Speed	110
01/22/2020 08:48:58	High Speed	Ack	Critical	Speed/High Speed	
01/22/2020 08:48:48	High Speed	Clear	Critical	Speed/High Speed	87
01/21/2020 11:47:32	High Speed	Active	Critical	Speed/High Speed	132
01/21/2020 11:47:32	High Speed	Ack	Critical	Speed/High Speed	
01/21/2020 11:47:15	High Speed	Clear	Critical	Speed/High Speed	88
Viewing instances of prov:default;/tag:Speed;/alm:High Speed					
View all events					
25 rows	1	2	3	4	5

Related Topics ...

- [Configuring Alarms](#)
- [Perspective - Alarm Journal Table](#)

Perspective Alarm Journal - Row Styles

The Perspective Alarm Journal Table allows you to customize row styles for different states and priorities of alarm history. Just like the Alarm Status Table, the [Alarm Journal Table](#) is preconfigured with a default set of colors under the **rowStyles** property for each of the alarm event states, and immediately ready for you to use. The default row colors are pink for Active, yellow for Acknowledged, blue for Clear, and white for System alarms events.

Event Time	Name	Source	Event State	Priority	Event Value
01/22/2020 15:00:45	High Speed	prov:default:/tag:Speed:/alm:High Speed	Ack	Critical	
01/22/2020 15:00:45	High Speed	prov:default:/tag:Speed:/alm:High Speed	Active	Critical	107
01/22/2020 15:00:33	High Speed	prov:default:/tag:Speed:/alm:High Speed	Clear	Critical	97
01/22/2020 15:00:11	High Level	prov:default:/tag:Sine/Sine1:/alm:High Level	Ack	High	
01/22/2020 15:00:11	High Level	prov:default:/tag:Sine/Sine1:/alm:High Level	Active	High	80.56041975080008
01/22/2020 14:57:56	High Level	prov:default:/tag:Sine/Sine1:/alm:High Level	Clear	High	79.17023895397026
01/22/2020 14:56:11	Low Level	prov:default:/tag:Sine/Sine2:/alm:Low Level	Ack	Critical	

On this page ...

- Customizing Row Styles for the Different Alarm States
- Example 1 - Modifying the `rowStyle.backgroundColor` Property for an Alarm State
- Example 2 - Add a Style Class to a `rowStyle`



INDUCTIVE
UNIVERSITY

Alarm Journal - Row Styles

[Watch the Video](#)

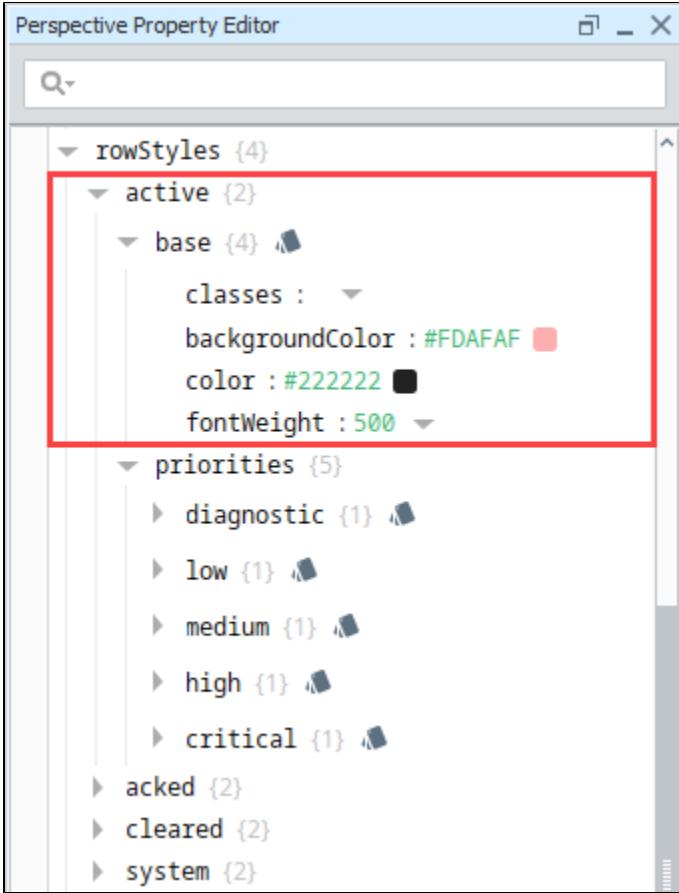
Customizing Row Styles for the Different Alarm States

The Alarm Journal Table comes with a default set of row colors associated with each of the different alarm states. You can modify an existing row style in the Designer by modifying the '`rowStyle`' properties for each of the alarm states. Each `alarm` state has the same default `rowStyle` properties, but their `backgroundColor` values are different.

Here is a list of default properties for each row:

- **base** - Opens a [Style Options](#) menu to configure `style elements` for Text, Background, Margin and Padding, Border, Shape and Misc.
- **classes** - Allows you to set a pre-defined `style class`.
- **backgroundColor** - Background color for each of the priorities (i.e., diagnostic, low, medium, high, critical) for an `alarm` state.
- **color** - Color of the Text.

- **fontWeight** - Font weight of the text.

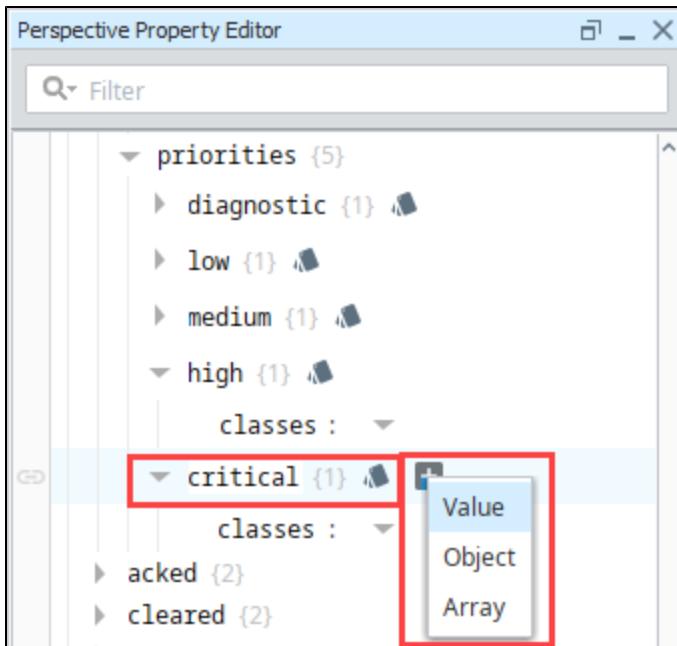


You can modify rowStyles properties in multiple places: using the '**base**' (i.e., [Style Options](#) menu) or [property data types](#) in the Property Editor. When you modify, add, or remove a style [property](#) from either Style Options menu or the Property Editor, the change will immediately be visible in both locations, as well as in the table. Here are two examples of how to modify rowStyles in an Alarm Journal.

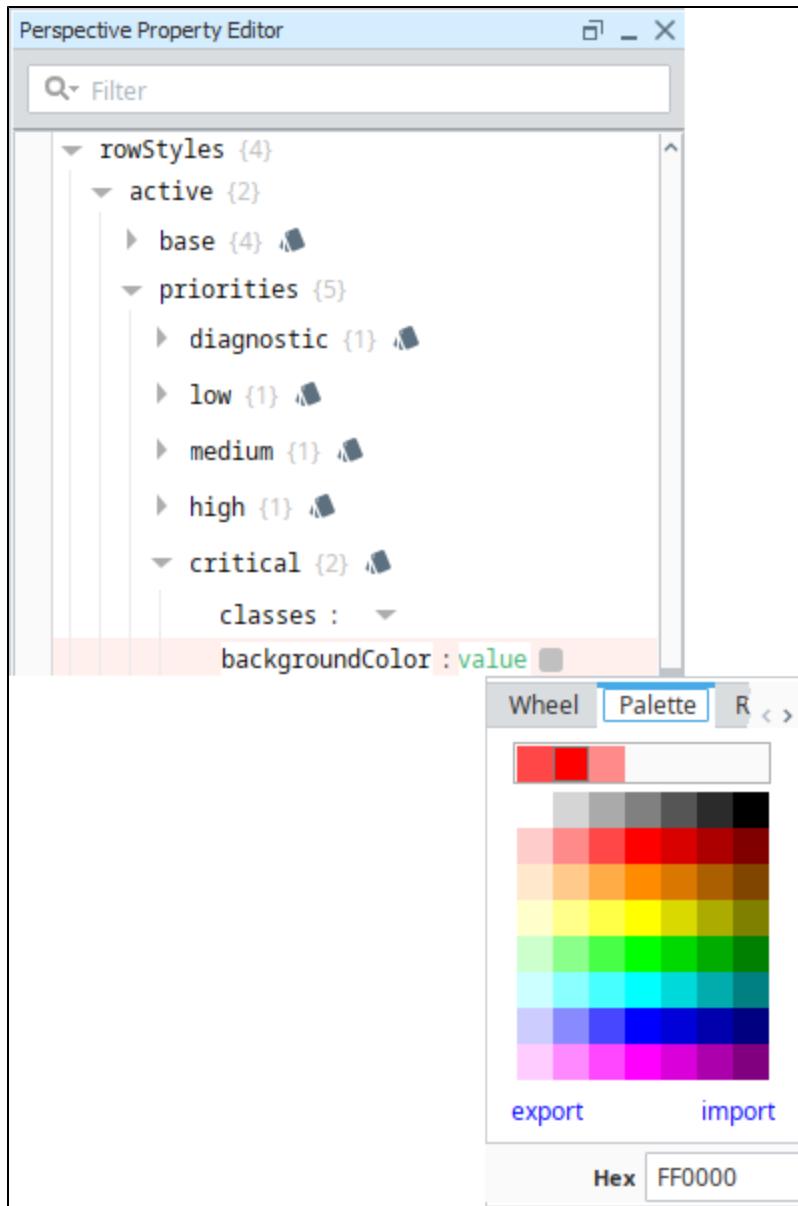
Example 1 - Modifying the rowStyle backgroundColor Property for an Alarm State

Let's change the background row color for the '**active**' state with a '**critical**' priority from pink to red.

1. In the Designer, select the Alarm Journal Table component.
2. In the Property Editor, expand the **rowStyles** property and the '**active**' state and '**priorities**' properties.
3. Expand the '**critical**' property. Mouse over the '**critical**' priority property and click on the plus icon.



4. A dropdown will appear and select 'value.'
5. The word '**key**' will be highlighted, enter the '**backgroundColor**' property.
6. Enter a value by clicking the gray square to open a popup color wheel or color palette, then choose a color (i.e., red).



7. This will change the **backgroundColor** property for the '**Active**' state and '**Critical**' priority from pink to red.

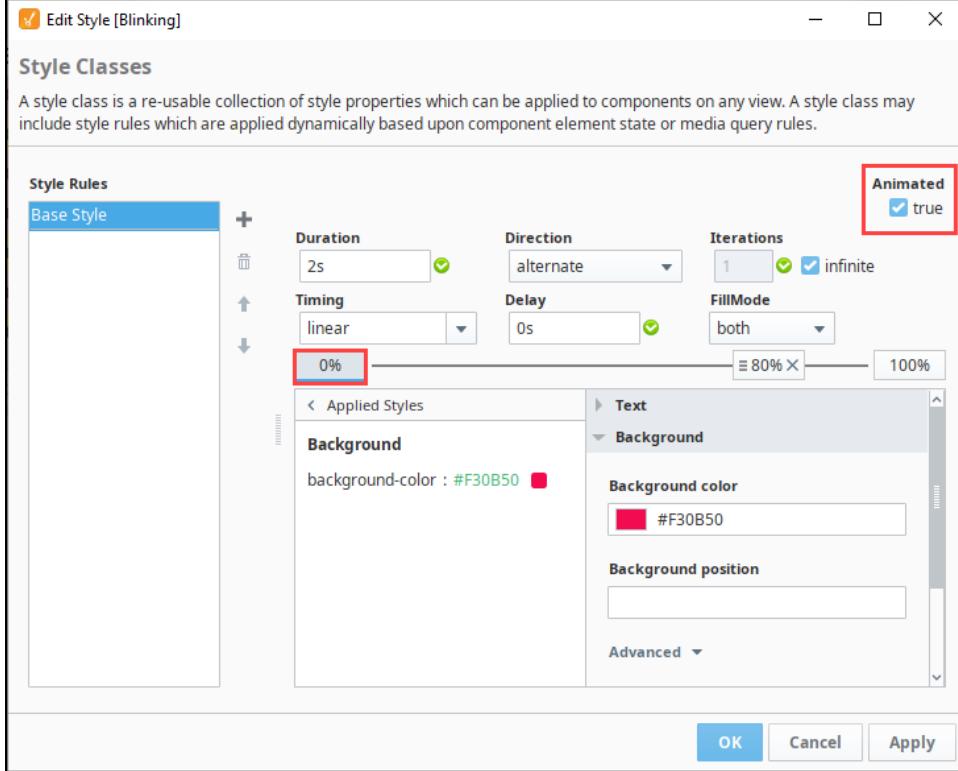
264 alarm events Last 8 hours					
FILTERS (7): Active Acknowledged Cleared Priority: Low Priority: Medium Priority: H > Remove All					
264 results within filters					
Event Time	Name	Source	Event State	Priority	
02/07/2020 08:40:41	High Temp	prov:default:/tag:Sine/Sine1:/alm:High Temp	Active	High	
02/07/2020 08:47:10	Low Level	prov:default:/tag:Sine/Sine2:/alm:Low Level	Active	Critical	
02/07/2020 08:47:50	High Temp	prov:default:/tag:Sine/Sine1:/alm:High Temp	Active	High	
02/07/2020 08:55:00	High Temp	prov:default:/tag:Sine/Sine1:/alm:High Temp	Active	High	
02/07/2020 08:55:30	Low Level	prov:default:/tag:Sine/Sine2:/alm:Low Level	Active	Critical	
02/07/2020 09:02:10	High Temp	prov:default:/tag:Sine/Sine1:/alm:High Temp	Active	High	
02/07/2020 09:03:50	Low Level	prov:default:/tag:Sine/Sine2:/alm:Low Level	Active	Critical	

Example 2 - Add a Style Class to a rowStyle

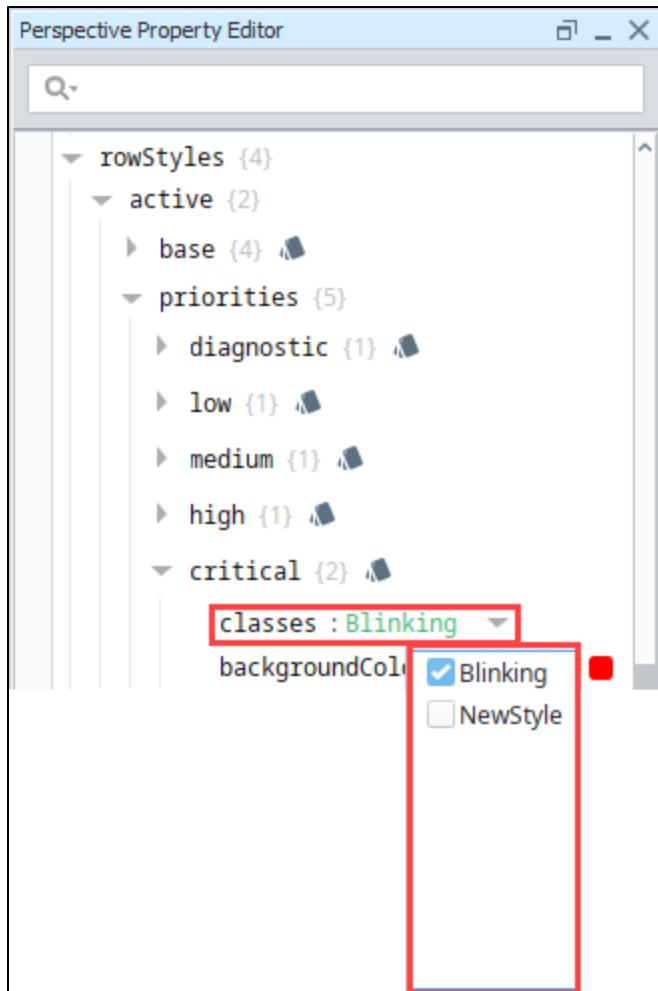
You can create a new style class to make a row style stand out and draw attention to that alarm state. In this example, let's create the 'active' state with the 'critical' priority blink between two colors, red and green.

1. To create a new Style Class, go to the **Project Browser**, right click on **Styles Folder**, and click on **New Style**.
2. Enter a name (i.e. Blinking) for your new style, and click **Create Style Class**. The **Edit Style** window will open.
3. Set **Animated** to 'true' in the upper right corner.
4. Click on **0%** to set the style for the beginning of the animation.
5. Click on **Background** to see Background settings for this style and choose a color. This example uses red.

Notice the other Animation properties on the Edit Style window. You can set Duration, Direction, number of Iterations, and more. For a detailed description of the Animated properties, refer to [Animated Style Classes](#).



6. Next, click on **100%** to set the style for the end of the animation.
7. Click on **Background** to expand the Background settings, and choose a color from the either the color palette or color wheel (i.e., green). When choosing a second color, make sure the foreground color will be readable when it's blinking.
8. To make one of the colors stay for a longer part of the duration, we can add another stop to the animation timeline. Right click on the horizontal line between the 0% and 100% boxes. Drag the stop to about **80%** to show the red (0% color) for longer than the green (100% color).
9. Set the Background color on this new 80% stop to match the red background color of the 0% stop.
10. Click **OK** to save your new style class.
11. Lastly, set the style class on the '**critical**' priority property to the name of the class (i.e., **Blinking**).



12. Now your '**active**' alarm state with a '**critical**' priority will blink red and green.

317 alarm events Last 8 hours					
FILTERS (7): Active Acknowledged Cleared Priority: Low Priority: Medium Priority: High Remove All					
317 results within filters					
Event Time	Name	Source	Event State	Priority	
02/07/2020 08:40:41	High Temp	prov:default:/tag:Sine/Sine1:/alm:High Temp	Active	High	
02/07/2020 08:47:10	Low Level	prov:default:/tag:Sine/Sine2:/alm:Low Level	Active	Critical	
02/07/2020 08:47:50	High Temp	prov:default:/tag:Sine/Sine1:/alm:High Temp	Active	High	
02/07/2020 08:55:00	High Temp	prov:default:/tag:Sine/Sine1:/alm:High Temp	Active	High	
02/07/2020 08:55:30	Low Level	prov:default:/tag:Sine/Sine2:/alm:Low Level	Active	Critical	
02/07/2020 09:02:10	High Temp	prov:default:/tag:Sine/Sine1:/alm:High Temp	Active	High	
02/07/2020 09:03:50	Low Level	prov:default:/tag:Sine/Sine2:/alm:Low Level	Active	Critical	

Reporting in Perspective

Note: The Report Module must be installed to use the Report Viewer component.

The Report Viewer component allows you to embed reports from the Reporting Module in a Perspective view, as well as view and print reports in PDF format. The Report Viewer located at the bottom of the Perspective Component palette in the Designer. To configure the Report Viewer, you must first create a report.

To learn more about creating reports, refer to the [Reporting](#) section.

On this page ...

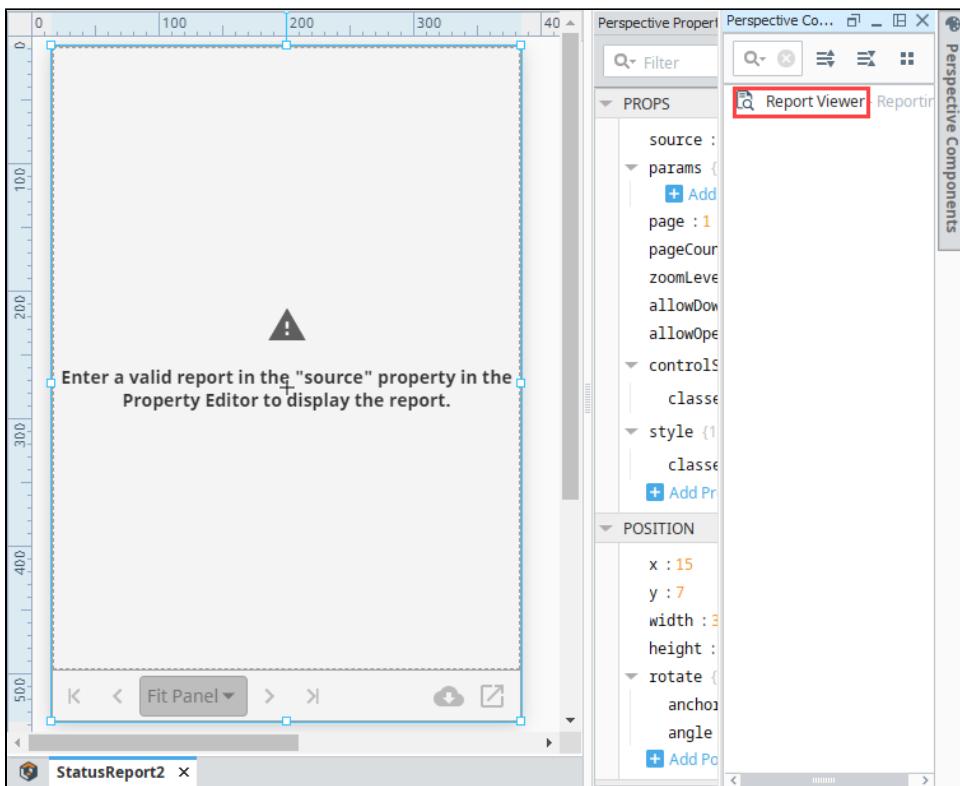
- [Configuring the Report Viewer](#)
- [Report Viewer Properties](#)
- [Using the Report Viewer](#)

Configuring the Report Viewer

The Report Viewer component provides an easy way to view and print reports in a Perspective View. Let's configure the Report Viewer for a report.

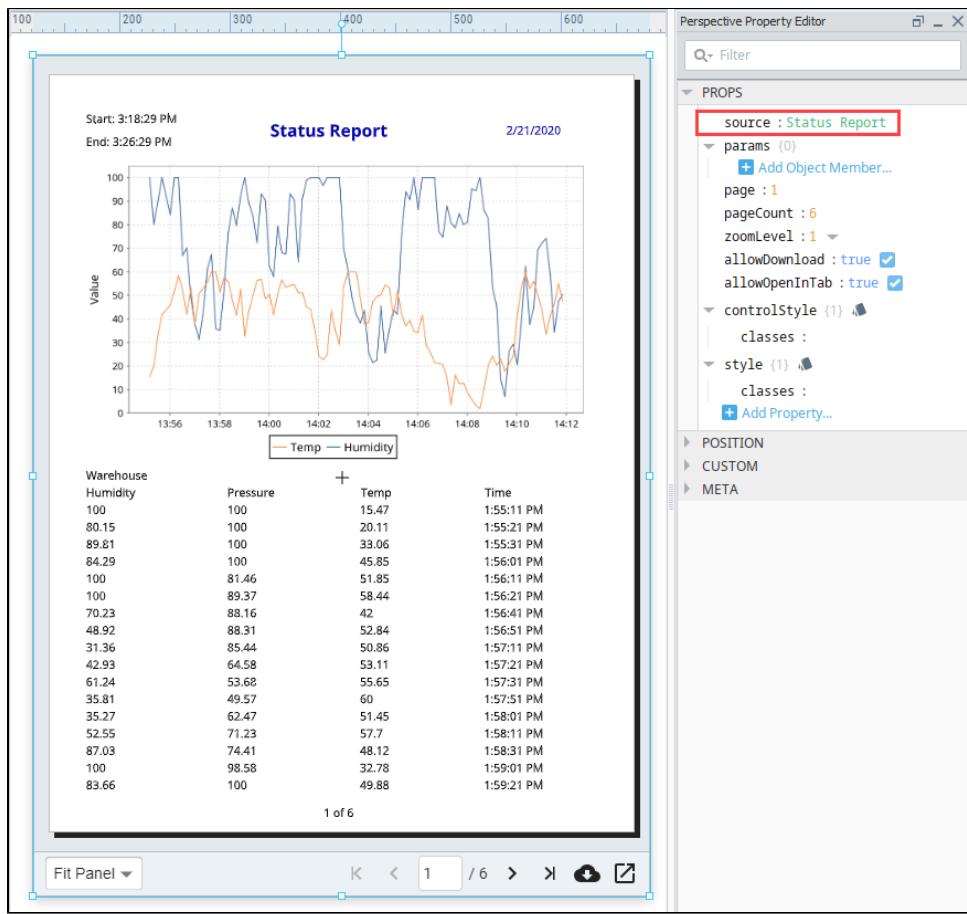
Note: This example requires that you already have a report created in your Gateway. You can learn more about creating reports [here](#).

1. Drag a **Report Viewer** component into a view.

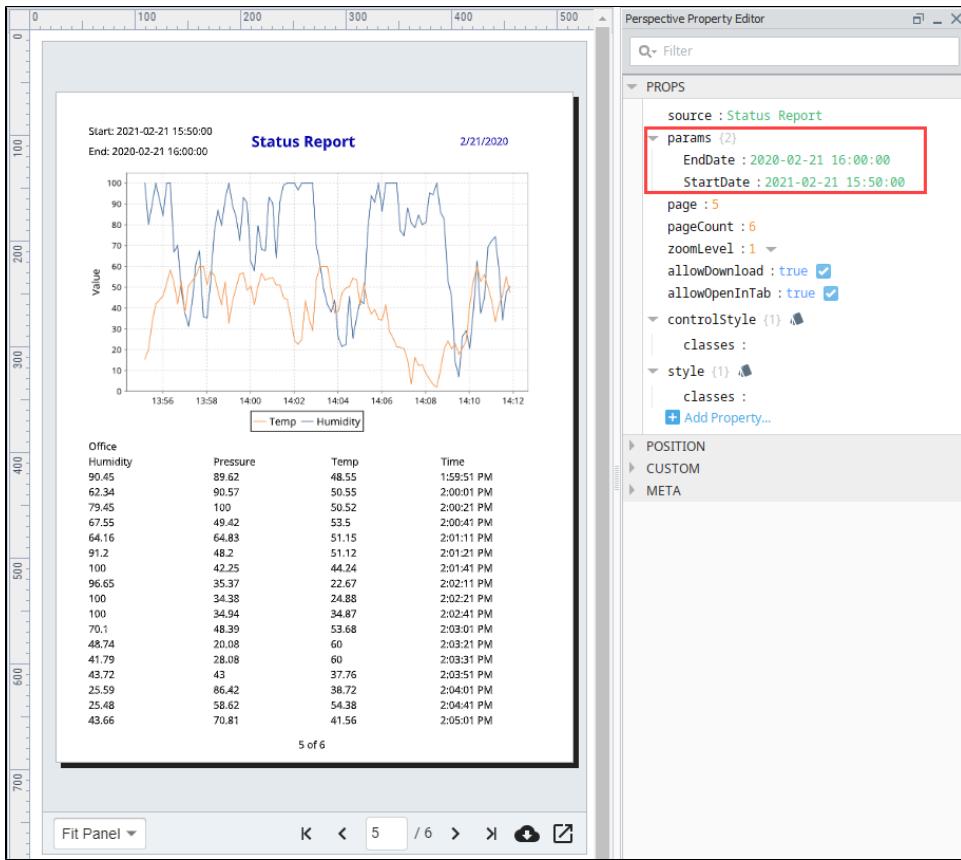


2. Enter the Path of the report you want to view in the **'source'** property of the Property Editor. The data from your report will immediately load into the Report Viewer.

Tip: You can right-click on any report in the designer Project Browser to Copy the full Path of the report.



- Parameters added during report creation are provided as properties in the Report Viewer. Add them under the **'params'** object. (The example below uses the EndDate and StartDate parameters). The parameter names must match exactly to the parameters in your Report Resource, and will override any default values set in the Report Resource. The Report Viewer also allows you to bind your report parameters in your view.



Report Viewer Properties

Once you are ready to view your report in the Report Viewer, set the '**source**' property to your report. If you have parameters defined in the Report Source, you can configure them under the '**params**' object, but the parameter names must match. The parameter values configured in the Report Viewer will override the values in the Report Source.

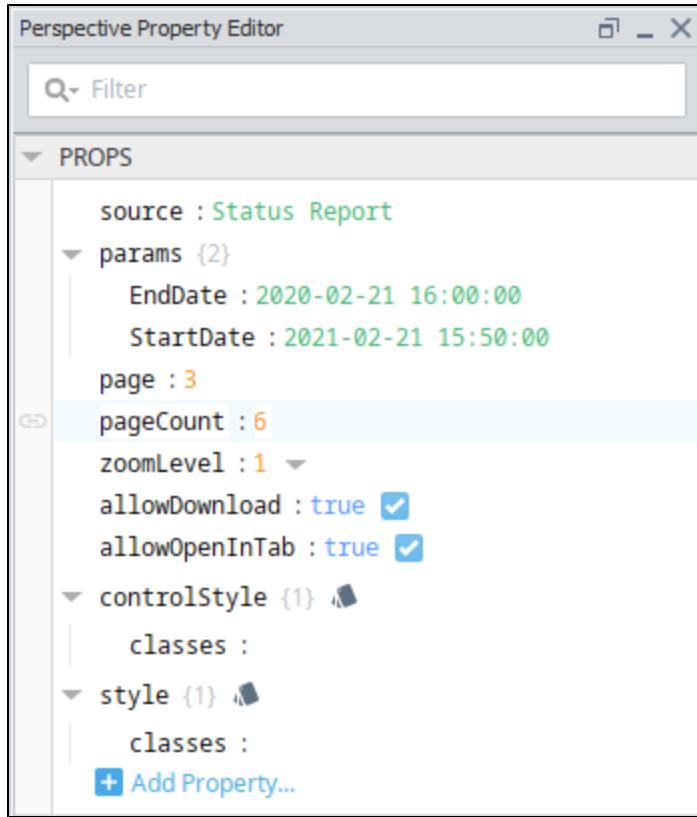
You can pick a starting page using the '**page**' property. Each time another page is viewed, the '**page**' property value will be updated.

You can specify a desired '**zoomLevel**' and every time the zoomLevel changes, the property value will also be updated.

You can customize the visual style of your report by creating a new style for your report such as changing the background colors of the viewer (not the report page), changing to a style class that was already defined in your project, or creating a new style.

The '**allowDownload**' and '**allowOpenInTab**' properties allow you to view and print your report.

To see the property descriptions, refer to the [Perspective - Report Viewer](#) page.



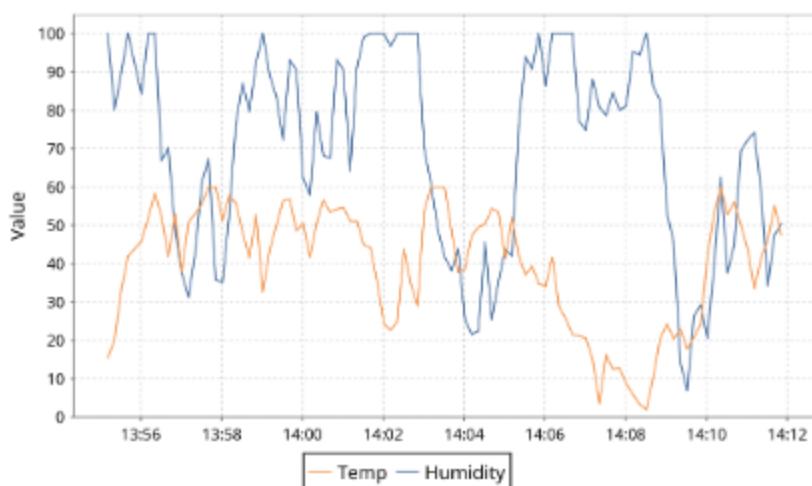
Using the Report Viewer

Once you have your report designed, it can be viewed one page at a time using the navigational controls across the bottom of the component. You have the option to download a report to your local device or print a report to your local printer using these built-in controls.

Start: 12:46:14 PM
End: 12:54:14 PM

Status Report

2/24/2020



Warehouse

Humidity	Pressure	Temp	Time
100	100	15.47	1:55:11 PM
80.15	100	20.11	1:55:21 PM
89.81	100	33.06	1:55:31 PM
84.29	100	45.85	1:56:01 PM
100	81.46	51.85	1:56:11 PM
100	89.37	58.44	1:56:21 PM
70.23	88.16	42	1:56:41 PM
48.92	88.31	52.84	1:56:51 PM
31.36	85.44	50.86	1:57:11 PM
42.93	64.58	53.11	1:57:21 PM
61.24	53.68	55.65	1:57:31 PM
35.81	49.57	60	1:57:51 PM
35.27	62.47	51.45	1:58:01 PM
52.55	71.23	57.7	1:58:11 PM
87.03	74.41	48.12	1:58:31 PM
100	98.58	32.78	1:59:01 PM
83.66	100	49.88	1:59:21 PM

1 of 6

Fit Panel ▾



/ 6



Related Topics ...

- [Reporting](#)
- [Perspective - Report Viewer](#)

Common Tasks in Perspective

This section contains examples for items we identified as "common" tasks: methods that many users are looking to utilize when first starting out with the Perspective Module or feature in Ignition. Additionally, this section aims to clarify some of the more complex or abstract tasks that our users may encounter.

The examples in this section are self-contained explanations that may touch upon many other areas of Ignition. While these examples are typically focused on a single goal or end result, they can easily be expanded or modified after the fact. In essence, they serve as a great starting point for users new to Ignition, as well as experienced users that need to get acquainted with a new or unfamiliar feature.

Below is a list of common tasks related to this section of the manual.

On this page ...

- [Popup Views](#)
- [Navigating with the Horizontal Menu](#)
- [Self-Hiding Navigation Drawer](#)
- [Configuring a Dashboard](#)
- [Displaying a Subview in a Table](#)
- [Download and Upload Files](#)
- [Table Column Configurations](#)

Popup Views

[Popup Views](#) are a great way to enable users to view more detailed information in your HMI. They are also relatively simple to create.

Navigating with the Horizontal Menu

The [Horizontal Menu](#) component is a great option for easy, quick navigation between pages. For an example, see [Navigating with the Horizontal Menu Component](#).

Self-Hiding Navigation Drawer

A [navigation drawer](#) is a special type of docked menu, usually appearing on the left side of a session. What makes a navigation drawer special is its responsive design. This example walks you through creating a self-hiding navigation drawer that is only displayed when the user needs it.

Configuring a Dashboard

The Dashboard exposes widgets to end users in a [Perspective Session](#) so they can customize their dashboard layout for their individual needs. Widgets are [views](#) that are pre-configured in the Designer and made available to Perspective Session users. For an example, see [Configuring a Dashboard](#).

Displaying a Subview in a Table

In a Perspective Table component, you have the option to enable subviews. When a subview is set up, you can click on the Expand icon in the table and have another view be displayed without closing the first view. For an example, see [Displaying a View in a Table](#).

Download and Upload Files

Downloading and uploading files from a Perspective session typically involves storing and retrieving files from a [database](#). A table will store all of the available files, and each row of the table represents a new file. This allows for long term storage that is accessible from any project. For examples, see [Download and Upload Files](#).

Table Column Configurations

In a Perspective Table component, you have the ability to replace a cell with something other than just text or a number. Instead, you can have the column render other objects altogether such as progress bars or a view. For an example, see [Table Column Configurations](#).

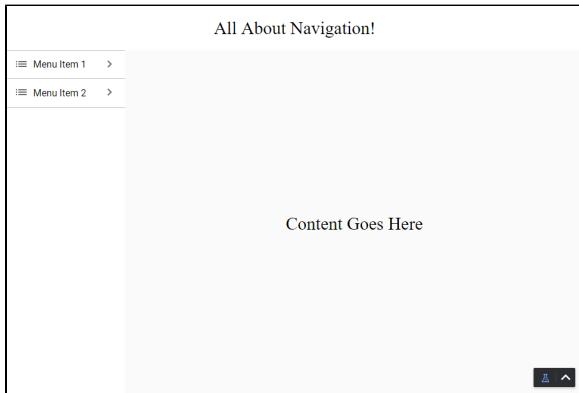
[In This Section ...](#)

Self-Hiding Navigation Drawer

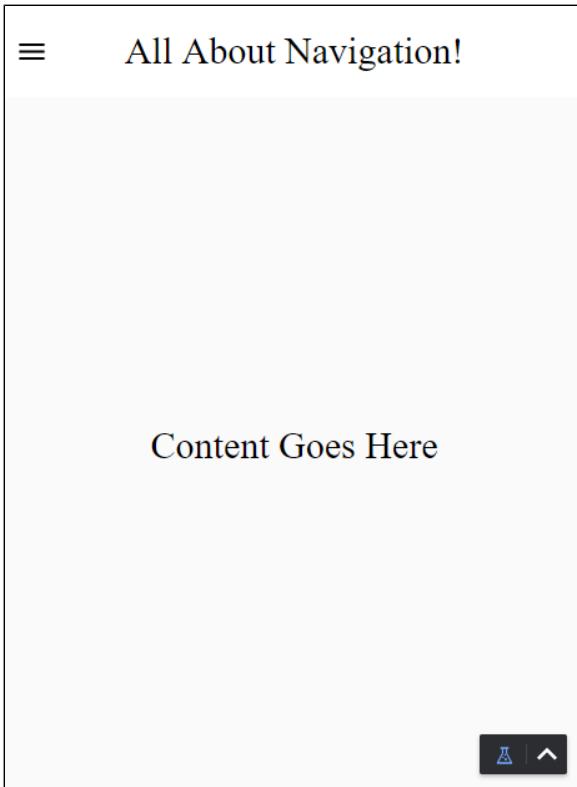
Navigation Drawer

A navigation drawer is a special type of docked menu, usually appearing on the left side of a session. What makes a navigation drawer special is its [responsive design](#). On smaller devices, this docked menu can hide itself and pop out when the user needs it. These drawers have become ubiquitous in User Interface (UI) design, particularly in apps.

Here's what one might look like on a computer monitor:



Here's what one might look like on a mobile device:



As the screen becomes smaller, the menu is hidden and an icon appears in the top left to allow us to toggle its visibility. This particular navigation drawer will probably need about 200 pixels horizontally, which on a desktop is fine, but on a mobile device takes up too much of the screen.

Note: This guide assumes a bit of knowledge about how views and components work. Please see those sections of this manual for more information as needed.

On this page ...

- [Navigation Drawer](#)
- [Configuring a Navigation Drawer](#)
 - [Navigation View](#)
 - [Header Views](#)
 - [Page Setup](#)
 - [Configure Menu Icon](#)

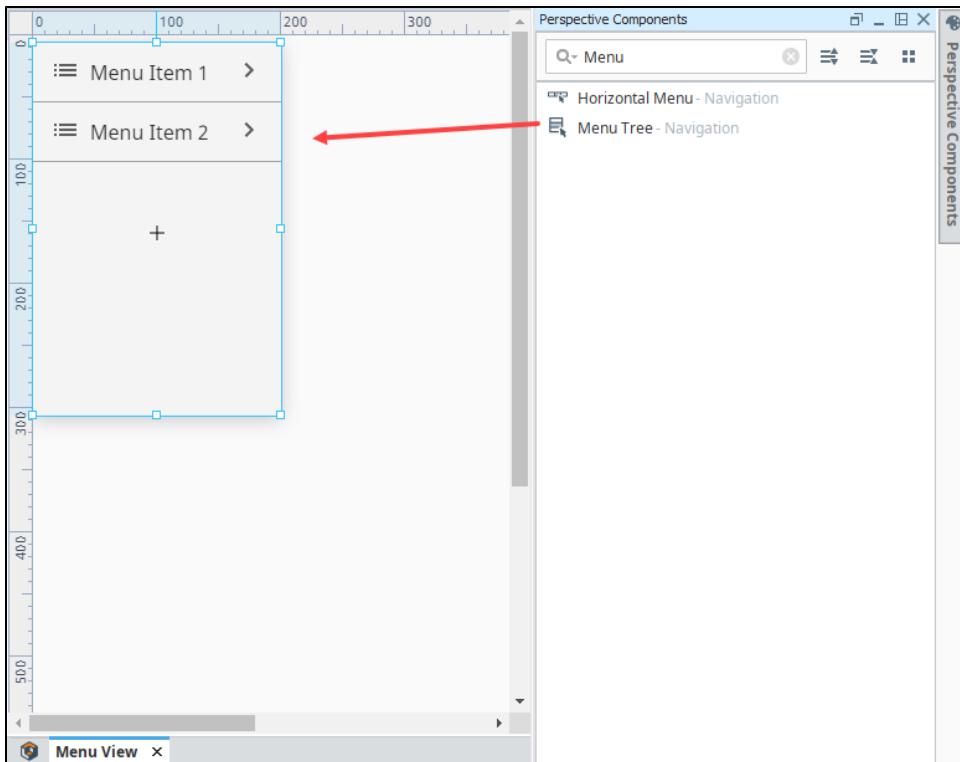
Configuring a Navigation Drawer

The following example walks through how to configure this self-hiding navigation drawer. There is no mention of setting up the Menu Tree or any content pages, it is strictly a guide to show you the layout type.

Navigation View

1. In the Project Browser, right click on **Views** and click **New View**.
2. Name the view **Menu View** and set the Root Container Type to **Coordinate**.
3. Click **Create View**.
4. Set the width of the view to **200 pixels**.
5. Drag a **MenuTree** component onto the **Menu View**. Configure the component as you would if you were using a standard docked view.

Note: You can set the root of the **Menu View** to use the Percent Mode. This way it is easy to make the **Menu Tree** fill all the space.

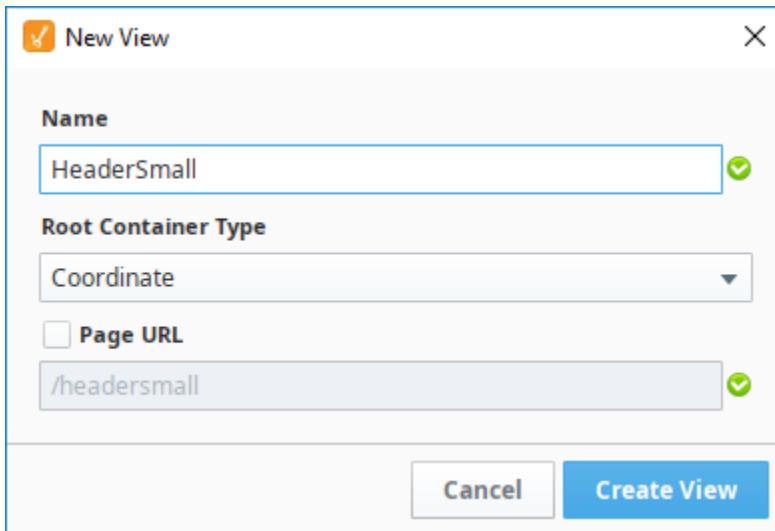


Header Views

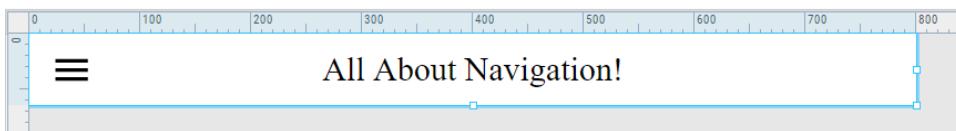
Next, we will create our header views. The two views will be setup in a **breakpoint** container, which will swap between a small header with an icon, and a big header without one.

1. Create a small header view for our mobile UI.
 - a. In the Project Browser, right click on **Views** and click **New View**.
 - b. Name the new view **HeaderSmall**, and set the Root container Type to **Coordinate**.

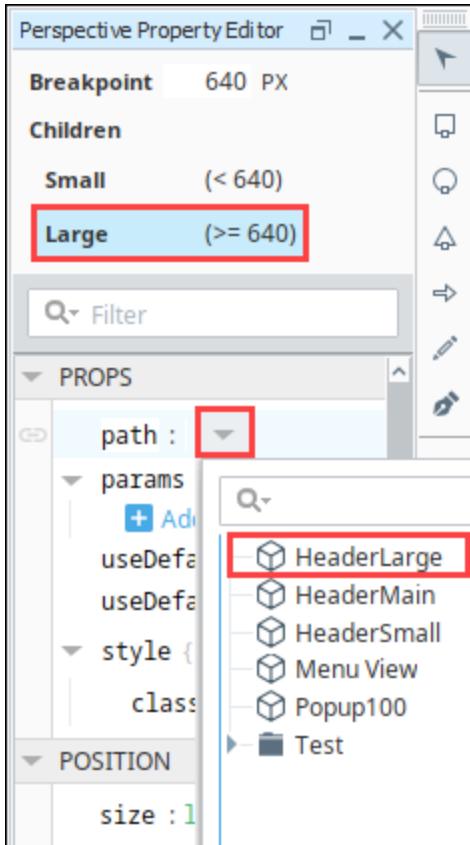
- c. Click **Create View**.



- d. Set the height of the view to **75 pixels**.
- e. Drag an Icon component onto the upper left side of the view, then click on the component to select it.
- f. In the Property Editor, set the path property to **material/menu**. You can of course use whatever icon you'd like; a list of all the icons in the material folder can be found [here](#).
- g. Add a title for the header. We used a Label component with the text "All About Navigation!"



2. Create a large header view for our desktop UI
 - a. Create another view called **HeaderLarge**, again with a layout of **Coordinate**.
 - b. Set the height to **75 pixels**.
 - c. Don't add the icon to this one, but add the same Label as above.
3. Finally, create a Breakpoint view to toggle between them.
 - a. Create a view called **HeaderMain** with a layout of **Breakpoint**.
 - b. Set the height to **75 pixels**.
 - c. In the Property Editor, click on **Large** (under Children).
 - d. Drag an Embedded View component on the HeaderMaster view.
 - e. Under PROPS, click on the **Expand ▾** icon next to the path property then select **HeaderLarge**.



- f. Now we'll do the same for the small child of this Breakpoint container. In the Property Editor, click on **Small** (under Children).
- g. Drag an Embedded View component on the HeaderMain view.
- h. In the Property Editor, click on the **Expand** ▼ icon next to the **path** property then select **HeaderSmall**.

Page Setup

Now that we have our header and menu views, we need to set up our pages to display the views properly.

1. Click on the **Perspective** icon in the bottom left of the Designer to access the Page Configuration menu. Select **Shared settings**. We're going to add our two docked views here, so they show up on every page.

Perspective [NewProject_SJP]

1.0.15-rc1 (b2020070813)

Create New View

Page Configuration

Shared settings

- / → Test/FirstView
- /breakpointtypeview → Test/BreakPointTypeView
- /columntypeview → Test/ColumnTypeView
- /drawingtypeview → Test/DrawingTypeView
- /flextypeview → Test/FlexTypeView
- /secondview → Test/SecondView
- /tabtypeview → Test/TabTypeView
- /thirdview → Test/ThirdView

Head

Content

left-right

Menu View

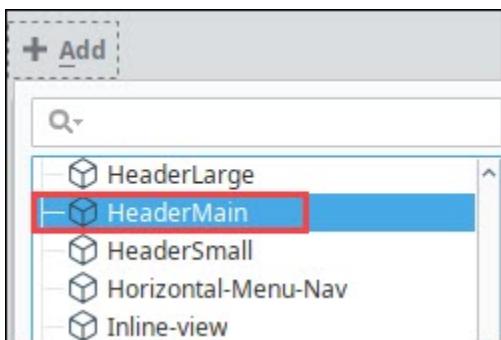
Add

Recently Modified Views

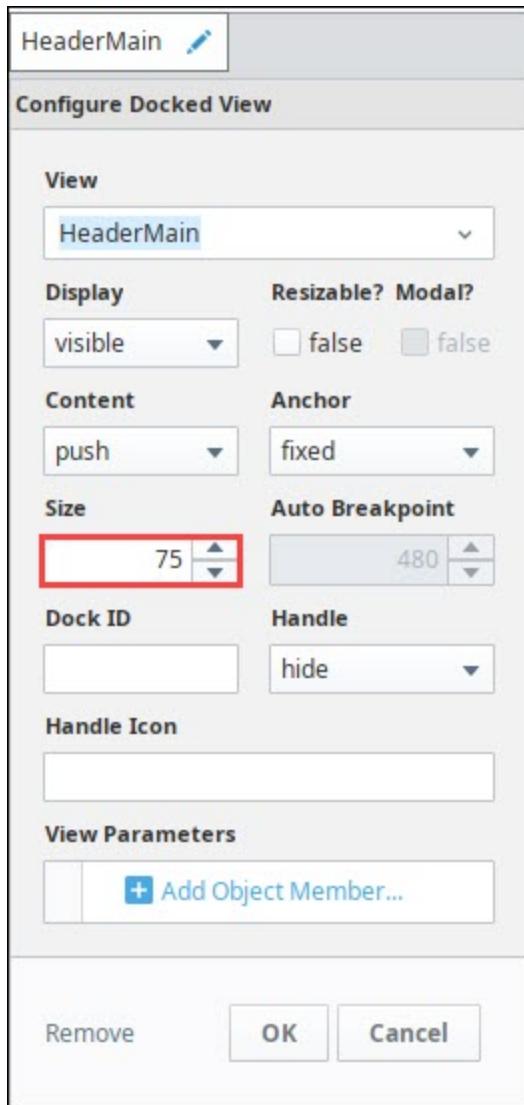
Menu View HeaderSmall HeaderLarge HeaderMain

This screenshot shows the 'Page Configuration' interface in a perspective view. On the left, a list of 'Shared settings' contains links to various test views like 'FirstView', 'BreakPointTypeView', etc. The main area is divided into sections: 'Head' at the top, 'Content' below it, and 'left-right' on the right. A 'Menu View' docked view is currently selected and visible. A 'Add' button is located in the bottom right of the configuration area. Below the configuration, a list of 'Recently Modified Views' includes 'Menu View', 'HeaderSmall', 'HeaderLarge', and 'HeaderMain'. The 'HeaderMain' view is highlighted with a red box.

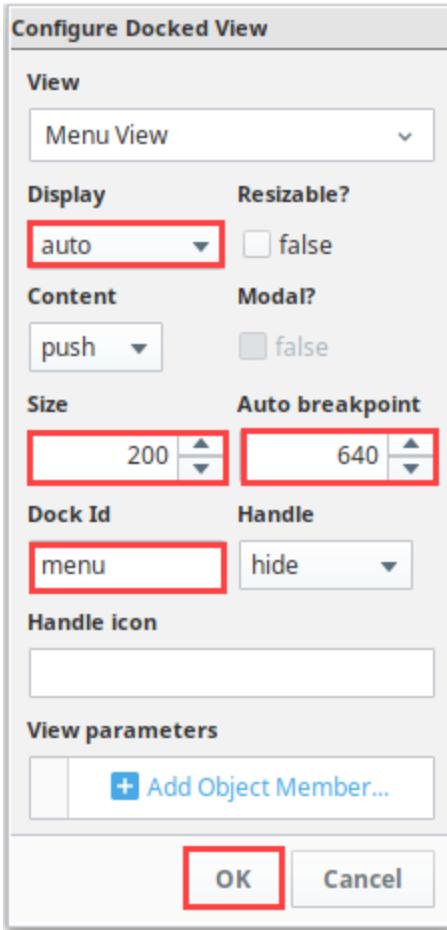
2. On the top dock, click the **Add** icon. Select the **HeaderMain** view and click **OK**.



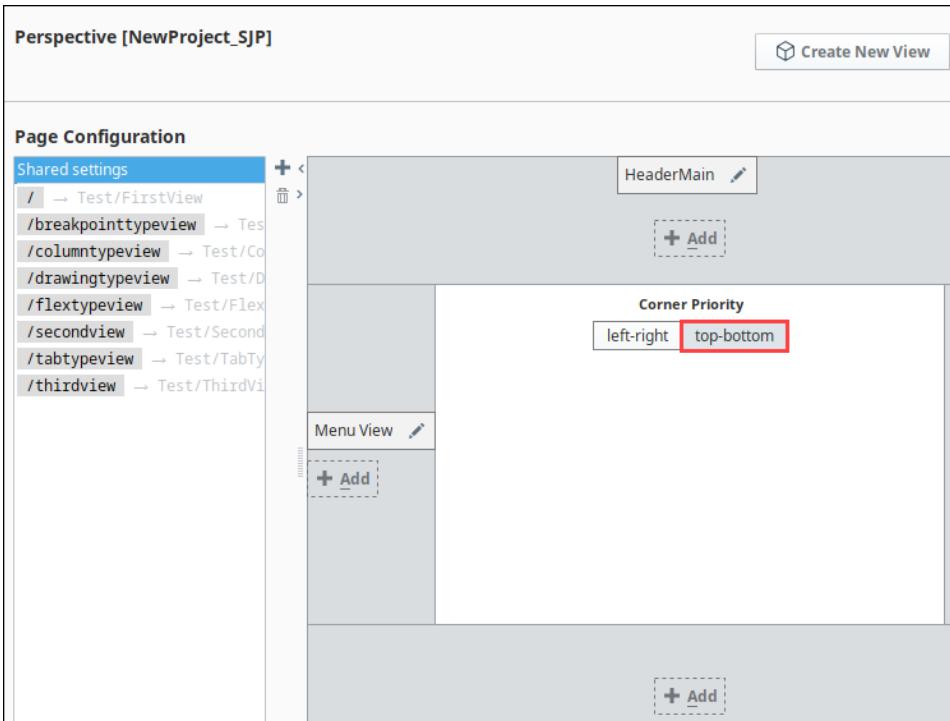
3. Click the **Edit** icon next to the HeaderMain docked view.
4. In the Configure Docked View menu, set the size to **75**.



5. On the left dock, click the **Add** icon. Add the **Menu View** and set the following:
Size: **200**
Display: **Auto** (This enables us to configure a breakpoint below)
Autobreakpoint: **640** (This is the same width that the breakpoint container on the HeaderMaster view is using)
Dock Id: **menu** (This will be used in our dock action on the menu icon to toggle the menu.)



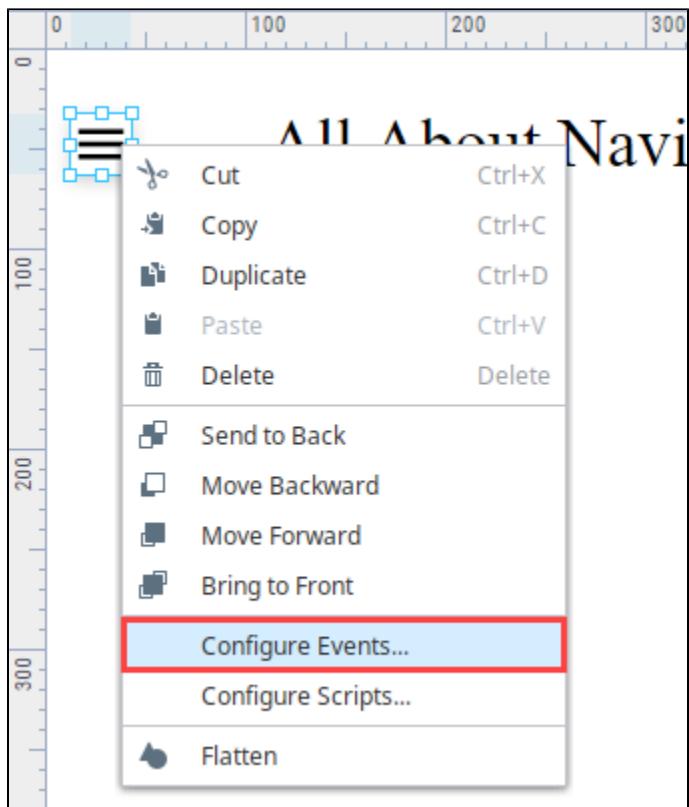
6. In the center of the Page Configuration menu, set the Corner Priority to **top-bottom**. The other option won't look quite right.



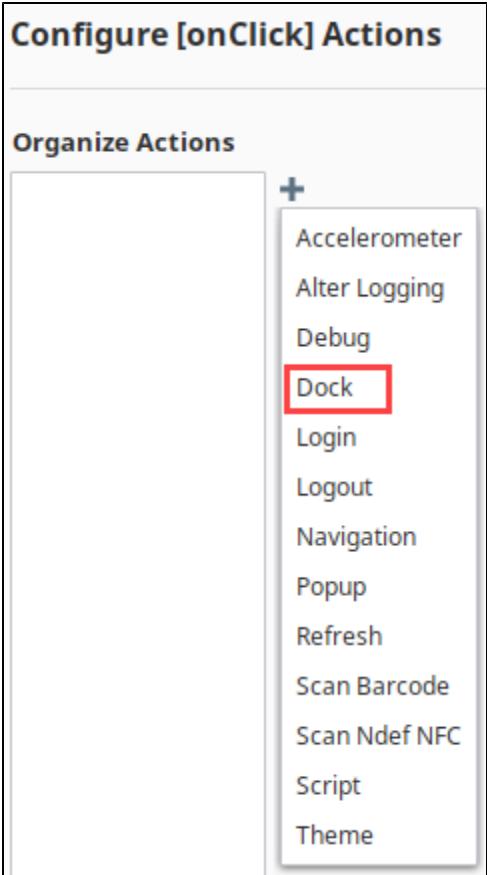
Configure Menu Icon

Now we need to configure the menu icon we created on HeaderSmall to pull up **Menu View**.

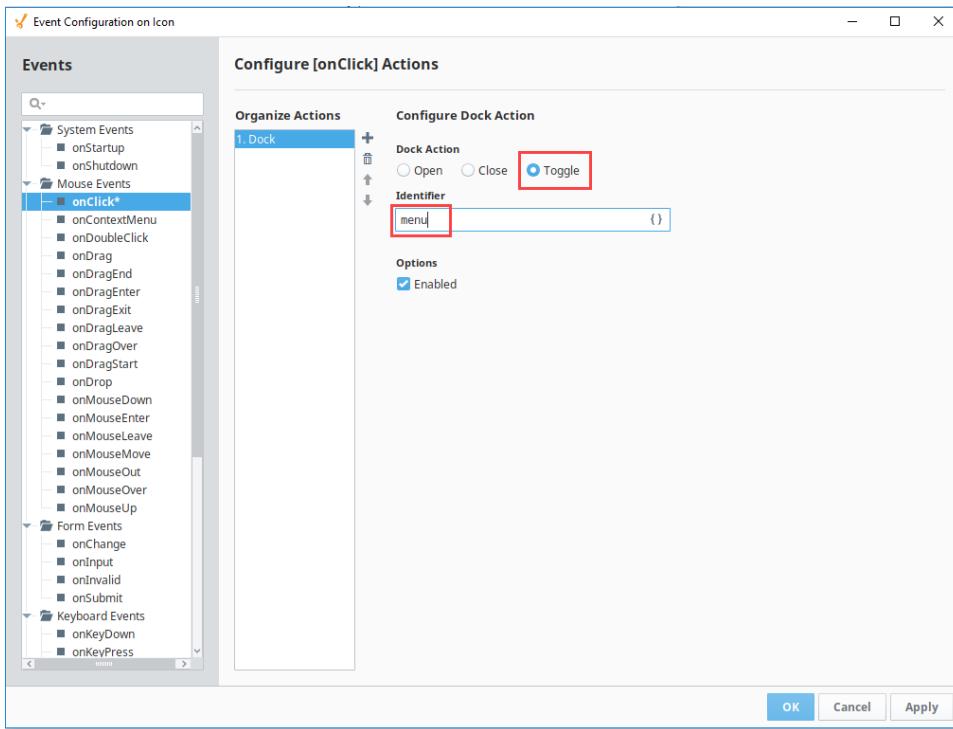
1. Open the HeaderSmall View.
2. Right-click on the Icon component, and select **Configure Events...**



3. Under Mouse Events, select **onClick**.
4. Next click the **Add +** icon to add an action. Select the **Dock** action.



5. This action needs the identifier we created. Set the Dock Action to **Toggle**, and the Identifier to **menu**. Click **OK**.



Now go test it out! It's easiest to open a browser on your desktop and change the size to toggle between the different views.

Using this strategy, you can configure a navigation drawer in combination with any basic navigation component or method.

OPC UA

OPC Specification

Open Process Connectivity (OPC) is a technology standard used to exchange information between hardware and software through specialized drivers.

Information is exchanged between items such as programmable logic controllers (PLCs), gauges, databases, and more. OPC is published and maintained by the OPC Foundation, an organization comprised of hundreds of member companies that strives to ensure interoperability on the plant floor and beyond.

The original OPC specifications used Microsoft Distributed Component Object Model (DCOM) technology to provide a uniform way for industrial applications to share data. There were several separate specifications that provided functions such as Data Access (OPC-DA), Alarms and Events (A&E), Historical data (HDA) and more.

DCOM always proved difficult to work with, and by 2004 it was clear that a more modern solution was needed. Therefore, a new specification was developed that used common networking principals (like TCP/IP) instead of DCOM, was platform independent, and combined the various separate specifications into one: Open Process Connectivity Unified Architecture (OPC UA).

OPC UA

OPC UA is the leading industrial standard for platform and vendor-neutral data access. Connecting any PLC device to Ignition is easy with OPC UA. Device connections are done over the Ethernet for those devices that have an Ignition device driver. The OPC UA Module makes Ignition act as an OPC UA server, serving data collected by its built in drivers to other Ignition modules, as well as to third-party OPC UA clients.

OPC UA is the latest revision of the OPC specification, which offers platform and vendor neutral transfer and use of industrial data. The specification plays a crucial role in Ignition, and is the primary data access specification used in the Gateway. Ignition supports connections to any number of OPC UA servers created by any manufacturer, provided that they are compliant to the specification. The data is then used to drive all aspects of the system. Creating connections to OPC UA servers is described below in [Connecting with OPC UA](#).

Distributed Systems with OPC UA

OPC UA breaks down boundaries and enables free data flow. Using standard TCP/IP instead of legacy DCOM, OPC UA makes it easy to securely transfer data between networks and though firewalls. All OPC UA connections are based on the same technology, which means that a connection to your local machine is not entirely different than a connection to a machine that's far away. This enables the creation of highly distributed system, and in combination with other features of Ignition can lead to much more connected enterprises.

For example, imagine a corporate network with an office in the center, and remote processes connected through a VPN, which would pass through a variety of connections. Each remote site could have an Ignition installation running only an OPC UA module that would report data back to a central facility and record it in a database. The overall system cost would be very low. The data could be managed centrally in a single location, and then made available to all interested parties through the Vision module or any application that could access the database.

Servers and Clients

When discussing OPC (as the specifications are often called collectively), it is common to hear about OPC **servers** and OPC **clients**. An OPC server is a piece of software that implements the OPC interface and provides data. An OPC client is an application which connects to an OPC server and uses the specification to retrieve and work with data.

The Ignition platform inherently offers OPC UA client functionality. Even with no modules installed, the Gateway can connect to any compliant OPC UA server and work with data. With the addition of the OPC UA module, Ignition becomes an OPC server as well, hosting device drivers that read and publish data.

The OPC COM module is available to provide client access to older, DCOM based, OPC-DA servers.

Technology

On this page ...

- [OPC Specification](#)
- [OPC UA](#)
- [Distributed Systems with OPC UA](#)
- [Servers and Clients](#)
- [Technology](#)
- [Connecting with OPC UA](#)
 - [Connecting to a Device](#)
 - [Connecting to a Server](#)
- [OPC Quick Client](#)
- [How Do I Get Data from My PLC?](#)
 - [Brief Summary of Device Connection in Ignition](#)
 - [Adding a Device to Ignition](#)
 - [Adding Connection to Third Party OPC Server Via OPC UA](#)
 - [Adding Connection to Third Party OPC Server Via OPC COM](#)

Steve Hechtman's Blog

Click the link below to see the blog:

[OPC UA goes mainstream](#)

The OPC UA specification offers a wide range of flexibility in choosing technologies, from the transport mechanism, to the way data is encoded, to the encryption used to secure the data. Ignition supports the UA/TCP transport with the UA/Binary encoding scheme for maximum performance.

Additionally, Ignition supports all of the common encryption schemes. This means that Ignition connects to OPC UA servers (and allows connections from clients) over TCP/IP using encryption, and sends data by first encoding it into an efficient format defined by the OPC UA specification. This is in contrast to other schemes outlined in the specification, which can use web services and XML encoding that are not as efficient.

Connecting with OPC UA

Connecting to a Device

With the Ignition OPC UA module and device drivers installed, connecting to a devices is simple. To quickly get connected to one of your devices, go to the **Config** tab of the Ignition Gateway and scroll down to **OPC UA > Device Connections**. The Device page will appear showing all of your installed devices.

To add a new device, click on **Create New Device**.... Ignition has several device drivers available, and the Device Type list is populated based on what Driver modules you have installed. Select the type of device connection you want and fill in the details for your device connection. See the device types below for more information on connecting to your specific device type:

- [Allen Bradley Ethernet](#)
- [Modbus](#)
- [Siemens](#)
- [UDP and TCP Driver](#)
- [DNP3](#)
- [Omron NJ Driver](#)
- [Simulators](#)

The screenshot shows the Ignition configuration interface. The left sidebar is dark blue with white text, showing navigation links for Home, Status, Config (which is selected and highlighted in orange), Networking, Security, and Databases. The main content area has a light gray background. At the top right are 'Help ?' and 'Get Designer' buttons. Below them is a breadcrumb trail: 'Config > Opcua > Devices'. The main content area lists seven device connection options, each in a separate box:

- Allen-Bradley CompactLogix (Legacy)**
Connect to CompactLogix firmware v20 and prior processors.
- Allen-Bradley ControlLogix (Legacy)**
Connect to ControlLogix firmware v20 and prior processors.
- Allen-Bradley Logix Driver**
Connect to Allen-Bradley Logix family devices, including devices with firmware v21+.
- Allen-Bradley MicroLogix**
Connect to MicroLogix 1100 and 1400 series PLCs.
- Allen-Bradley PLC5**
Connect to PLC5s via Ethernet.
- Allen-Bradley SLC**
Connect to SLC 5/05s via Ethernet.
- DNP3 Driver**
Connect to a DNP3 outstation.

Connecting to a Server

If you don't see the type of device you want, then you can always connect with another OPC UA or OPC DA server.

- [Third Party OPC Servers](#)
- [OPC COM](#)

OPC Quick Client

After you connected to a device, you can access the OPC Quick Client in the Gateway. It allows for quick, simple testing of any devices connected to the server.

You can find the Quick Client under the **OPC Connections** section of the Ignition Gateway **Config** tab. You can browse by expanding the tree nodes and read from or write to OPC tags by clicking on the [r] and [w] buttons next to those tags.

Subscriptions can be made by clicking on the [s] button. Clicking on the **enable live values** link will automatically refresh subscriptions and show live value changes (if there are any).

The screenshot shows the Ignition Gateway configuration interface. The left sidebar has a dark blue theme with sections for SYSTEM, NETWORKING, SECURITY, DATABASES, and ALARMING. The 'Config' icon is highlighted. The main area is titled 'OPC Quick Client' and shows a hierarchical tree view of OPC tags. A table below the tree lists tags with columns for Type, Action, and Title. Some tags have [s][r][w] buttons next to them. At the bottom, there's a 'Subscription 1' configuration panel with fields for Subscription name (Subscription 1), Rate (ms) (1000), and a Set button.

TYPE	ACTION	TITLE
Server	refresh	Ignition OPC UA Server
Object		Server
Object		ServerConfiguration
Object		ServerCapabilities
Object		ServerDiagnostics
Object		VendorServerInfo
Object		ServerRedundancy
Object		Namespaces
Tag	[s][r][w]	ServerArray
Tag	[s][r][w]	NamespaceArray
Tag	[s][r][w]	ServerStatus
Tag	[s][r][w]	ServiceLevel
Tag	[s][r][w]	Auditing
Tag	[s][r][w]	EstimatedReturnTime

How Do I Get Data from My PLC?

Getting data from your PLC into Ignition is a two step process:

1. Add a device, see [Connecting to a Device](#).
2. Add some tags, see [Creating Tags](#).

It requires you to touch both the Ignition Gateway and the Ignition Designer. There are also some limitations as to what kind of devices you can connect to Ignition and these are explained throughout the user manual, however, included below is an overview of what you can expect when it comes to compatibility.

Brief Summary of Device Connection in Ignition

- Ignition can only connect directly to devices over Ethernet.
- Ignition can only connect directly to devices for which there is an Ignition device driver. Visit the [OPC UA Drivers](#) page for an overview of supported devices and drivers.
- Ignition can connect to third party OPC servers via OPC UA or OPC-DA (using the OPC-COM module) for devices that do not have a supported driver.

Adding a Device to Ignition

Ignition Supported OPC UA Device

Most commonly you will be adding a device that is supported by one of the built-in device drivers. The first step is connecting your device to Ignition. This is done through the Ignition Gateway Config tab under the OPC UA > Device Connections page.

1. Click **Create new Device...**
2. Select the driver for the device you wish to add. Click **Next**.
3. When adding a device you will notice that there are some common settings that are shared by all devices. You can find an explanation of these settings on the individual page for each [OPC UA Driver](#).
4. Specify any of the required device specific settings for the device (for example, hostname, etc.)
5. Check the status of your device to see if it is connected.

As long as all the device information you entered was correct you should see your device in a connected state. The only exception to this is if you chose to add a Siemens or Modbus device. Since these devices don't support the browsing of Tags, you will have to create and address some Tags in the Ignition Designer before the device will stop cycling from a connected to disconnected state.

If you need to address your Tags for your Siemens or Modbus device, you'll want to read about adding Tags in the Ignition Designer as well as how addressing works for the different protocols. You will have to first add a Tag in the Ignition Designer and then edit the OPC Item Path of the Tag using the appropriate addressing scheme.

Adding Connection to Third Party OPC Server Via OPC UA

If your device does not have an Ignition driver, you can use a third party OPC server to connect to your device and then have Ignition connect to the server as a client. If the OPC server offers support for OPC UA, you can add a new OPC UA server connection in the Ignition Gateway. For more information, see the [OPC UA Client Connection Settings](#) page. Additionally, the [Connecting to Kepware OPC UA](#) is useful when connecting to Kepserver.

Adding Connection to Third Party OPC Server Via OPC COM

When attempting to connect Ignition to an OPC COM server, the [OPC COM](#) module must be installed. More information about configuring OPC COM connections can be found on the [OPC COM](#) page.

OPC UA Client Connection Settings

Configuring an OPC Client Connection

An OPC UA connection is used to communicate with an OPC UA compliant server, such as the one the [OPC UA Module](#) provides.

The following steps walk through connecting Ignition (as an OPC UA client) to a OPC UA server.

On this page ...

- [Configuring an OPC Client Connection](#)
- [OPC UA Client Connection Settings](#)
 - [Failover Versus Backup Properties](#)

1. On the Config tab of the Gateway Webpage, go to **OPC Client > OPC Connections**. The OPC Server Connections page is displayed.
2. Click on the **Create new OPC Server Connection** link.

The screenshot shows the 'OPC Connections' page with a single entry:

Name	Type	Description	Read Only	Status	More	edit
Ignition OPC UA Server	OPC UA	A "loopback" connection to the Ignition OPC UA server running on this gateway.	false	Connected	More	edit

Below the table, there is a link to 'Create new OPC Connection...' and a note: 'Note: For details about a connection's status, see the [OPC Connection Status](#) page.'

3. Select **OPC UA Connection** from the list and click **Next**. The Endpoint Discovery page appears.
4. Enter an OPC UA endpoint URL for the OPC UA server Ignition should connect to. The format should be as follows:

```
opc.tcp://IpAddress:Port
```

Instead of an IP address, a host name can be used:

```
opc.tcp://myServer:12345
```

An "Advanced Configuration" link was added to this flow, allowing you to manually configure connection settings. This is useful in cases where a server doesn't allow anonymous endpoint access, but provides separate discovery endpoints.

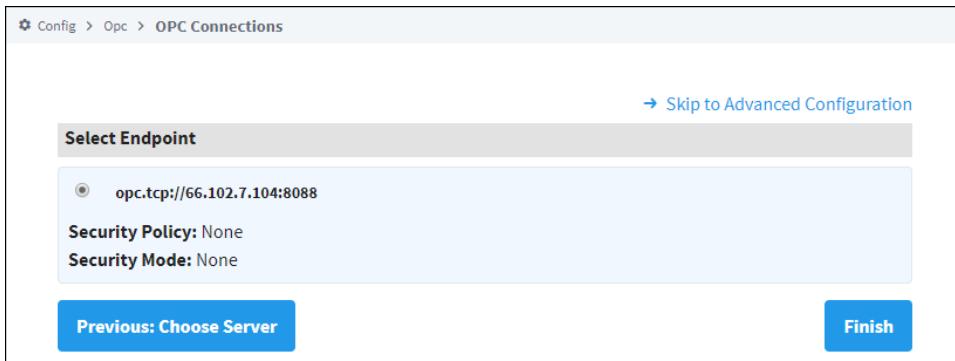
5. Choose a server, then click **Next: Select Endpoint**.

The screenshot shows the 'Choose a Server' step of the configuration flow. It displays a list of servers:

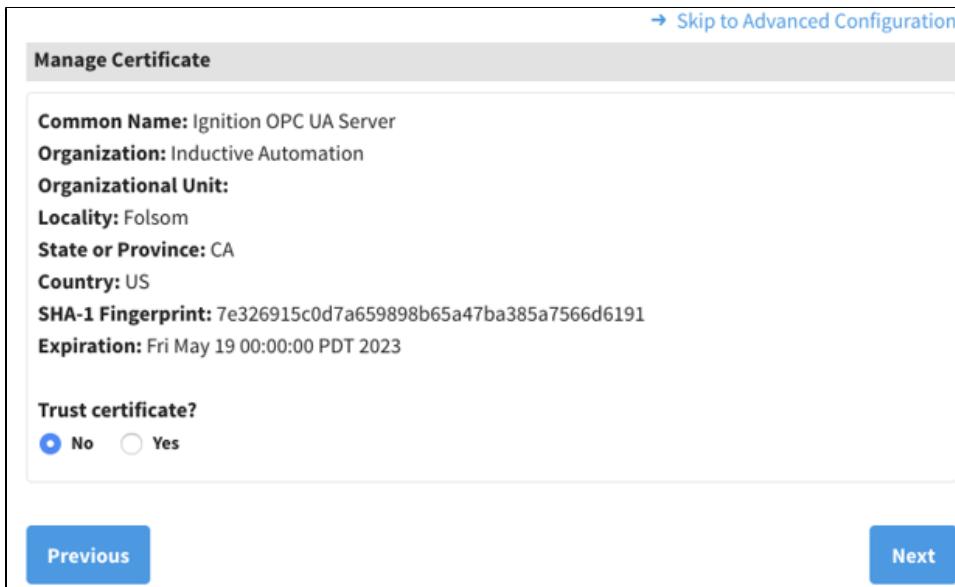
- Ignition OPC UA Server
- opc.tcp://66.102.7.104:8088

At the top right, there is a link to 'Skip to Advanced Configuration'. At the bottom, there are two buttons: 'Previous: Endpoint Discovery' and 'Next: Select Endpoint'.

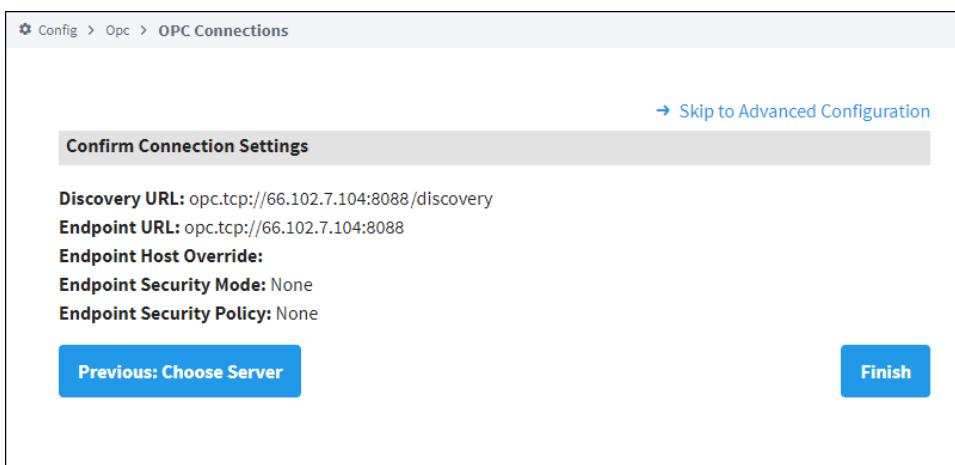
6. Click **Next: Select Endpoint**. A list of available Security Policies and Message Security options will appear.



7. A Manage Certificate window will open if you haven't previously trusted a certificate. If the certificate that the Endpoint sends you to is already in Ignition's trust store, this step is skipped. Click **Next**.



8. If you entered a discovery URL in step 4, you also have an option to enter another URL if the host is unreachable.
9. Select a Security Policy and Message Security configuration to use when connecting to the endpoint, then click **Finish**. The policies that appear here are determined by the server.
10. A confirmation page is displayed. Click **Finish**.



11. On the **New OPC UA Connection Settings** page, give the connection a name. Some OPC UA servers may require a username and password, but this is not always the case. Check with the OPC UA server's documentation for more details. Credentials for Ignition's OPC UA server can be found on the [OPC UA Server Settings](#) page.

The screenshot shows the Ignition configuration interface for creating a new OPC connection. The 'Main' tab is active, displaying fields for 'Name' (set to 'New Name OPC Server'), 'Description' (empty), 'Enabled' (checked), and 'Read Only' (unchecked). The 'Authentication' tab is also visible, showing fields for 'Username' (empty), 'Password' (empty), and 'Re-type password' (empty). At the bottom, there is a 'Create New OPC Connection' button.

12. Once credentials have been entered, click the **Create New OPC Connection** button.

Ignition is now connected to the OPC UA server.

OPC UA Client Connection Settings

The following table describes all the available properties.

Main	
Name	A name used to identify this connection.
Description	Short description of this connection.
Enabled	Disable the connection to the OPC server.
Read-only	Puts the connection into read-only mode. All writes sent to this server will fail.
Authentication	
Username	A username the connection will use when authenticating with the UA server.
Password Fields	The password to use when authenticating with the UA server.
Advanced	
Host Override	When specified, if the endpoint address returned by the OPC server has a different IP address or hostname than the discovered endpoint, the overridden value will be used. Expects just an IP address or hostname, for example: 192.168.1.10
Connect Timeout	The timeout, in milliseconds, when opening a socket connection to a remote host. Default is 5,000.
Acknowledge Timeout	The timeout, in milliseconds, to wait for an Acknowledge message in response to the client's Hello message. Default is 5,000.
Request Timeout	Maximum amount of time, in milliseconds, to wait for the response to a request. Default is 120,000.

Session Timeout	Requested session timeout value, in milliseconds. Default is 120,000.
Max Per Operation	Specify the maximum number of nodes to read, write, subscribe, or unsubscribe to in any given UA server request. Default is 8,192.
Max References Per Node	<p>Configures the number of references per node. A "node" in this case is any item inside of a UA server, so items like tags and folders would qualify as a node, while a References is simply a reference to another node. This setting is useful in situations where the address space is completely flat, so a large number of adjacent nodes could potentially run into a maximum message size. In these cases increasing the value of this property can be useful.</p> <p>However, most systems will not need to change this setting. Defaults to 8,192 references.</p>
Max Pending Publish Requests	The number of concurrent Publish Requests allowed to be pending at any given time. Default is 2.
Max Notifications Per Publish	The maximum number of notifications per publish. Default is 65,535.
Max Message Size	The maximum allowable size of an OPC UA application layer message. Default is 33,554,432.
Max Array Length	The maximum allowable size for arrays. Default is 2,147,483,647.
Max String Length	The maximum allowable size for strings. Default is 2,147,483,647.
Type Dictionary Fragment Size	The fragment size to request when reading the server's type dictionary. Default is 8,192.
Keep-Alive Failures Allowed	Number of consecutive failures allowed before disconnecting. Setting this to <= 0 means consecutive failures will not cause a disconnect. Default is 1.
Keep-Alive Interval	Interval, in milliseconds, between keep-alive requests.
Keep-Alive Timeout	Max duration, in milliseconds, to wait for a response to a keep-alive request. Default is 10,000.
Browser Origin	The Node that browsing should originate from. Options are OBJECTS_FOLDER or ROOT_FOLDER. Most OPC UA Servers use OBJECTS_FOLDER, but some non-standard servers may require ROOT_FOLDER to browse correctly. Ignition's OPC UA Servers uses OBJECTS_FOLDER.
Failover	
Failover Enabled	Enable failover on the connection, allowing the UA client to switch to a backup server in the event the primary server is unavailable.
Failover Threshold	The number of retry attempts before the failover connection is used. The default is 3.
Failover Discovery URL	<p>The discovery URL for the backup server's OPC UA server. Expects the following format:</p> <pre>opc.tcp://hostname:port</pre>
Failover Endpoint URL	<p>The endpoint of the failover server. Example:</p> <pre>opc.tcp://192.168.1.0:62541</pre>
Failover Host Override	When specified, if the endpoint address returned by the failover OPC server has a different IP address or hostname than the discovered endpoint, the overridden value will be used. Expects just an IP address or hostname. Example: 192.168.1.10
Security	
Certificate	

Validation Enabled	<p>This feature is new in Ignition version 8.1.0 Click here to check out the other new features</p> <p>Enables validation of server certificates. This is required by the OPC UA specification, but it may be disabled for troubleshooting or temporarily connecting to servers with invalid or untrusted certificates. Default is true.</p> <p>Caution: Disabling certificate validation compromises the security of the connection.</p>
Keystore Alias	The alias of the certificate and private key stored in the client key store.
Password Fields	The password to use when authenticating with the UA server.

Failover Versus Backup Properties

The Failover properties should be used when a single Ignition Gateway needs to connect to a pair of redundant OPC UA servers. The failover OPC UA server will be used in the event the primary OPC server goes down. To enable failover, set the **Failover Enabled** property to true, and specify the **Failover Endpoint**. The **Failover Threshold** can be adjusted if desired.

Note: Failover events are "sticky." That means once control has moved to a backup OPC UA server, it stays there until that server fails.

Related Topics ...

- [OPC UA Server Settings](#)
- [Tag Browser](#)

OPC UA Server Settings

Ignition's OPC UA server, provided by the [OPC UA module](#), allows an ignition installation to utilize Ignition's various device driver modules. In addition, with the module installed, OPC UA clients can connect to Ignition's UA server, exposing any connected devices to 3rd party systems.

Settings for the server can be found under the **Config** section of the Gateway Webpage. On the sidebar, locate **OPC UA > Server Settings**.

Default Credentials

Ignition's OPC UA server does not initially support anonymous access, but can be configured to do so (see the settings table below). Authenticated connection require the following credentials:

Username	opcuauser
Password	password

New installations of Ignition will automatically create the user above, allowing the Gateway to initially connect as a UA client to its own UA server.

Connecting with UA Discovery

Ignition's OPC UA server is initially, and intentionally, difficult to discover on new installations. To aid with discovery attempts, a separate unsecured endpoint is available, allowing UA clients a means of finding the server. When attempting to discover the server, the endpoint UR should include "/discovery" at the end:

```
opc.tcp://192.168.2.134:62541/discovery
```

OPC UA Server Settings

Note: Changes made to any of the following OPC UA Server settings requires a restart (of either the Gateway or the OPC UA module) before they changes will take effect.

The table below represents settings on Ignition's OPC UA server. They'll only become available if the [OPC UA Module](#) is installed on the Gateway.

Setting	Description	Default value
Endpoint Configuration		
Bind Port	The port the UA server will bind to.	62,541
Bind Addresses	The address the server will bind to. If you want to expose the OPC UA server to external sources, you need to use 0.0.0.0 or the IP address of the computer.	localhost
Endpoint Addresses	A common separated list of endpoint addresses that the UA server can be reached at. It is important that this is set to addresses that can be reached by any UA clients attempting to connect to the server. When entering addresses into this property, they can be just an IP address or hostname: <code>10.10.10.100</code>	<hostname>, <localhost>
	Alternatively, angled brackets can be used. When applied to an address, the server attempts to find the hostname, or resolve the value to as many addresses or hostnames as it can find. <code><10.10.10.100></code>	

Security Policies	<p>A comma separated list of acceptable security policies. Available policies are:</p> <ul style="list-style-type: none"> • None • Basic256Sha256 • Aes128_Sha256_RsaOaep • Aes256_Sha256_RsaPss <p>In addition, the following deprecated policies are available, but not recommended:</p> <ul style="list-style-type: none"> • Basic128Rsa15 (deprecated) • Basic256 (deprecated) 	Basic256Sha256
-------------------	--	----------------

Authentication

Anonymous Access Allowed	Specifies if UA clients are allowed to connect to this server anonymously. While false, client connections are required to authenticate with the server.	false
User Source	Which user source contains the initial user for authenticated access. Credentials for the initial user can be found above.	Attempts to use the 'opcua-module' user sources

Advanced

Expose Tag Providers	When enabled, Ignition Tag Providers will be exposed through the UA server, allowing third-party UA clients to access tags in the provider.	false
----------------------	---	-------

Redundancy

Backup Bind Addresses	The local addresses that the UA server will attempt to bind to while the backup server in a redundant pair .	localhost
Backup Endpoint Addresses	The endpoint addresses that the UA server can be reached at while the server is configured as the backup in a redundant pair . The notation on this property is similar to the Endpoint Addresses property above, in that angled brackets can be used with each hostname and IP address.	<hostname>, <localhost>
Read-only When Inactive Node	When enabled, this server switches to a read-only state while its Gateway is the inactive node in a redundant pair .	false

Related Topics ...

- [OPC UA Client Connection Settings](#)
- [Tag Browser](#)

Third Party OPC Servers

Ignition has a built-in OPC UA server that includes all of the drivers in this section, but if your devices aren't listed here, there is another way to connect to them. Ignition can quickly and easily connect to third party OPC servers via OPC UA or OPC DA (using the OPC COM module) for devices that do not have a supported driver. This opens up the possibilities to connect Ignition to any device with servers like Kepware, Matricon, etc. With over 100 driver suites each, you are sure to be able to connect anything to Ignition.

OPC Server Connections

Name	Type	Description	Read-only	Status	More	Edit
Ignition OPC-UA Server	OPC-UA	The default connection to Ignition's OPC-UA server.	false	Connected	More	edit
Kepware	OPC-UA		false	Connected	More	edit

[→ Create new OPC Server Connection...](#)

Note: For details about a connection's status, see the [OPC Connection Status page](#).

OPC UA

OPC UA is the new Unified Architecture for connecting to OPC devices. Probably the best feature of this new standard is that it can be used on any OS, it is not tied to Microsoft like its predecessor is. Many companies make an OPC UA server, and between them grant access to hundreds of driver suites, giving you access to thousands of available devices through Ethernet, serial, and other types of communication. If Ignition doesn't have it already, you can still get your data in.

Connecting to an OPC-UA server is simple (locally or remote), but because of the increased security, you need access to both the OPC server and Ignition to complete the setup. For step-by-step instructions to connect to a Kepware OPC server, see [Connecting to Kepware OPC UA](#). All other OPC UA servers will have extremely similar connection steps.

OPC COM (DA)

[OPC COM](#) (often called OPC-DA) has been the way to connect to OPC devices for many years, but the world has since out-grown it with the creation of OPC-UA. It relies on the .NET framework and because of this is restricted to Microsoft operating systems. There is no way to install an OPC COM server on Mac or Linux. There are many still in use today so it cannot be abandoned completely yet, but Ignition has an [OPC COM module](#) to connect to them.

OPC COM servers are notoriously difficult to connect to, especially if you want to connect to one on a remote server. If you need to do so, using Ignition's [OPC COM Tunneller Module](#) is highly preferred over setting up a Remote COM connection.

In This Section ...

Connecting to Kepware OPC UA

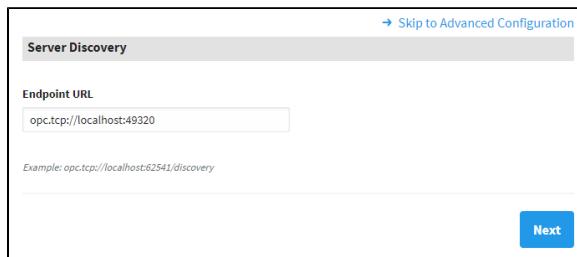
OPC UA makes connecting to third party OPC servers quick and easy without all the headaches associated with COM. This is a detailed step-by-step guide to connecting to KEPServerEX from Ignition using OPC UA.

Connect to KEPServerEX from Ignition using OPC UA

1. In the **Config** section of the Gateway, go to **OPC Client > OPC Connections**. The OPC Connections page is displayed showing the OPC UA servers your Ignition is connected to.
2. Click on **Create new OPC Connection....**
3. Choose **OPC UA** as the connection type, and click **Next**.
4. On the Server Discovery page, enter the endpoint of the OPC UA server Ignition should connect to. For example:

Sample Format	Localhost Example, Default Port	Remote Example, Custom Port
opc.tcp://IpAddress: Port	opc.tcp://localhost:493 20	opc.tcp://10.1.1.10:44 44

Click the **Next** button to continue.



Server Discovery → Skip to Advanced Configuration

Endpoint URL
opc.tcp://localhost:49320

Example: opc.tcp://localhost:62541/discovery

Next

On this page ...

- [Connect to KEPServerEX from Ignition using OPC UA](#)
 - Troubleshooting
 - Failover
 - No Anonymous Token Policy Found
- [Other UA Servers](#)
 - [Download a Client Certificate Manually](#)



Connecting to Kepware OPC-UA

[Watch the Video](#)

5. Select the Server you want and click **Next**.

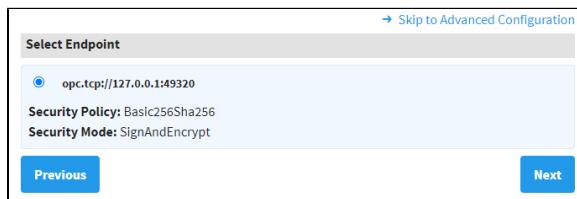


Choose a Server → Skip to Advanced Configuration

KEPServerEX/UA@TR-0690157-SB.ia.local
 opc.tcp://127.0.0.1:49320

Previous Next

6. A list of available Endpoints with Security Policies and Security Modes options appears.



Select Endpoint → Skip to Advanced Configuration

opc.tcp://127.0.0.1:49320

Security Policy: Basic256Sha256
Security Mode: SignAndEncrypt

Previous Next

Once an endpoint configuration has been selected, click the **Next** button.

7. On the Manage Certificate page select **Yes** for **Trust Certificate?** and click Next.

Common Name: KEPServerEX/UA Server
Organization: Unknown
Organizational Unit:
Locality:
State or Province:
Country: US
SHA-1 Fingerprint: 60df905dc826cbce78594cd590428e37c5e2bef1
Expiration: Sat Jun 24 15:36:00 PDT 2023

Trust certificate?
 No Yes

[Previous](#) [Next](#)

8. Confirm your settings and click **Finish**.

Discovery URL: opc.tcp://127.0.0.1:49320
Endpoint URL: opc.tcp://127.0.0.1:49320
Endpoint Host Override:
Endpoint Security Mode: SignAndEncrypt
Endpoint Security Policy: Basic256Sha256

[Previous](#) [Finish](#)

9. This takes you to the new OPC Connection screen. Fill in the **Username** and **Password** if your KEPServer connection requires it.

Most current installations of KEPServer require a login and will not connect without one. See the [No Anonymous Token Policy Found](#) section below.

Main	
Name	KEPServerEX/UA
Description	
Enabled	<input checked="" type="checkbox"/> (default: true)
Read Only	<input type="checkbox"/> If selected, the connection to this OPC server will be read-only; all calls to write will fail. (default: false)

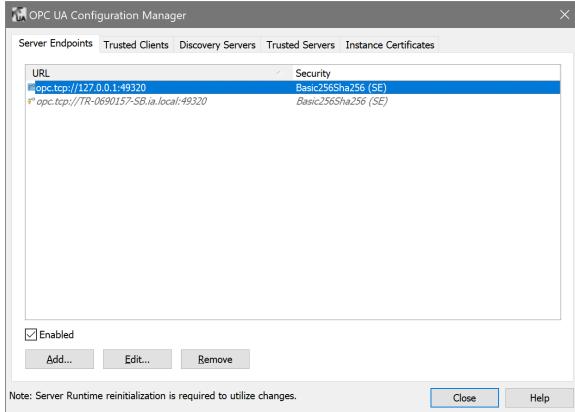
Authentication	
Username	
Password	
Re-type Password	

Show advanced properties

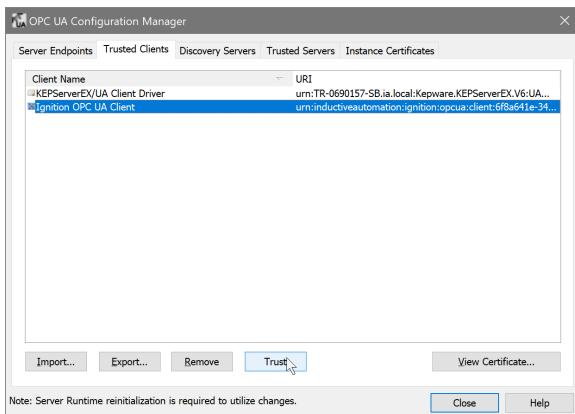
[Create New OPC Connection](#)

10. Click **Create New OPC Connection**.

11. The connection will appear as **Faulted**. This is expected because KEPServerEX is denying access to the Ignition OPC UA Client. The next step is to have KEPServerEX trust the Ignition OPC UA Client.
12. On the computer that KEPServer is installed on, right-click on the **KEPServerEX** icon on the desktop KEPServerEx is installed on, and from the menu select **OPC UA Configuration**. The **OPC UA Configuration Manager** will appear.



13. On the OPC UA Configuration Manager window, go to the **Trusted Clients** tab.
14. Click on **Ignition OPC UA Client**, click the **Trust** button, and click **Close**. Now the OPC Server Connections page shows the Status of Kepware to be Connected.



15. Again, right-click on the **KEPServerEX** icon on the desktop KEPserverEx is installed on, and from the menu select **Reinitialize**.
16. Go back to the Ignition Gateway Webpage. In the **Config** section, go to the **OPC UA > Security** page.
17. Under the Client Security tab, you will find your new connection listed as Quarantined. Click on the **Trust** button on the far right.

Common Name	SHA-1 Fingerprint	Expiration	Action
Ignition OPC UA Ser...	fd012b875173fa53e...	Jun 24, 2023	Delete
KEPServerEX/UA Ser...	60df905dc826cbce7...	Jun 24, 2023	Delete

Common Name	SHA-1 Fingerprint	Expiration	Action
KEPServerEX/UA Ser...	60df905dc826cbce7...	Jun 24, 2023	Trust

18. Go back to the **Config** section of the Gateway, to OPC Connections > Servers. The Status of your KEPserver connection should be **Connected**.
19. To test your tag connections, go to the OPC Connections > Quick Client in the Configure section of the Gateway. Expand the **KEPserver** object until you find tags.

Troubleshooting

If **Status** does not read **Connected**, click the **edit** link next to the server connection, scroll down to the bottom of the connection configuration page, and click **Save**. If **Status** is still reading something other than **Connected**, click the **OPC Connection Status** link at the bottom of the **OPC Server Connections** page and see if there are any useful messages to help troubleshoot the issue. Also, ensure your firewall is not blocking traffic on the port that KEPServerEX is using to communicate.

Failover

The failover Kepware OPC UA server works the same as the OPC UA server with the exception that you need to have two copies of Kepware setup, preferably on different servers. The failover Kepware OPC UA server will be used in the event the primary Kepware server goes down. To enable failover, check the box to **Show advanced properties** in the New OPC UA Connection Settings, set the **Failover Enabled** property to '**true**', and specify the **Failover Endpoint**.

The **Backup** properties should be used when a pair of redundant Ignition Gateways are trying to look at the same Kepware OPC UA server. Both the **Backup Discovery URL** and **Backup Endpoint URL** properties need to be configured.

For additional information on Failover, refer to [OPC UA Client Connection Settings](#).

No Anonymous Token Policy Found

When connecting to KepServer, some versions may not allow anonymous connections by default. This typically means you need to specify user credentials for Ignition to use in the OPC UA server connection. Alternatively, individual Kepware Projects can allow anonymous login. For more information, look into allowing "anonymous login" in [KepServer's OPC UA Configuration Manager documentation](#).

Other UA Servers

While the above example is specific to KEPServerEX, the same concepts apply to connecting to any other third party OPC server that accepts OPC UA client connections. The only difference may be in the way that the certificates are accepted on the server.

The Ignition OPC UA server sends the client certificate to the third party OPC server when it tries to make the connection, however if the OPC server is not designed to expect these certificates then there may not be a straight forward way to accept them. In these cases, you can manually download a client ticket from Ignition and supply it to the OPC server in the appropriate manner.

Download a Client Certificate Manually

1. Go to **Config** section in the Gateway Webpage.
2. Select **OPC UA > Certificate** from the left side of the page. The Manage Certificates page is displayed.
3. In the **This Gateway** tab, click the download link under **Ignition OPC UA Client**, and save the certificate somewhere to disk. This certificate is then supplied to your third-party OPC server in a way specific to that server. For more information, check the respective server's documentation.

Related Topics ...

- [OPC COM](#)

OPC COM

Connecting to OPC Classic (COM)

Note: Classic OPC is based on COM, which is a technology in Microsoft Windows. Therefore, the information in this section only applies to Ignition Gateways installed on Windows. For other operating systems, OPC UA must be used.

The OPC-COM module provides the ability to connect to OPC servers that only communicate using the older COM based OPC-DA standard. If you have an OPC server that is not capable of accepting OPC UA connections and you need to talk to a PLC for which Ignition has no supported driver, you'll have to use the OPC-COM module to make your device data available in Ignition. Connections to OPC servers will be held open while the Ignition Gateway is running. All subscriptions to the server will use the same connection.

This section provides a brief walk-through of how to set-up a new Local or Remote OPC-DA server connection using the COM module. Due to the complications that Windows DCOM security settings can cause, this set-up guide is followed by the Troubleshooting OPC-COM Connections section that deals with an overview of how to deal with a faulted server connection due to DCOM security settings as well as other possibilities.

Install OPC Core Components

1. Register at www.opcfoundation.org.
The OPC-COM module relies on a .dll package provided by the OPC Foundation (www.opcfoundation.org) called the OPC Core Components. You can download the OPC Core Components Redistributable from the OPC Foundation's website under the downloads section. Registration with the OPC Foundation is required before you can download the package, but the registration process is free and painless.
2. Download appropriate OPC Core Components Redistributable package.
There are two packages to choose from, the 32-bit (x86) and the 64-bit (x64), make sure you get the correct one for the version of Java and Ignition you are running. 64-bit Java and Ignition needs the 64-bit Core Components package and likewise 32-bit installations needs the 32-bit package. You may have to look around a bit for the redistributable. At last update of this page (2019), the download was listed under [Resources Samples and Tools classic](#).
3. Install Core Components on Ignition server.
It should be noted that if you are going to connect to an OPC server on a remote machine, you must also install the appropriate version of the Core Components on that server as well. The version type, 64-bit or 32-bit, does not need to be the same across the two servers. Just be sure to install the version that is appropriate for the OPC Server and Windows architecture.
4. (Remote) Install Core Components on remote machine running the OPC-DA server.
Once you have the correct package downloaded you can extract the contents of the .zip file and then run the installer. With the core components installed you can now proceed to setting up your OPC-DA server connection in Ignition.

Connecting to OPC-DA Server

With the OPC Core Components now installed the next step is creating/configuring a new OPC-DA server connection.

Install OPC-DA Server Connection

1. Go to the Ignition Gateway Config section (<http://localhost:8088/main/web/config>).
2. Go to OPC Connections > Servers and then select **Create new OPC Server Connection....**
3. Choose the **OPC-DA COM Connection** and then select whether you want to make a Localconnection or if the OPC server resides on a Remote machine. For the most part, setting up a local or remote connection to an OPC-DA server is the same. There are only a couple of differences for a remote connection that will be highlighted along the way.
Local - Selecting a local connection takes you to a screen that contains a list of the available and running OPC servers located on the local machine.
Remote - For a remote connection you first have to specify the host name or IP address of the machine the the OPC server resides on and then (as of Ignition 7.4) you are redirected to the available servers list.
4. Select the OPC server that you wish to connect to from the list. In the case where your server is not listed, see the **OPC server is not listed...** the Troubleshooting OPC-COM Connection section.

Unique Remote Connection Settings - Remote connections have a few unique settings that you can specify. You can get to these settings by selecting the **Show advanced properties** check box. As of Ignition 7.4 these should all be set for you (except for the CLSID which should no longer be necessary but is still available for you to set if you wish).

Remote Server - Specifies that the server is remote and that a DCOM connection will be used.

Host Machine - The computer name or IP address of the machine on which the remote server is running.

CLSID - This is no longer required as of Ignition 7.4, but it is still made available for you. It can be used in place of the ProgId because the ProgId is really just used to lookup the CLSID in the registry. This id can be found in the registry of the machine hosting the server under:

HKEY_CLASSES_ROOT\OPCServerName\CLSID

5. All of the settings for the server connection are rather straight forward and each property has a description of its functionality. Most of these settings should be fine when left at their default values. The only setting that could possibly give you some trouble is the ProgId. If you selected your OPC server from the list on the **Choose OPC-DA Server** page, this will be filled in for you. However, if for whatever reason your server wasn't listed and you choose the **Other Server** option, you will have to know the ProgId for your server and specify it here. The ProgId is used to look up the CLSID of the OPC Server in the Windows Registry and without this a connection cannot be made.
6. When you are finished fine tuning these settings click **Create new OPC Server Connection**. You will be redirected to the OPC Server Connections page and your new server connection should be listed. The status of your connection will read Connected if Ignition was able to successfully connect to the third-party OPC server.

Connection Is Faulted

In the case where your connection status is reporting Faulted, the troubleshooting process begins. As previously stated, configuring the DCOM settings on your machine can be a headache. The [Troubleshooting OPC-COM Connections](#) section next is an attempt to ease the process of determining why your connection is faulted and how to go about fixing the issue. If after exhausting the options presented to you, you are still having issues getting your server connection up, give our Inductive Automation tech support line a call and one of our representatives will be happy to assist you.

Troubleshooting OPC-COM Connections

This section provides you with a list of common OPC-COM connection problems with their possible solutions. It would be impossible to give an exhaustive list of everything that can go wrong but this should give you a good start on the troubleshooting process. If you do not see your problem listed and your connection status is faulted, try following the steps outlined in the [Ignition Server DCOM Settings](#) and [OPC Server DCOM Settings](#) sections.

Common Problems

OPC Server Is not Listed in Choose OPC-DA Server List when First Creating a Connection

There are some cases in which an OPC Server that is installed will not show up in the generated list. This list is generated by the OPC Server Enumerator which is part of the OPC Core Components, so when a server you have installed on the machine does not appear in this list it is likely due to the OPC Core Components not being installed correctly.

Try reinstalling the Core Components and going through the process of creating a new server connection in Ignition again. If the server still does not appear and you have the ProgId (or the CLSID for a remote connection) for the OPC server, you can just select the **Other Server** option and then click **Next**. In this situation you will have to enter the ProgId manually on the [New OPC-DA Server](#) page.

With all the correct information about the OPC server we can sometimes still make a valid connection to the OPC Server even when it is not detected automatically. This however is rare. Most of the time when the server is not detected, any connection attempts Ignition makes will fail.

Connection Status Is Connected but Data Quality Is Bad or the Connection Goes Faulted after Trying to Read Tag Data

Usually this occurs when the DCOM settings for the machine on which Ignition is running are not correctly configured. DCOM connections go in both directions. Ignition must be able to send requests to the OPC server and the OPC server must also be able to callback to Ignition. If the DCOM settings on the Ignition server are not configured correctly those callbacks will fail and the server connection that initially had a status of "Connected" will either fault or all the Tags that you have configured will come back with bad quality.

This is a problem that can affect both local and remote server connections.

Follow the steps outlined in the "Ignition Server DCOM Settings" section to ensure that you have correctly configured the DCOM security settings on the Ignition server machine.

Ignition Launches Second Instance of an Already Running OPC Server and Is Unable to See Any Data

It is important to note that Ignition runs as a service under the Windows System account. This can cause some issues with OPC servers that are meant to run interactively, meaning they run under the user account that is currently logged on. When Ignition attempts to make a connection to the OPC server, it will attempt to find an instance running under the same account and if it doesn't find one it will launch its own instance under the System account. Even if there are other instances running, Ignition will choose the one that was launched under the System account for its connection.

Many OPC servers maintain an instance running under the interactive user account that has been configured by the user and maintains all of the device connection information. When Ignition launches a new instance, this configuration information is lacking and none of the desired data can be seen or accessed. To get around this problem, you must specify in the DCOM settings for the OPC server that it always identify itself with the interactive user. Essentially this will force Ignition to use the currently running instance of the OPC server.

Set the OPC Server to Run as Interactive User

1. The DCOM settings are found in the Component Services manager. Right-click the entry for your OPC server under the DCOM Config folder and select properties from the popup menu.
2. Select the Identity tab. Select the option that reads **The interactive user**, and click **OK**.

3. Close out of component services and kill any extra instances of the OPC server you see running in the Task Manager.
4. Go edit and save the OPC server connection in the Ignition Gateway.

Faulted Status with E_CLASSNOTREG Error Reported on OPC Connections Status Page

This is almost always caused by the OPC Core Components not being installed correctly. Download and install the correct version(s) for your system (s) from the OPC Foundation (www.opcfoundation.org). Remember, if you are making a remote connection you must install these components on both the Ignition server as well as the machine on which the OPC server is running.

DCOM Settings

Ignition Server DCOM Settings

Follow these steps to open up the DCOM security settings on the machine that is running Ignition:

1. Open the **Windows Component Services**, located in the Administrative Tools section of the Control Panel.
2. Browse down through the Component Services tree until you see My Computer, right-click and select **Properties**.
3. We want to focus on the COM Security tab. There are two sections, **Access Permissions** and **Launch** and **Activation Permissions**. Each section has an **Edit Limits...** and **Edit Defaults...** button. You must add the ANONYMOUS and Everyone accounts under each of the four areas making sure that the **Allow** option is checked for each of the permission settings. If you skip adding both of these to either the limits or defaults areas under either of the two sections there is a good chance your connection will not be successful.
4. You can also try setting the Default Authentication Level to **None** and the Default Impersonation Level to **Identify** on the Default Properties tab. This isn't always necessary but it can sometimes help.

OPC Server DCOM Settings

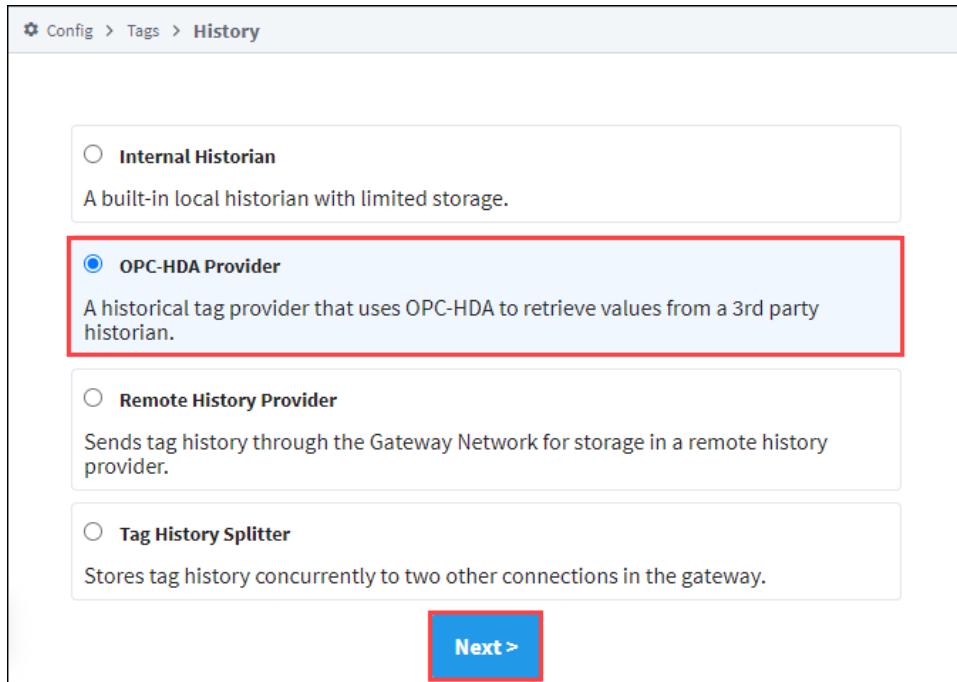
Follow these steps to open up the DCOM security settings on the machine that is running the OPC server:

1. Open up **Windows Component Services**, located in the Administrative Tools section of the Control Panel.
2. Browse down through the Component Services tree until get to the **DCOM Config** folder.
3. Locate the entry for your OPC server that you wish to make a connection to, right-click and select properties.
4. Click the **Security** tab and you will see three sections: Launch and Activation Permissions, Access Permissions, and Configuration Permissions. There are two options to choose from for each section. If you already added the ANONYMOUS and Everyone accounts to the COM Security section from the **Ignition Server DCOM Settings** section then you can go ahead and just select the **Use Default** option for each of the three areas. The second option is to edit each of the groups that have **Customize** selected. You will have to add both the **ANONYMOUS** and **Everyone** accounts with all privileges.
5. Now select the **Identity** tab. You will notice that you can choose which account you want to run the OPC server under. Select the **Interactive User** option. This ensures that if Ignition launches an instance of the OPC server, it will run under whichever user is currently logged into the system.

Creating an OPC-HDA Connection

The process of connecting to an OPC HDA server is similar to that of a DA server. Instead of going to the "OPC Connections" section, however, you define the server as a Tag History Provider.

1. Navigate to the Gateway Config Webpage, as outlined above.
2. Under Tags, History, select **Create new Historical Tag Provider...**
3. Select OPC HDA Provider and click **Next**.



4. Follow the step outlined above, for DA connections.
5. Once complete, the status on the Tags>History screen will show the state of the connection. If Connected, you should now be able to browse and query the server through the Ignition designer.

OPC-HDA Properties

OPC-HDA Provider	
Provides a connection to an OPC-HDA server.	
Provider Name	Name of the Tag History Provider.
Enabled	If the check box is selected (enabled), the provider is turned on and will expose historical data
Description	A description of the provider.
ProgId	A description of the provider.
Use Flat Browsing	Flat browsing returns all items at once. This is less efficient than normal browsing, but if a server only supports flat browsing, then this needs to be checked.
Remote Server	If selected, DCOM will be used to connect to the server on the specified Host with the given ProgId or CLSID.
Host Machine	The name or IP address of the machine hosting the server. Leave empty for local machine.
CLSID	The CLSID of the server. If not specified, will be obtained using the ProgId.

Example - Adding OPC-HDA data to a chart

1. Open the Designer, and create or open a project.
2. Create a window, and add an Easy Chart component.
3. Double-click on the chart, or right click and select Customizers>Easy Chart Customizer to bring up the chart customization window.
4. Next to the "Tag History Pens" table, select the first button, **Browse for Tags**. This will display a tree for browsing all historical Tags.
5. Browse through your defined HDA server. Once you find a Tag, select "ok" to add it to the chart.
6. You may edit the Tag to alter its aggregation mode, though the HDA provider will select a supported mode automatically if the specified mode does not exist in the server.
7. Once you save the configuration, the chart should update with the requested data.

A similar procedure can be used anywhere Tag History can be bound or used.

OPC UA Security

On the OPC UA security page you can manage [OPC UA](#) certificates for the client and server. Trusted certificates can be imported and quarantined certificates can be marked as trusted.



On this page ...

- [Trusted Certificates on the Client](#)
- [Trusted Certificates on the Server](#)
 - [Upload a Trusted Certificate](#)
 - [Download a Trusted Certificate](#)
 - [Delete a Trusted Certificate](#)
 - [OPC UA Security Page Details](#)
- [Quarantined Certificates](#)
 - [Accept a Quarantined Certificate](#)
- [Download Current Certificates](#)
- [Regenerate Current Certificates](#)
 - [Regenerate a Client Certificate](#)
 - [Regenerate a Server Certificate](#)

Trusted Certificates on the Client

When viewing the **Client** tab, you're viewing the certificates trusted by the Gateway, as a UA client. In the screenshot below, you can see that this client trusts the certificate named "Ignition OPC UA Server."

Common Name	SHA-1 Fingerprint	Expiration	Action
Ignition OPC UA Server	a233eddf3a52a46f22c5789d27...	Sep 16, 2023	Delete Download

Trusted Certificates on the Server

When viewing the **Server** tab, you're viewing the certificates trusted by the server, meaning the Gateway's OPC UA server. In the screenshot below, you can see that this server trusts the certificate named "Ignition OPC UA Client."

The screenshot shows the Ignition OPC UA Security configuration page. The top navigation bar includes 'Config > Opcua > Security'. Below this, there are three tabs: 'Client' (blue), 'Server' (red box), and 'Certificates'. The 'Server' tab is active. The main content area is titled 'Server Security' and contains a section for 'Trusted Certificates'. A table lists one certificate: 'Ignition OPC UA Client' with SHA-1 Fingerprint '390c157ef05a7b97f8cc899d6...' and Expiration 'Sep 16, 2023'. Actions include a 'Delete' button and a blue download icon. Below this is a section for 'Upload Trusted Certificate:' with a 'Browse' button and a 'Drag files here' placeholder. The 'Quarantined Certificates' section shows no results.

Upload a Trusted Certificate

The steps for uploading trusted certificates are the same whether you're on the Client tab or the Server tab. To upload a trusted Certificate, do the following.

1. On the Gateway Webpage, select **OPC UA > Security**.
2. Click the **Client** tab or **Server** tab, depending on what certificate you're uploading.
3. Click the **Browse** button.
4. Navigate to the location of the certificate on your system and click **Open**. (Alternatively, you can drag the certificate file onto the page where it says "Drag files here".)
5. If the upload was successful, you'll see the name of the certificate and the message "Upload Successful!" The certificate will appear in the Trusted Certificates list.

This screenshot shows the Ignition OPC UA Security configuration page with the 'Client' tab selected (red box). The main content area is titled 'Client Security' and contains a section for 'Trusted Certificates'. A table lists one certificate: 'Ignition OPC UA Server' with SHA-1 Fingerprint '8aa496b1f50ac579fb8b45adab0cb3635...' and Expiration 'Mar 9, 2023'. Actions include a 'Delete' button and a blue download icon. Below this is a section for 'Upload Trusted Certificate:' with a 'Browse' button and a 'Drag files here' placeholder. A red box highlights the 'Browse' button. At the bottom, a message box shows 'ed9ea756a105d27335ce23602cce0fae4... ✓' and 'Upload Successful!'.

Download a Trusted Certificate

To download a trusted certificate, do the following.

1. Next to the certificate name, click the **Download** icon.
2. The certificate is downloaded to your system by your web browser.



Delete a Trusted Certificate

To delete a trusted certificate, do the following.

1. Next to the certificate name, click the **Delete** action button.
2. The certificate is deleted.



To view more information about a trusted certificate, click the **More Info** icon.

Common Name	SHA-1 Fingerprint	Expiration	Action
Ignition OPC UA Client	6af7b24fc... (truncated)	Mar 9, 2023	Delete
CN	O	OU	L
Subject	Ignition OPC UA Client	Inductive Automation	Folsom
Issuer	Ignition OPC UA Client	Inductive Automation	CA
			US

OPC UA Security Page Details

Trusted Certificates	
Common Name	Name of the certificate.
SHA-1 Fingerprint	The SHA-1 (Secure Hash Algorithm 1) fingerprint is the unique identifier of the certificate.
Expiration	Date the certificate will expire.
Additional Information	
C	Common Name
O	Organization, usually the legal incorporated name of a company.
OU	Organizational Unit
L	Locality (Town or City)
ST	State
C	Country, the two-letter ISO code for the country where the organization is located (i.e., CA for California).

Quarantined Certificates

If you import a certificate that is not trusted, it will appear on the Quarantined Certificates list.

The screenshot shows the Ignition OPC UA Security configuration interface. The top navigation bar includes 'Config > Opcua > Security'. Below this, there are three tabs: 'Client' (disabled), 'Server' (selected and highlighted with a red border), and 'Certificates'. The main content area is titled 'Server Security'.

Trusted Certificates

Common Name	SHA-1 Fingerprint	Expiration	Action
Ignition OPC UA Client	fe37b96f9129a0980...	Sep 19, 2022	<button>Delete</button> <button>Download</button>

Upload Trusted Certificate:

or Drag files here

Quarantined Certificates

Common Name	SHA-1 Fingerprint	Expiration	Action
Ignition OPC UA Client	120fsd30980d455b3...	Sep 19, 2022	<button>Trust</button> <button>Delete</button> <button>Download</button>

Accept a Quarantined Certificate

To accept a quarantined certificate, do the following:

1. Next to the certificate name, click the **Trust** action button.
2. The certificate is accepted and will appear in the Trusted Certificates list.

This feature is new in Ignition version **8.1.0**
[Click here](#) to check out the other new features

Download Current Certificates

You can download a certificate for the OPC UA client that's currently running on your Gateway as follows:

1. On the Gateway Webpage, go to the Config tab and select **OPC UA > Security**.
2. Click the **Certificates** tab. You'll see the current Client and Server certificates.
3. To download a current certificate, click the **Download** button. In this example, we download a client certificate.
4. The certificate is downloaded to your system.

Config > Opcua > Security

Client Server Certificates

Manage Certificates

Client Certificate

Manage the certificate for the OPC UA Client running on this Gateway

Name	Signature	Expiration	Actions
Ignition OPC UA Client	638acd09e33c2b9ca3f508986ece9256b85523de	Sep 21, 2023	Download Regenerate

Server Certificate

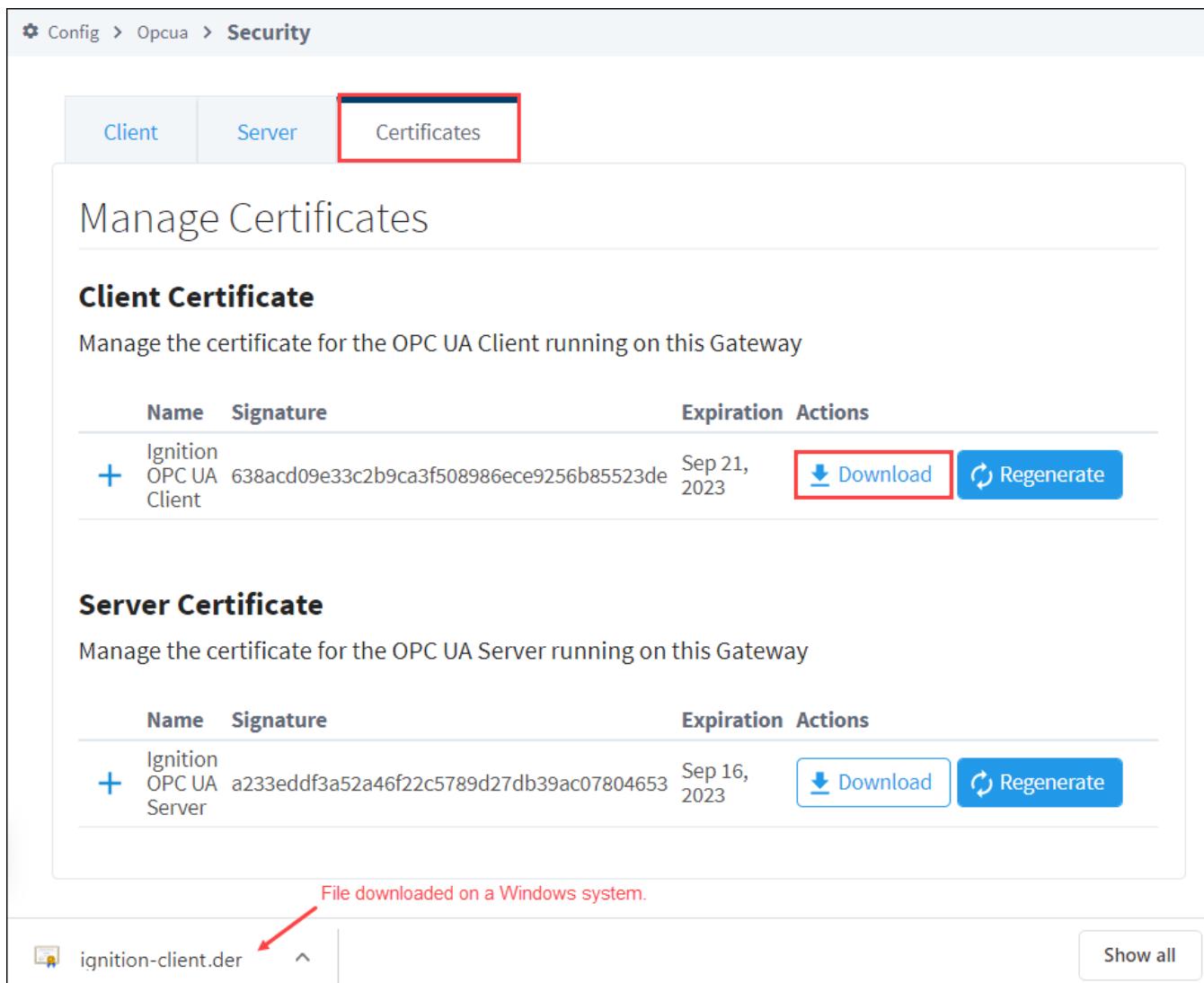
Manage the certificate for the OPC UA Server running on this Gateway

Name	Signature	Expiration	Actions
Ignition OPC UA Server	a233eddf3a52a46f22c5789d27db39ac07804653	Sep 16, 2023	Download Regenerate

File downloaded on a Windows system.

ignition-client.der

Show all



This feature is new in Ignition version **8.1.0**
[Click here](#) to check out the other new features

Regenerate Current Certificates

All SSL certificates have a definitive live span. For example, the default life span for an Ignition-generated OPC UA certificate is three years. The limited lifespan helps ensure that certificates keep up with the latest security standards. Regenerating the certificates resets the expiration date to extend it another three years with an entirely new certificate. If your private key is somehow compromised, regenerating a Client or Server certificate ensures that the private key will no longer work with the Ignition Gateway.

Newly regenerated certificates are automatically trusted by the Gateway issuing them.

Regenerate a Client Certificate

1. On the Gateway Webpage, go to the Config tab and select **OPC UA > Security**.
2. Click the **Certificates** tab. You'll see the current Client and Server certificates.
3. Next to the client certificate you want to regenerate, click the Regenerate button.

Config > Opcua > Security

Client Server Certificates

Manage Certificates

Client Certificate

Manage the certificate for the OPC UA Client running on this Gateway

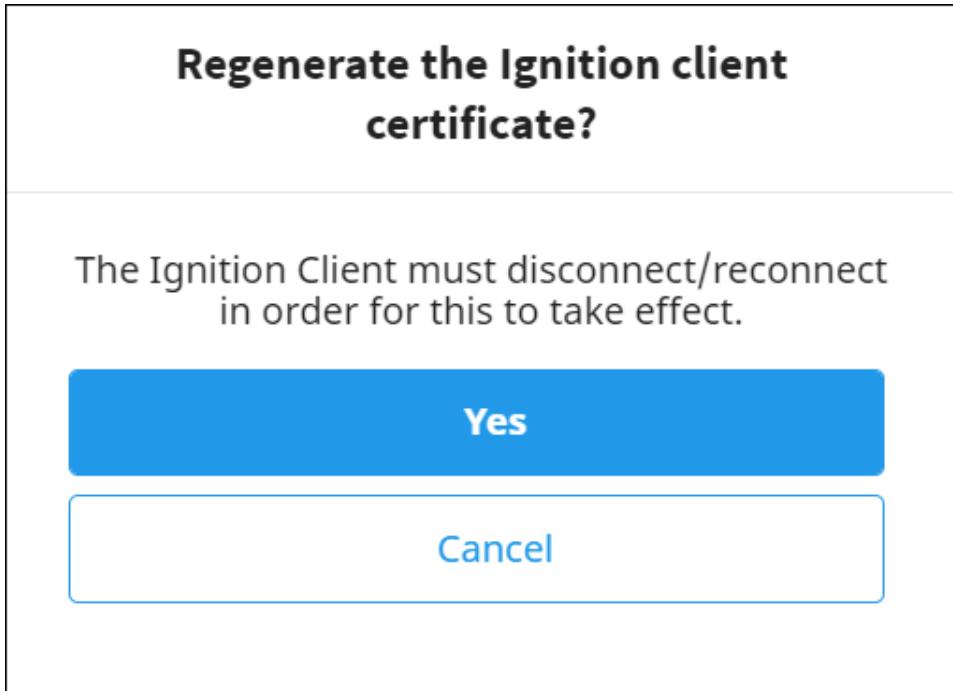
Name	Signature	Expiration	Actions
Ignition OPC UA Client	638acd09e33c2b9ca3f508986ece9256b85523de	Sep 21, 2023	Download Regenerate

Server Certificate

Manage the certificate for the OPC UA Server running on this Gateway

Name	Signature	Expiration	Actions
Ignition OPC UA Server	a233eddf3a52a46f22c5789d27db39ac07804653	Sep 16, 2023	Download Regenerate

4. You will see a confirmation message. Click **Yes** to regenerate the certificate.



5. You must disconnect and reconnect all OPC clients for this to take effect. Go to the Config. On the Gateway Webpage, go to the Config tab and select **OPC Client > OPC Connections**.
6. Click the Edit button for the OPC Client you want to restart.

OPC Connections

Name	Type	Description	Read Only	Status	More	edit
Ignition OPC UA Server	OPC UA	A "loopback" connection to the Ignition OPC UA server running on this gateway.	false	Connected	More	edit

→ [Create new OPC Connection...](#)

Note: For details about a connection's status, see the [OPC Connection Status](#) page.

7. Click **Save**. The Client is restarted.

Regenerate a Server Certificate

1. On the Gateway Webpage, go to the Config tab and select **OPC UA > Security**.
2. Click the **Certificates** tab. You'll see the current Client and Server certificates.
3. Next to the server certificate you want to regenerate, click the Regenerate button.

Security

Certificates

Client Certificate

Manage the certificate for the OPC UA Client running on this Gateway

Name	Signature	Expiration	Actions
Ignition OPC UA Client	638acd09e33c2b9ca3f508986ece9256b85523de	Sep 21, 2023	Download Regenerate

Server Certificate

Manage the certificate for the OPC UA Server running on this Gateway

Name	Signature	Expiration	Actions
Ignition OPC UA Server	a233eddf3a52a46f22c5789d27db39ac07804653	Sep 16, 2023	Download Regenerate

4. You will see a confirmation message. Click Yes to regenerate the server certificate.

Regenerate the Ignition server certificate?

The OPC UA module must be restarted in order for this to take effect.

Yes

Cancel

5. To restart the OPC UA module, go to the Config tab and select **System > Modules**.
6. Scroll down to the OPC-UA module and click **Restart**.

Config > System > Module Configuration					
OPC-UA	9.1.1-SNAPSHOT (b2020092102)	Provides Ignition's OPC UA client and server functionality.	Trial	Running	More ▾ restart
OpcCom	6.1.1-SNAPSHOT (b2020092102)	Bridge that exposes COM based OPC-DA servers to the system.	Trial	Running	More ▾ restart
Perspective	2.1.1-SNAPSHOT (b2020092102)	A module that provides modern, responsive html based graphical interfaces for Ignition projects.	Trial	Running	More ▾ restart

OPC UA Drivers

Ignition has a few drivers in the form of modules that allow you to connect to specific types of devices over OPC UA. Each module provides the ability to connect to different types of devices.

Allen Bradley Drivers

The [Allen Bradley Driver](#) module and Logix Driver module provides the ability to use any of our Allen Bradley suite of drivers. One of the more popular options, it contains drivers to connect to:

- Logix family devices with firmware versions 21 and newer
- MicroLogix 1100 and 1400
- PLC5
- SLC 5/05
- CompactLogix and [ControlLogix](#) with legacy firmware 20.18 and prior

On this page ...

- [Allen Bradley Drivers](#)
- [Modbus Drivers](#)
- [Siemens Drivers](#)
- [UDP and TCP Drivers](#)
- [DNP3 Driver](#)
- [Omron NJ Driver](#)
- [Omron FINS Driver](#)
- Ignition now supports Omron FINS devices. This driver does support both UDP and TCP transport protocols.
- [Simulator Drivers](#)

Modbus Drivers

The Modbus Driver module allow you to connect to a wide range of devices using the [Modbus protocol](#). While this typically means figuring out the Modbus specific addresses for each register, [address mapping](#) makes this process simpler by mapping a range of addresses for you.

Siemens Drivers

The Siemens Drivers module allows you to connect to a number of [Siemens S7](#) devices. The available device drivers are:

- S7-300
- S7-400
- S7-1200
- S7-1500

UDP and TCP Drivers

The [UDP and TCP Drivers](#) module are very basic drivers that can connect to one or more ports of a given IP address and read in data. This is typically used for barcode scanners or scales.

DNP3 Driver

The DNP3 Driver module allows you to connect to a [DNP3 outstation](#). Many different types of devices support DNP3.

Omron NJ Driver

The Omron Driver module allows you to connect to the [Omron NJ](#) series of devices.

Omron FINS Driver

Ignition now supports Omron FINS devices. This driver does support both UDP and TCP transport protocols.

Simulator Drivers

There are a number of drivers that do not connect to any device, but instead provide simulated values for testing. These simulators are a part of the Allen Bradley Driver module.

In This Section ...

Allen Bradley Ethernet

Connecting to Allen Bradley Devices

Ignition contains drivers for several Allen Bradley devices. These drivers can connect directly through the Gateway to devices that support Ethernet communications and to devices that have access to an Ethernet card such as an ENBT or EN2T. The device drivers in the Allen Bradley Ethernet Module are:

- [Logix](#) ControlLogix and CompactLogix firmware v21 and newer
- [ControlLogix](#) firmware v20.18 and prior
- [CompactLogix](#) firmware v20.18 and prior
- [PLC-5](#)
- [MicroLogix](#) 1100, 1200, 1400, 1500
- [SLC](#) 5/05

Note: The Allen Bradley Ethernet drivers only support firmware versions 16 and up. Older firmware versions may work, but are unsupported. If your device does not meet these requirements, another OPC Server can be used to connect to them.

Caution: Direct connections to RSLinx Emulate 5000 are not supported, and may not work properly.

Verifying Device Connectivity

Device connectivity can be verified in the following places:

- In the **Config** page of the Gateway under **OPC UA > Device Connections**
- In the **Status** page of the Gateway under **Overview > click the Devices box**
- In the **Status** page of the Gateway under **Connections > Devices**
- In the **Status** page of the Gateway under **Connections > OPC Connections**

Allen Bradley Connection Paths Explained

All of the Allen Bradley drivers have a **Connection Path** property that is used when the device uses a special Ethernet card (instead of having an Ethernet connection built in) or if you are using another device to act as a bridge. That is, connections to ControlLogix, CompactLogix, PLC-5, MicroLogix and SLC Allen-Bradley processors bridged through a ControlLogix Gateway require a connection path. The connection path is unique to your setup and is dependent on what modules the connection is being routed through. With there being nearly an endless number of ways to route your connection from device to device it is impossible to give an example of every possible connection path, but in general there is a pattern to how the connection path is specified.

The following is a basic outline for figuring out your connection path. For more specific information on individual device types, see the individual connection pages.

Follow the Path

A connection path is exactly what it sounds like. It is a path that when followed will lead a processor residing in a numbered slot of a chassis somewhere on site. You merely have to follow the path and build the connection path as you go. The first connection point between Ignition and the device is a ControlLogix Ethernet module such as an ENET, ENBT or EN2T module. The slot number of this module doesn't matter and there is no need to specify it in the connection path. The first entry in any connection path will be a 1, which specifies moving to the back plane. You then specify the slot of the module you wish to move to, followed by the port or channel of that module that you wish to exit through. Finally you specify the address of your entry point to the next module and the process starts all over again. This process may sound complicated at first but after some practice it will get easier.

Connection Path Steps

1. Move to the backplane.
2. Specify the slot number of the module you are moving to.
3. Specify the exit port or channel.
4. Specify address of entry point (DH+ Station Number / ControlNet Address / IP Address of Ethernet module).
5. Move to the backplane.
6. Specify processor slot number or the slot number of the module you wish to exit through.

Connection Path Entries for Different Module Types

How you specify your exit point from a module differs depending on which module type you are using. You can only move in two directions once you are "in" a module: out to the backplane or out through the module port/channel. Ethernet modules have Ethernet ports and an IP address; ControlNet modules have ControlNet Ports and ControlNet addresses; DHRIO modules have channels and station numbers. Below is a list of different kinds of modules and what numbers you specify in the connection path when you are exiting or entering those modules. When in a module, an entry of 1 will always take you to the backplane.

ENET, ENBT, and EN2T:

Exiting

- 1 = Backplane
- 2 = Ethernet Port

Entering

IP Address

CNB:

Exiting

- 1 = Backplane
- 2 = ControlNet Port

Entering

ControlNet Address

DHRIO

Exiting

- 1 = Backplane
- 2 = DH+ Channel A
- 3 = DH+ Channel B

Entering

DH+ Station Number (an octal value between 0-77)

You use these numbers to specify how to move out of the module, then you specify where you are moving to by either specifying the DH+ station number, ControlNet address, or the IP address of another Ethernet module. Your connection path will always be an even number of entries due to the fact that you always move in two steps: out of a module and then in to another module. So if your connection path ends up with an odd number of entries you have missed a step somewhere and you'll have to go back and trace the path again.

[In This Section ...](#)

Connecting to CompactLogix

Note: This driver is made for CompactLogix firmware up to version 20.18.

Connect to an Allen-Bradley CompactLogix Device

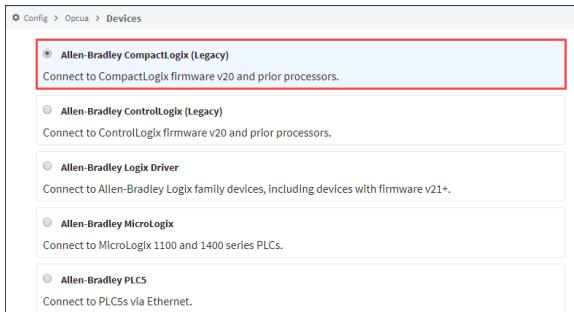
1. Go to the **Config** section of the Gateway Webpage.
2. Scroll down and select **OPC UA > Device Connections**.



Connecting to CompactLogix

[Watch the Video](#)

3. On the **Devices** page, find the blue arrow and click on **Create new Device**.
4. Select **Legacy Allen-Bradley CompactLogix**, and click **Next**.



5. On the **New Device** page, leave all the default values and type in the following fields:
Name: **CompactLogix**
Hostname: type the **IP address** for the PLC, for example 10.20.4.55.

A screenshot of a configuration form for a new device. The 'General' tab shows 'Name' set to 'CompactLogix'. The 'Connectivity' tab shows 'Hostname' set to '10.20.4.55'. At the bottom, there is a 'Create New Device' button.

6. Click **Create New Device**.
The **Devices** page is displayed showing the **CompactLogix** device is added to Ignition. The **Status** will show as Disconnected and then Connected.

Config > Opcua > Devices					
Successfully created new Device "CompactLogix"					
Name	Type	Description	Enabled	Status	
Sim	Simulator: Generic		true	Running	<button>delete</button> <button>edit</button>
CompactLogix	Allen-Bradley CompactLogix (Legacy)		true	Connected: Protocol: EIP Run Mode	<button>More</button> <button>edit</button>
Sim	Simulator: Generic		true	Running	<button>delete</button> <button>edit</button>

7. To see all the Tags, go to **OPC Client > Quick Client** in the **Config** section. On the **OPC Quick Client** page, expand the **CompactLogix** folder and in the **Global** folder you can see all the tags.

Device Connection Settings

The general settings are common to all Allen Bradley devices. The connectivity and advanced settings are device dependent.

General	
Name	The user-defined name for this Device. The Device Name must be alphanumeric.
Description	The user-defined description for the device. This is only used as a note to differentiate between devices.
Enable Device	Only devices that are enabled appear in Connections > Devices of the Status page of the Gateway and thus have their tags available for use. Default is true.
Connectivity	
Hostname	This is the IP Address of the Ethernet module to route through to connect a CompactLogix processor.
Timeout	The timeout settings refer to the communication between the device driver and the OPC-UA server and usually can be left at their default values.
Connection Path	The Connection Path value is used to define the route to connect to the processor. The Connection Path format contains 4 numbers separated by commas. The first number is always 1 and tells the Ethernet module to route through the backplane. The rest of the numbers vary by device type, for a more in depth explanation of connection paths, see Allen Bradley Connection Paths Explained section .
Current Requests	The number of requests that Ignition will try to send to the device at the same time. Increasing this number can sometimes help with your request throughput, however increasing this too much can overwhelm the device and hurt communications with the device.
Advanced	
Disable Automatic Browse	Disables the automatic browse setting. Default is false.
Show String Arrays	Disables the Show String Arrays setting. Default is false.
Status Request Poll Rate	Controls the poll rate for status requests, in milliseconds. Default is 1,000.

Related Topics ...

- [Connecting to MicroLogix](#)

Connecting to ControlLogix

Note: This driver is made for CompactLogix firmware up to version 20.18.

Connect to an Allen-Bradley ControlLogix Device

1. Go to the **Config** section of the Gateway Webpage.
2. Scroll down and select **OPC UA > Device Connections**.



3. On the **Devices** page, find the blue arrow and click on **Create new Device**.
4. Select **Legacy Allen-Bradley ControlLogix**, and click **Next**.



5. On the **Devices** page, leave all the default values and type in the following fields:
Name: **CLX**
Hostname: Enter the **IP address** for the PLC, for example 10.20.4.50.

A screenshot of the 'Devices' configuration form. It has two sections: 'General' and 'Connectivity'. In the 'General' section, the 'Name' field is set to 'CLX'. In the 'Connectivity' section, the 'Hostname' field is set to '10.20.4.50'. Other fields like 'Timeout', 'Slot Number', 'Connection Path', and 'Concurrent Requests' have their default values checked.

6. Click **Create New Device**.
The **Devices** page is displayed showing the **ControlLogix** device is added to Ignition. The **Status** will show as Disconnected and then Connected.

On this page ...

- [Connect to an Allen-Bradley ControlLogix Device](#)
- [Device Connection Settings](#)
 - [Supported Connection Methods](#)



Connecting to ControlLogix

[Watch the Video](#)



7. To see all the Tags, go to **OPC Client > Quick Client** in the **Config** section. On the **OPC Quick Client** page, expand the **ControlLogix** folder and in the **Global** folder you can see all the tags.

Device Connection Settings

The **General** settings are common to all Allen Bradley devices, and the **Connectivity** settings are device dependent.

General	
Name	The user-defined name for this Device. The name chosen will show up in OPC Item Paths and under OPC-UA Server > Devices of the Configure page of the Gateway. The Device Name must be alphanumeric.
Description	The user-defined description for the device. This is only used as a note to differentiate between devices.
Enable Device	Only devices that are enabled appear in Connections > Devices of the Status page of the Gateway and thus have their tags available for use.
Connectivity	
Hostname	This is the IP Address of the ControlLogix Ethernet module (1756-ENET) to route through to connect a ControlLogix processor. EthernetIP protocol on TCP port 44818 (0xAF12) is used to communicate to ControlLogix processors.
Timeout	After sending a request to the ControlLogix processor, the Communication Timeout setting is the amount of time in msec to wait for a response before treating it as a failure.
Slot Number	The Slot Number value is the zero based ControlLogix chassis slot number of the ControlLogix processor to connect to.
Connection Path	The Connection Path value is used to define the route of the PLC-5 processor to connect to. Currently routing through the ControlLogix Ethernet Communication Interface Module (1756-ENET) to the ControlLogix Data Highway Plus-Remote I/O Communication Interface Module (1756-DHRI0) and on to a PLC-5 processor of the DH+ network is supported.
Concurrent Requests	The number of requests that Ignition will try to send to the device at the same time. Increasing this number can sometimes help with your request throughput, however increasing this too much can overwhelm the device and hurt your communications with the device.
Advanced	
Disable Automatic Browse	Disables the automatic browse setting. (Default is false.)
Show String Arrays	Disables the Show String Arrays setting. (Default is false.)
Status Request Poll Rate	Controls the poll rate for status requests, in milliseconds. (Default is 1,000.)

Supported Connection Methods

ControlLogix 5500 connected through 1756-ENET/A or 1756-ENET/B.

Related Topics ...

- [Connecting to CompactLogix](#)

Connecting to Logix

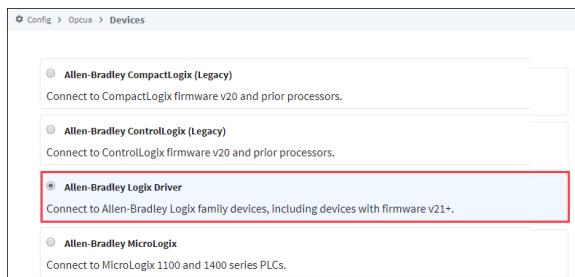
Note: This driver is optimized for Logix family devices with firmware version 21+, but supports earlier firmware versions with significantly reduced performance.

Connect to an Allen-Bradley Logix Device

1. Go to the **Config** section of the Gateway Webpage.
2. Scroll down and select **OPC UA > Device Connections**.



3. On the **Devices** page, find the blue arrow and click on **Create new Device**.
4. Select **Allen-Bradley Logix Driver**, and click **Next**.



5. Fill in the following fields:

Name: **CLX**
Hostname: type the **IP address** for the PLC, for example 10.20.1.54.

Leave the default values in the remaining fields.

A screenshot of the device configuration form. The 'General' tab shows:

Name	CLX
Description	[empty]
Enabled	<input checked="" type="checkbox"/> (default: true)

The 'Connectivity' tab shows:

Hostname	10.20.1.54
Port	44818
Timeout	2000
Max Concurrent Requests	2
Slot Number	0
Connection Path	[empty]

Each field has a small description below it.

On this page ...

- [Connect to an Allen-Bradley Logix Device](#)
- [Driver Implementation](#)
- [Device Connection Settings](#)



INDUCTIVE
UNIVERSIT

Connecting to ControlLogix v21

[Watch the Video](#)

- Click **Create New Device**.
- The **Devices** page is displayed showing the **ControlLogix** device is added to Ignition. The **Status** will show as Disconnected and then Connected.

Config > Opcua > Devices					
Successfully created new Device "CLX"					
Name	Type	Description	Enabled	Status	
CLX	Allen-Bradley Logix Driver		true	ReconnectWait	<button>delete</button> <button>edit</button>
Sim_New_Programmable	Programmable Device Simulator		true	Running	<button>More</button> <button>edit</button>

- To see all the Tags, go to **OPC Client > Quick Client** in the **Config** section. On the **OPC Quick Client** page, expand the **CompactLogix** folder and in the **Global** folder you can see all the tags.

Driver Implementation

One noteworthy implementation of the Logix driver is how boolean arrays are handled. Boolean arrays items are browsed as members of a 32-bit DWORD. Thus, a BOOL[64] in the PLC is implemented as DWORD[2] in the driver.

Below is a table representing how members of a boolean array items are mapped in the PLC, compared to the Logix driver's implementation.

PLC Mapping	Driver Implementation
boolTag[0]	boolTag[0].0
boolTag[31]	boolTag[0].31
boolTag[32]	boolTag[1].0
boolTag[63]	boolTag[1].31

Device Connection Settings

The General settings are common to all Allen Bradley devices, and the Connectivity settings are device dependent.

General	
Name	The user-defined name for this Device. The name chosen will show up in OPC Item Paths and under OPC-UA Server > Devices of the Configure page of the Gateway. The Device Name must be alphanumeric.
Description	The user-defined description for the device. This is only used as a note to differentiate between devices.
Enable Device	Only devices that are enabled appear in Connections > Devices of the Status page of the Gateway and thus have their tags available for use.
Connectivity	
Hostname	This is the IP Address of the ControlLogix Ethernet module (1756-ENET) to route through to connect a ControlLogix processor. EthernetIP protocol on TCP port 44818 (0xAF12) is used to communicate to ControlLogix processors.
Port	Port to connect to the remote device.
Timeout	After sending a request to the ControlLogix processor, the Communication Timeout setting is the amount of time in msec to wait for a response before treating it as a failure.
Max Concurrent Requests	The number of requests that Ignition will try to send to the device at the same time. Increasing this number can sometimes help with your request throughput, however increasing this too much can overwhelm the device and hurt your communications with the device.
Slot Number	The Slot Number value is the zero based ControlLogix chassis slot number of the ControlLogix processor to connect to.
Advanced	

Automatic Rebrowse	Monitor for tag additions and UDT changes and automatically initiate a re-browse when detected. If this is disabled tags will only be browsed when connecting and reconnecting. (Default is true.)
Identity Request Frequency	<p>This feature is new in Ignition version 8.1.3 Click here to check out the other new features</p> <p>Frequency, in milliseconds, that the request to read CIP Identity Object attributes occurs at. (Default is 5,000.)</p>
CIP Connection Size	The CIP connection size to use during Forward Open requests. (Default is 500.)

Related Topics ...

- [Connecting to ControlLogix](#)

Connecting to MicroLogix

Connect to a Device

1. Go to the **Config** section of the Gateway Webpage.
2. Scroll down and select **OPC UA > Device Connections**.



3. On the **Devices** page, find the blue arrow and click on **Create new Device**.
4. Select **Allen-Bradley MicroLogix**, and click **Next**.



5. On the **New Device** page, leave all the default values and type in the following fields:
Name: **MLX**
Hostname: Enter **IP address** for the PLC, for example **10.20.7.77**

A screenshot of the Ignition Config New Device configuration form. It shows fields for General (Name: MLX, Enabled checked) and Connectivity (Hostname: 10.20.7.77). At the bottom, there is a 'Create New Device' button.

6. Click **Create New Device**.
7. The **Devices** page is displayed showing the **MicroLogix** device is added to Ignition. The **Status** will show as Disconnected and then Connected.

A screenshot of the Ignition Config Devices page. It shows a table with one row for the newly created device 'MLX'. The table columns are Name, Type, Description, Enabled, and Status. The device is listed as 'Connected: Protocol: EIP'.

On this page ...

- [Connect to a Device](#)
- [Device Connection Settings](#)
- [Supported MicroLogix Connection Methods](#)



Connecting to MicroLogix

[Watch the Video](#)

- To see all the Tags, go to **OPC Client > Quick Client** in the **Config** section. On the **OPC Quick Client** page expand the **MicroLogix** folder which contains all the folders with the individual Tags.

Device Connection Settings

The **General** settings are common to all Allen Bradley devices, and the **Connectivity** settings are device dependent.

General	
Name	The user-defined name for this Device. The name chosen will show up in OPC Item Paths and under OPC-UA Server > Devices of the Configure page of the Gateway. The Device Name must be alphanumeric.
Description	The user-defined description for the device. This is only used as a note to differentiate between devices.
Enable Device	Only devices that are enabled appear in Connections > Devices of the Status page of the Gateway and thus have their tags available for use.
Connectivity	
Hostname	The Hostname value is the IP Address of the MicroLogix 1100 processor, MicroLogix 1400 processor or 1761-NET-ENI Ethernet interface. EthernetIP protocol on TCP port 44818 (0xAF12) is used to communicate to the listed devices.
Communication Timeout	After sending a request to the MicroLogix processor, the Communication Timeout setting is the amount of time in msec to wait for a response before treating it as a failure.
Browse Cache Timeout	When the data table layout is read from the MicroLogix processor, the Browse Cache Timeout value is the amount of time in msec to cache the results.

Supported MicroLogix Connection Methods

- MicroLogix 1100 and 1400 direct
- MicroLogix 1100 and 1400 connected through 1761-NET-ENI
- MicroLogix 1100/1400 connected through Spectrum Controls WebPort 500

Caution:

MicroLogix 1200 and 1500 are not fully supported. Browsing is not available on these devices, so the 'Disable Processor Browse' advanced property will need to be set to True on the device connection.

Additionally, the MicroLogix driver can not access Input Parameters.

Note: Some Micrologix devices may get stuck in a 'Browse Pending' status. In this case setting the connection path property to 1,0 should resolve the issue.

Related Topics ...

- [Connecting to PLC5](#)

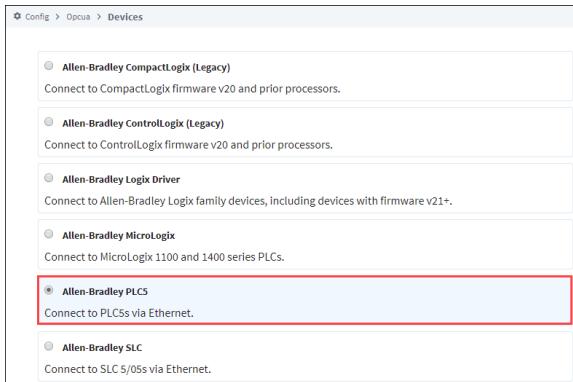
Connecting to PLC5

Connect to a Device

1. Go to the **Config** section of the Gateway Webpage.
2. Scroll down and select **OPC UA > Device Connections**.



3. On the **Devices** page, click on **Create new Device**.
4. Select **Allen-Bradley PLC5**, and click **Next**.



5. Fill in the following fields:
Name: **PLC5**
Hostname: type the **IP address** for the PLC, for example 10.20.4.56.

Leave the default values for the remaining fields.

A screenshot of a "Create New Device" form for a PLC5. The form has two main sections: "General" and "Connectivity".

General

Name	PLC5
Description	[Empty]
Enabled	<input checked="" type="checkbox"/> (default: true)

Connectivity

Hostname	10.20.4.56
Timeout	2000 (default: 2,000)
Browse Cache Timeout	240000 (default: 240,000)
Connection Path	[Empty]

Show advanced properties

Create New Device

6. Click **Create New Device**.
7. The **Devices** page is displayed showing the **PLC5** device is added to Ignition. The **Status** will show as Disconnected and then Connected.

On this page ...

- [Connect to a Device](#)
- [Device Connection Settings](#)
- [More Information On Connection Path](#)
 - [Supported PLC-5 Connection Methods](#)



INDUCTIVE
UNIVERSITY

Connecting to PLC5

[Watch the Video](#)



- To see all the Tags, go to **OPC Client > Quick Client** in the **Config** section. On the **OPC Quick Client** page, expand the **PLC5** folder and in the **Global** folder you can see all the tags.

Device Connection Settings

The **General** settings are common to all Allen Bradley devices, and the **Connectivity** settings are device dependent.

General	
Name	The user-defined name for this Device. The name chosen will show up in OPC Item Paths and under OPC-UA Server > Devices of the Configure page of the Gateway. The Device Name must be alphanumeric.
Description	The user-defined description for the device. This is only used as a note to differentiate between devices.
Enable Device	Only devices that are enabled appear in Connections > Devices of the Status page of the Gateway and thus have their tags available for use.
Connectivity	
Hostname	The Hostname value is the IP Address of the PLC-5 processor. The protocol that the PLC-5 processor supports is automatically detected. It will use either CSP protocol on port 2222 (0x8AE) or EthernetIP protocol on port 44818 (0xAF12).
Communication Timeout	After sending a request to the PLC-5 processor, the Communication Timeout setting is the amount of time in milliseconds to wait for a response before treating it as a failure.
Browse Cache Timeout	When the data table layout is read from the PLC-5 processor, the Browse Cache Timeout value is the amount of time in milliseconds to cache the results.
Connection Path	The Connection Path value is used to define the route of the PLC-5 processor to connect to. Currently routing through the ControlLogix Ethernet Communication Interface Module (1756-ENET) to the ControlLogix Data Highway Plus-Remote I/O Communication Interface Module (1756-DHRIOD) and on to a PLC-5 processor of the DH+ network is supported.
Advanced	
Disable Processor Browse	Disables the processor browse setting. (Default is false.)
Zero TNS Connection	Disables the Zero TNS connection setting. (Default is false.)

More Information On Connection Path

The Connection Path format contains 4 numbers separated by commas. The first number is always 1 and tells the 1756-ENET module to route through the backplane. The second number is the slot number of the 1756-DHRIOD module of the DH+ network the PLC-5 processor is connected to. The third number is the channel of the 1756-DHRIOD module that the PLC-5 processor is connected to. Use 2 for channel A and 3 for channel B. The final and fourth number is the DH+ node number. This number is in octal and is the same as configured in the PLC-5 processor. See the **ControlLogix Ethernet Communication Interface Module** User Manual for more information.

Connection Path Format: 1,<1756-DHRIOD slot number>,<1756-DHRIOD channel>,<DH+ node number>

The valid range for the 1756-DHRIOD slot number is between 0 and 16 but depends on the chassis size. The 1756-DHRIOD channel is either 2 for channel A or 3 for channel B. The DH+ node number range is from 00 to 77 octal.

Supported PLC-5 Connection Methods

PLC-5 L/20E, L/40E, L/80E direct
All PLC-5 processors connected through DH+ via the 1756-DHRIOD module.

Note: ASCII data types from a PLC5 are not supported by the Allen-Bradley PLC5 driver.

Related Topics ...

- [Connecting to SLC](#)

Connecting to SLC

Connect to an Allen-Bradley SLC Device

1. Go to the **Config** section of the Gateway Webpage.
2. Scroll down and select **OPC UA > Device Connections**.



3. On the **Devices** page, click on **Create new Device**.
4. Select **Allen-Bradley SLC**, and click **Next**.

Note: These PLCs do not have a native Ethernet connection, therefore another device like a Net ENI or an ENBT must be used for the connection.

The screenshot shows a list of device types under the 'Devices' category. The 'Allen-Bradley SLC' option is selected and highlighted with a red border. Other options include 'Allen-Bradley MicroLogix', 'Allen-Bradley PLCs', 'DNP3 Driver', 'Modbus RTU over TCP', and 'Modbus TCP'. Each option has a brief description below it.

5. Fill in the following fields:

Name: **SLC**

Hostname: **IP address** for the PLC, for example 10.20.4.56.

Leave the default values in the remaining fields.

The screenshot shows the configuration interface for the 'SLC' device. It includes two main sections: 'General' and 'Connectivity'. In the 'General' tab, the 'Name' field is filled with 'SLC', 'Enabled' is checked, and there are empty fields for 'Description' and 'Notes'. In the 'Connectivity' tab, the 'Hostname' is set to '10.20.4.56', 'Timeout' is '2000', 'Browse Cache Timeout' is '240000', and 'Connection Path' is empty.

On this page ...

- Connect to an Allen-Bradley SLC Device
- Device Connection Settings
- More Information On Connection Path
 - Supported SLC Connection Methods



Connecting to SLC

[Watch the Video](#)

6. Click **Create New Device**.

The **Devices** page is displayed showing the **SLC** device is added to Ignition. The **Status** will show as Disconnected and then Connected.

Config > Opcua > Devices					
Name	Type	Description	Enabled	Status	
Sim	Simulator: Generic		true	Running	<button>delete</button> <button>edit</button>
SLC	Allen-Bradley SLC		true	Connecting: Browse pending..	<button>More</button> <button>edit</button>

7. To see all the Tags, go to **OPC Client > Quick Client** in the **Config** section. On the **OPC Quick Client** page, expand the **SLC** folder and in the **Global** folder you can see all the tags.

Device Connection Settings

The general settings are common to all Allen Bradley devices, and the connectivity and advanced settings are device dependent.

General	
Name	The user-defined name for this Device. The name chosen will show up in OPC Item Paths and under OPC-UA Server > Devices of the Configure page of the Gateway. The Device Name must be alphanumeric.
Description	The user-defined description for the device. This is only used as a note to differentiate between devices.
Enable Device	Only devices that are enabled appear in Connections > Devices of the Status page of the Gateway and thus have their tags available for use.
Connectivity	
Hostname	The Hostname value is the IP Address of the SLC processor. The protocol that the SLC processor supports is automatically detected. It will use either CSP protocol on port 2222 (0x8AE) or EthernetIP protocol on port 44818 (0xAF12).
Communication Timeout	After sending a request to the SLC processor, the Communication Timeout setting is the amount of time in milliseconds to wait for a response before treating it as a failure.
Browse Cache Timeout	When the data table layout is read from the SLC processor, the Browse Cache Timeout value is the amount of time in milliseconds to cache the results.
Connection Path	The Connection Path value is used to define the route of the SLC processor to connect to. Currently routing through the ControlLogix Ethernet Communication Interface Module (1756-ENET) to the ControlLogix Data Highway Plus-Remote I/O Communication Interface Module (1756-DHRI0) and on to a SLC processor of the DH+ network is supported.
Advanced Options	
Disable Processor Browse	Disables the processor browse setting. (Default is false.)
Zero TNS Connection	Disables the Zero TNS connection setting. (Default is false.)

More Information On Connection Path

The Connection Path format contains 4 numbers separated by commas. The first number is always 1 and tells the 1756-ENET module to route through the backplane. The second number is the slot number of the 1756-DHRI0 module of the DH+ network the SLC processor is connected to. The third number is the channel of the 1756-DHRI0 module that the SLC processor is connected to. Use 2 for channel A and 3 for channel B. The final and fourth number is the DH+ node number. This number is in octal and is the same as configured in the SLC processor. See the ControlLogix Ethernet Communication interface Module User Manual for more information.

Connection Path Format: 1,<1756-DHRI0 slot number>,<1756-DHRI0 channel>,<DH+ node number>

The valid range for the 1756-DHRI0 slot number is between 0 and 16 but depends on the chassis size. The 1756-DHRI0 channel is either 2 for channel A or 3 for channel B. The DH+ node number range is from 00 to 77 octal.

Supported SLC Connection Methods

SLC505 direct
SLC505, SLC504, SLC503 connected through 1761-NET-ENI
SLC504 connected through 1756-DHRIO
SLC505, SLC504, SLC503 connected through Spectrum Controls WebPort 500

Allen-Bradley Connection Paths

Connections to ControlLogix, CompactLogix, PLC-5, MicroLogix and SLC Allen-Bradley processors through a ControlLogix Gateway require a connection path. The connection path is unique to your setup and is dependent on what modules the connection is being routed through. However, each connection path follows a basic set of rules outlined below.

Follow the Path

A connection path is a path that when followed leads from the ControlLogix gateway to a processor residing in a numbered slot of a chassis somewhere on site. You only have to build the connection path as you go.

On this page ...

- Follow the Path
 - Setting Up the Device Connection
 - Examples

Setting Up the Device Connection

The first connection point between Ignition and the device is a ControlLogix Ethernet module such as an ENET, ENBT, or EN2T module. This means that while the driver type used is the same type as the PLC you want to connect to, the hostname actually needs to point to the ControlLogix Gateway. The connection path is then followed to go out of the ControlLogix Gateway and into the PLC you are connecting to.

You need to have 6 numbers/entries to specify the connection path. The way you find each number is described in the following table:

1 st Number	Is 1 and means move to the back plane
2 nd Number	Is the slot number of the module you want to move to
3 rd Number	Is the exit port or channel of that module that you want to exit through
4 th Number	Is the address of entry point to the next module (DH+ Station Number / ControlNet Address / IP Address of ethernet module)
5 th Number	Is 1 and means move to the back plane (from this 5 th Number, it starts repeating as the 1 st Number)
6 th Number	Is the processor slot number OR the slot number of the module you want to move to

The process of coming up with these numbers may sound complicated at first but after some practice it gets easier.

Examples

Below there are a few examples that go over the differences with using specific modules to route the connection. Note that typically, the end result PLC does not matter. So while the example may show connecting to a PLC5 through ControlNet, the PLC5 could easily be swapped out with another PLC, and the connection path would remain the same. The big change when using different end result PLCs is what device driver to use for the connection.

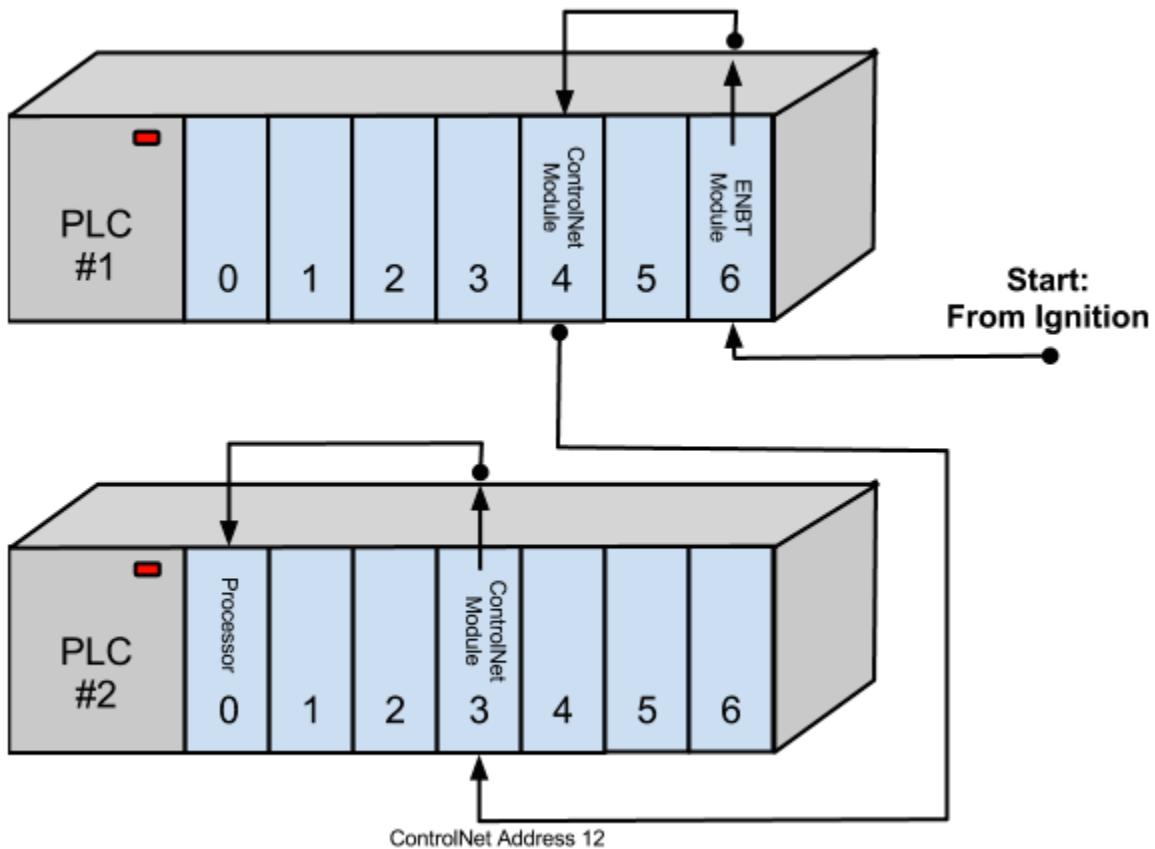
Note: Understanding how the Connection Paths are built listed above is important to understand the specifics of using different modules in the examples below.

Connecting to a PLC through ControlNet

The example below connects to a PLC5 using ControlNet that has a **connection path** of **1,4,2,12,1,0** going from **PLC 1** to **PLC 2**. The device driver is the Allen-Bradley PLC5, and the hostname is the IP address of the ControlLogix Gateway (PLC #1).

1	Move out of ENBT Module to the backplane
4	Move to ControlNet Module in Slot 4

2	Move out ControlNet Port
12	Move in ControlNet Module at ControlNet Address 12
1	Move out of ControlNet to the backplane
0	Move to the Processor in Slot 0



ControlNet Example

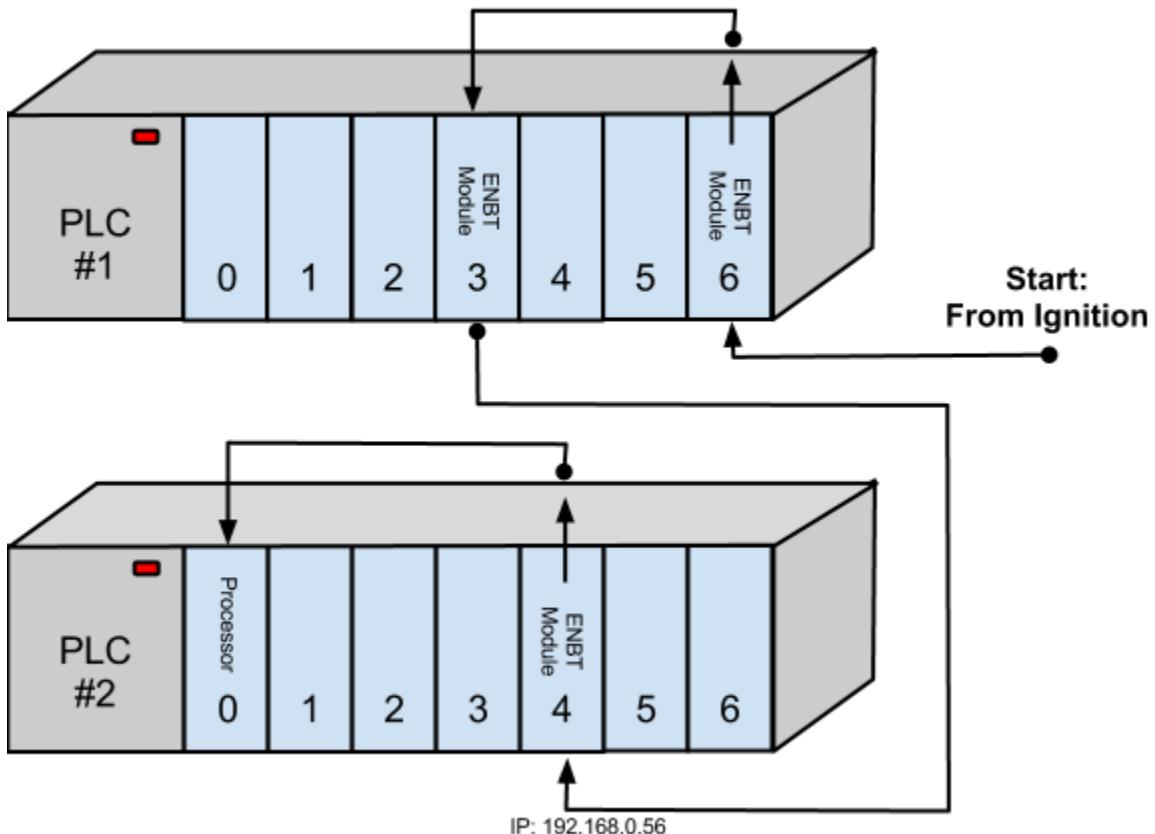
[Watch the Video](#)

Connecting to a PLC using ENBT

The example below connects to a ControlLogix using ENBT that has a **connection path** of **1,3,2,192.168.0.56,1,0** going from **PLC 1** to **PLC 2**. The device driver is the Allen-Bradley ControlLogix, and the hostname is the IP address of the ControlLogix Gateway (PLC #1).

1	Move out of ENBT Module to the backplane
3	Move to ENBT Module in Slot 3

2	Move out the Ethernet Port
192.168.0.56	Move in EBNT Module at IP Address 192.168.0.56
1	Move out of EBNT to the backplane
0	Move to the Processor in Slot 0



An ENBT Example

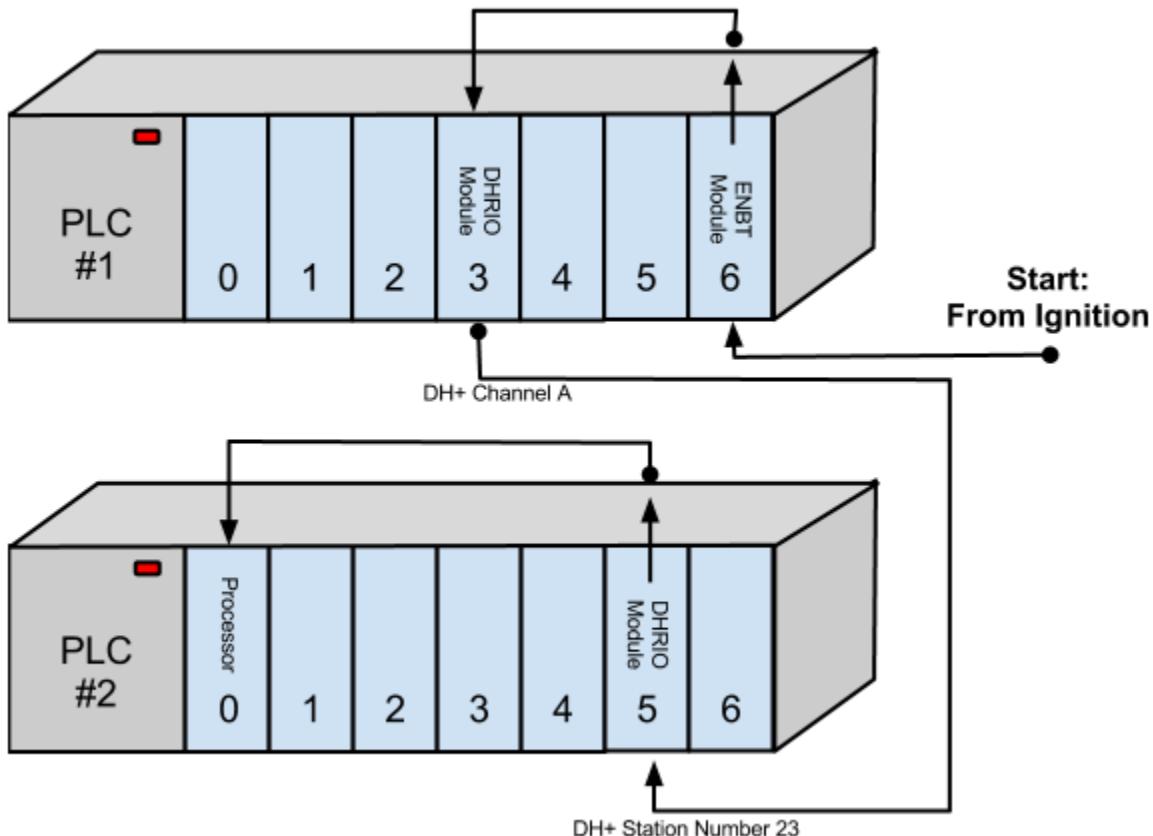
[Watch the Video](#)

Connecting to a ControlLogix using Data Highway Plus (DH+)

The example below connects to a ControlLogix using Data Highway Plus that has a **connection path** of **1,3,2,23,1,0** going from PLC 1 to PLC 2. The device driver is the Allen-Bradley ControlLogix, and the hostname is the IP address of the ControlLogix Gateway (PLC #1).

1	Move out of ENBT Module to the backplane
3	Move to DHRIO Module in Slot 3

2	Move out DH+ Channel A
23	Move in DH+ Station Number 23
1	Move out of DHRIO Module to the backplane
0	Move to the Processor in Slot 0



Connecting to multiple SLCs using Data Highway Plus (DH+)

The example below connects to 3 different SLCs using Data Highway Plus. Each SLC connection would have their own device connection, each with a unique connection path.

The connection paths are:

- SLC X 1,3,2,21
- SLC Y 1,3,3,40
- SLC Z 1,3,3,41

The device driver is the Allen-Bradley ControlLogix, and the hostname is the IP address of the ControlLogix Gateway (PLC #1).

SLC X

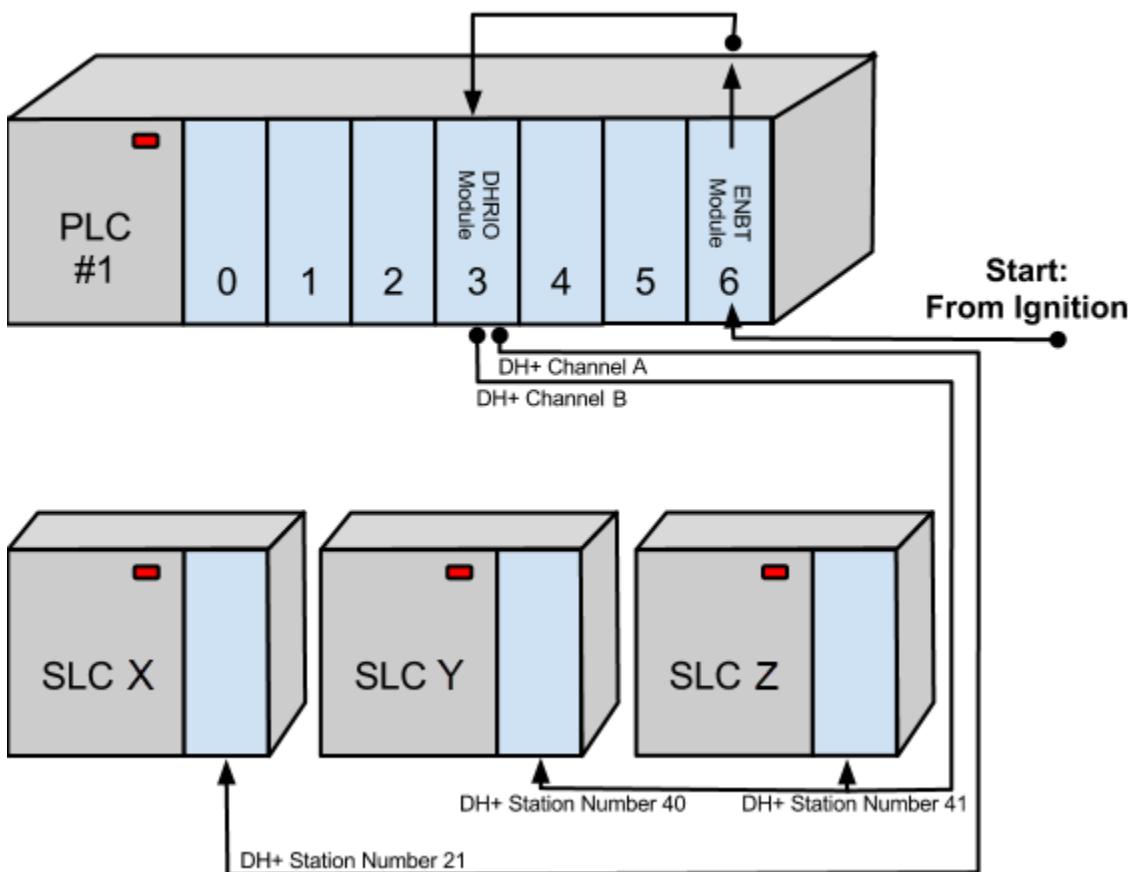
1	Move out of ENBT Module to the backplane
3	Move to DHRIO Module in Slot 3
2	Move out DH+ Channel A
21	Move in DH+ Station Number 23

SLC Y

1	Move out of ENBT Module to the backplane
3	Move to DHRIO Module in Slot 3
3	Move out DH+ Channel B
40	Move in DH+ Station Number 40

SLC Z

1	Move out of ENBT Module to the backplane
3	Move to DHRIO Module in Slot 3
3	Move out DH+ Channel B
41	Move in DH+ Station Number 41



INDUCTIVE
UNIVERSITY

A DH+ Example

[Watch the Video](#)

Modbus

Modbus Devices

Unlike some of the other drivers (like Allen Bradley, Siemens) Modbus is a standard and not specific to a brand of device. Modbus is a protocol used for connecting to many types of devices. This is a huge benefit, but can lead to a lot of confusion because different device manufacturers may design their devices differently, and the documentation for the different manufacturers varies wildly in quality and availability. The main idea behind the Modbus standard is that you should have a regular pattern for tags inside the device, and use the same connection methods.

Our generic Modbus driver allows the Ignition OPC-UA server to communicate with any device that supports the Modbus TCP protocol or the RTU over TCP protocol. Because of this, the Modbus driver can connect directly to any devices that support Ethernet communications, even if they use the older RTU standard. If you're not sure which connection type to use, it's probably not RTU. However, it's not difficult to just try both and see which works.

Connecting to a Device

The [Connecting to Modbus Device](#) section contains step-by-step instructions on how to connect to a Modbus device. It is important to only add one Modbus device connection to Ignition for each IP address, regardless of how many devices are using that IP address. When communicating to multiple Modbus devices on the same IP address where each has a unique unit ID, either include the unit ID in the Modbus specific address or set it in the [address mapping](#) for the device.

It's easy to get connected to a Modbus device, but figuring out the addressing can be time consuming. Even with poor (or missing) device documentation, it's just a little bit of trial and error to get your addressing set.

Modbus Addresses

Getting access to your Modbus tags can be confusing because of all the different options available in the device connection. For example, you could have 0/1 based addressing or reversed word order that isn't clearly documented in the device instructions. It's helpful to manually create a few Tags in Ignition and change your connection settings until your values are correct.

One important setting to note is the **Max Holding Registers Per Request**. This setting can cause all of your test tags to go to bad quality when only one is bad. Change this setting (under the Advanced properties) to 1 while you are testing, and set it back when you are done. If you leave it at 1, this will cause a huge strain on your system after you have added hundreds or thousands of tags from your device.

Manually Addressing

[Manually creating Tags](#) in Ignition is pretty easy. If you want to look at Holding Register 1 (a common address in Modbus), the OPC Item Path is "[devicename]HR1". It's that simple! Try setting up HR0, HR1, and HR2 while you are testing to help figure out your connection settings.

Address Mapping

Creating an [Address Map](#) in Ignition for your device allows you to drag and drop tags just like any of the browseable device connections (like an Allen Bradley ControlLogix). You can even import and export your maps to make it quick and easy to set up many devices. Once you have your mapping set up, just drag your Tags into Ignition and start designing. Make sure to test a few tag values when you get the Address Mapping set up. Modbus devices cannot verify that your mapping is correct, it just creates a list of tags for you.

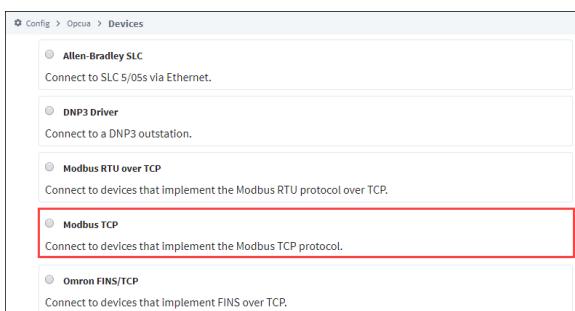
In This Section ...

Connecting to Modbus Device

1. Go to the **Config** section of the Gateway Webpage.
2. Scroll down and select **OPC UA > Device Connections**.



3. On the **Devices** page, find the blue arrow and click on **Create new Device**. There are two modules to choose from
 - **Modbus RTU over TCP**
Which connects to devices that implement the Modbus RTU protocol over TCP.
 - **Modbus TCP**
Which connects to devices that implement the Modbus TCP protocol.
 - **Modbus RTU** (only available if the **Serial Support Gateway** module is installed)
Connects to a device via Modbus RTU over serial
4. In this example we'll select **Modbus TCP**, and click **Next**.



5. On the **New Device** page, leave all the default values and type in the following fields:
Name: **Modbus**
Hostname: type the **IP address**, for example 10.20.8.117.

A screenshot of the 'New Device' configuration form. It has two main sections: 'General' and 'Connectivity'.
General:

- Name: Modbus
- Description: (empty)
- Enabled: checked (default: true)

Connectivity:

- Hostname: 10.20.8.117
- Timeout: 2000 (default: 2,000)
- Connection Path: (empty)
- Concurrent Requests: 2 (default: 2)

At the bottom, there are two buttons: 'Show advanced properties' (unchecked) and a prominent blue 'Create New Device' button.

On this page ...

- Modbus Drivers using Ethernet
 - Supported Functions Codes
 - Device Connection Settings



INDUCTIVE
UNIVERSIT

Connecting to Modbus Device

[Watch the Video](#)

6. You can check the box for **Show advanced properties?** to see the additional settings, or you can keep all the defaults.

7. Click **Create New Device**.

The **Devices** page is displayed showing the **Modbus** device is successfully created and added to Ignition. The **Status** will show as Disconnected and then Connected.

Config > Opcua > Devices				
Successfully created new Device "Modbus"				
Name	Type	Description	Enabled	Status
SIM	Simulator: Generic		true	Running
Modbus	Allen-Bradley CompactLogix (Legacy)		true	Connecting

Unlike other PLCs, Modbus devices do not support browsing, therefore you can not browse the Tags by going to the **OPC Connections > Quick Client** in the **Configure** section of the Gateway. To see and browse the Tags, you need to create the Tags as described in the [Modbus Addressing](#) section.

Modbus Drivers using Ethernet

The generic Modbus driver allows the Ignition OPC-UA server to communicate with any device that supports Modbus TCP protocol. The Modbus driver can connect directly to devices that support Ethernet communications. It can also connect to Modbus devices through a Gateway.

It is important to only add one Modbus device in the Ignition Device List per IP address. When communicating to multiple Modbus devices through a Gateway each with a unique unit ID, either include the unit ID in the [Modbus specific address](#) or set it in the [address mapping](#) for the device.

Supported Functions Codes

The Modbus driver supports the functions codes listed below. In some cases, a device may not allow certain function codes. To remedy this, advanced properties on the device connection can restrict or force specific functions codes. See the **Driver Properties** table on this page for more details.

Function Name	Code	Hex
Read Discrete Inputs	02	0x02
Read Coils	01	0x01
Write Single Coil	05	0x05
Write Multiple Coils	15	0x0F
Read Input Register	04	0x04
Read Holding Register	03	0x03
Write Single Register	06	0x06
Write Multiple registers	16	0x10

Device Connection Settings

General	
Name	The user-defined name for this Device. The name chosen will show up in OPC Item Paths and under OPC-UA Server > Devices of the Configure page of the Gateway . The Device Name must be alphanumeric.
Description	The user-defined description for the device. This is only used as a note to differentiate between devices.
Enable Device	Only devices that are enabled appear in Connections > Devices of the Status page of the Gateway and thus have their Tags available for use.

Connectivity (Modbus RTU over TCP and Modbus TCP connections)

Hostname	Is the IP Address of the Modbus device.
Port	Is the port to use when connecting to a Modbus device. The Modbus TCP port specified in the Modbus specification is 502, but it can be changed to a different port.

Communication Timeout	<p>Is the amount of time in milliseconds to wait for a response before treating it as a failure, after sending a request to the Modbus device.</p> <p>Note: When working with a Modbus RTU over TCP connection, each "device" would be an individual Modbus device on the Modbus network.</p>
-----------------------	--

Connectivity (Modbus RTU connections)	
Serial Port	The name of the Serial Port (i.e COM1).
Bit Rate	The bit rate for the connection.
Data Bits	Data bits for the connection.
Parity	Parity configuration for the connection.
Stop Bits	Determines the stop bits for the connection.
Handshake	Determines the handshake for the connection.
Communication Timeout	Determines the maximum amount of time the connection will wait for a response.

Advanced	
Concurrent Requests	<p>This feature is new in Ignition version 8.1.0 Click here to check out the other new features</p> <p>The number of requests that Ignition will try to send to the device at the same time. Increasing this number can sometimes help with your request throughput, however increasing this too much can overwhelm the device and negatively impact communications with the device.</p>
Max Holding Registers per Request	The maximum number of Holding Registers the device can handle. Because some Modbus devices cannot handle the default of requesting 125 Holding Registers in one request, to accommodate this limitation you can change this setting.
Max Input Registers per Request	The maximum number of Input Registers the device can handle. Because some Modbus devices cannot handle the default of requesting 125 Input Registers in one request, to accommodate this limitation you can change this setting.
Max Coils per Request	The maximum number of Coils the device can handle. Because some Modbus devices cannot handle the default of requesting 2000 Coils in one request, to accommodate this limitation you can change this setting.
Max Discrete Inputs per Request	The maximum number of Discrete Inputs the device can handle. Because some Modbus devices cannot handle the default of requesting 2000 Discrete Inputs in one request, to accommodate this limitation you can change this setting.
Reverse Word Order	When reading and writing 32bit values from/to a Modbus device, the low word comes before the high word. By checking this option, the high word comes before the low word. The Modbus specification does not include a section for reading and writing 32bit values and as a result device manufacturers have implemented both methods.
Zero-based Addressing	<p>When this option is checked, the address range for each area starts at 0. If unchecked, the range starts at 1.</p> <p>The Modbus specification states that Modbus addresses are to be zero based. Meaning Modbus addresses start at 0 instead of 1. To read a value from Modbus address 1024, 1023 is sent to the device. When connecting to devices that do not adhere to zero based addressing, make sure this option is not selected.</p>
Span Gaps	When this option is checked, it spans address gaps when optimizing requests, reducing the number of requests but increasing the amount of data requested at once. If unchecked, it does not span the address gaps.
Allow Write Multiple Registers Request	<p>Enable or disable Modbus function code 0x10, Write Multiple Registers. Some devices may not support this function code.</p> <p>Caution: Disabling this option will break the ability to write 32-bit and String values correctly to registers.</p>
	Force the use of Modbus function code 0x10, Write Multiple Registers, on write requests.

Force Multiple Register Writes	
Allow Write Multiple Coils Request	Enable or disable Modbus function code 0x0F, Write Multiple Coils. Some devices may not support this function code.
Allow Read Multiple Registers Request	If disabled all registers will be read in individual read requests. Disable with caution.
Allow Read Multiple Coils	If disabled all coils will be read in individual read requests. Disable with caution.
Allow Read Multiple Discrete Inputs	If disabled all discrete inputs will be read in individual read requests. Disable with caution. Note: Function code 0x02 is always used to read Discrete Inputs, regardless what this property is set to.
Reconnect After Consecutive Tries	
Max Retry Count	Maximum number of times to retry a read request after receiving a response containing an allowable Exception Code (GatewayPathUnavailable, GatewayTargetDeviceFailToRespond, SlaveDeviceBusy, or SlaveDeviceFailure). Default is 1.

String Handling	
Reverse String Byte Order	When reading and writing string values from/to a Modbus device, the low byte comes before the high byte. By checking this option the high byte comes before the low byte. If reading a string value from a device should read ABCD but BADC appears in Ignition, then check this option.
Right Justify Strings	Strings stored in a Modbus device may contain leading spaces or trailing spaces. This can produce unwanted results so that Modbus driver removes spaces or zeros when reading string values. By default, left justify string handling is used when reading and writing strings. When you check this option, right justify string handling is used.
Read Raw Strings	Whether or not to read the entire length of a string, ignoring any null bytes encountered.

Related Topics ...

- [Modbus Addressing](#)

Modbus Addressing

Manually Addressing a Modbus Device

Modbus doesn't support tag browsing, this means you cannot view the Tags in the Tag Creator of the Designer or from the OPC Connections > Quick Client in the Config section of the Gateway.

There are two ways you can create Tags so that you can browse them:

1. By manually specifying each address
This is done from the Designer by entering Modbus Specific Addresses into the OPC Item Path of an OPC Tag. See below for detailed information.
2. By specifying the address mapping
This is done from the Gateway, see the [Modbus Address Mapping](#) section.

On this page ...

- [Manually Addressing a Modbus Device](#)
- [Manually Specify Each Address](#)
- [Modbus Specific Addressing](#)
 - [Manually Create an Address for a Single Tag](#)
 - [Bit-Level Addressing](#)



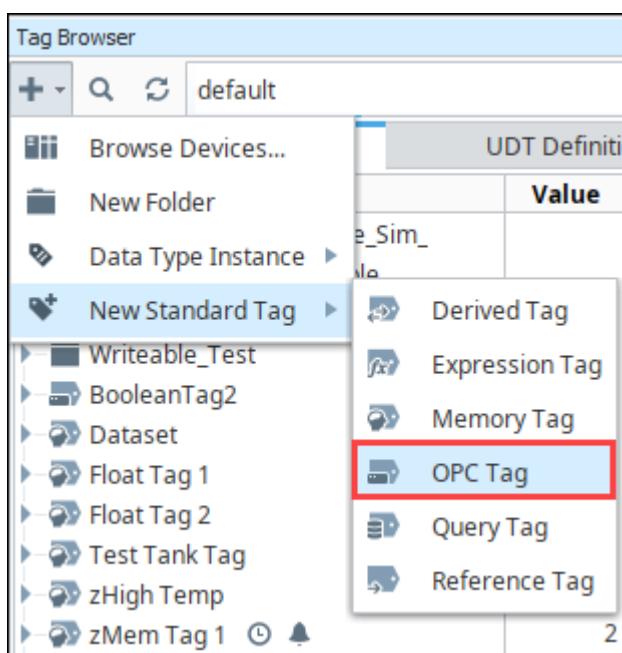
About Modbus Addressing

[Watch the Video](#)

Manually Specify Each Address

You can enter Modbus specific addresses into the OPC Item Path of an OPC Tag by using the following designators along with the Modbus address:

1. In the **Tag Browser**, click the Add icon.
2. Select **New Standard Tag** then **OPC Tag**.



3. In the **Tag Editor** window, as an example, you can set the following values:
Name: **Temp**

Data Type: **Integer**

OPC Server: Choose **Ignition OPC UA Server** from the dropdown

OPC Item Path: **[Modbus]HR1**. The **Modbus** device name goes in the square brackets. Next you give the address to PLC which in this case is the **HR** designator plus **1** as the Modbus address. The **Modbus Specific Addressing** section below, explains how your can construct these addresses.

4. Click **OK**.

Now you can see the Temp tag in the Tag Browser.

Modbus Specific Addressing

Per the Modbus protocol specification, the following four basic types of addresses can be read from a device:

- Holding Registers (read/write 16 bit words)
- Input Registers (read only 16 bit words)
- Coils (read/write bits)
- Discrete Inputs (read only bits associated with device input points)

Manually Create an Address for a Single Tag

To manually enter Modbus Specific Addresses into the **OPC Item Path** of the **Tag Editor** window, use one of the following designators plus the Modbus address:

Note: Other OPC servers represent each type by starting the OPC address with a number, for example, 4 for holding registers.

Designator	Description
HR	for 16 bit signed Holding Register (HR1, equivalent to 40001 in other applications)
IR	for 16 bit signed Input Register (IR1, equivalent to 30001)
C	for Coil (C1, equivalent to 00001)
DI	for Discrete Input (DI1, equivalent to 10001)

For example, to use these designators with the Modbus address you would enter **HR1** in the OPC Item Path of an OPC Tag in the Tag Editor window, which is the **HR** designator plus the Modbus address **1**.

Because some devices that support Modbus protocol store data in **BCD format**, there are two additional designators. These designators convert the data from BCD format to decimal when reading data from the device and convert data from decimal to BCD when writing to the device.

Designator	Description
HRBCD	for Holding Register with BCD conversion.
HRBCD_32	for 2 consecutive Holding Registers with BCD conversion.
IRBCD	for Input Register with BCD conversion.
IRBCD_32	for 2 consecutive Input Registers with BCD conversion.

To accommodate other data encoding commonly used by Modbus supported devices, the following designators are available for Modbus specific addressing:

Description	Holding Register Designator	Input Register Designator
Float/Double		
2 consecutive Registers with Float conversion.	HRF	IRF
4 consecutive Registers with Double conversion.	HRD	IRD
Integer		
Holding Registers with 16 bit unsigned integer conversion.	HRUS	IRUS
2 consecutive Registers with 32 bit integer conversion.	HRI	IRI

2 consecutive Registers with 32 bit unsigned integer conversion.	HRUI	IRUI
4 consecutive Registers with 64 bit integer conversion.	HRI_64	IRI_64
4 consecutive Registers with 64 bit unsigned integer conversion.	HRUI_64	IRUI_64

To read or write string values from/to a Modbus device, the following designation is available for Modbus specific addressing:

Designator	Description
HRS	read or write consecutive Holding Registers as a string value.

Note: There are two characters for each word and the order of which character comes first is controlled by the Reverse String Byte Order device setting as described in the [Connecting to Modbus Device](#) section. Because two characters are stored in a word, the string length must be an even number of characters.

HRS FORMAT: HRS<Modbus address>:<length>

Examples	Description
[DL240]HR1024	Read 16bit integer value from Holding Register 1024.
[DL240]HRBCD1024	Read BCD value from Holding Register 1024.
[DL240]IR512	Read 16bit integer value from Input Register 512.
[DL240]C3072	Read bit value from Coil 3072.
[DL240]IR0	Read 16bit integer value from Input Register 0.
[DL240]HRS1024:20	Read 20 character string value starting at Holding Register 1024.

Unit ID

You can also specify the Modbus unit ID by pre-pending it to the Modbus address. For example, to access Modbus unit ID 3 and read HR1024, the full OPC path is as follows:

[DL240]3.HR1024

Bit-Level Addressing

For bit-level addressing, you just append a period and the bit number you want to read and write to a bit. Your Modbus device must support the Mask Write command (your device documentation should specify if it does).

To read or write to a specific bit within a holding register, simply append the location of the bit as demonstrated in these examples:

[DL240]HR1024.0 will read and write to the first bit of the holding register.

[DL240]HR1024.10 will read and write to the 11th bit of the holding register.

Related Topics ...

- [Modbus Address Mapping](#)

Modbus Address Mapping

Because it can be tedious to manually enter OPC Tag information one-by-one, the driver offers an address mapping feature. This feature allows entering blocks of common addresses and the driver will create the individual addresses and display them in the Tag Creator.

Another benefit of address mapping is that the addresses inside a device can have a different numbering scheme than the Modbus address. The Direct Automation DL240 is a perfect example of this. Address V2000, capable of holding a 16 bit integer, is Modbus Holding Register 1024. In addition, the DL240 addressing is in octal meaning there are no 8 or 9s. The sequence of addresses are: V2000, V2001, V2002, V2003, V2004, V2005, V2006, V2007, V2010, V2011.... V3777. This is not very straight forward.

On this page ...

- [Address Mapping Properties](#)
- [Simple Mapping Demonstration](#)
- [Specify the Address Mapping](#)
- [Address Mapping Multiple Devices](#)
- [Floating Point or 32-bit Address Mapping](#)
 - [Mapping Float Point Addresses](#)
- [Import / Export Address Mapping](#)



INDUCTIVE
UNIVERSITY

About Modbus Address Mapping

[Watch the Video](#)

Address Mapping Properties

Name	Description
Prefix	A prefix applied to each mapped address as they appear in the Tag Creator. Must compose of letters, numbers, and underscore characters. The following values are reserved, and may not be used: HR , IR , C , or DI
Start and End	Numerical values will be assigned to each mapped addresses. These properties determine the range of the numerical assignments. Follows the Prefix. The difference between these two values determines how many mapped addresses will be created.
Step	When enabled, adjacent addresses will be combined. This is commonly used to combine two words (16-bit addresses) into a double word (32-bit addresses). Please see the Floating Point or 32-bit Address Mapping for more details.
Radix	The base number of unique digits for modbus addresses. Determines what format addresses in the device are incremented and labeled (HR0, HR1,...HR9, HR10, HR11). Common values are 10 (decimal system) or 16 (hexadecimal).
Unit ID	The Unit Id for the device to use. When several Modbus devices are connected to a single IP address, the step determines which device the mapping should be applied against. A value of 0 means the first device, and should be used when only a single Modbus device is connected. See Address Mapping Multiple Devices for more details.
Modbus Type	The table each mapped address should run against, as well as the type and size of each address.
Modbus Address	The address in the device that mapping will begin at. Since the Modbus Type property denotes which table the address will run against, the value here does not need to start with the entity number. Thus, when attempting to start a mapping at the 100th Holding Register, the value on this property would be 100 (assuming one-based addressing), not 40100; the Modbus Type property determines what the leading number is.

Simple Mapping Demonstration

Temperature readings are being stored in 10 16-bit addresses: 40,010 - 40,019. There is a single Modbus device at the IP address (unit ID 0), and the addresses are decimal. The mapped addresses should appear in the Tag Creator as "Temp1", "Temp2", and so-on. The following configuration would be used:

Prefix: Temp
Start: 1
End: 10
Step: False
Unit ID: 0
Modbus Type: Holding Register (Int16)
Modbus Address: 10

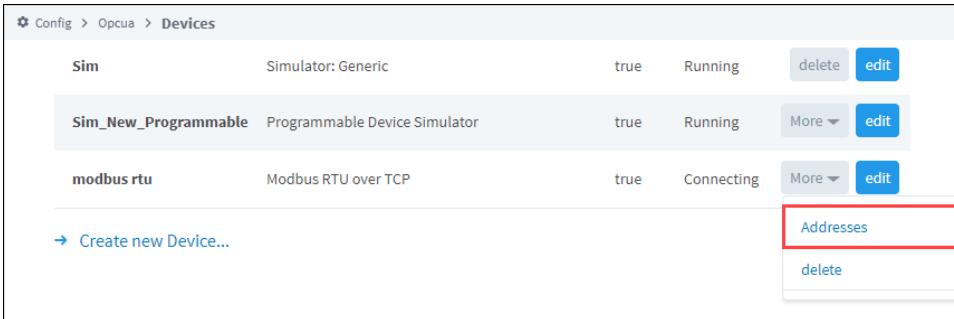
Note: Address 40,010 (HR10) uses Modbus Address 10 as a starting point and not 40,010. The leading 4 is automatically entered for you when you select Holding Register from the dropdown. The same is true for all other types of Tags: Inputs are 30,000, Discrete are 10,000, etc.

Prefix	Start	End	Step	Unit ID	Modbus Type	Modbus Address
Temp	1	10		0	Holding Register (Int16)	10 [delete]
Radix	10	[]				

The above configuration results in the items Temp1 through Temp10 appearing in the [Tag Creator](#)

Specify the Address Mapping

1. Go to the **Config** section of the Gateway Webpage.
2. Scroll down to **OPC-UA > Device Connection**.
3. Click on the **More** button and select **Addresses** to the right of your Modbus device.



The screenshot shows the 'Devices' section of the OPC-UA configuration. It lists three devices: 'Sim' (Simulator: Generic), 'Sim_New_Programmable' (Programmable Device Simulator), and 'modbus rtu' (Modbus RTU over TCP). The 'modbus rtu' row has a status of 'Connecting'. To the right of the device list, there is a button labeled 'Addresses' which is highlighted with a red box. Below the device list, there is a link '→ Create new Device...'.

4. Click on **Add Row**.

Config > Opcua > Devices

Address Configuration

Choose File No file chosen

Import Configuration

No file chosen

Export Configuration

Prefix	Start	End	Step	Unit ID	Modbus Type	Modbus Address
Radix	10					

There are currently no address mappings configured...

Add Row

Save

5. Enter the mapping as follows:

Prefix: V

Start: 2000

End: 3777

Modbus Type: Holding Register (Int16)

Modbus Address: 1024

Radix: 8 (8 causes the addresses to be in octal, also known as base 8)

Config > Opcua > Devices

Address Configuration

Choose File No file chosen

Import Configuration

Export Configuration

Prefix	Start	End	Step	Unit ID	Modbus Type	Modbus Address
V	2000	3777		0	Holding Register (Int16)	1024 [delete]
Radix	8					

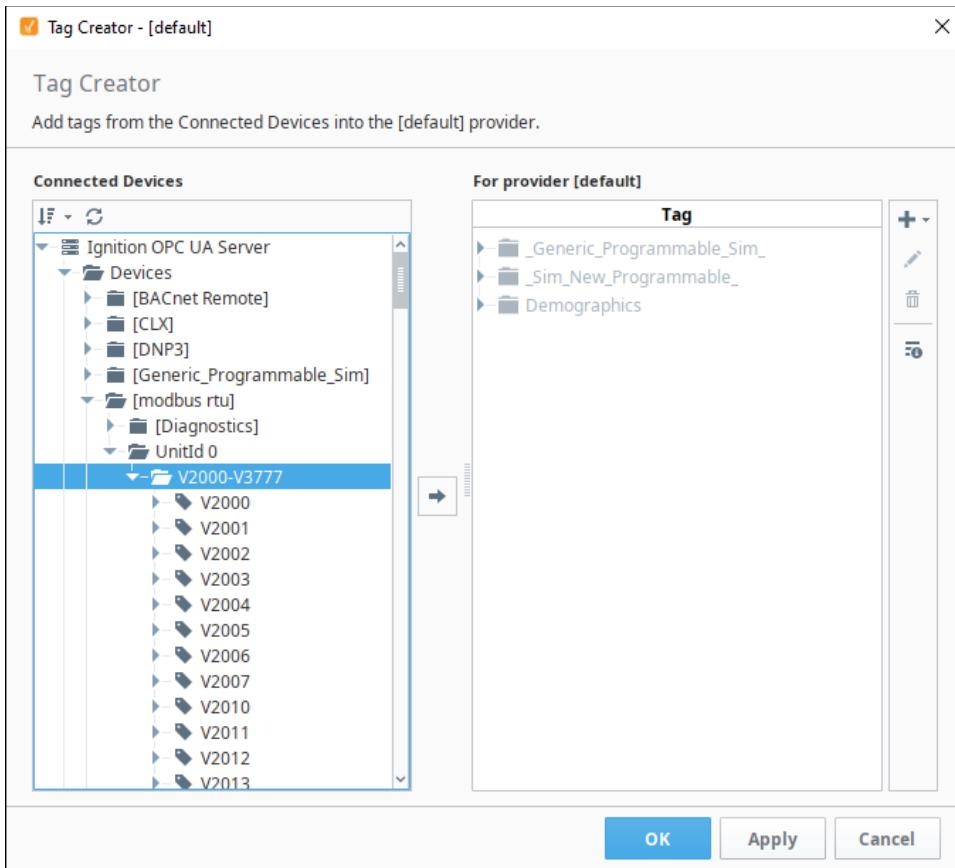
These settings map the **Modbus address range V2000 to V3777** in octal to **Modbus Holding Register addresses 1024 to 2047**.

Note: The mappings for string data types cannot be entered. Strings can only be read or written using Modbus Specific Addressing.

6. Click **Save**.

7. Go to the Tag Creator in Designer or the OPC Connections > Quick Client (on Gateway), and open the **Modbus** folder.

You can now see all the Modbus addresses from V2000 to V3777 in octal.



Here is an example of mapping for all of the Modbus DL240 addressing.

Prefix	Start	End	Step	Unit ID	Modbus Type	Modbus Address	
V	2000	3777		0	Holding Register (Int16)	1024	[delete]
X	0	477		0	Discrete Input	2048	[delete]
Y	0	477		0	Coil	2048	[delete]
TAO	0	127		0	Input Register (BCD16)	0	[delete]
Radix	8						

Address Mapping Multiple Devices

It is not recommended to communicate to multiple Modbus devices through a Modbus Gateway where Gateway has the same address. Therefore, do not add multiple Modbus devices with the same IP address.

Only add one Modbus device to the Ignition OPC UA Server device list for Gateway and specify the different unit IDs in the address mapping. The unit ID is specified for each entry in the address mapping for the Modbus device. Notice in the example below, the Prefix, Start, End, Modbus Type and Modbus Address can be the same for two entries provided that the Unit IDs are different.

Prefix	Start	End	Step	Unit ID	Modbus Type	Modbus Address
V	2000	3777	<input type="checkbox"/>	0	Holding Register (Int16)	1024 [delete]
V	2000	3777	<input type="checkbox"/>	1	Holding Register (Int16)	2024 [delete]
Radix	8					

Now when browsing the Modbus device, the unit ID will show as a folder and the OPC Tag path includes the unit ID. This only happens when more than one unit ID is specified in the address mapping otherwise the unit ID is eliminated.

Floating Point or 32-bit Address Mapping

Modbus only supports reading and writing to memory types of bits and 16-bit words. This is not very useful when reading from or writing to float point or 32-bit integers.

To work around this problem, the Modbus driver is designed to read two consecutive 16-bit words and encode it into the desired data type.

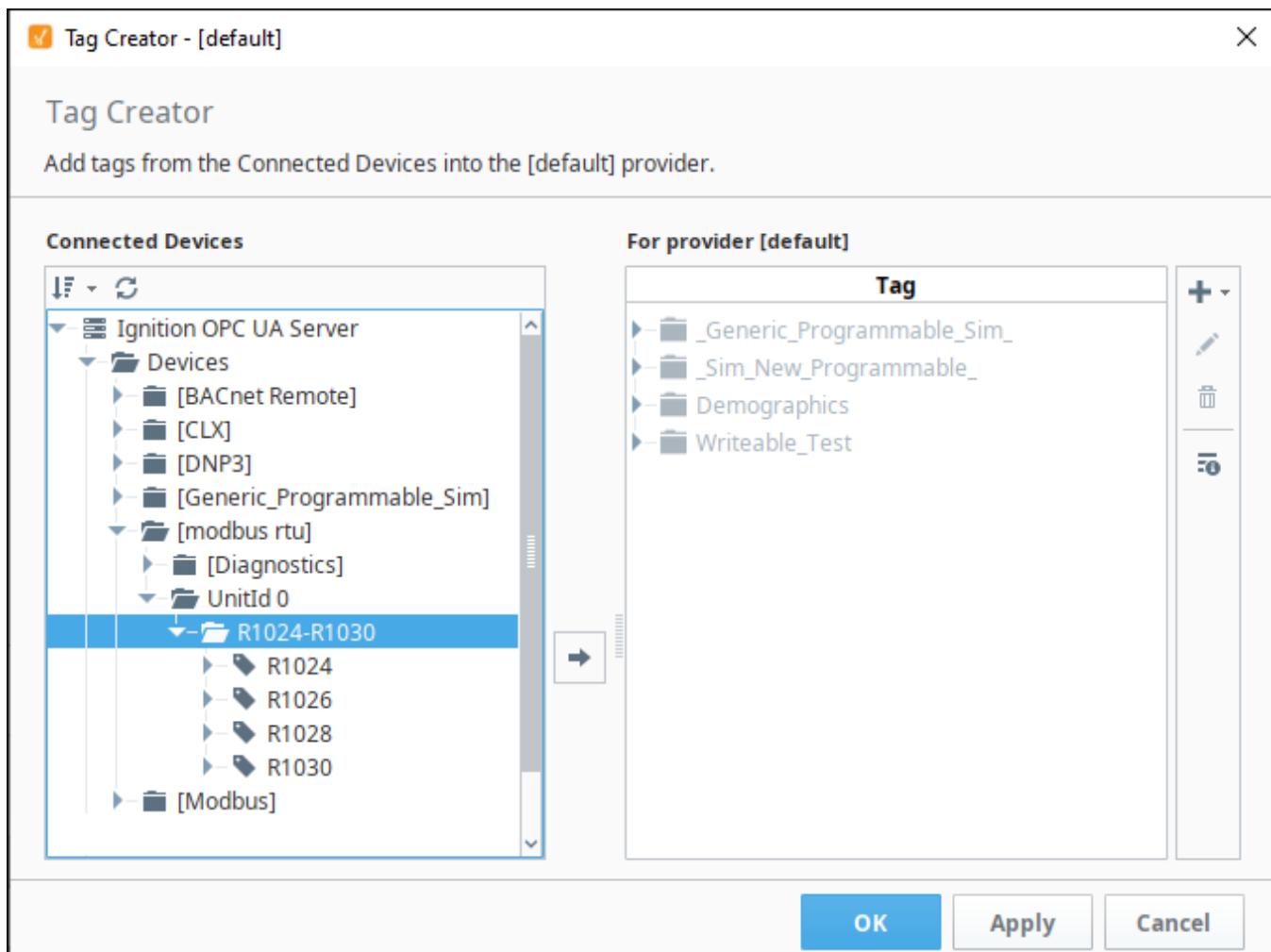
Mapping Float Point Addresses

The Modbus address mapping below shows how to map float point addresses starting at 1024 and ending at 1030. With the box in the **Step** column checked, the addresses on the Ignition side will index by 2. In this case, R1024, R1026, R1028 and R1030 will be created.

Because **Modbus Type of Holding Register (Float)** is selected, the driver will read two consecutive 16-bit words and convert it to a floating point value. It also indexes the Modbus Address by 2 for each entry. In this case, R1024 reads from Modbus addresses 1024 and 1025 and converts them into a floating point value. When writing, the reverse of converting a floating point value into two 16-bit words is done before sending them to the device.

Prefix	Start	End	Step	Unit ID	Modbus Type	Modbus Address
R	1024	1030	<input checked="" type="checkbox"/>	0	Input Register (Float)	1024 [delete]
Radix	10					

The following window shows what is displayed in the Tag Creator. Notice that the numbering is indexed by two and that it matches the Modbus address. With some devices, this allows the addresses displaying in the Tag Creator to match the addresses in the device.



Import / Export Address Mapping

The mapping configuration can be exported to a comma separated values (CSV) file. The CSV file can later be imported in other Ignition installations or similar devices. You can find a few examples of CSV files on our website. To see the examples, go to:

<https://inductiveautomation.com/downloads/extra-material>

Scroll down to **Modbus Templates** and double-click on the template files to see an example of the CSV file.

Siemens

The Siemens drivers in Ignition support basic connections to S7 devices. Ignition connects to these PLCs via TCP/IP using the S7 protocol. Similar to Modbus and some Allen Bradley connections, the Siemens S7 devices do not support Tag browsing. You can create S7 Tags manually in Ignition, or use Ignition's Tag import/export to create all of your Tags quickly in Excel or another spreadsheet program. Currently, Ignition has drivers for the following Siemens PLCs:

- S7-300
- S7-400
- S7-1200
- S7-1500

Considerations for 1200 and 1500 Devices

The following considerations and configurations changes must be made when using the S7-1200 and S7-1500 drivers:

1. Only global DBs can be accessed.
2. The optimized block access must be turned off.
3. The access level must be "full" and the "connection mechanism" must allow GET/PUT.
4. Reads and Writes can not be used in TM/CT areas.

Connect to a Siemens Device

1. Go to the **Config** section of the Gateway Webpage.
2. Scroll down and select **OPC UA > Device Connections**.
3. On the Devices page, click on **Create new Device**.
4. On the Add Device Step 1: Choose Type page, select **Siemens S7-1200**, and click **Next**.
5. There are four modules to choose from
 - **Siemens S7-1500**
Which connects to Siemens S7-1500 PLCs over Ethernet.
 - **Siemens S7-1200**
Which connects to Siemens S7-1200 PLCs over Ethernet.
 - **Siemens S7-300**
Which connects to Siemens S7-300 PLCs over Ethernet.
 - **Siemens S7-400**
Which connects to Siemens S7-400 PLCs over Ethernet.
6. On the New Device page, leave all the default values and type in the following fields:
Name: **S71200**
Hostname: type the **IP address**, for example 10.20.4.71
7. You can check the box for **Show advanced properties?** to see the additional settings, but you can keep all the defaults.
8. Click **Create New Device**.
The Devices page is displayed showing the Siemens device is successfully created and added to Ignition. The Status will show as Disconnected and then Connected.

Siemens devices do not support browsing, therefore you can not browse the Tags by going to the **OPC Connections > Quick Client** in the **Configure** section of the Gateway. To see and browse the Tags, you need to create the Tags manually as described in the [Configuring Siemens Addressing](#) section.

Configuring Siemens Addressing

The S7 protocol does not support Tag browsing. Therefore, you must configure all Tags in the Designer. This can be done either manually, as needed, or by importing bulk using the Tags CSV import functionality.

To Manually Specify Each Address

1. From the Designer, in the Tag Browser, click the **Add +** icon.
2. Select **New Standard Tag**, then **OPC Tag**.



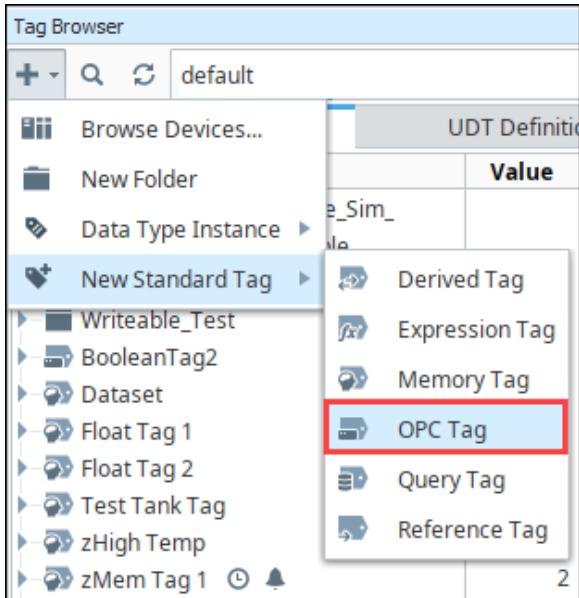
Connecting to S7 Devices

[Watch the Video](#)



About Siemens Addressing

[Watch the Video](#)



- In the Tag Editor window, as an example, you can set the following values:

Name: **Tag**

Data Type: **Int4**

OPC Server: choose **Ignition OPC-UA Server** with the edit button on the right.

OPC Item Path: **[S71200]IW0**, the **S71200** device name goes in the square brackets then you give the address to PLC which in this example is **IW0** (Word at Offset 0 in the Inputs area). The [Configuring Siemens Addressing](#) section, explains how you can construct these addresses.

- Click **OK**.

Now you can see the Temp Tag in the Tag Browser.

Address Syntax

You need a device name plus a Tag address to create a Tag. The device name is a known, but the Tag address needs to be configured. Once you have both the device name and Tag address you enter them in the **OPC Item Path** field of the **Tag Editor** window using the **[device_name]address** format, where **device_name** is the name of the device and **address** is the configured Tag address which is described here.

Tag addresses are made up of three different components: Area, Data Type and Offset.

Area Syntax		
DataBlocks	DBn	
Inputs	I	
Outputs	Q	
Flags	M	
Timers	T	
Counters	C	
Data Type Syntax		Signedness
Bit	X	N/A
Byte	B	Unsigned
Char	C	Signed
Word	W	Unsigned
Int	I	Signed
DWord	D	Unsigned
DInt	DI	Signed
Real	REAL	Signed

String	STRING or STRING.LEN	N/A
--------	----------------------	-----

To form an address, you combine syntax for the desired Area and Data Type with an Offset into that area.

Examples	
Area+Data Type+Offset	
IB0	Byte at Offset 0 in the Inputs area
IW0	Word at Offset 0 in the Inputs area
DB500,DI8	DInt at Offset 8 in DataBlock 500
ISTRING24.50	A String of length 50 starting at offset 24 in the Inputs area
IX20.3	Bit 3 of the Byte at Offset 20 in the Inputs area
T0	Timer at offset 0 (No data type is specified for timers)
C0	Counter at offset 0 (No data type is specified for counters)

Offsets

It is important to note that offsets are absolute. **IW0** and **IW1** share a byte. To get two consecutive, non-overlapping words you need to address **IW0** and **IW2**.

Bits

Bits are addressed by using the Bit data type (X) and appending .bit to the end, where bit is in the range [0-7]. When addressing a bit at a given offset, that offset is always treated as a byte.

Strings

Strings are assumed to be in the S7 string format and have a max length of 210.

Timers

Timers are scaled up to a DWord and converted from S5 time format so they can represent the time in milliseconds without requiring any multipliers. When you write to a timer it is automatically converted from milliseconds into S5 time format for you. A data type is not specified when accessing timers.

Counters

Counters in the PLC are stored in BCD. The driver automatically converts to/from BCD for you and exposes any counter Tags as UInt16 values. A data type is not specified when accessing counters.

Device Settings

General	
Name	The name of the device connection
Description	A description for the device connection. The description will appear on the Devices page on the Gateway.
Enabled	Whether or not the connection is active. Disabling this setting terminates communication with the device.
Connectivity	
Hostname	The hostname or IP address of the device.
Timeout	The request timeout, specified in milliseconds. The default is 2,000.
Advanced	
Port	The port to use when connecting to the device. The default is 102.
PDU Size	Number of bytes to fit into PDU block of a single packet. Increasing this number can improve request throughput only if the

	processor supports a higher PDU Size. Varies from 240 and up to 960 depending on the device. The default is 240.
Rack Number	The number of the rack that the device is positioned in. The default is 0.
CPU Slot Number	The slot number assigned to the CPU. The default is 2.
Reconnect After Consecutive Timeouts	After several consecutive timeouts, the device connection will attempt to reconnect to the device. This setting determines how many consecutive timeouts must occur before reconnecting.

UDP and TCP Driver

Ignition's UDP and TCP drivers allows Ignition to communicate to various devices like barcode scanners, scales, and more. These are not catch-all drivers that will talk to any PLC that communicates over TCP, but rather very basic drivers that will communicate over TCP or UDP. The drivers are configured to connect to one or more ports on a given IP address and bring in any data there (ASCII characters, etc) as a value of a Tag. Both drivers can act as strictly passive listeners: meaning they do not write back or make any requests to the device. This is very powerful for the devices that are designed to function in this way (like barcode scanners) because the device just needs to constantly update data on a port for this driver to work.

The TCP driver has the option of writing back to the device. When configured for Writeback, a Tag is exposed in Ignition's OPC server that will handle writing: any writes made to the Tag are sent to the device.

Structure in the Address Space

A device using the UDP or TCP driver appears in the **Devices** folder of the OPC-UA server with the name it was configured to use. Browsing the device will yield one folder per port configured to listen on. Browsing the port folder will yield one variable node containing the entire message received as well as an additional variable node per field configured. A device configured with a field count of four would have five nodes total: one for the message and four for the fields.

Connecting to a Device

Instead of connection to a device directly, this driver will connect to a port (often on the host computer or a computer connected directly to the device), and that device will be configured to post data to that same host/port. Rules are configured that dictate how the incoming data is interpreted. You can configure multiple ports for each device connection.

Connect to a Barcode Scanner or Scale

You can connect to a barcode scanner or scale by using Ignition's UDP and TCP driver.

1. Go to the Config section of the Gateway Webpage.
2. Scroll down and select **OPC UA > Device Connections**.
3. On the Devices page, click on **Create new Device**.
4. On the Add Device Step 1: Choose Type page, scroll all the way down and select **TCP Driver**, and click **Next**.
5. On the New Device page, leave all the default values and type in the following fields:
Name: the name you specify here will appear under **Devices** folder on the Quick Client page in the Gateway.
Port(s): **12345**, as an example
Hostname: type the **IP address**, for example **10.20.3.456**

The screenshot shows the Ignition Configuration interface. The path is 'Config > Opcua > Devices'. A new device is being created with the following settings:

General	
Name	Scale
Description	(empty)
Enabled	<input checked="" type="checkbox"/> (default: true)

Connectivity	
Port(s)	12345 The port(s) to connect to.
Address	10.20.3.456 The IP address to connect to.
Inactivity Timeout	0 The number of milliseconds without receiving data from the source before a disconnect/reconnect is made. Set to 0 to disable. (default: 0)

6. You can check the box for **Show advanced properties?** to see the additional settings, but you can keep all the defaults.

On this page ...

- [Structure in the Address Space](#)

[Connecting to a Device](#)

- [Connect to a Barcode Scanner or Scale](#)

[Device Properties](#)

[UPD and TCP Device Tags](#)

When configured, both drivers have explicit sets of Tags they use to communicate. The Tags are listed below.

- [TCP Tags](#)
- [UDP Tags](#)



Connecting to TCP Device

[Watch the Video](#)

7. Click **Create New Device**.
The Devices page is displayed showing the Scale device is successfully created and added to Ignition. The **Status** will show as **1/1 Connected**.
8. Go to the **OPC Connections > Quick Client** in the Config section of the Gateway Webpage.
9. Under the **Devices>[Scale]>12345** folder you will see the Last Receive Time and the Message folders.
Next to each Tag, under the **Action** column, you will see **[s][r][w]**.
10. Click on **[s]** which means Subscription. You will be able to see the Value of the Tag displayed on this OPC Quick Client page.

Device Properties

The properties on the **New Device** page of the Gateway for the TCP and UDP devices are as follows:

General	
Name	Name of the device using this driver. This name will appear in the Devices folder when browsing the OPC-UA server.
Enables	When selected, the device is enabled. When not selected, disabled devices will not make a connection attempt.
Connectivity	
Port(s)	On the UDP driver, this is the port(s) to listen on. On the TCP driver, this is the port(s) to connect to. Separate multiple ports with a comma.
Address	On the UDP driver, this is the IP address to listen to. On the TCP driver, this is the IP address to connect to.
Inactivity Timeout	The number of milliseconds without receiving data from the source before a disconnect/reconnect is made. Set to 0 to disable.
Message	
Message Delimiter Type	Sets the method used to determine how much or what data length constitutes a full message . Packet Based: Assumes that whatever arrives in one packet, regardless of length or content, is the message. Character Based: Content is appended to a message buffer until the given character or set of characters arrives, at which point the contents of the buffer are considered the message. Fixed Size Content is appended to a message buffer until some fixed number of bytes is received, at which point the contents of the buffer are considered the message.
Message Delimiter	If the message delimiter type is Character Based , this will be the character or set of characters used to identify a message. If the type is Fixed Size , this will be the size used to identify a message.
Field Count	The number of fields within a message must be fixed. This property dictates how many fields will be present in each message. When the number of fields received does not match the designated count, all nodes will receive quality BAD_CONFIG_ERROR .
Field Delimiter	This is the character(s) that are used as field delimiters. For example, the message a b c d with a field delimiter of " " would be split into four fields: a , b , c , and d . The field count would have to be set at 4.

Both drivers have unique Advanced Properties

TCP Driver	
Writeback Enabled	Enable writeback capabilities for the device.
Writeback Message Delimiter	The delimiter expected by the device signaling the end of an incoming message.
UDP Driver	
Message Buffer Size	The size of the message buffer in bytes.
Multicast	If the connection should be enabled for multicast.

Tag Polling Speed

Tag values update no faster than the scan class a Tag is assigned to. If additional messages are received over TCP or UDP at a faster rate, the Tag will not show these additional values. Generally it is recommended to use these drivers with updates that come through at a 1 second rate or slower. The absolute maximum rate for the TCP driver is 25ms, although most systems cannot reliably execute this quickly. The UDP driver does not

have a maximum rate, but a practical maximum rate will depend on hardware, and likely will be significantly slower than 25ms as well. Most "fast" scan rates fall in the 100ms-200ms range.

UPD and TCP Device Tags

When configured, both drivers have explicit sets of Tags they use to communicate. The Tags are listed below.

TCP Tags

Name	Description
Last Receive Time	A datetime representing when the last message was received.
Message	A string interpretation of the last received message.
MessageBytes	<p>This feature is new in Ignition version 8.1.2 Click here to check out the other new features</p> <p>A binary representation of the last received message.</p>
Writable	Only available when the TCP device connection has Writeback Enabled . Writing to this Tag from Ignition will send a write request to the device. The content will be sent as a string.
WritableBytes	<p>This feature is new in Ignition version 8.1.2 Click here to check out the other new features</p> <p>Only available when the TCP device connection has Writeback Enabled. Writing to this Tag from Ignition will send a write request to the device. The content will be sent as binary data.</p>

UDP Tags

Name	Description
Last Receive Time	A datetime representing the when the last message we received.
Message	A string interpretation of the last received message.
MessageBytes	<p>This feature is new in Ignition version 8.1.2 Click here to check out the other new features</p> <p>A binary representation of the last received message.</p>

DNP3

DNP3 is a protocol used commonly in utilities like water and electric companies. It is commonly used to connect to one master station (device) that is then connected to several other devices. This creates a web of devices without taxing the network too much. DNP3 is similar to the Modbus protocol in that it is more device agnostic, but it's newer, more robust, and because of that, more complex. You can use it to connect to many modern devices, check your documentation to see whether your device supports DNP3.

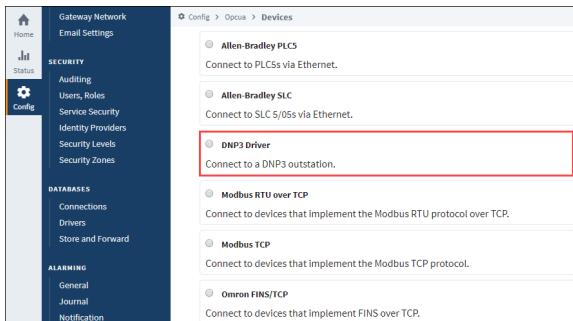
Connecting to a Device

Ignition's DNP3 driver can connect directly to any devices that support Ethernet communication through the master station. It is important to make a new device connection for each of the outstations (remote devices), setting the source and destination addresses for each in Ignition's device connection.

1. Go to the **Config** section of the Gateway Webpage.
2. Scroll down and select **OPC UA > Device Connections**.



3. On the **Devices** page, find the blue arrow and click on **Create new Device**.
4. On the **Add Device Step 1: Choose Type** page, select **DNP3 Driver**, and click **Next**.



5. On the **New Device** page, leave all the default values and type in the following fields:
Name: **DNP3**
Hostname: Enter IP address, for example 10.20.8.51
6. You can check the box for **Show advanced properties?** to see the additional settings, but you can keep all the defaults.
7. Click **Create New Device**.
The **Devices** page is displayed showing the **DNP3** device is successfully created and added to Ignition. The **Status** will show as Disconnected and then Connected or Idle, depending on the status of the device.

Connection Settings

General	
Name	The name of this DNP3 device connection.
Description	Device connection description (optional). Can be used to provide any useful information / comments about this connection.
Enabled	If True (checked), the connection is enabled; if False (unchecked), the connection is disabled.

On this page ...

- [Connecting to a Device](#)
- [Connection Settings](#)
 - [Internal Indicators](#)
 - [Terminology](#)
- [Aliased Points](#)
- [Buffered Events](#)
 - [Support for this Feature](#)
- [Browsing DNP3 Points](#)
 - [Point Types](#)



Connecting to DNP3 Devices

[Watch the Video](#)

Main	
Hostname	The IP Address of the device.
Port	The port to use when connecting to a DNP3 device. The default port is 20000.
Source Address	The address of the master station, default is 3.
Destination Address	The address of the outstation, default is 4.
Integrity Poll Interval	The interval at which to perform an integrity poll, in millis, default is 3,600,000.
Direct Operate Enabled	When true, the Direct-Operate function code is used on a write, otherwise Select-Operate is used, default is true.
Unsolicited Messages Enabled	When true, the outstation may send unsolicited messages for Class 1, 2, and 3 data, default is false.
Advanced	
Message Fragment Size	The maximum size of a message fragment in the application layer, default is 249.
Message Timeout	The amount of time to wait for a message response from the outstation, default is 5,000.
Retries	The number of retries on a message timeout, default is 0.
Link Layer Confirmation	When true, a link layer confirmation will be required from the outstation when sending messages, default is false
Default Outstation Conformance level	The default DNP3 Application Layer level subset to use when communicating with the outstation.
Default Value Types	
Analog Input Points	The default value type to use when reading an analog input point. default is INTEGER
Analog Input Frozen Points	The default value type to use when reading a frozen analog input point. default is INTEGER
Analog Output Points	The default value type to use when reading/writing an analog output point. default is INTEGER
Counter Points	The default value type to use when reading a counter point. default is INTEGER
Counter Frozen Points	The default value type to use when reading a frozen counter point. default is INTEGER
Binary Input Points	The default value type to use when reading a binary input point. default is WITH_FLAGS
Double-Bit Binary Input Points	The default value type to use when reading a double-bit binary input point. default is WITH_FLAGS
Binary Output Points	The default value type to use when reading a binary output point. default is WITH_FLAGS

Notes

- **Source Address and Destination Address:** These addresses are assigned to the computers and should be the same across all settings, because of this the settings in Ignition and the settings in the device are the opposite of each other. For example, if the device is configured with an address of 4 and Ignition has an address of 3:
 - the settings in Ignition should have the Source Address set to 3 and the Destination Address set to 4.
 - the settings in the Device should have the Source Address set to 4 and the Destination Address set to 3.
- **Unsolicited messages enabled** property: setting this property to True (checking the box) allows the outstation to send unsolicited messages to Ignition. This means that Ignition will connect to the outstation, but not request any data from it. Ignition waits for the outstation to send data. Not all devices support this option; those that do need to be configured to use it. Please refer to your device's documentation for more information.

Internal Indicators

Each response received from a connected outstation will contain an Internal Indication (IIN) bit field. This field indicates certain states or error conditions in the outstation. IINs are mapped to read-only points, indicating the following:

- Broadcast message received (**Broadcast**)
- Additional Class 1, 2, or 3 event data is available (**Class 1 Events**, **Class 2 Events**, **Class 3 Events**)
- Time synchronization required in the outstation (**Need Time**)
- Some output points are in local mode (**Local Control**)
- An abnormal condition exists (**Device Trouble**)
- The outstation device has restarted (**Device Restart**)
- Function code not implemented (**No Func Code Support**)
- Object Unknown (**Object Unknown**)
- Request parameter error (**Parameter Error**)
- Outstation event buffer overflow (**Event Buffer Overflow**)
- An operation is already executing (**Already Executing**)

- Configuration corrupt (**Config Corrupt**)

Terminology

- **unsolicited response:** An Application Layer message from an outstation to a master for which no explicit request was received. The request is implied by the act of a master enabling unsolicited reporting of various points within an outstation.
- **integrity poll:** Requests all event data, followed by the static data of all points assigned to one of the four classes (static Class 0 or event Class 1, 2, or 3).
- **DNP3TIME:** Universal Coordinated Time (UTC) time expressed as the number of milliseconds since the start of January 1, 1970. The effective date for using the UTC time base is January 1, 2008. Prior to this, DNP3 did not require a specific time reference.

Aliased Points

Aliased points allow the user to assign meaningful names and descriptions to DNP3 points. They are also useful for addressing any points that were not returned by the initial integrity-poll on connection.

Point Address	The group, variation, and index that fully describe a point. A full address consists of all three parts: <ul style="list-style-type: none"> • Group – An integer prefixed with g. For example, g40 • Variation – An integer prefixed with v. For example, v2 • Index – An integer prefixed with i. For example, i5 Example: g30v1i20
Path	A "/" separated folder hierarchy in which to create the aliased point. Example: Facility1/Voltage
Description	A user-defined description of the point mapping.

Buffered Events

Buffered events can be enabled by setting the Queue Size property on the corresponding Tag Group to a number greater than 1. (See Queue Size on the [Tag Groups](#) page.) The number represents the number of buffer events that will be handled by the driver. The queue always keeps the most recent events, dropping older events. For example, if Queue Size were set to 20, and 30 value changes occur on the device while disconnected, then the 10 oldest events would be ignored by the driver.

After recovering from a disconnect, the driver will playback missed events from the driver, and update the value on the corresponding Ignition Tags by cycling through each event quickly, from oldest event in the buffer to the most recent. This value change will trigger value changes in certain systems. Specifically:

- alarms events (recent events as well as alarm journal records)
- Tag History records (when the History **Sample Mode** is set to "On Change")
- Tag Events Scripts

Once finished cycling through the buffered values, the Tag will resume showing the live value.

Support for this Feature

This Buffered Events feature can only be used by devices that support Sequence of Events (SoE), and have the option enabled for points Ignition is subscribed to. Note that the data type for SoE may differ from the data type that is received by normal subscription. This is generally due to a configuration on the Default Variation property in the device.

Browsing DNP3 Points

When the driver (master) connects to a device (outstation), an integrity poll is performed. Any DNP3 objects returned in the response that fall under the Point Type categories listed in the table on the right are mapped to the OPC server with the appropriate index. (For example, g40v1i2 corresponds to an AnalogOutput point, variation 1, index 2.)

To see the points mapped, you can go to the Designer, and open the [Tag Creator](#).

Point Types

Type Name	Group	Supported Variations	
SinglBitBinaryInput	1	1 - Packed format	2 - With flags
DoubleBitBinaryInput	3	1 - Packed format	2 - With flags
BinaryOutput	10	1 - Packed format	2 - With flags



**INDUCTIVE
UNIVERSITY**

**About DNP3
Addressing**

[Watch the Video](#)

Counter	20	1 - 32-bit with flags 2 - 16-bit with flags	5 - 32-bit 6 - 16-bit
FrozenCounter	21	1 - 32-bit with flags 2 - 16-bit with flags 5 - 32-bit with flags and time	6 - 16-bit with flags and time 9 - 32-bit 10 - 16-bit
AnalogInput	30	1 - 32-bit with flags 2 - 16-bit with flags 3 - 32-bit	4 - 16-bit 5 - Float with flags 6 - Double with flags
FrozenAnalogInput	31	1 - 32-bit with flags 2 - 16-bit with flags 3 - 32-bit with time of freeze 4 - 16-bit with time of freeze	5 - 32-bit 6 - 16-bit 7 - Float with flags 8 - Double with flags
AnalogOutput	40	1 - 32-bit with flags 2 - 16-bit with flags	3 - Float with flags 4 - Double with flags
OctetString	110	0 - 255	

Omron NJ Driver

Connect Ignition to an Omron NJ Device

1. Go to the **Config** section of the **Gateway** Webpage.
2. Scroll down and select **OPC UA > Device Connections**.



3. On the **Devices** page, click on **Create new Device**.
4. Select **Omron NJ Driver**, and click **Next**.
5. On the **New Device** page, leave all the default values and type in the following fields:

Name: **Omron**

Hostname: type the IP address, for example 74.125.224.72

Check the box for **Show advanced properties?** to see the additional settings, but you can keep all the defaults.

6. Click **Create New Device**.
The **Devices** page is displayed showing the **Omron** device is successfully created and added to Ignition.
7. On the **Devices** page, click the Tags link next to the newly created device.
The **Manage Tags** page is displayed, allowing you to configure which variables in the device will show up as Tags in Ignition.

On this page ...

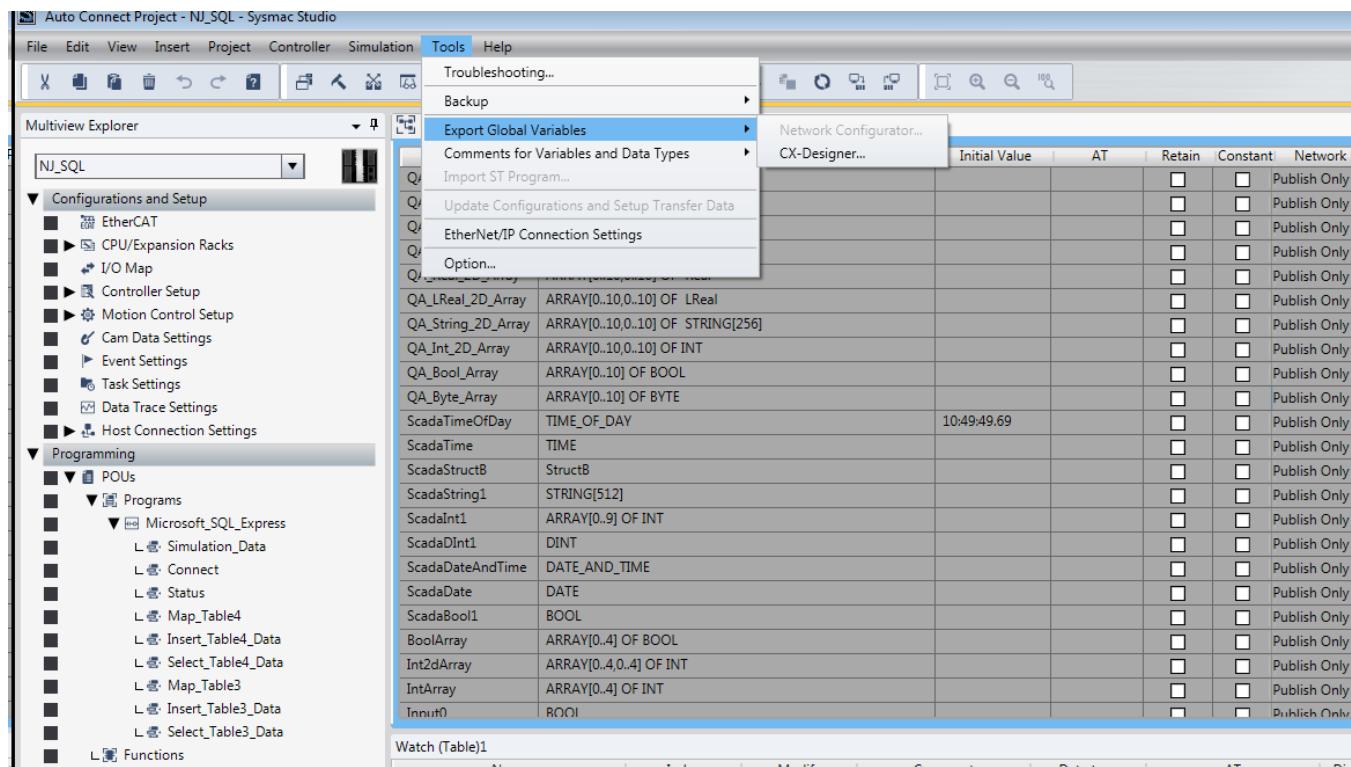
- Connect Ignition to an Omron NJ Device
 - Device Settings
 - Exporting from the Device
- Managing Tags
- Importing Tags
 - Addressing
 - Scalars
 - Arrays
 - Multidimensional Arrays
 - Strings

Device Settings

General	
Name	Device name.
Description	Description of the device connection.
Enabled	Default is set to true.
Main	
Hostname	The hostname or IP address of the device.
Timeout	The request timeout, specified in milliseconds. The default is 2,000.
Concurrency	The number of concurrently issued requests allowed. There is a 1:1 correlation between concurrency and the number of CIP connections used. The default is 2.
Advanced	
Connection Size	The CIP connection size to use. The default (and maximum) is 1,994 bytes.
Date/Time Offset	<p>This feature is new in Ignition version 8.1.1 Click here to check out the other new features</p> <p>Offset in hours for Date/Time values. Default is 0. The driver needs an advanced setting for an offset to be applied when reading or writing date/time values.</p>
Slot Number	The slot number in the backplane in which the CPU is located. The default is 0.

Exporting from the Device

To export variables from Sysmac Studio, navigate to the global variables and select **Tools > Export Global Variables > CX-Designer**.



The variables will be saved to the clipboard in tab-separated format. You can now paste the contents into an empty text file for use with importing into the Ignition Gateway.

Managing Tags

In order to browse Tags in the Designer, you must first create a mapping for the device in the Gateway. The **Manage Tags** page can be accessed by navigating to the Omron device and clicking the **Tags** link.

Devices

Name	Type	Description	Enabled	Status	More	edit
Omron NJ Driver	Omron NJ Driver		true	Idle	tags	More edit
SLC	Simulators SLC Simulator		true	Connected	delete	delete edit

[→ Create new Device...](#)

Importing Tags

Once on the **Manage Tags** page, you can manually enter the Tags, or import them from a tab-separated file. When importing, first choose a file, then click the **Import** button. The default option when importing is to replace the **Tags** table with Tags from the import. Select the append option to append Tags to the table.

Manage Tags

Import a List of Tags.

No file chosen

Tags

<input type="checkbox"/> Name	DataType	Chars	Elements	R/W
<input type="checkbox"/> ex. Facility.Amps.Amp1	UINT_BCD			RW

<< < 1 > >>

Manage Tags

Import a List of Tags.

No file chosen

Tags

Name	DataType	Chars	Elements	R/W
ScadaString1	STRING	512		RW
ScadaDInt1	DINT			RW
Int2dArray	INT		0..4,0..4	RW
IntArray	INT		0..4	RW
BoolArray	BOOL		0..4	RW

<< <1 > >>

Once you save any changes made to the Tag mapping, you can view the Tags in the [Tag Creator](#).

Addressing

In the **Tags** table of the **Manage Tags** page, we have four columns of configuration per Tag:

- **Name** - The corresponding address of the variable found in the Omron device. Struct members are separated with periods.
- **Datatype** - The datatype of the variable found in the Omron device.
- **Chars** - The maximum number of characters that a String Tag will contain.
- **Elements** - Denotes whether the Tag is considered a scalar or array. See below for more detail on specifying the number of elements to read from the device.
- **R/W** - Specifies read / write access permissions on the Tag.

Support for the following data types is included:

- TIME_NSEC
- DATE_NSEC
- TIME_OF_DAY_NSEC
- DATE_AND_TIME_NSEC

Scalars

Leaving the **Elements** column blank will result in a scalar Tag. When reading from the device, only one element will be requested.

ScadaBool1	BOOL		RW
------------	------	--	----

Arrays

Specify the number of elements in an array in the form of **0..N**. The initial index 0 is always included, so an array mapped with 0..4 elements is a 5 element array.

BoolArray	BOOL	0..4	RW
-----------	------	------	----

- i** The number range specified in the Elements field can deviate from the range specified in the device's program. Thus, if an array was configured with a range of 0 - 4, but the mapping on the Ignition Gateway is set to 3 - 7, then the resulting items would be offset as follows:

PLC Program	Tag in Ignition
0	BoolArray_3_
1	BoolArray_4_
2	BoolArray_5_
3	BoolArray_6_
4	BoolArray_7_

This is because the driver always assumes that the lowest configured element on the mapping page matches up with the lowest element in the PLC program. As seen above, this can cause some confusion if the mapping on the Ignition Gateway is configured with a different range.

For this reason, it is **highly recommended** to configure the Elements field on the Ignition Gateway to match the range used in the PLC program.

Note, that this also applies to **Multidimensional Arrays**.

Multidimensional Arrays

Multidimensional arrays are specified in the same way as arrays with each group of indices separated by a comma.

Int2dArray	INT	0..4,0..4	RW
------------	-----	-----------	----

Optional Format

Array elements may also be specified with a single number equaling the total number of elements.

Int2dArray	INT	5, 5	RW
------------	-----	------	----

Strings

The number of characters for String variables is specified in the **Chars** field.

<input type="checkbox"/> ScadaString1	STRING	▼	512	<input type="text"/>	RW	▼
---------------------------------------	--------	---	-----	----------------------	----	---

String arrays are mapped using both the **Chars** and **Elements** fields.

<input type="checkbox"/> StringArray	STRING	▼	512	0..4	RW	▼
--------------------------------------	--------	---	-----	------	----	---

Omron FINS Driver

Ignition supports Omron FINS devices. This driver does support both UDP and TCP transport protocols.

Connect Ignition to an Omron FINS Device via TCP

1. Go to the **Config** section of the **Gateway** Webpage.
2. Scroll down and select **OPC UA > Device Connections**.



3. On the **Devices** page, click on **Create new Device**.
4. Select **Omron FINS/TCP**, and click **Next**.
5. Fill in the following fields:

Name: **FINS TCP**

Hostname: Enter the **IP address**, for example 74.125.224.72

Leave the default values in the remaining fields.

6. Click **Create New Device**.

The **Devices** page is displayed showing the **FINS TCP** device is successfully created and added to Ignition.

On this page ...

- Connect Ignition to an Omron FINS Device via TCP
- Connect Ignition to an Omron FINS Device via UDP
- Device Settings
- Import Addresses
- Addressing
 - Data Areas
 - EM Area
 - Examples
- Ignition OPC UA Server Node Configuration
 - Example File

Connect Ignition to an Omron FINS Device via UDP

1. Go to the **Config** section of the **Gateway** Webpage.
2. Scroll down and select **OPC UA > Device Connections**.
3. On the **Devices** page, click on **Create new Device**.
4. Select **Omron FINS/UDP**, and click **Next**.
5. Fill in the following fields:

Name: **FINS UDP**

Bind Address: Enter the **IP address** of your Gateway, for example 74.125.225.10

Remote Address: Enter the **IP address** of your Omron FINS device, for example 74.125.224.72

6. Under FINS Settings set the following:

FINS Source Node: The last octet of your IP address, for example 10 using the bind address above.

FINS Destination Network: The address number of your FINS device typically configured in the routing table for your device. The valid range is 0-128.

FINS Destination Node: The node number of your FINS device. This is typically the last octet of the IP address of the device, for example 72 in the example above.

Fins Destination Unit: The unit number of your FINS device.

7. Click **Create New Device**.

The **Devices** page is displayed showing the **FINS UDP** device is successfully created and added to Ignition.

Device Settings

General	
Name	Device name.
Description	Device description.

Enabled Default is set to true.

Connectivity		
Hostname	The hostname or IP address of the device.	TCP Only
Port	The port your Omron device is listening on. The default is 9600.	TCP Only
Local Address	Address of network adapter to connect from.	TCP Only
Bind Address	The hostname or IP address of the Gateway.	UDP Only
Bind Port	The port you wish to create a UDP connection on for the Gateway. The default is 9600.	UDP Only
Remote Address	The hostname or IP address of the device.	UDP Only
Remote Port	The port your Omron device is listening on. The default is 9600.	UDP Only
Timeout	The request timeout, specified in milliseconds. The default is 2,000.	UDP & TCP

FINS Settings		
FINS Source Network	The address number of the source network (the Ignition Gateway).	Valid value is an integer between 0 and 127. The default setting is 0.
FINS Source Node	The number of the source node (the Ignition Gateway), typically the last octet of the IP address.	Valid value in an integer between 0 and 254. The default setting is 0.
FINS Source Unit	The unit number of the source device (the Ignition Gateway).	Valid value is an integer between 0 and 255. The default setting is 0
FINS Destination Network	The address number of the destination network (the FINS device).	Valid value is an integer between 0 and 127. The default setting is 0.
FINS Destination Node	The number of the destination node (the FINS device), typically the last octet of the IP address and set via dials on the front of the device.	Valid value in an integer between 0 and 254. The default setting is 0.
Fins Destination Unit	The unit number of the destination device (the FINS device).	Valid value is an integer between 0 and 255. The default setting is 0

Request Optimization	
Concurrent Requests	The number of requests that Ignition will try to send to the device at the same time. Increasing this number can sometimes help with your request throughput, however increasing this too much can overwhelm the device and hurt communications with the device.
Max Request Size	The maximum size of a request in bytes.
Max Gap Size	The maximum address "gap" to span when optimizing requests to contiguous address locations.
Write Priority Ratio	The number of write requests to execute for every read request, when an abundance of both are queued for execution.

Import Addresses

There is a user interface in the Gateway to create import addresses. After import, you'll be able to browse the Tags in the Designer.

1. Go to the **Config** section of the **Gateway** Webpage.
2. Scroll down and select **OPC UA > Device Connections**.
3. On the **Devices** page, scroll to the **OMRON FINS** device. Click the **More** dropdown and choose **addresses**.

OMRON FINS UDP	Omron FINS/UDP	true	UNBOUND	More ▾	edit
OMRON NJ	Omron NJ Driver	true	Idle	addresses	delete

4. On the next page, load the addresses from a CSV, TSV, or XML file. Click the **Load Configuration** button and choose a file.

This feature is new in Ignition version **8.1.2**
[Click here](#) to check out the other new features

Release 8.1.2 added support for XML configuration files.

5. Click **Load**.

Config > Opcua > Devices

< back

Device Connections / Omron FINS TCP

Load Configuration File

Address Description

Load configuration from file (*.csv, *.tsv)
 Omron.csv

Append to current configuration?

Save Configuration

You'll see the values load onto the screen.

Config > Opcua > Devices

< back

Device Connections / Omron FINS TCP

Load Configuration File

Address Description

Tag Path	Address	Description	Remove
Tag1	CIO1	here	Remove
Tag2	CIO2	there	Remove
Tag3	CIO3	everywhere	Remove
Tag4	CIO4	over here	Remove
Tag5	CIO5	over there	Remove
Tag6	CIO6	all over everywhere	Remove

Add Row **Save Configuration** <<< 1 >>>

6. Click **Save Configuration**.

7. Once you save any changes made to the Tag mapping, you can view the Tags in the [Tag Creator](#).

Addressing

You also have the option to manually address tags via an Ignition OPC tag in the Ignition Designer. See [Ignition OPC UA Server Node Configuration](#) for more information.

Data Areas

Area	Ignition Code	Omron Symbol Code	Native Size
CIO Area	CIO		Int16/UInt16
Work Area	WR or W	W	Int16/UInt16
Holding Bit Area	HR or H	H	Int16/UInt16
Auxiliary Bit Area	AR or A	A	Int16/UInt16
Timer Area	TIM or T	T	Int16/UInt16 or Bool
Counter Area	CNT or C	C	Int16/UInt16 or Bool
Index Register Area	IR	IR	Int32/UInt32
Data Register Area	DR	DR	Int16/UInt16
DM area	DM or D	D	Int16/UInt16
EM area	EM or Exx	E	Int16/UInt16

EM Area

Use `EM` for the current bank, or `Exxx` for bank xx, where xx is 2 hex digits. Valid banks are E00 through E18 (25 total).

Data Types

Bit	Bool	Int16
Int32	Int64	UInt16
UInt32	Float	Double
String	BCD16	BCD32

Data Type Modifiers

Modifier	Order
@BE	Big Endian Byte Order (default when not specified)
@LE	Little Endian Byte Order
@HL	High-Low Word Order
@LH	Low-High Word Order (default when not specified)

Examples

Syntax Example

Note: Items in curly brackets {} are optional.

Below is an example OPC Item path, representing how to manually specify an address in the device.

```
ns=1;s=[DeviceName]Area{<DataType>}Offset{.Bit}
```

When typing Addresses in using the Device's Address page on the gateway, the syntax looks like the following:

```
Area{<DataType>}Offset{.Bit}
```

More Examples

Item Path (OPC UA NodeId)	FINS Request
CIO0	An Int16 at offset 0 in CIO area.
CIO<Int32>1	An Int32 at offset 1 in CIO area.
CIO<Int64>3.0	Bit 0 of an Int64 at offset 3 in CIO area.
CIO<Int64@HL>3	An Int64 at offset 3 in CIO area in High-Low word order.
TIM0	Present Value (16-bit) of Timer at offset 0.
TIM<Bool>0	Completion Status (Bool) of Timer at offset 0.
CNT0	Present Value (16-bit) of Counter at offset 0.
CNT<Bool>0	Completion Status (Bool) of Counter at offset 0.
DR0	An Int32 at offset 0 in DR area (32-bit by default).
DR<Int64>0	An Int64 at offset in DR area.
CIO<Int16[3]>0	An Int16 array of size 3 at offset 0 in CIO area.
CIO<Int16[3]>0[1]	Element 1 of an Int16 array of size 3 at offset 0 in CIO area.
CIO<String10>0	String of up to length 10 (string length in bytes) at offset 0 in CIO area.
E001000	An Int16 from EM bank 0 at offset 1000.
E00<Int16>1000	Also an Int16 from EM bank 0 at offset 1000, but with an explicit DataType to make the address look less confusing.

Ignition OPC UA Server Node Configuration

You can also create OPC Nodes in Ignition's OPC UA server manually via a CSV file on the Ignition Gateway itself with the following settings:

File Name: tags.csv

File Location:\$ignitionHome/data/opcua/devices/\$deviceName

File Format: Comma separated value file with the following columns:

- Browse Path
- Address
- Description

Example File

```
Tag1, CIO0, "First tag" Foo/Tag2, CIO1, "Second tag, in a folder" Foo/Bar/Tag3, CIO2, "In a couple of
folders"
```

Use quotes on the description when it contains a comma (or whenever you like).

When adding or making changes to this file, a restart of the device is necessary for the changes to be picked up. This is done by editing the device and clicking **Save**.

Programmable Device Simulator

The Programmable Device Simulator allows you to add simulator devices that act as if they are connected to a real device. It allows users to read and write Tags without any network or PLC connection.

The Programmable Device Simulator provides functionality that allows you to create your own simulator program with outputs you define. Users are able to create Tags that can be used in their own development process, thus providing the flexibility to simulate devices with a Tag structure they are building for their project. There are a number of built-in functions that will result in unique Tag outputs rather than static values. We also provide the ability to set specific values at different points of the program should the functions not provide exactly what you need.

Note: The Programmable Device Simulator replaced the previous built-in simulators in earlier versions of Ignition, and combined the Dairy Demo, Generic and SLC simulator device connections into a single driver. [Click here](#) for the previous simulator documentation.

On this page ...

- [Connecting to a Programmable Device Simulator](#)
 - [Simulator Settings](#)
- [Program Instructions](#)
 - [Value Source Functions](#)
 - [Example Instruction Files](#)
 - [Preloaded Programs](#)
- [Meta Tags](#)
- [Export Instruction File](#)

Connecting to a Programmable Device Simulator

Connecting to a Programmable Device Simulator and loading the programs are simple and will give you some values that change on their own. These simulator devices can be used for realtime values, history and alarms.

1. From the Gateway webpage, go to the **Config** section.
2. Scroll down and select **OPC UA > Device Connections**.
3. Click on **Create new Device...**
4. Select **Programmable Device Simulator**, and click **Next**.
5. Give the device a name (i.e., GenSim), and click **Create New Device** button.

The screenshot shows the Ignition configuration interface for creating a new device. The top navigation bar shows 'Config > Opcua > Devices'. The main area has two tabs: 'General' and 'Simulator Program Settings'. In the 'General' tab, the 'Name' field is filled with 'GenSim' and the 'Enabled' checkbox is checked. In the 'Simulator Program Settings' tab, the 'Repeat Program' checkbox is checked and the 'Base Rate (ms)' input field is set to '1000'. At the bottom of the screen, there is a large blue button labeled 'Create New Device'.

6. The window will refresh and you'll see your device was successfully created with a status of "Running". Now you can set the program for the simulator by clicking **More > edit program**.

The screenshot shows a configuration interface for 'Devices'. At the top, there's a breadcrumb navigation: 'Config > Opcua > Devices'. A green success message box displays: 'Successfully created new Device "GenSim"'. Below this, a table lists devices. The first row shows 'GenSim' as a 'Programmable Device Simulator' with 'true' status and 'Running' status. To the right of this row is a 'More' dropdown menu with options: 'edit program' (highlighted with a red box) and 'delete'. Below the table is a link: '→ Create new Device...'

7. Create a program by either clicking the Add Instruction link to add a new line of instruction to the program, or loading a predefined program from the **Load Program** dropdown (i.e., Generic Program) and clicking the **Load Simulator Program** button.

The screenshot shows the 'GenSim' configuration page. At the top, there's a back link: '< back'. The main section is titled 'GenSim' with a 'Load Program' dropdown. The dropdown menu is open, showing several options: 'Choose One', 'Choose One SLC Program', 'Generic Program' (which is highlighted with a red box), 'Dairy Simulator', and 'Load from CSV'. To the right of the dropdown are two buttons: 'Export Instruction File' and 'Clear All'. Below the dropdown are four input fields: 'Time Interval', 'Browse Path', 'Value Source', and 'Data Type'. At the bottom left is a 'Save Program' button.

8. Once you have some instructions, click the **Save Program** button. These instructions determine what Tags get made, and what their values are.

Config > Opcua > Devices

< back

Program loaded successfully.

GenSim

Load Program

Generic Program

Load Simulator Program

Export Instruction File **Clear All**

Time Interval	Browse Path	Value Source	Data Type	Remove
0	Ramp/Ramp0	ramp(0.0, 10.0, 100.0, true)	Double	Remove
0	Ramp/Ramp1	ramp(5.0, 123.0, 230.0, true)	Double	Remove
0	Ramp/Ramp2	ramp(1.0, 2.0, 50.0, true)	Double	Remove
0	Ramp/Ramp3	ramp(-10.0, 10.0, 300.0, true)	Double	Remove
0	Ramp/Ramp4	ramp(0.0, 500.0, 1500.0, true)	Double	Remove
0	Ramp/Ramp5	ramp(0.0, 700.0, 2000.0, true)	Double	Remove
0	Ramp/Ramp6	ramp(0.0, 800.0, 2500.0, true)	Double	Remove
0	Ramp/Ramp7	ramp(0.0, 900.0, 3000.0, true)	Double	Remove
0	Ramp/Ramp8	ramp(0.0, 110.0, 3500.0, true)	Double	Remove
0	Ramp/Ramp9	ramp(0.0, 5000.0, 4000.0, true)	Double	Remove

Add Instruction <<< 1 2 3 4 5 6 7 8 >>>

Save Program

Simulator Settings

Setting	Description
Repeat Program	If True, the program will repeat indefinitely, meaning that once the last defined Time Interval has been reached, the program will start over again at Time Interval 0.
Base Rate (ms)	The number of milliseconds determining how quickly the program should move between each Time Interval. The initial duration of the Base Rate is defined in the Program Device Simulator settings should you need this value to persist after restarts.

Program Instructions

The simulator gets its values from a Program made up of various instructions. The program runs through the instructions in order and executes. Each Program Device Simulator contains only one program. Each instruction has a Time Interval and Tag path. If there is more than one Time Interval for a Tag, it will step through those values in order. You can alternatively have a single time Interval and a function in the Value Source field.

Term	Description
Time Interval	Represents a "step" or point in time for the program. A program will start at interval 0, and set the Value Source on the Tag. The program will wait for a number of milliseconds (determined by the Base Rate, defined below) before moving to the new Time Interval.
Browse Path	The full path to the Tag you want to create in the simulator. Forward slashes are used to specify folders, as well as act as a delimiter for additional folders in the path.
Value Source	Determines how the value for the node at the Browse Path is generated. Either a static value or a function will be used to generate the value.

Data Type	<p>The Data Type for a node. Valid types are:</p> <ul style="list-style-type: none"> • boolean • int16 • uint16 • int32 • uint32 • int64 • float • double • string • datetime • uint64 (Not all values fully supported by Ignition at this time)
-----------	---

Value Source Functions

Users are allowed to define a 'function' for the value source. This is not a Python or Expression function. Rather, the value in the cell is a string that looks like a function definition; eg., `foo(x,x,x)`. The driver will attempt to parse the string, and then derive a value based on the function. Valid functions are in the following table.

Function	Descriptions
<code>sine(min, max, period, repeat)</code>	Sine wave. The value moves between the min value and max value, and back to the min. The total duration of this wave is based on the period parameter. If no parameters are specified, then the function should produce a wave that starts at 0, reaches an upper bound of 100, and returns to 0. This series should occur over the default period ($(10 * \text{Base Rate}) * 1000 \text{ ms} = 10,000 \text{ ms}$).
<code>cosine(min, max, period, repeat)</code>	Cosine wave.
<code>square(min, max, period, repeat)</code>	Square wave.
<code>triangle(min, max, period, repeat)</code>	Triangle wave.
<code>ramp(min, max, period, repeat)</code>	Ramp signals starts from the min value going up to max value based on the period parameter. When the value reaches its upper limit, it is reset to the min value.
<code>realistic(setPoint, proportion, integral, derivative, repeat)</code>	A realistic number generator driven by a PID system.
<code>random(min, max, repeat)</code>	Random values starting at the min value and working up to the max value.
<code>list(value1, value2, etc..., valueN, repeat)</code>	Accepts any number of parameters and walks through each value in this list.
<code>qv(value, qualityCode)</code>	Sets a value and quality code. Allows users to simulate a value with bad quality.
<code>readonly(value)</code>	Sets a static value for read only purpose.

Each of the functions above expects certain parameters.

Parameter	Description	Default Value
<code>min</code>	Minimum value for the function.	0
<code>max</code>	Maximum value for the function.	100
<code>period</code>	Represents a period of time as a number of Time Intervals. The default value of 10 represents "10 Time Intervals."	10
<code>setPoint</code>	The setpoint of the PID Control Loop. Also known as the desired position	100
<code>proportion</code>	The proportional value of the PID Control Loop.	1.2

integral	The integral value of the PID Control Loop.	0.06
derivative	The derivative value of the PID Control Loop.	0.25
qualityCode	Allows users to set the quality on a Tag (assuming the function used contains a quality code parameter). Valid values are: Good, Bad, Uncertain.	Good
repeat	Does the function repeat for each time slice of the program, or does it report a single value when the instruction is run. The default value of true will ensure that a function will continue to run as long as the program itself is running. A value of false will only update a value when it reaches the specified Time Interval in the program.	true

Example Instruction Files

Simple Function Example

TimeInterval	BrowsePath	ValueSource	DataType
0	TagA	sine(0,100,10,true)	Double

The table above shows us function usage with parameters. Because the example is using default values, it is functionally equivalent to the following:

TimeInteval	BrowsePath	ValueSource	DataType
0	TagA	sine()	Double

Running this program should yield the following results (assuming repeat is enabled):

1. A Tag at root named "TagA" will be created.
2. Since only a single Time Interval was defined, the program ends quickly. The delta time between the start of the program and the end should be around one Base Rate interval.
3. The simulator will use a internal clock to ensure that the value reported is representative of the actual period for the function requested and if history is enabled, a full wave would be graphed.

Advanced Program Example

TimeInterval	BrowsePath	ValueSource	DataType
0	TagA	sine()	Double
0	myFolder/TagB	0	Int32
3	myFolder/TagB	4	Int32
4	another_folder/TagC	100	Int32
20	TagA	ramp()	Double
30	myFolder/TagB	55	Int32

The program above should yield the following results:

1. TagA is initialized with the Sine function.
2. TagB (which is located under **myFolder** in the simulator's structure) is initialized with a value of 0.
3. TagC isn't defined at the start, but that doesn't matter; the program creates a TagC initially because it will exist in the device. If this is the first time the program ran, then TagC started with a value of 0. If it is the second time through, it maintains the value at the end of the program until it reaches an interval where something changes.
4. Time Interval 1. The program doesn't state that any changes should be made, so we're done evaluating this Time Interval.
5. Time Interval 2. Again, nothing to do here, so we wait another Base Rate duration.
6. Time Interval 3, the value on TagB to 4.
7. Time Interval 4. The value of TagC changes to 100. If this program has executed at least once before, then TagC could already have a value of 100, since this is the only entry in the program that changes the value on Tag C. However, it is possible that the user (or something else, such as a binding or script) wrote another value to this Tag. At this point, our program is setting the value back to 100

8. Time Interval 20, TagA switches to the Ramp function. This whole time it has been using the Sine function to generate some moving numbers, but now we're telling it to use the Ramp function, which change the value (starting with the minimum value for the function) and uses a different method to determine its value.
9. Time Interval 30. The value of TagB changes to 55.
10. We're now at the end of the program. If Repeat was enabled, we'll move back to Time Interval 0 and start the whole process over again. If not, then the program ends for all Tags.

Preloaded Programs

The Programmable Device Simulator comes with preloaded programs for Generic, Dairy and SLC simulators, as well as the ability to load from a CSV.

Generic Simulator Program

The Generic Simulator Program provides a variety of Tags that offer different data types and value generation styles. For example, there are ramps, sine waves, and random values. Additionally, there is a set of static writable Tags whose values will persist while the device is running.

Dairy Simulator Program

The Dairy Simulator Program has a ControlLogix like structure with Compressor, Tank, Motor Tags and more. The folders are split into an Overview and Refrigeration section with Tags multiple levels deep that mimic a UDT in a ControlLogix device.

SLC Simulator

The SLC Simulator Program creates a simple device whose address structure mimics a basic SLC structure. These Tags are readable and writeable.

Load From CSV

The Load from CSV option allows you to select a CSV file to load a predefined program from. The CSV file must have four columns in a specific order (Time Interval, Browse Path, Value Source, Data Type), with each row representing a different instruction in the program.

Meta Tags

Each Programmable Device Simulator will have Meta Tags that allow users to interact with the program from the Designer/Session/Client. All Tags will be listed under the parent folder **[Controls]** within the device. Altered Meta Tag values do not persist after restart.

Term	Description
Base Rate	The number of milliseconds determining how quickly the program should move between each Time Interval. The initial duration of the Base Rate is defined in the Program Device Simulator settings should you need this value to persist after restarts.
Pause	Setting this Tag to 'True' will stop the program at its current Time Interval. Setting it back to 'False' will cause the program to resume from that point.
Program Counter	A count that tracks the Time Interval of the currently running program. This counter continues to increment for the duration of the program and resets when a program repeats.
Repeat	If True, the program will repeat indefinitely, meaning that once the last defined Time Interval has been reached, the program will start over again at Time Interval 0.
Reset	If True, the program is immediately reset to Time Interval 0 and the value will immediately change back to 'False.'

Export Instruction File

The ability to create custom instructions in the driver was added providing an opportunity to create a custom simulator device. You have the option of exporting the instruction file to a CSV formatted file and loading it once it's modified. You also have the ability to add and remove individual instructions. **Clear All** removes all instructions in the program.

Generic

Load Program

Generic Program

Load Simulator Program

Export Instruction File

Clear All

Time Interval	Browse Path	Value Source	Data Type	
0	Ramp/Ramp0	ramp(0.0, 10.0, 100.0, true)	Double	Remove
0	Ramp/Ramp1	ramp(5.0, 123.0, 230.0, true)	Double	Remove
0	Ramp/Ramp2	ramp(1.0, 2.0, 50.0, true)	Double	Remove
0	Ramp/Ramp3	ramp(-10.0, 10.0, 300.0, true)	Double	Remove
0	Ramp/Ramp4	ramp(0.0, 500.0, 1500.0, true)	Double	Remove
0	Ramp/Ramp5	ramp(0.0, 700.0, 2000.0, true)	Double	Remove
0	Ramp/Ramp6	ramp(0.0, 800.0, 2500.0, true)	Double	Remove
0	Ramp/Ramp7	ramp(0.0, 900.0, 3000.0, true)	Double	Remove
0	Ramp/Ramp8	ramp(0.0, 110.0, 3500.0, true)	Double	Remove
0	Ramp/Ramp9	ramp(0.0, 5000.0, 4000.0, true)	Double	Remove

[Add Instruction](#)

<< < 1 2 3 4 5 6 7 8 > >>

Save Program

BACnet

BACnet

Ignition provides a driver for BACnet (a data communication protocol for building automation and control networks). The first step before setting up devices and using the driver is to download the module. Go to the [Ignition downloads page](#) then refer to [Installing or Upgrading a Module](#).

The driver implements BACnet/IP over UDP. Any access to devices on other network media types (Ethernet (ISO-8802-3), MSTP, ARCNET, etc...) must be done through a gateway/router device.

On this page ...

- [BACnet](#)
- [Setup Process](#)
- [Creating Tags](#)
 - [Status_Flags and Document Tags](#)
 - [Writing BACnet Null Values](#)

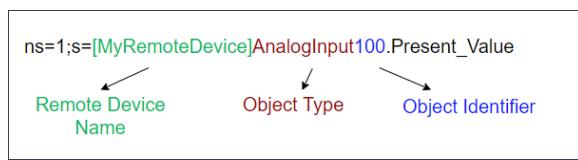
Setup Process

The process for setting up BACnet differs from other drivers in that you have to first set up a local device. After that you can configure a remote device.

1. Set up a [local device](#).
2. Set up a [remote device](#).

Creating Tags

The BACnet driver offers browsing support, making Tag creation as easy as dragging-and-dropping from the Tag Browser. However understanding how the OPC Item Path is read by the driver can be useful. The image below calls out some important parts of the item path.



Remote Device Name

The name of the Remote Device, as known by the OPC server.

Object Type

The BACnet object type. Currently supported types are:

- Device
- AnalogInput
- AnalogOutput
- AnalogValue
- LargeAnalogValue
- BinaryInput
- BinaryOutput
- BinaryValue
- MultiStateInput
- MultiStateOutput
- MultiStateValue

Object Identifier

The numerical identifier of the object.

Status_Flags and Document Tags

The Status_Flags Tags on each object are turn a JSON document, which requires parsing before you can see the values on the flags.



BACnet Addressing

[Watch the Video](#)

Tag Browser

Tag	Value	Data Type	Traits
Tags			
Data Types			
Present_Value OPC	-147.6	Float	
Status_Flags OPC	{"Value": [0], "ValidBits": [15]}	Document	
System			
All Providers			

While the individual flags can be accessed by manually creating Tags for each flag, in some cases you may want to handle the parsing with an expression instead of creating separate OPC Tags. This can easily be done with the [jsonGet](#) and [getBit](#) functions. Assuming an Expression Tag was created in the same directory as a Status_Flags Tag, we could use the following expression to extract the first bit of Status_Flags (which represents In_Alarm):

```
getBit(jsonGet({[.]Status_Flags}, 'Value[0]'), 0)
```

This feature is new in Ignition version **8.1.4**.
[Click here](#) to check out the other new features

Writing BACnet Null Values

As of release 8.1.4, you can write a `None` value (via Python) to a Tag that's subscribing to an address in a BACnet device.

In This Section ...

BACnet Local Device

Before you can start communicating with a remote device you must first configure a local device, representing Ignition's presence as a BACnet device on a network.

Local Devices are configured by specifying a local bind address, port, broadcast address, BACnet network, and BACnet device number. A Local Device can communicate with many remote devices as long as they are reachable on the same IP network.

Note: If you haven't downloaded the BACnet module yet, go to the [BACnet](#) page.

On this page ...

- [Device Configuration Strategies](#)
- [Configure a Local BACnet Device](#)
- [Device Settings](#)

Device Configuration Strategies

One strategy for configuring local devices is to configure one per network adapter that will be used for communication between the Ignition Gateway and remote BACnet devices.

You can also configure a local device to bind to a wildcard address such as 0.0.0.0, and in fact, this is required to receive broadcast packets from any of the remote devices you intend to communicate with. We've found that some devices will only respond to the BACnet Who-Is request with an I-Am that is sent to a broadcast address regardless of whether the original Who-Is was sent unicast or broadcast.

Another situation where you may need multiple local devices is when registration as foreign device with a BACnet Broadcast Management Device (BBMD) is necessary to bridge traffic between networks. Each Local device instance can only be registered with one BBMD.

Configure a Local BACnet Device

1. Go to the **Config** section of the **Gateway** Webpage.
2. Scroll down and select **BACnet > Local Devices**.



3. On the **Devices** page, click on **Create new Device**.

4. Select **BACnet/IP**, and click **Next**.
5. Fill in the following fields:

Name: **BACnet Local**

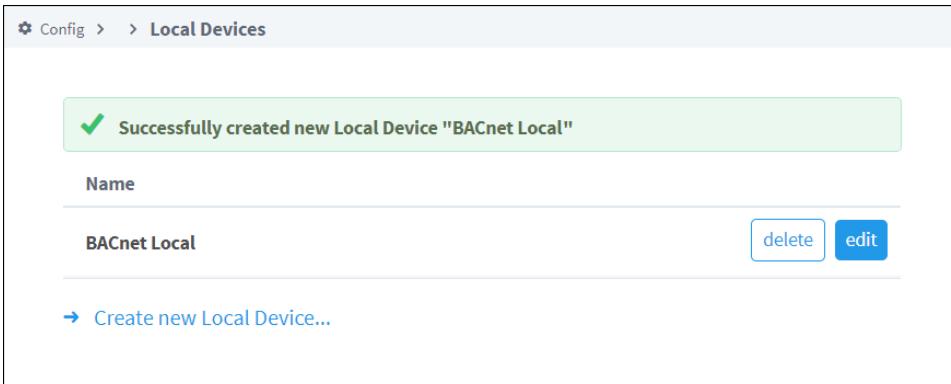
Bind Address: Enter the **IP address**, for example 10.10.###.##

Bind Port: Enter **local port** or leave default of 47,808.

Broadcast Address: Enter the **IP address**, for example 10.10.###.##

6. Leave the default values in the remaining fields.

7. Click **Create New Local Device**. You'll see the message "Successfully created new Local Device "BACnet Local"."



Now you can configure a driver for the [remote device](#).

Device Settings

General	
Name	Device name.
Bind Address	The local IP address to bind to. If this Local Device is going to receive broadcast packets (as opposed to unicast) from a Remote Device connection, then this must be set to a wildcard address (0.0.0.0).
Bind Port	This is the port the Device you are setting up will listen on. The local port to bind to. Default is 47,808.
Broadcast Address	A network address shared by other BACnet devices to send and receive UDP data.
Network Prefix Length	Default is 24.
Device Number	Default is 1,000.
Network Number	Default is 1.
Foreign Device Registration Enabled	If true, register as a foreign device with the configured BBMD (BACnet Broadcast Management Device). Default is false.
BBMD Address	Address of the BACnet Broadcast Management Device (BBMD) to register with. If you're planning on having this Local Device configuration work in conjunctions with a Remote Device via BBMD, then the address here should be the address of the BBMD on the same subnet as the Remote Device.
BBMD Port	Port the BACnet Broadcast Management Device to register with is listening on. Default is 47,808.

BACnet Remote Device

Once a [local device](#) has been configured a remote BACnet device can be added. You must know the IP address and device number of the remote device to configure a connection.

UDP is a connectionless protocol so a device is considered connected or initialized once Ignition has sent a Who-Is, received an I-Am, and then interrogated the remote device for some basic properties, including its object list and the supported properties of each object.

The object and properties are cached and subsequent connection attempts will only re-read them if the Database_Revision property changes. You can invalidate the browse data under the "More" menu for a given device (Config > OPC UA > Device Connections) on the devices page in the Ignition Gateway.

On this page ...

- [Connect to a Remote BACnet Device](#)
- [Device Settings](#)

Connect to a Remote BACnet Device

1. Go to the Config section of the Gateway Webpage.
2. Scroll down and select **OPC UA > Device Connections**.



3. On the Devices page, click on **Create new Device**.
4. Select **BACnet/IP**, and click **Next**.
5. Select the local device from the dropdown. If no local devices are shown, you'll need to set one up first. See [BACnet Local Device](#).
6. Fill in the following fields:

Name: **BACnet Remote**

Hostname: Enter the **IP address**, for example 10.10.###.##

Remote Device Number: The instance number of the remote device you're setting up.

7. Leave the default values in the remaining fields.
8. Click **Create New Device**.
9. The Devices page is displayed showing the **BACnet/IP** device is successfully created and added to Ignition.

Config > Opcua > Devices					
Successfully created new Device "BACnet Remote"					
Name	Type	Description	Enabled	Status	
BACnet Remote	BACnet/IP		true	Initialized	More ▾ edit
CLX	Allen-Bradley Logix Driver		true	Reconnect Wait	delete edit
DNP3	DNP3 Driver		true	Reconnect Wait	More ▾ edit

Device Settings

General	
Name	Device name.

Description	Device description.
Enabled	Default is set to true.

Other	
Local Device	The name of the local BACnet device instance that will be used when communicating with the remote device. If no local devices are shown, you'll need to set one up first. See BACnet Local Device .
Remote Address	The hostname or IP address of the remote device.
Remote Port	The port that the remote device is listening on. Default is 47,808.
Remote Device Number	The instance number of the remote device you're setting up. Default is 1.
Write Priority	The priority to use when writing to commandable properties. Default is 8.
COV Enabled	If true a Change of Value (COV) subscription is used for properties that support it. If false all properties are polled. Default is true. Note: COV subscriptions are only applied to Present_Value and Status_Flags.
COV Heartbeat Interval	Interval, in seconds, between reading the System_Status property of the Device object as a heartbeat. If three consecutive attempts fail, all COV items will be marked with uncertain quality. Default is 5.
COV Subscription Lifetime	Lifetime, in seconds, of COV subscriptions. Use 0 for indefinite. When non-zero COV subscriptions are renewed at a rate of 75% of the lifetime. Default is 900.
Confirmed Notifications Enabled	If true COV subscriptions use confirmed notifications. If false COV subscriptions use unconfirmed notifications. Default is false.
Discovery Timeout	This feature is new in Ignition version 8.1.4 Click here to check out the other new features Duration, in seconds, to wait for the remote device discovery process to complete. Default is 5.

Tag Historian

The Tag Historian module provides power and flexibility for storing and accessing historical data. When history is enabled on an Ignition Tag, data is stored automatically in your SQL database in an efficient format. This data is then available for querying through scripting, historical bindings, and reporting. Options for partitioning and deleting old data help to ensure the system stays maintained with minimal extra work. Also, you can drag-and-drop Tags directly onto an Easy Chart component to create trends or onto a table to display historical values. Tag Historian's robust querying features provide you great flexibility in how you retrieve the stored data.

Tag Historian Querying

While the data is stored openly in the database, it does not lend itself well to direct querying. Ignition offers a range of built-in querying options that are very powerful and flexible. In addition to simple on-change querying, the system can perform advanced functions such as querying many Tags from multiple providers, calculating their quality, interpolating their values, and coordinating their timestamps to provide fixed resolution returns. Tag history bindings allow you to pull Tag history data that is stored in the database into a component through a binding. The binding type, which is only available for Dataset type properties, runs a query against the Tag Historian.

For more information, see [Tag History Bindings in Perspective](#) or [Tag History Bindings in Vision](#).

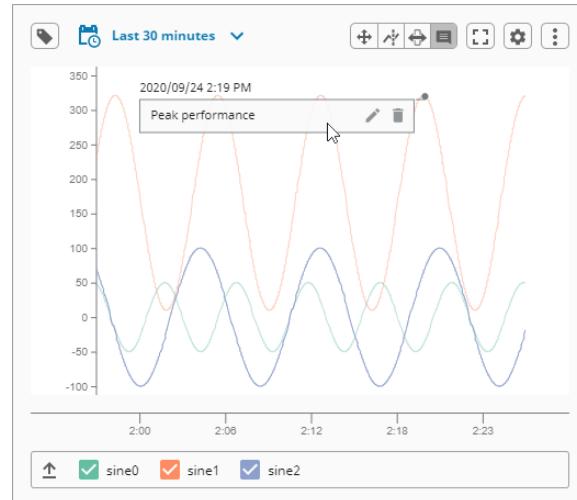
Querying can be performed on tables and charts through the Historical binding, Nested Queries, and through scripting. You can also query Tags from the [Reporting module](#).

On this page ...

- [Tag Historian Querying](#)
- [Charts that Display Historian Data](#)
- [Store and Forward](#)
- [Data storage](#)

Charts that Display Historian Data

With the Vision module [Easy Chart component](#) you can create powerful timeseries charts from Tag Historian data. As of 8.1.0, the Perspective module has a [Power Chart](#) component with similar capabilities. Both components enable you to drag and drop history-enabled Tags onto a chart to create chart pens and data trends. Your charts and graphs can include subplots, axes, digital offsets, and moving averages. You can quickly and easily turn your historical and realtime data into dynamic charts and graphs for your users. These charts can be configured in the runtime to give users quick access to data in the time range they need.



Store and Forward

The [Store and Forward](#) system provides a reliable way for Ignition to store historical data to the database. The Store and Forward system is not exclusively part of Tag history, but systems such as the Tag Historian and Transaction Groups use it to prevent data loss and store data efficiently using a record cache.

Data storage

Historical Tag values pass through the [Store and Forward](#) system before they are stored in the database connection associated with the historian provider. The data is stored according to its data type directly to a table in the SQL database, with its quality and a millisecond resolution timestamp. The data is only stored on-change, according to the value mode and deadband settings on each Tag, thereby avoiding duplicate and unnecessary

data storage. The storage of scan class execution statistics ensures the integrity of the data. While advanced users may change the table according to their database to be more efficient (for example, using a compression engine), Ignition does not perform binary compression or encrypt the data.

[In This Section ...](#)

Tag History Gateway Settings

Configuring Tag History Settings

Tag History storage is easy to set up quickly, but there are also some settings that can be adjusted to allow for differences in database storage space and performance needs.

Table Partitioning

Ignition has the ability to automatically break up data into different tables of fixed duration. This can help make data maintenance easier by preventing tables from becoming too large. Tables can easily be deleted in order to prune old data, and the database is able to better optimize access to frequently retrieved rows. The built-in partitioning feature can be used with any database.

It is important to note the difference between this feature and any partitioning options that the database might provide. Most modern databases offer their own faculties for defining "partitions", offering similar and greater benefits. While Ignition cannot use these features directly, advanced users may choose to apply these features on top of what Ignition currently offers.

Data Compression

Ignition does not perform any binary compression on the data. That is, values are stored directly in standard database tables. However, to reduce the number of values stored, Ignition offers two different algorithms for pre-compressing the data (trimming unnecessary values). The two modes correspond to the value mode property of the Tag. The value mode (**Discrete** or **Analog**) dictates the type of value that the Tag represents, affects how the deadband is applied to values, and how interpolation is performed when querying.

- **Discrete**

The value uses a simple deadband and is only stored when a new value is +/- the deadband value away from the previously stored value.

- **Analog**

The deadband is used to form a corridor along the trajectory of the value. A new value is only stored when it falls outside the previous corridor. When this occurs, the trajectory is recalculated and a new corridor is formed.

Typically, Discrete is used for boolean or integers that represent state, and Analog is used for floats or integers that change more often (which is why you want to perform compression). While advanced users can change the table according to their database to be more efficient (for example, using a compressed engine), Ignition does not perform binary compression or encrypt the data in any way. See Deadband Style, in [Tag Properties Table](#) for more information about the difference between Discrete and Analog values.

Datasource History Providers

Datasource History Providers can not be created or deleted, but are instead tied to a database connection. They are automatically added when connecting to a new database and removed after the database connection is removed. It comes pre-configured to partition every month, but the provider can be edited to change its behavior.

There are two other major options to configure on the provider: pre-processed partitions and data pruning. With pre-processed partitions, the data that is stored is summarized and then placed into another table in the database. While this takes up more space in the database, it can improve query speed by reducing the amount of data points that must be loaded. Data pruning will automatically remove old data from your system after it reaches an age that you set. It will only remove whole tables though. If each partitioned table represents a month and the pruning system removes data that is three months old, it will wait until all the data in the oldest table is three months old before pruning it.

Editing Datasource History Providers

The following table lists the settings for the Datasource History Providers. To access these settings, go to the **Config** tab of the Gateway Webpage and select **Tags > History**. Then click the **Edit** button for the provider you want to update.

Main	
Provider Name	Name of the Tag History Provider. By default, this will match up with the name of the database connection.
Enabled	If the check box is selected (enabled), the provider is turned on and accepts tag history data. If disabled, the database is not shown in the list of history providers when configuring tag history from the Designer. Also, any data logged to the provider, will error out and be quarantined by the store and forward engine, if possible.

On this page ...

- Configuring Tag History Settings
 - Table Partitioning
 - Data Compression
- Datasource History Providers
 - Editing Datasource History Providers
 - OPC-HDA Provider
 - Internal History Provider
 - Remote History Provider
 - Tag History Splitter

Description	A description of the provider.
Data Partitioning	
Enable Partitioning	To improve query performance, Tag Historian can partition the data based on time. Partitions will only be queried if the query time range includes their data, thereby avoiding partitions that aren't applicable and reducing database processing. On the other hand, the system must execute a query per partition. It is therefore best to avoid both very large partitions, and partitions that are too small and fragment the data too much. When choosing a partition size, it is also useful to examine the most common time span of queries.
Partition Length and Units	The size of each partition, the default is one month. Many systems whose primary goal is to show only recent data might use smaller values, such as a week, or even a day.
Enabled Pre-processed Partitions	Pre-processed partitions will use more space in the database, but can improve query speed by summarizing data, reducing the amount that must be loaded.
Pre-processed Window Size (seconds)	When pre-processing is turned on, the data will be summarized into blocks of this size.
Data Pruning	
Enable Data Pruning	Partitions with data older than a specific age are deleted. The check box is not selected/enabled by default. Note: Data pruning works by deleting old partitions. Therefore, data will only be removed when a partition has no data younger than the prune age.
Prune Age and Units	The maximum age of data. As mentioned, the data is deleted by the partition, and could therefore surpass this threshold by quite a bit before all of the data in the partition is old enough to be dropped.
Advanced	
Enable Stale Data Detection	If enabled, tracks scan class executions to determine the difference between unchanging values, and values that are flat due to the system not running.
Stale Detection Multiplier	The multiplier for scan class rate used to determine when values are stale. If scan class execution is not recorded within this amount of time, values will be considered bad on query.

OPC-HDA Provider

Establishes a connection to an [OPC-HDA Server](#) to read history data that may be stored there from a third party. Ignition can not write to this type of history provider.

Note: This requires the [OPC COM](#) to be installed.

Internal History Provider

The Edge Historian is an internal history provider, is available on standard Ignition Gateways.

To set up an Edge history provider, do the steps the follow:

1. Go to the Config section of the Gateway Webpage and select **Tags > History**.
2. Click **Create New Historical Tag Provider**.

3. Select the **Internal Historian** radio button and click **Next**.

The screenshot shows the Ignition configuration interface. On the left, there's a sidebar with 'Config' selected. The main area is titled 'Config > Tags > History'. It lists four provider options:

- Internal Historian** (selected, highlighted with a red box): A built-in local historian with limited storage.
- OPC-HDA Provider**: A historical tag provider that uses OPC-HDA to retrieve values from a 3rd party historian.
- Remote History Provider**: Sends tag history through the Gateway Network for storage in a remote history provider.
- Tag History Splitter**: Stores tag history concurrently to two other connections in the gateway.

A 'Next >' button is located at the bottom right of the main panel.

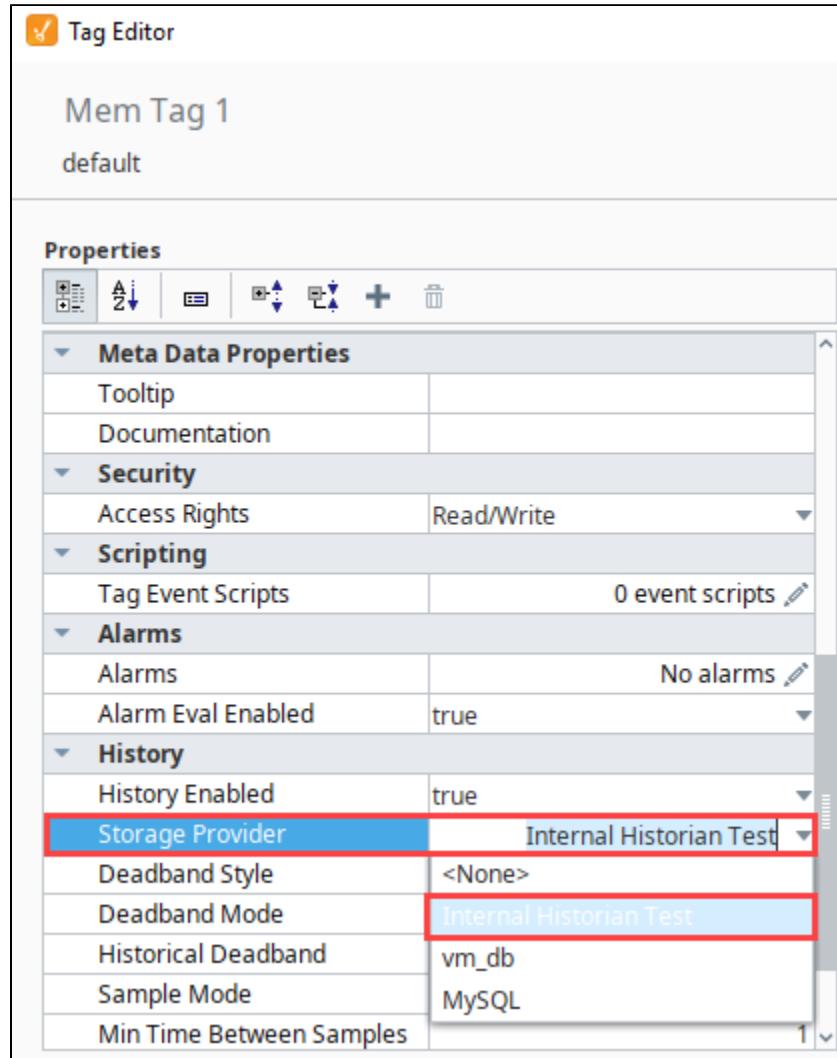
4. Fill in the properties in the table.

Main	
Provider Name	Name of the Edge History Provider.
Enabled	If the check box is selected (enabled), the provider is turned on and accepts tag history data. Default is true.
Description	A description of the provider.
Limits (Requires Tag Historian Module License)	
Time Limited Enabled?	Whether or not time limit is enabled. Default is true.
Time Limit Size	Size of the time limit. Unit (seconds, weeks, etc) is set in the Time Limit UnitsDefault is 1.
Time Limit Units?	Options are milliseconds, seconds, minutes, hours, days, weeks, months, or years. Default is WEEK.
Point Limit Enabled?	Whether or not point limit is enabled. Default is true.
Point Limit Size	Maximum number of data points the historian will store. Default is 10,000,000.
Sync Frequency	The frequency with which data will be sent to the remote Gateway. This setting will be used in conjunction with the sync schedule, if enabled. Default is 10.
Sync Frequency Units	The unit of time that will be used with the Sync Frequency. Options are milliseconds, seconds, minutes, hours, days, weeks, months, or years. Default is SEC.
Max Batch Size	The maximum number of data points that will be sent per batch to the remote Gateway. Default is 10,000.
Enable Schedule	If enabled, the data will only be synchronized during the times specified by the pattern provided. Default is false.
Schedule Pattern	A comma separated list of time ranges. Examples: <ul style="list-style-type: none"> • 9:00-15:00 • 9pm-5am • 20.30-04.30
Sync Settings (Requires EAM Module Licensing)	

Remote Sync Enabled	Allows you to turn Tag History Synchronization on or off. Default is false.
Remote Gateway Name	The Gateway to target for remote synchronization. Must have the Tag Historian module installed, and allow remote storage. The Ignition Gateway's security settings will also need to be configured to allow remote storage.
Remote Provider Name	The remote history provider to sync data to.

5. After filling in the properties in the table as desired, click **Create New Historical Tag Provider**.

Once an Edge History Provider is set up, you can select it as the storage provider for your tags. For example, in the following image, and Edge History Provider named "Internal Historian Test" is selected for storing history on a memory tag.



Remote History Provider

A Remote History Provider is a link to a historical provider on another Gateway. Since it is grabbing historical Tag data from another provider, its only configuration is to ensure it is pointed at the correct Tag provider. You can't change any of the settings like partition length and prune age, but would instead have to change those settings on the original history provider on the remote Gateway. By default, the remote history provider will fall under the [Default Security Zone](#) and be read only.

To set up a Remote History Provider, do the steps that follow:

1. Go to the **Config** section of the Gateway Webpage and select **Tags > History**.
2. Select the **Remote History Provider** radio button.

3. A list of known Gateways appears. If the Gateway is not currently available or displayed here, you can specify its name manually.

4. Select a Gateway and click **Next**.
 5. Fill in the properties in the table.

Main	
Provider Name	Name of the Tag History Provider.
Enabled	If the check box is selected (enabled), the provider is turned on and accepts Tag history data. Default is true. If disabled, the database is not shown in the list of history providers when configuring Tag history from the Designer. Also, any data logged to the provider, will error out and be quarantined by the store and forward engine, if possible.
Description	A description of the provider.
Remote Gateway	
Remote Gateway Name	The name of the remote Gateway.
Remote History Provider	The name of the provider on the remote Gateway. This does not have to match the provider name on the local Gateway.

Storage	
Allow Storage	If false, the provider will only be used for querying historical data. If true, the provider will create a store and forward pipeline for sending data to a remote Gateway. Default is true.
Max Bundle Size	The maximum number of data points that can be sent per request. This value is used in conjunction with the store and forward settings to dictate how much data is sent at once. 0=unlimited

6. After filling in the properties in the table, click **Create New Historical Tag Provider**.

Tag History Splitter

This provider combines two separate providers into a single new provider. When setting up a Tag to store history, selecting this provider will write the same data to both providers that it has selected. The Tag History Splitter is useful for automatically creating a backup of your data, or for reading history from two separate providers. Learn more about setting up the [Tag History Splitter](#) here.

Related Topics ...

- [Configuring Tag History](#)

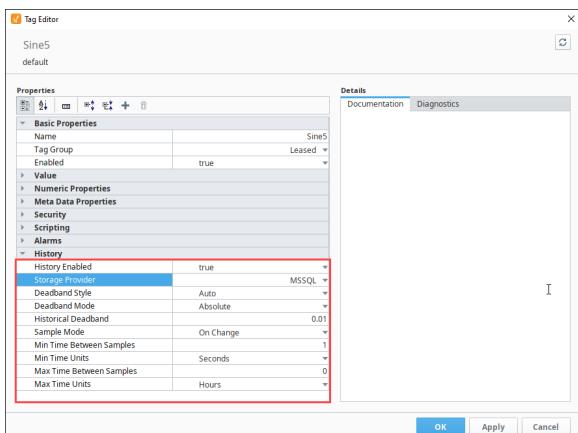
Configuring Tag History

Logging data is easy with [Tag Historian](#). Once you have a database connection, all you do is set the Tags to store history and Ignition takes care of the work. Ignition creates the tables, logs the data, and maintains the database.

The historical Tag values pass through the store-and-forward engine before ultimately being stored in the database connection associated with the historian provider. The data is stored according to its data type directly to a table in the SQL database, along with its quality and a millisecond resolution time stamp. The data is only stored on-change, according to the value mode and deadband settings on each Tag, thereby avoiding duplicate and unnecessary data storage. The storage of scan class execution statistics ensures the integrity of the data.

Tag Configuration

The first step to storing historical data is to configure Tags to record values. This is done from the [History](#) section of the Tag Editor in the Designer. Select the **History Enabled** property to turn on history. The properties include an Historical [Tag group](#) that will be used to check for new values. Once values surpass the specified deadband, they are reported to the history system, which then places them in the proper store and forward engine. Complete information on the History properties (and all properties in the Tag Editor), can be found on the [Tag Properties Table](#).



Sample Mode

The Sample Mode setting determines how often a historical record should be collected.

- **On Change** - Collects a record whenever the value on the Tag changes.
- **Periodic** - Collects a record based on the **Sample Rate** and **Sample Rate Units** properties.
- **Tag Group** - Collects a record based on the Tag Group specified under the **Historical Tag Group** property.

Historical Tag Group

Historical Tag Group setting shows up with Sample Mode is set to Tag Group. Historical Tag Group setting determines how often to record the value on the Tag. It uses the same [Tag Groups](#) that dictate how often your Tags should execute. Typically, the Historical Tag Group should execute at the same rate as the Tag's Tag Group or slower: if a Tag's Tag Group is set to update at a 1,000ms rate, but the Historical Tag Group is set to a Tag Group that runs at 500ms rate, then the Tag History system will be checking the Tag's value twice between normal value changes, which is unnecessary.

Max and Min Time Between Samples

Normally Tag Historian only stores records when values change. By default, an "unlimited" amount of time can pass between records – if the value doesn't change, a new row is never inserted in the database. By modifying these settings, it is possible to specify the maximum number of scan class execution cycles that can occur before a value is recorded. Setting the value to 1, for example, would cause the Tag value to be inserted each execution, even if it has not changed. Given the amount of extra data in the database that this would lead to, it's important to only change this property when necessary.

Deadband and Analog Compression

The deadband value is used differently depending on whether the Tag is configured as a Discrete Tag or as an Analog Tag. Its use with discrete values is straight forward, registered a change any time the value moves +/- the specified amount from the last stored value. With Analog Tags,

On this page ...

- [Tag Configuration](#)
 - [Sample Mode](#)
 - [Max and Min Time Between Samples](#)
 - [Deadband and Analog Compression](#)
 - [Log Tag History Data](#)
 - [Setting a UDT to Log History Data](#)



Configuring Tag History

[Watch the Video](#)

however, the deadband value is used more as a compression threshold, in an algorithm similar to that employed in other Historian packages. It is a modified version of the 'Sliding Window' algorithm. Its behavior may not be immediately clear, so the following images show the process in action, comparing a raw value trend to a "compressed" trend.

The Deadband Style property sets the: Auto, Analog, or Discrete.

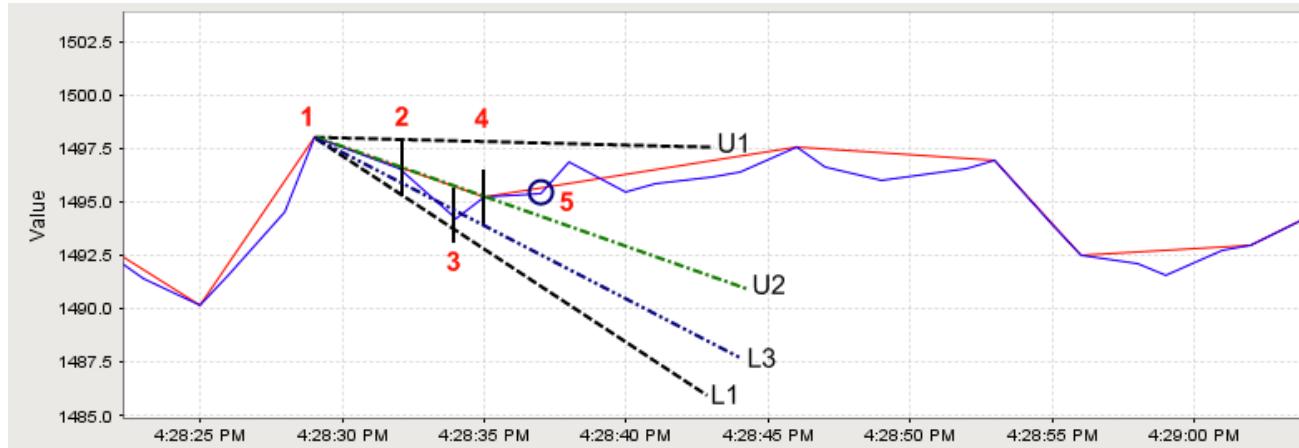
When set to Auto, this setting will automatically pick from Analog or Discrete, based on the data type of the Tag.

- If the data type of the Tag is set to a float or double, then Auto will use the Analog Style.
- If the data type of the Tag is any other type, then the Discrete style will be used.



In this image, an analog value has been stored. The graph has been zoomed in to show detail; the value changes often and ranges over time +/- 10 points from around 1490.0. The compressed value was stored using a deadband value of 1.0, which is only about .06% of the raw value, or about 5% of the effective range. The raw value was stored using the Analog Tag mode, but with a deadband of 0.0. While not exactly pertinent to the explanation of the algorithm, it is worth noting that the data size of the compressed value, in this instance, was 54% less than that of the raw value.

By looking at one specific sequence, we can see how the algorithm works:



The sequence starts with the second stored compressed value on the chart.

1. A value is stored. No further action is taken.
2. The next value arrives. A line is made through the value, with the size of the specified deadband value. A line is projected from the last stored value to the upper (line U1), and lower (line L1), bounds of this new value line. This establishes the initial corridor.
3. A new value arrives. The same procedure is taken, and new lines are created. However, only lines that are more restrictive than the previous are used. In this case, that means only line U2, the new upper line.
4. Another value arrives, causing a new lower line (L3) to be used.
5. Finally, a value arrives that falls outside of our corridor. The last received value (value 4) is stored, and the process is started again from that point.

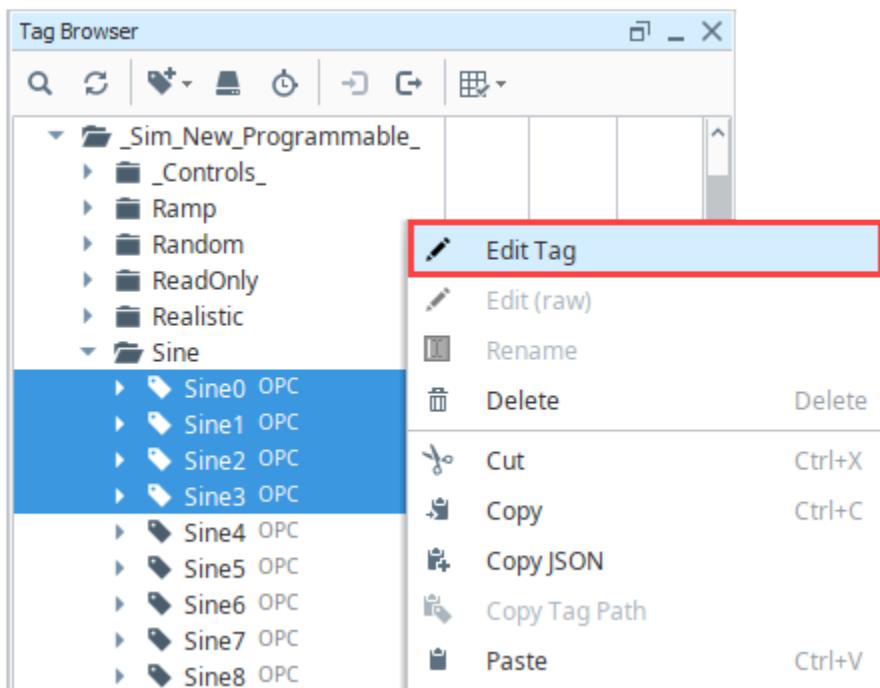
Log Tag History Data

Do the following steps to log history data for your Tags:

1. In the Tag Browser, select one or more Tags. For example, we selected several **Sine** Tags in the Sine folder.

2. Right-click on the selected Tags, and then select **Edit Tag**.

The Tag Editor window is displayed. Here, you can edit the Tag and change the name, data type, scaling options, metadata, permissions, history, and alarming.

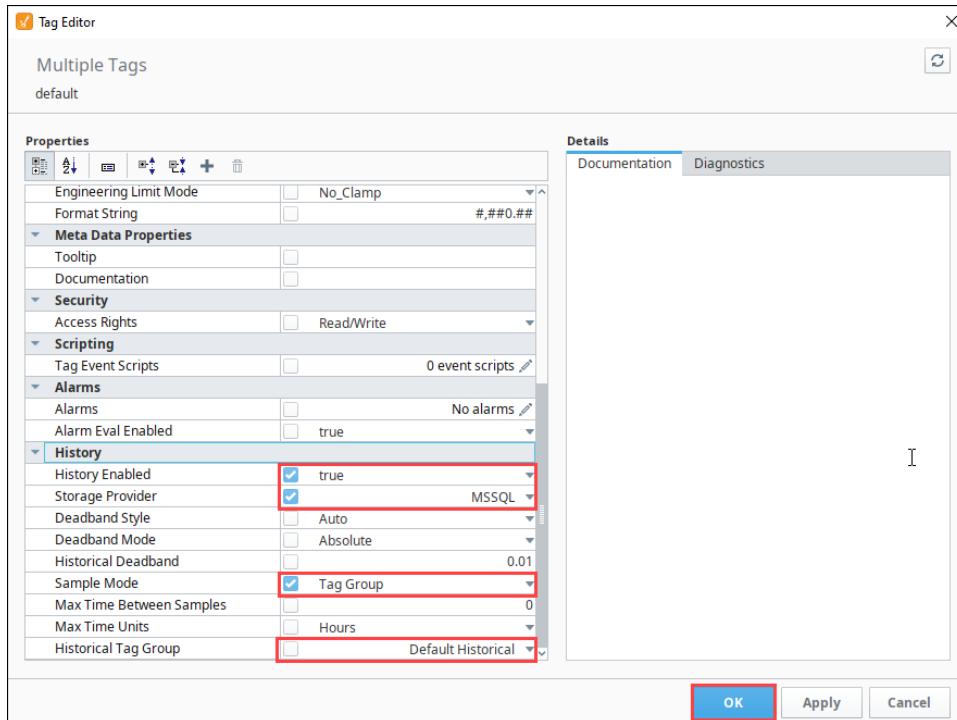


3. Scroll down to the History section of the Tag Editor. Select the **History Enabled** check box.

4. Choose a database (for example, MySQL) from the **Storage Provider** dropdown.

5. Set the Sample Mode to **Tag Group**.

6. Set the Historical Tag Group to **Default Historical**.



7. Click **OK**. Now look in the Tag Browser. To the right of each Sine Tag that is storing history, a **History** icon appears letting you know it is set up.

			History
▶	ReadOnly		
▶	Realistic		
▼	Sine		
▶	Sine0 OPC	-94.78	Double
▶	Sine1 OPC	9.3	Double
▶	Sine2 OPC	0.92	Double
▶	Sine3 OPC	38.12	Double
▶	Sine4 OPC	-29.42	Double
▶	Sine5 OPC	-94.78	Double

If you were to look in your database, you can see all the tables and data Ignition has created for you.

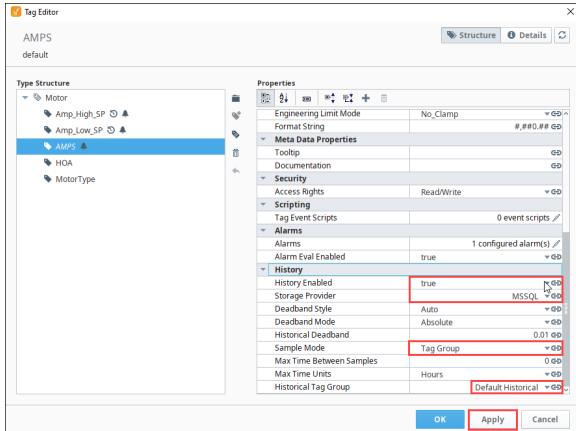
Setting a UDT to Log History Data

You can set a UDT to log history data, then all the instances of that UDT will log data without having to edit the individual instances.

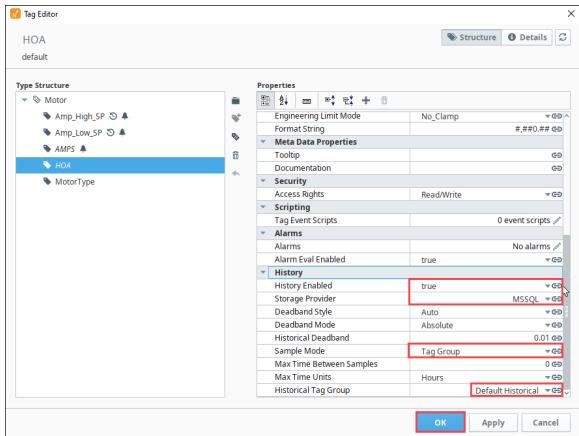
1. In the Tag browser, right-click on the UDT (for example, a Motor UDT) and select the **Edit Tag** icon.
The Tag Editor is displayed.
2. In Tag Editor, click a Tag (for example, the AMPS Tag). Scroll down to the **History** section.
3. Set the following properties in the Tag Editor:
 - History Enabled:** true
 - Storage Provider:** MSSQL
 - Sample Mode:** Tag Group
 - Historical Tag Group:** Default Historical

Add History to Tags in UDT

[Watch the Video](#)



4. Click **Apply**.
5. Next, select the HOA Tag.
6. Set the following properties in the Tag Editor:
 - History Enabled: **true**
 - Storage Provider: **MSSQL**
 - Sample Mode: **Tag Group**
 - Historical Tag Group: **Default Historical**
7. Click **OK** to save the changes to the UDT. Now every motor instance automatically starts logging data.



Data Partitioning and Pruning

Tag Historian will partition data into separate tables according to the time setting so that one table doesn't grow indefinitely, and then will delete old data to ensure the system is maintained for query performance. By default, partitioning is enabled to improve query performance. Tag Historian partitions and breaks up the data into separate tables based on time. Partitions will only be queried if the query time range includes their data, thereby avoiding partitions that aren't applicable and reducing database processing. On the other hand, the system must execute a query per partition. It is therefore best to avoid both very large partitions, and partitions that are too small and fragment the data too much. When choosing a partition size, it is also useful to examine the most common time span of queries. The data prune feature will delete partitions with data older than a specific age/time.

On this page ...

- [Partition and Prune Data](#)
- [History Table Timestamps](#)



Data Partitioning and Pruning

[Watch the Video](#)

Partition and Prune Data

1. Go to the **Config** tab of the Gateway.
2. Select **Tags > History** from the menu on the left.

The **Historical Tag Providers** page is displayed. You can see the Databases that have Enabled tag history on and their Status shows as Running.

The screenshot shows a table with columns: Provider Name, Enabled, Type, Description, and Status. There are two rows:

Provider Name	Enabled	Type	Description	Status
MySQL	true	Datasource History Provider		Running edit
New History Provider	true	Tag History Splitter		Running delete edit

Below the table is a link to "Create new Historical Tag Provider..." and a note: "Note: For details about a provider's status, see the [Tag Providers Status](#) page."

3. Click on **edit** at the far right of the provider you want to update.
4. Once you've made changes, click **Save Changes** at the bottom of the screen.

The following table describes all the settings available for Tag History:

Main	
Provider Name	Name of the Tag History Provider, for example, MySQL.
Enabled	By default, the check box is selected (enabled) meaning the provider is turned on and accepts tag history data.
Description	Description of the Tag History Provider (optional).

Data Partitioning	
Enable Partitioning	The built-in partitioning system breaks up data into separate tables of a specified time frame. This can improve performance and make certain maintenance tasks easier. Default is true.
Partition Length	The size of each partition, the default is one table per month. Many systems whose primary goal is to show only recent data might use smaller values, such as a week, or even a day. Default is 1.
Partition Units	Unit of time for the partition length. Options are: Milliseconds, Seconds, Minutes, Hours, Days, Weeks, Months, and Years. Default is Months.
Enable Pre-processed Partitions	Pre-processed partitions will use more space in the database, but can improve query speed by summarizing data, reducing the amount that must be loaded. Default is false.
Pre-processed Window Size (seconds)	When pre-processing is turned on, the data will be summarized into blocks of this size. Default is 60.
Data Pruning	
Enable Data Pruning	Partitions with data older than a specific age are deleted and if the data is not archived, the data is then lost. Default is false. Note: Data pruning works by deleting old partitions. Therefore, data will only be removed when a partition has no data younger than the prune age.
Prune Age	The maximum age of data. As mentioned, the data is deleted by the partition, and could therefore surpass this threshold by quite a bit before all of the data in the partition is old enough to be dropped. Default is 1.
Prune Age Units	Unit of time for the prune age. Options are: Milliseconds, Seconds, Minutes, Hours, Days, Weeks, Months, and Years. Default is Years.
Show Advanced Properties	Select this option to display the Advanced properties below:
Advanced	
Enable Stale Data Detection	If enabled, tracks tag group executions to determine the difference between unchanging values, and values that are flat due to the system not running. Default is true.
Stale Detection Multiplier	The multiplier for tag group rate used to determine when values are stale. If tag group execution is not recorded within this amount of time, values will be considered bad on query. Default is 2.

History Table Timestamps

If you've looked behind the scenes of SQLTags Historian, you've probably noticed the timestamps are not stored as standard SQL timestamps. They are stored in a variant of Unix time, or the number of milliseconds since January 1, 1970 00:00:00. The time may come when you need to convert that timestamp to a more human-readable format. The following describes how to do it in MySQL and MSSQL.

Both examples below assume the partition table is named 'sqlt_data_1_2016_08'.

MySQL

It's pretty easy to deal with Unix timestamp in MySQL because they have a built-in function for doing so. The FROM_UNIXTIME() function will take in a Unix timestamp and return the current timestamp.

Usage:

```
SELECT FROM_UNIXTIME(t_stamp/1000) FROM sqlt_data_1_2016_08
```

MSSQL

In Microsoft SQL Server, it's a little more verbose. We use the DATEADD() function to figure out the timestamp.
Usage:

```
SELECT DATEADD(s,t_stamp/1000,'1970-01-01 00:00:00') FROM sqlt_data_1_2016_08
```


Custom Tag History Aggregates

Python Aggregation Functions

The Tag History system has many built-in aggregate function, such as Average, Sum, and Count. However a custom aggregate may be defined via Python scripting. These functions are used for calculations across timeframes, and they process multiple values in a "window" into a single result value.

For example, if a query defines a single row result, but covers an hour of time (either by requesting a single row, or using the Tag Calculations feature), the system must decide how to combine the values. There are many built in functions, such as Average, Sum, Count, etc. Using a custom Python aggregate, however, allows you to extend these functions and perform any type of calculation.

Note: When calling a custom tag history aggregate, the returnSize argument must be set to natural (returnSize = 0). If the returnSize is set to -1, or left with its default value, the the custom aggregate will be ignored.

On this page ...

- [Python Aggregation Functions](#)
- [Description](#)
 - [Parameters](#)
 - [Return Value](#)
- [Usage](#)
- [Examples](#)
 - [Creating an Aggregate Function on the Fly](#)

Description

As values come in, they will be delivered to this function. The interpolator will create and deliver values. For each window (or "data block", the terms are used synonymously), the function will get a fresh copy of blockContext. The blockContext is a dictionary that can be used to as a memory space. The function should not use global variables. If values must be persisted across blocks, they can be stored in the queryContext, which is also a dictionary.

The function can choose what data to include, such as allowing interpolation or not, and allowing bad quality or not.

The window will receive the following values, many of which are generally interpolated (unless a raw value happens to fall exactly at the time):

- The start of the window
- 1 ms before each raw value (due to the difference between discrete and analog interpolation. A value equal to the previous raw value indicates discrete interpolation)
- The raw value
- The end of the window

At the end of the window, the function will be called with "finished=true". The function should return the calculated value(s). The resulting value will have a timestamp that corresponds to the beginning of the block timeframe.

Parameters

- qval - The incoming QualifiedValue. This has:
 - value : Object
 - quality : Quality (which has 'name', 'isGood()')
 - timestamp : Date
- interpolated - Boolean indicating if the value is interpolated (true) or raw (false)
- finished - Boolean indicating that the window is finished. If true, the return of this particular call is what will be used for the results. If false, the return will be ignored.
- blockContext - A dictionary created fresh for this particular window. The function may use this as temporary storage for calculations. This object also has:
 - blockId - Integer roughly indicating the row id (doesn't take into account aggregates that return multiple rows)
 - blockStart - Long UTC time of the start of the window
 - blockEnd - Long UTC time of the end of the window
 - previousRawValue - QualifiedValue, the previous non-interpolated value received before this window
 - previousBlockResults - QualifiedValue[], the results of the previous window.
 - insideBlock(long) - Returns boolean indicating if the time is covered by this window.
 - get(key, default) - A helper function that conforms to python's dictionary "get with default return".
- queryContext - A dictionary that is shared by all windows in a query. It also has:
 - queryId - String, an id that can be used to identify this query in logging
 - blockSize - Long, time in ms covered by each window
 - queryStart - Long, the start time of the query
 - queryEnd - Long, the end time of the query
 - logTrace(), logDebug(), logInfo() - all take (formatString, Object... args).

Return Value

- Object - Turned into Good Quality qualified value

- List - Used to return up to 2 values per window
- Tuple - (value, quality_int)
- List of quality tuples

Usage

Custom Python aggregates can be used in two ways:

1. Defined as a Project Library script, where the full path to the function is passed to the query.
2. Defined as a string, prefaced with "python:", and passed to the query.

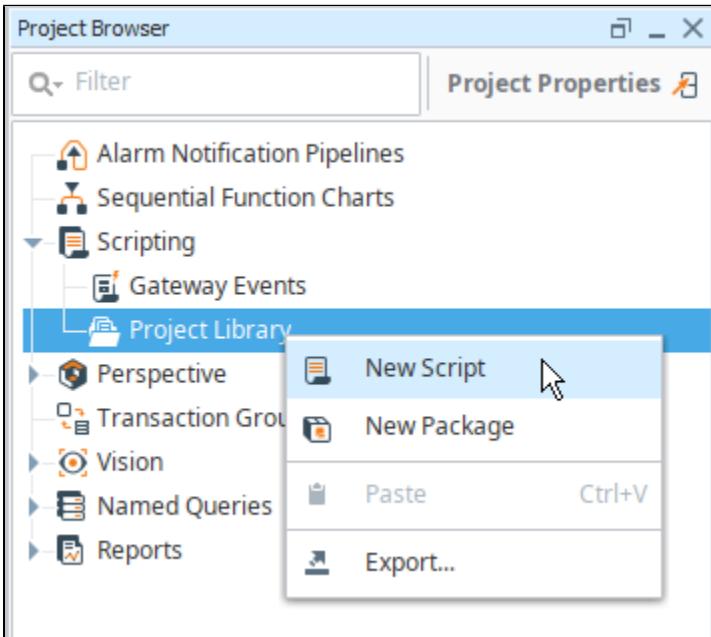
Currently both options are only available through the system.tag.queryTagHistory/queryTagCalculations functions. Later versions of Ignition will expose this functionality through the graphical binding interfaces.

Both of these options are used with the "aggregationMode" and "aggregationModes" parameters to system.tag.queryTagHistory, and the "calculations" parameter of system.tag.queryTagCalculations. If the value is not an Enum value from the defined AggregationModes, it will be assumed to be a custom aggregate. The system will first see if it's the path to a Project Library script, and if not, will then try to compile it as a full function.

For performance reasons, it is generally recommended to use the Project Library script whenever possible. For more information, see [Project Library](#).

Examples

1. Add a project script, by right clicking the **Project Library** item in the Project Browser, and choosing the **New Script** option.

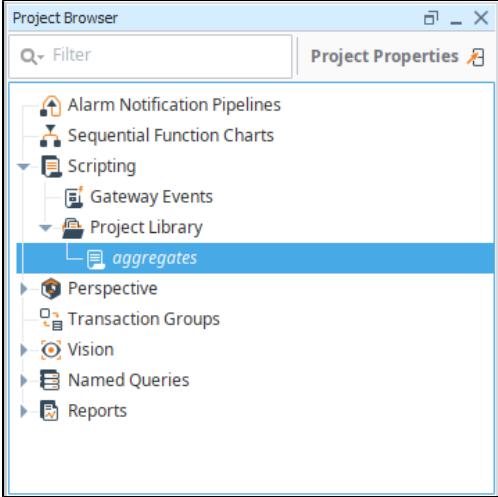


2. Enter a name for the script and click **Create Script** (we named ours aggregates in the example).
3. Enter the following code block:

Example

```
# this is a simple count function, called for each value in a time window
def myCount(qval, interpolated, finished, blockContext, queryContext):
    cnt = blockContext.getOrDefault('cnt',0)
    if qval.quality.isGood():
        blockContext['cnt']=int(cnt)+1

    if finished:
        return blockContext.getOrDefault('cnt', 0)
```



```

aggregates
1 # this is a simple count function, called for each value in a time window
2 def myCount(qval, interpolated, finished, blockContext, queryContext):
3     cnt = blockContext.getOrDefault('cnt',0)
4     if qval.quality.isGood():
5         blockContext['cnt']=int(cnt)+1
6
7     if finished:
8         return blockContext.getOrDefault('cnt', 0)

```

The custom function could be used by using the example below:

Note: Note the use of the "shared" prefix to our aggregate function reference. This is required for the use of custom aggregate functions even though it is not normally required to invoke Project Library defined functions.

Example

```

#Return tag history using a custom aggregate function you wrote.

system.tag.queryTagHistory(paths=['MyTag'], rangeHours=1, aggregationModes=['shared.aggregates.myCount'],
returnSize = 0)

```

Creating an Aggregate Function on the Fly

Example

```

#create a function on the fly to pass in as a custom aggregate.

wrapper = """\
python:def wrapper(qval, interpolated, finished, blockContext, queryContext):
    return shared.aggregates.customFunction(qval, interpolated, finished, blockContext, queryContext)
"""

system.tag.queryTagHistory(paths=['MyTag'], rangeHours=1, aggregationModes=[wrapper], returnSize = 0)

```

Tag History Splitter

The Tag History Module has a provider type called the Tag History Splitter. Like the Remote History Provider, it doesn't store history on its own, it relies on having other providers already set up. A Splitter provider simply logs Tag History into multiple existing History Providers.

Some users prefer to have data recorded by the Tag Historian sent to multiple databases: project specifications require redundant logging, or users at another facility would like to have a copy of the data in their local database. In cases like this, the Tag History Splitter Provider is ideal.

The Gateway that these Tags reside in must have multiple Tag History Providers configured. Should one of the providers fault, the Store and Forward system will kick in to maintain the data on the faulted connection. Since each database connection has its own Store and Forward engine, the data is always forwarded to the correct database.



Tag History Splitter Provider Properties

Below are the properties available on the Historical Tag Provider.

Main	
Provider Name	Name of the connection.
Enabled	Enables and disables the connection.
Description	Description of the connection. The description appears on the Historical Tag Providers page of the Gateway.
Storage	
First Connection	Data is stored to both connections equally. However, all tag history queries (tag history bindings, system.tag.queryTagHistory() calls, reporting tag historian queries, etc.) execute against the first connection, unless a limit is imposed using the settings below, or the first connection is unavailable.
Second Connection	The second connection to store Tag history.
Querying	
Limit First Connection Query	If enabled, only queries that are within the time frame specified below will be executed against the first connection. Queries that go further back will execute against the second connection.
Limit Length and Units	The unit and length of the time frame limitation mentioned above.

Set Up a Tag History Splitter

To create the additional Tag History provider, do the following steps:

1. Go to the **Config** section of the Gateway Webpage, and choose **Tags > History**.
2. Click **Create new Historical Tag Provider**.
3. Select **Tag History Splitter** and click **Next**. The New Historical Tag Provider page is displayed.

Config > Tags > History

- Internal Historian**
A built-in local historian with limited storage.
- OPC-HDA Provider**
A historical tag provider that uses OPC-HDA to retrieve values from a 3rd party historian.
- Remote History Provider**
Sends tag history through the Gateway Network for storage in a remote history provider.
- Tag History Splitter**
Stores tag history concurrently to two other connections in the gateway.

Next >

4. Enter a name for the **Provider Name**. From the dropdown choose a database for the **First Connection** (for the primary data) and one for the **Second Connection** (for secondary data).

Config > Tags > History

Main	
Provider Name	New History Provider
Enabled	<input checked="" type="checkbox"/> Enable this tag history provider (default: true)
Description	

Storage	
First Connection	MySQL <input type="button" value="▼"/> <ul style="list-style-type: none"> Choose One MySQL
Second Connection	Choose One <input type="button" value="▼"/>

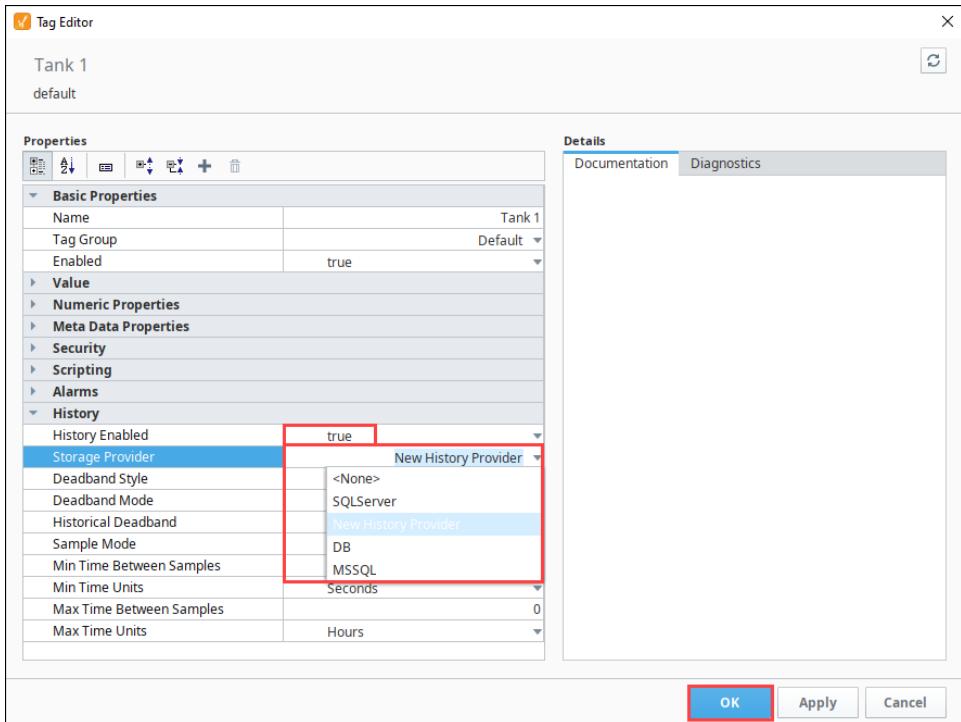
Querying	
Limit First Connection Query	<input type="checkbox"/> If enabled, only queries that are within the time frame specified below will be executed against the first connection. Queries that go further back will execute against the second connection. (default: false)
Limit Length	1 (default: 1)
Limit Units	Months <input type="button" value="▼"/> (default: MONTH)

Create New Historical Tag Provider

5. Click **Create New Historical Tag Provider**.

Now the Tag History Splitter provider is created, and you can use it to log the Tag History data in the Designer. To test this, open your project in the Designer.

1. In the Tag Browser, selecting a Tag and right-click to select the **Edit Tag**  option.
2. On the Tag Editor window scroll down to the **History** section and set **History Enabled** to true.
3. In the dropdown list for Storage Provider, select the new provider.



4. Click **OK** to save the change.

Historian Simulator

This feature is new in Ignition version **8.1.0**
[Click here](#) to check out the other new features

The Historian Simulator is a provider that will generate simulated historical records, without the need for an external database or data source. The Simulator doesn't actually store any records. Rather, subsystems in Ignition can make requests to the simulator, and it will generate a result set based on the Tag path(s) provided.

The simulator is great for testing, as the data generated is reliably aligned to the specified period. Each generation of data for a given set of parameters and given time range will be repeatable. In addition, the "resolution" of the raw data, or the frequency of the generated datapoints, is configurable allowing for as dense or sparse raw values.

On this page ...

- [Configuring a Historian Simulator Provider](#)
- [Retrieving Simulated Data](#)
- [Simulator Path Syntax](#)
 - [Editing the Paths](#)
 - [Function Examples](#)



Historian Simulator

[Watch the Video](#)

Configuring a Historian Simulator Provider

1. On the Gateway Webpage, navigate to the Config section.
2. Under the **TAGS** heading in the sidebar, click on **History**.
3. On the History Provider listing, click **Create new Historical Tag Provider**.
4. From the listing of available types, select **Simulator**, and click **Next**.

A screenshot of the Ignition Config interface. The top navigation bar shows 'Config > Tags > History'. A green banner at the top indicates 'Trial Mode 0:18:39' and has a 'Activate Ignition' button. Below this, a list of history providers is shown in a card-based layout. The 'Simulator' option is selected and highlighted with a red border. Other options include 'DB Table Historian', 'Internal Historian', 'Remote History Provider', and 'Tag History Splitter'. At the bottom right of the card is a blue 'Next >' button.

<input type="radio"/> DB Table Historian	Adds tag historian style query support to standard wide database tables.
<input type="radio"/> Internal Historian	A built-in local historian with limited storage.
<input type="radio"/> Remote History Provider	Sends tag history through the Gateway Network for storage in a remote history provider.
<input checked="" type="radio"/> Simulator	Provides simulation data for testing without an external data source.
<input type="radio"/> Tag History Splitter	Stores tag history concurrently to two other connections in the gateway.

Next >

5. The simulator requires a unique name from other historical providers on the same Gateway, but otherwise does not require any other configuration settings. The bulk of configuration occurs when attempting to request records from the simulator. For now just provide a name, and click the **Create New Historical Tag Provider** button to finish configuring the provider.

The screenshot shows the Ignition Config interface with the following details:

- Header:** Shows "Config > Tags > History" and "Trial Mode 0:49:52" on the left, and "Activate Ignition" on the right.
- Main Section:** A large grey header labeled "Main".
- Provider Name:** A row with "Provider Name" and a text input field containing "Simulated Provider".
- Enabled:** A row with "Enabled" and a checked checkbox labeled "Enable this tag history provider (default: true)".
- Description:** A row with "Description" and a large empty text area.
- Buttons:** A blue button at the bottom labeled "Create New Historical Tag Provider".

6. You'll be redirected back to the History Provider listing. You're now ready to use the simulator.

Retrieving Simulated Data

Once a Simulator Provider has been configured, you need a Tag Historian Query to start retrieving records from it. Lots of different interfaces can request Tag Historian Queries, but for the sake of simplicity we'll only focus on [Tag history bindings in Vision](#). Browsing for available historical Tags using a Vision Tag History binding reveals the following Tags:

Property Binding: Root Container.Power Table

Tag History
Queries the tag history system for time-series tag history data

Available Historical Tags

- Simulated Provider
 - cos_1m_100_1000ms
 - function_period_amplitude_resolution
 - ramp_60s_100_1s
 - realistic_1d
 - sine_10s_20_500ms
 - square_1h_10_10s

Use fully-qualified paths

Date Range Most Recent

Realtime 1

Aggregation Mode Return Format Sample Size

No Binding

To get started quickly, you can use the default Tags in the list above. The one exception would be the "function_period_amplitude_resolution" Tag, as that's simply there for syntax reference. The rest of the Tags utilizes the simulator's pathing syntax.

Simulator Path Syntax

The built-in "Tags" provided by the simulator are a great example of the simulator can do, but it is worth knowing that the data returned by the simulator can be customized. Doing so involves modifying the Tag path in the Tag history query.

As a refresher, history providers require a path that contains multiple components. Together, the components identify where the data came from. For example, a historical path could have a `histprov` component that represents the history provider associated with the data, and a `Tag` component that represents the Tag path of the Tag. A hypothetical historical path would look like the following:

```
histprov:my_provider:/tag:my_tag_path
```

With that understanding, retrieving records from the simulator involves providing the name of the simulator configuration on the gateway (the `histprov`) and a Tag path. Again, the simulator doesn't store any records. Rather, it looks at the Tag path provided, parses it, and returns a result set based off of contents in the path.

The simulator expects Tag paths to use the following notation, with each "parameter" delimited by an underscore:

```
function_periodTime_amplitude_resolutionTime
```

Each parameter is described below:

parameter	description
function	The function to use when generating the raw data. The following functions are available:

	<ul style="list-style-type: none"> Ramp - Start at zero, and increment up to Amplitude over a the periodTime. Once Amplitude has been reached, start over at zero and repeat. Sine - Generates a Sine wave over the period of time, with Amplitude representing the peak deviations from zero. Cos - Generate a Cosine wave over a period of time, with Amplitude representing the peak deviations from zero. Square - Half the time is spent at zero, the other half is spent at amplitude, across periodTime. Realistic - Generates 400 random but somewhat realistic data points that repeat each period. Note that this function ignores both the amplitude and resolutionTime parameters.
periodTime	How often the data sequence starts over. Values for periodTime need a time unit.
amplitude	Used as a peak value by most of the functions.
resolutionTime	How often raw data points are generated. Like periodTime, this parameter requires a time unit.

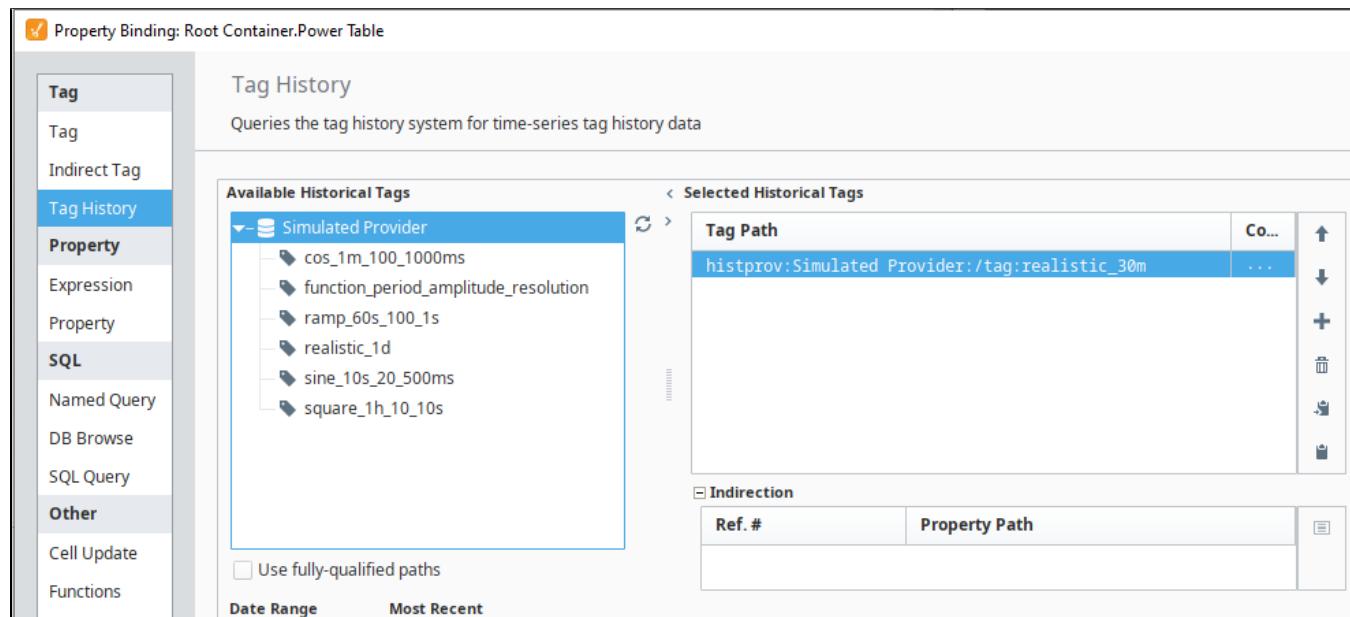
For those functions utilizing a time unit represented by certain notation. A description of each notation can be found below:

Notation	Time Unit
ms	Milliseconds
s	Seconds
m	Minutes
h	Hours
d	Days

Editing the Paths

With the knowledge of how the paths work, you can easily use the existing Tag history interfaces to request datasets with custom paths, usually by simply typing in the desired parameters.

For example, the image below shows Vision's Tag History Binding interface, but we modified the Selected Historical Tag path to use the realistic function over a period of 30 minutes, instead of the default one day.



Function Examples

For example, the Tag in the path below utilizes a sine function, over a period of 10 seconds, with an amplitude of 20, and a raw point will be generated every 500 milliseconds.

```
tag:sine_10s_20_500ms
```

The following would generate a square wave, with an amplitude of 10, repeating every day, while generating a raw value for every 5 minutes.

```
tag:square_1d_10_5m
```

The realistic function is unique in that it only utilizes the periodTime parameter, therefore:

```
tag:realistic_30m
```

DB Table Historian Provider

This feature is new in Ignition version **8.1.0**
[Click here](#) to check out the other new features

The DB Table Historian Provider acts as a bridge between tables in a database connection and the Tag Historian Module, mapping columns in the tables as "available Tags", thus allowing Tag History Queries to access the content of the table. Usually Tag History Queries can interact only with tables created by the Tag Historian. Thus, a table created with a Transaction Group, or by some other means (e.g., manually creating the database tables, tables generated by third party systems, etc) couldn't be accessed via things like a Tag History Binding. The DB Table Historian Provider solves this problem.

Utilizing this provider requires that a **Datasource History Provider**. Meaning, the type of historical provider that is automatically created whenever a Database connection is configured. You can check the type of any History Provider under the **Config** section of the Gateway, on the **Tags > History** page.

Provider Name	Enabled	Type	Description	Status	
DB_Table_Historian	true	DB Table Historian		Running	delete edit
Internal_Historian	true	Internal Historian		Running	delete edit
Maria_DB	true	Datasource History Provider		Running	edit
Simulated_Provider	true	Simulator		Running	delete edit
internal_DB	true	Datasource History Provider		Running	edit

[→ Create new Historical Tag Provider...](#)

Note: For details about a provider's status, see the [Tag Providers Status](#) page.

On this page ...

- Configuring a DB Table Historian Provider
- Retrieving Records
- Path Components
 - Example



DB Table Historian

[Watch the Video](#)

Configuring a DB Table Historian Provider

1. On the Gateway Webpage, navigate to the Config section.
2. Under the TAGS heading in the sidebar, click on **History**.
3. On the History Provider's listing, click **Create new Historical Tag Provider**.
4. From the listing of available types, select **DB Table Historian**, and click **Next**.

The screenshot shows the Ignition configuration interface. The left sidebar is titled 'SYSTEM' and includes options like Home, Status, Config (which is selected), Overview, Backup/Restore, Ignition Exchange, Licensing, Modules, Projects, Redundancy, Gateway Settings, NETWORKING (Web Server, Gateway Network, Email Settings), SECURITY (Auditing, Users, Roles, Service Security, Identity Providers, Security Levels, Security Zones), and a search bar. The main content area is titled 'Config > Tags > History' and shows 'Trial Mode 1:37:27' and an 'Activate Ignition' button. A red box highlights the first option in a list of providers:

- DB Table Historian**
Adds tag historian style query support to standard wide database tables.
- Internal Historian**
A built-in local historian with limited storage.
- Remote History Provider**
Sends tag history through the Gateway Network for storage in a remote history provider.
- Simulator**
Provides simulation data for testing without an external data source.
- Tag History Splitter**
Stores tag history concurrently to two other connections in the gateway.

A blue 'Next >' button is at the bottom right of the provider list.

- Enter a unique name for the provider in the Provider Name field.
- In the Data source field, select the backing data source. You would select which ever Datasource History Provider contains the tables you want exposed to the historian system. Once ready, click the **Create New Historical Tag Provider** button, which completes the configuration process.

The screenshot shows a configuration dialog for creating a new historical tag provider. The 'Main' tab is active, displaying the following fields:

Provider Name	DB_Table_Historian
Enabled	<input checked="" type="checkbox"/> Enable this tag history provider (default: true)
Description	

The 'Other' tab is also visible, showing a 'Data source' dropdown set to 'Maria_DB'.

A large blue 'Create New Historical Tag Provider' button is located at the bottom of the dialog.

Retrieving Records

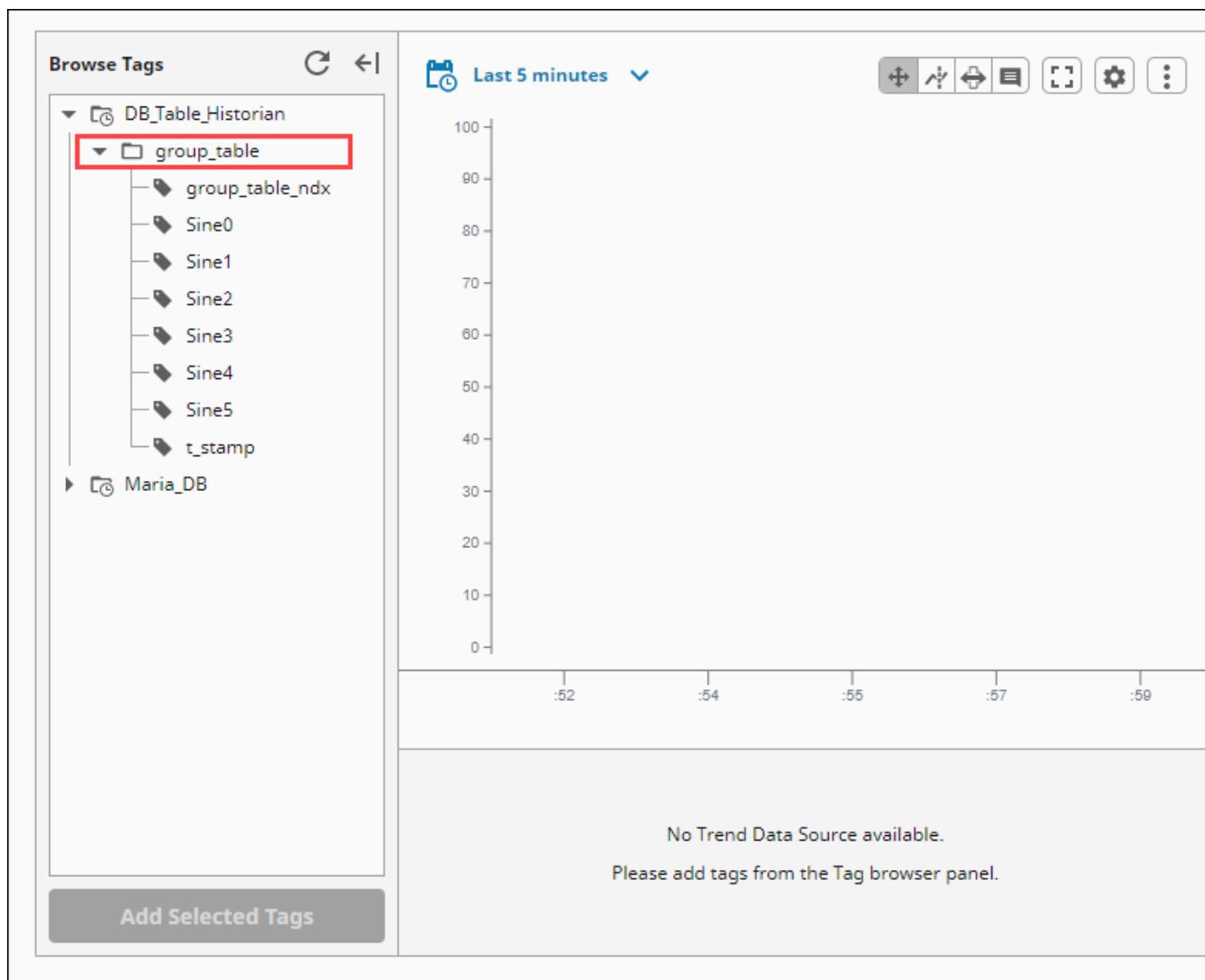
Once configured, the DB Table Historian will expose any tables found in the associated data source (specifically, the **Data Source** listed on the configuration page of the DB Table Historian). From this point on, any tables found in the data source will be available for browsing via the various Tag History interfaces found throughout Ignition.

For example, we could store some records in a database connection with a Transaction Group, which would create a table like the following:

The screenshot shows a 'Schema' browser window with a 'History' tab selected. Under the 'group_table' entry, the following columns are listed:

- group_table_ndx (INTEGER)
- Sine0 (REAL)
- Sine1 (REAL)
- Sine2 (REAL)
- Sine3 (REAL)
- Sine4 (INTEGER)
- Sine5 (INTEGER)
- t_stamp (TEXT)

Then, we could use the Perspective [Power Chart](#) built-in Tag browser panel to detect the table, which exposes the columns as "tags":



Path Components

The DB Table Historian Provider attempts to map each column in a database table to a Tag. When querying the results, the historical Tag path used is composed of multiple components, which each represent some identification of the data source. The DB Table Historian is molded after the following:

```
histprov:[historyProvider]:/table:[tableName]:/column:[columnName]:/timestamp:[timestampColumnName]:  
/keycolumn:[keyColumnName]:/keyvalue:[keyColumnNameValue]
```

Each of the components are described below.

Component	Description
histprov	The name of the history provider that should be queried.
table	The name of the database table in the history provider.
column	The name of the column on the table, in the history provider.
timestamp	A column on the table that will be used as the source of the timestamp for the query. By default, the query will look for a column named t_stamp to use for the timestamp component. It's highly recommended to include this component if the table doesn't contain a timestamp column named "t_stamp", otherwise the query will fail.
keycolumn	An optional component that allows you to specify a single column on the table to use for simple filtering. Used in conjunction with keyvalue. Value must be an integer. There can only ever be a single keycolumn for any given path. More complex filtering can be accomplished by instead using a Named Query .
keyvalue	An optional component that works with keycolumn, allowing the query to only return rows if the keycolumn contains the value specified on this component.

As far as historical pathing goes, Tags created by this provider may look something like below, with dedicated **table** and **column** components in the path.

```
histprov:DB_Table_Historian:/table:group_table:/column:Sine1:/timestamp:my_timestamp_column
```

Example

Say we have a database table like the following.

Database Query Browser

```
SELECT machine_id, process_value, time FROM machine_values
```

Limit SELECT to: 1000 rows

Resultset 1

machine_id	process_val...	time
1	111	2020-09-10 21:44:35
1	100	2020-09-10 21:44:41
2	22	2020-09-10 21:45:01
2	222	2020-09-10 21:45:15

4 rows fetched in 0.026s

We could use the DB Table Historian provider to expose our machine_values table to a [Vision Tag History Binding](#).

Property Binding: Root Container.Power Table

Tag History
Queries the tag history system for time-series tag history data

Available Historical Tags

- DB Table Historian
 - group_table
 - machine_values
 - machine_id
 - process_value
 - time
 - internalDB

Selected Historical Tags

Tag Path	Column Name
histprov:DB Table Historian:/tbl	process_value

Use fully-qualified paths

Date Range **Most Recent**

Realtime 10 min

Indirection

Ref. #	Property Path

OK **Cancel**

The resulting historical Tag path for process_value (after dragging it over to the **Selected Historical Tags** table) would look like the following:

```
histprov:DB Table Historian:/table:machine_values:/column:process_value
```

We would need to explicitly state that the "time" column should be used by the binding, so we could double click on the cell under the "Tag Path" header, and change the Tag Path to the following, which would allow the query to feature the records accurately.

```
histprov:DB Table Historian:/table:machine_values:/column:process_value:/timestamp:time
```

We could further modify this Tag path, so that only entries with a "machine_id" of 1 are returned, but changing the path to the following:

```
histprov:DB Table_Historian:/table:machine_values:/column:process_value:/timestamp:time:/keycolumn:  
machine_id:/keyvalue:1
```

SQL Bridge (Transaction Groups)

Overview

The SQL Bridge Module enables the creation of Transaction Groups that synchronize data between PLCs and databases. You can use Transaction Groups to easily log from PLCs to the database, move data from the database back to PLCs, and even keep the two synchronized. Drag and drop functionality makes setup of Transaction Groups quick and easy.

Originally conceived as an easy data storage method, Transaction Groups have become a core feature of Ignition. In their simplest form, they regularly read values from OPC addresses and store them into a SQL database. While data collection is still their primary use, they have grown in functionality over time.

To set up and use Transaction Groups, SQL knowledge is not required. Ignition can automatically create and manage the database table for each group. Prior experience writing SQL queries or creating database tables are not required to log data.

On this page ...

- Overview
- Types of Transaction Groups
 - Historical Data Logging
 - Database and OPC Synchronization
 - Large Data Block Storage
 - Stored Procedures
- Centralizing Data Collection

The screenshot shows the Ignition configuration interface for setting up a Transaction Group. The main window displays three tables: 'Basic OPC/Group Items (7)', 'Run-Always Expression Items (ignore trigger) (0)', and 'Triggered Expression Items (0)'. The 'Basic OPC/Group Items' table lists seven items (B3:0 to B3:3) with target names B3_0 to B3_3 and Int2 data type. The 'Run-Always Expression Items' and 'Triggered Expression Items' tables are currently empty. On the right side, the 'Action' tab is selected, showing execution scheduling (Timer at 1 second), a data source dropdown set to '<Default>', and a table name input field containing 'group_table'. Advanced options include checkboxes for 'Automatically create table', 'Use custom index column' (with value 'group_table_ndx'), 'Store timestamp to' (with value 't_stamp'), 'Store quality code to' (with value 'quality_code'), and 'Delete records older than' (with value '1 day(s)').

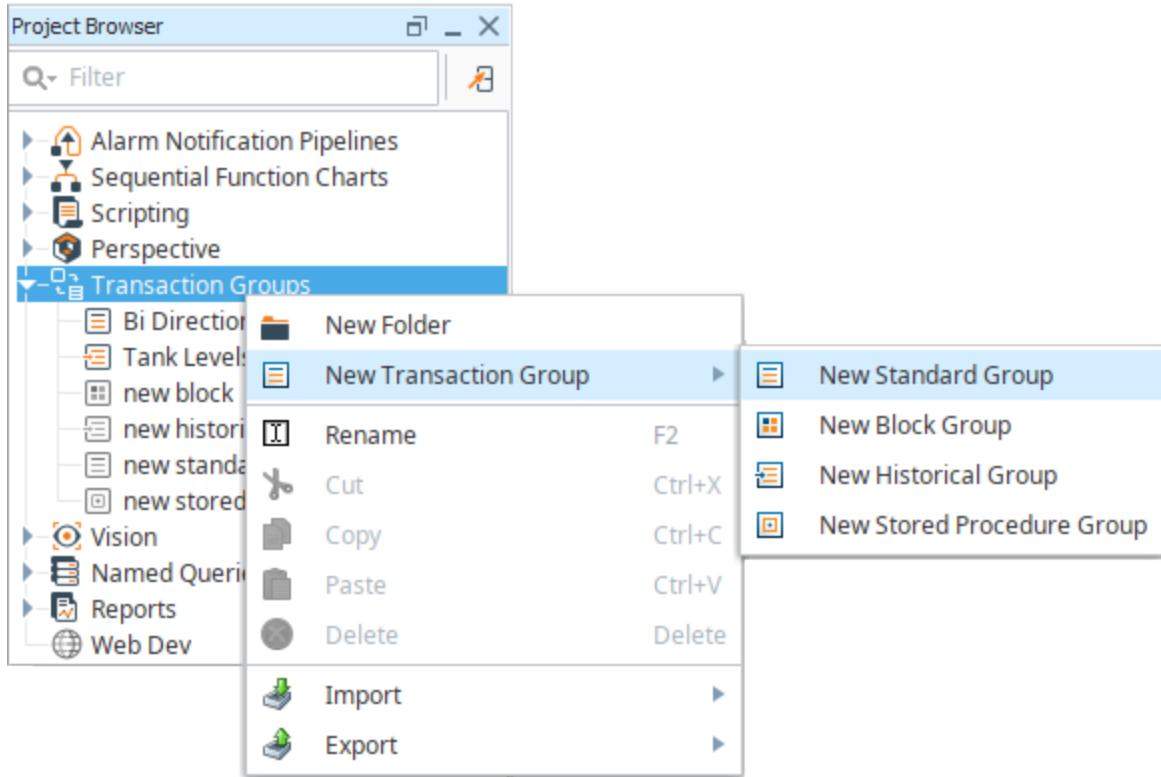
Types of Transaction Groups

There are four types of Transaction Groups, and they each handle data a little differently:

- **Historical Groups** - Quick and easy group that collects historical records
- **Standard Groups** - An improved version of the historical group that can reverse the flow of data, writing database values directly to Tags.
- **Block Group** - Records "blocks" of data, allowing you to record multiple values per execution in a tall format.
- **Stored Procedure Group** - Invokes a stored procedure in the database, returning the results of any OUT or INOUT parameters to Tags.

Learn more about each type of group on the [Understanding Transaction Groups](#) page.

All Transaction Groups can execute at a set rate or on a schedule. A [trigger](#) can be used to determine when the group should record. You can use Ignition's [expression language](#) in the trigger to allow complex logic to determine when logging occurs, making precision execution easy.



Historical Data Logging

Historical Groups quickly and easily store data from the plant floor into any kind of SQL database. Items from any or all devices can be included in the same group, just drag a few Tags over and start the group running. Ignition will log the data until you tell it to stop.

The screenshot displays the Ignition configuration interface for historical data logging. On the left, a table titled 'Basic OPC/Group Items (6)' lists six items, each mapping a source tag to a target tag in a MySQL database:

Item Name	Source Va...	Latched ...	Target Name	Data Type	Properti...
Sim_Generic/Ramp/Ramp0	344.427	344.427	Ramp0	Float8	
Sim_Generic/Ramp/Ramp1	58.320	58.320	Ramp1	Float8	
Sim_Generic/Ramp/Ramp2	119.427	119.427	Ramp2	Float8	
Sim_Generic/Ramp/Ramp3	247.480	247.480	Ramp3	Float8	
Sim_Generic/Ramp/Ramp5	465.267	465.267	Ramp5	Float8	
Sim_Generic/Ramp/Ramp4	158.320	158.320	Ramp4	Float8	

The 'Action' tab on the right shows the following configuration:

- Execution Scheduling: Timer (selected), 1 second(s)
- Data source: MySQL
- Table name: ramp_historical
- Automatically create table: checked

Below this, the 'Database Query Browser' window is open, showing a query results table and a history pane:

Resultset 1							
ramp_historical...	Ramp0	Ramp1	Ramp2	Ramp3	Ramp5	Ramp4	t_stamp
1	794.84	96.13	169.84	304.195	496.775	396.13	2019-06-18 10:2
2	808.187	6.14	-16.813	19.21	5.117	6.14	2019-06-18 10:2
3	821.52	16.14	-3.48	34.21	13.45	16.14	2019-06-18 10:2
4	834.867	26.15	9.867	49.225	21.792	26.15	2019-06-18 10:2
5	848.2	36.15	23.2	64.225	30.125	36.15	2019-06-18 10:2
6	861.547	46.16	36.547	79.24	38.467	46.16	2019-06-18 10:2
7	874.88	56.16	49.88	94.24	46.8	56.16	2019-06-18 10:2
8	888.227	66.17	63.227	109.255	55.142	66.17	2019-06-18 10:2
9	901.56	76.17	76.56	124.255	63.475	76.17	2019-06-18 10:2
10	914.92	86.19	89.92	139.285	71.825	86.19	2019-06-18 10:2

The history pane shows previous queries:

- SELECT * FROM Tank_Histo
- SELECT * FROM ramp_hist

At the bottom, a message indicates "268 rows fetched in 0.013s".

Database and OPC Synchronization

Standard Groups are the most flexible group. They are capable of not only storing OPC values in the database, but can also write database values to OPC addresses or synchronize data changes between both the database and PLC. With this group you can create true realtime value tables in the database, and allow anything that can talk to the database to push values to a PLC. This is often used to create [Recipe](#) systems where the recipe values are stored in the database, and a user can select a recipe to write all your settings directly to Tags. Changing recipes is as easy as changing a Tag value or selecting a name.

The screenshot shows the 'Line 1 Recipe' configuration window. At the top, there are three status buttons: 'Enabled' (green play icon), 'Disabled' (red stop icon), and 'Pause' (blue pause icon). Below the buttons, the status is listed as 'Running'. The main area is titled 'Basic OPC/Group Items (4)' and contains a table with the following data:

Item Name	Source Va...	Latched V...	Mode	Target Name	Data Type	Properties
CaseCount	96.716	95.882	—	CaseCount	Float8	
CurrentOrder	0	0	↙	CurrentOrder	Int2	
CurrentRun	0	0	↙	CurrentRun	Int2	
RunControl	0	0	↙	RunControl	Int4	

Large Data Block Storage

Transfer large amounts of data very efficiently with the [Block Group](#). This groups allows you to send whole arrays of data to and from the database. It works just like the Standard group, but on a much larger scale.

The screenshot shows the 'T4 ACC' configuration window. At the top, there are three status buttons: 'Enabled' (green play icon), 'Disabled' (red stop icon), and 'Pause' (blue pause icon). Below the buttons, the status is listed as 'Execution Disabled'. The main area is titled 'Block Items (0)' and contains a table with the following data:

Item Name	Source ...	Latche...	Mode	Target Name	Data Type	Propre	Size
Item_T4_0	ns=1;s=[SLC]_Meta:T4/T4:0/T4:0.ACC	0	—	T4_0_ACC	String		2
	ns=1;s=[SLC]_Meta:T4/T4:0/T4:0.DN	false	—				
Item_T4_1			—	T4_3_ACC	String		2
Item_T4_2			—	T4_2_ACC	String		2
Item_T4_3			—	T4_1_ACC	String		2

Stored Procedures

The Stored Procedure Group allows you to use PLC data as inputs and outputs for your existing [Stored Procedures](#). With the Stored Procedure Group, your IT department can have control over how data is entered and returned from the database.

SP All Params

Errored

Enabled Disabled Pause

Basic OPC/Group Items (2)

Item Name	Source V...	Latched ...	Target Name	Output	Data Type	Properties
In_Tag	50	50	myInParam	None	Int4	
Out_Tag	20	20	myCounts	None	Int4	

Run-Always Expression Items (ignore trigger) (0)

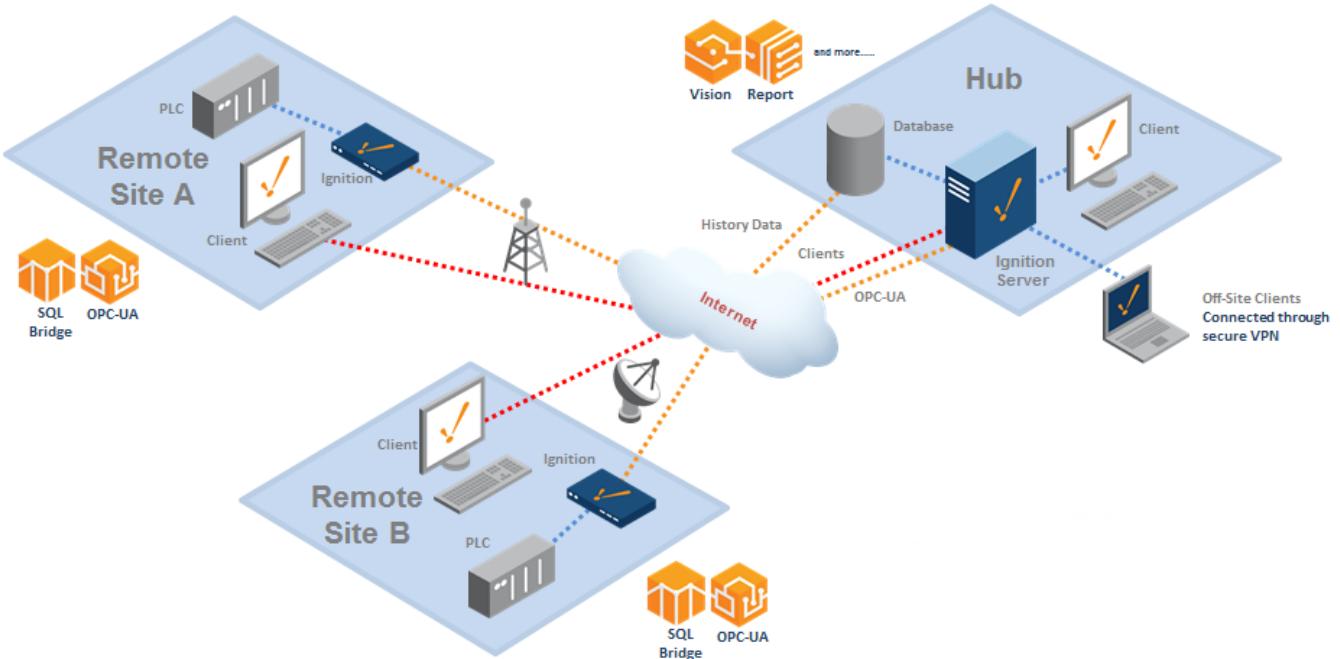
Item Name	Source Value	Latched Val...	Target Name	Data Type	Properties

Centralizing Data Collection

In Distributed systems, PLCs can be spread out over great distances to remote sites. Collecting and centralizing data from each can be difficult and time consuming. To combat this problem, Transaction Groups are used as the cornerstone of our **Hub and Spoke** architecture. Historical Groups can be applied locally to each PLC for a minimal cost, and forward all data into a single, central, database.

Remote Site with Central Hub

Remote Logging



In This Section ...

Understanding Transaction Groups

Transaction Groups are the heart of the SQL Bridge module. They are units of execution that perform actions such as storing data historically, synchronizing database values to OPC, or loading recipe values. A variety of group types, items types, and options means that Transaction Groups can be configured to accomplish almost any task.

Transaction Groups are configured in the Ignition Designer. Each Transaction Group is associated with a table in a database Ignition is connected to, and is made up of one or more Items. The group will then execute at a specific interval of time, or on a user-defined schedule. Generally each execution will create a new row in the database table with a separate column for each Item in the group. However, it is possible for some types of Transaction Groups to take values from the database and write to specific Tags.

Additionally, the Transaction Group can be configured to conditionally synchronize values by using a trigger. The trigger is evaluated every execution, and if the trigger condition is met, then synchronization will occur. If the trigger condition has not been met, then the group will wait until the next execution to re-evaluate the trigger.

There are four [types of Transaction Groups](#): Standard, Block, Historical, and Stored Procedure. Each offers different functionality. For example, the Historical Group allows you to quickly configure a group that reads OPC data and push it to the database. While the additional flexibility of a Standard Group allows you take values from the database and write them to a PLC.

Transaction Groups enable you to perform tasks such as database to OPC synchronization, recipe management, and historical data logging.

The Transaction Group Welcome tab allows you to create any one of the four types of transaction groups. Each one of the transaction groups is basically a template to help you get started creating your transaction group. Once you select a Transaction Group, enter a name, and press 'create', and the specific transaction group template will open. All you have to do next is enter the tags for the values you want to collect. The Transaction Group Welcome tab will show you any recently modified transaction groups along with the date it was modified and who modified it. You can even double click on a recently modified chart and open it.

The Transaction Group Welcome tab provides a quick way to create new transaction groups and update existing ones.

On this page ...

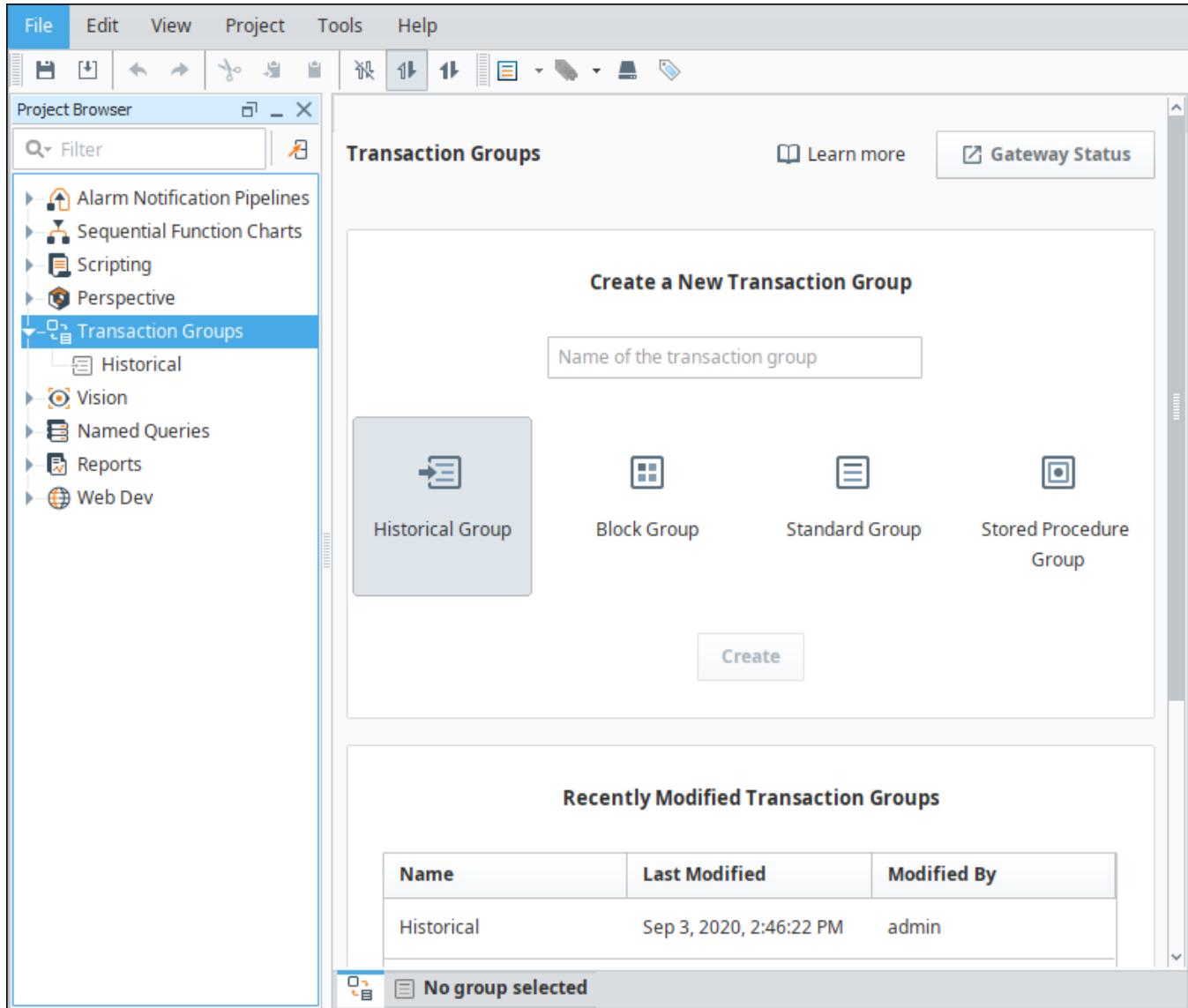
- [Transaction Group Workspace](#)
 - [Items](#)
 - [Enabling Group Execution](#)
 - [Editing Group Settings](#)
- [Action Settings](#)
- [Group Update Rate](#)
 - [Timer](#)
 - [Schedule](#)
 - [Execution Cycle](#)
- [Trigger and Handshake Settings](#)
- [Advanced Settings](#)
- [Creating a Transaction Group](#)



INDUCTIVE
UNIVERSITY

Transaction Group Introduction

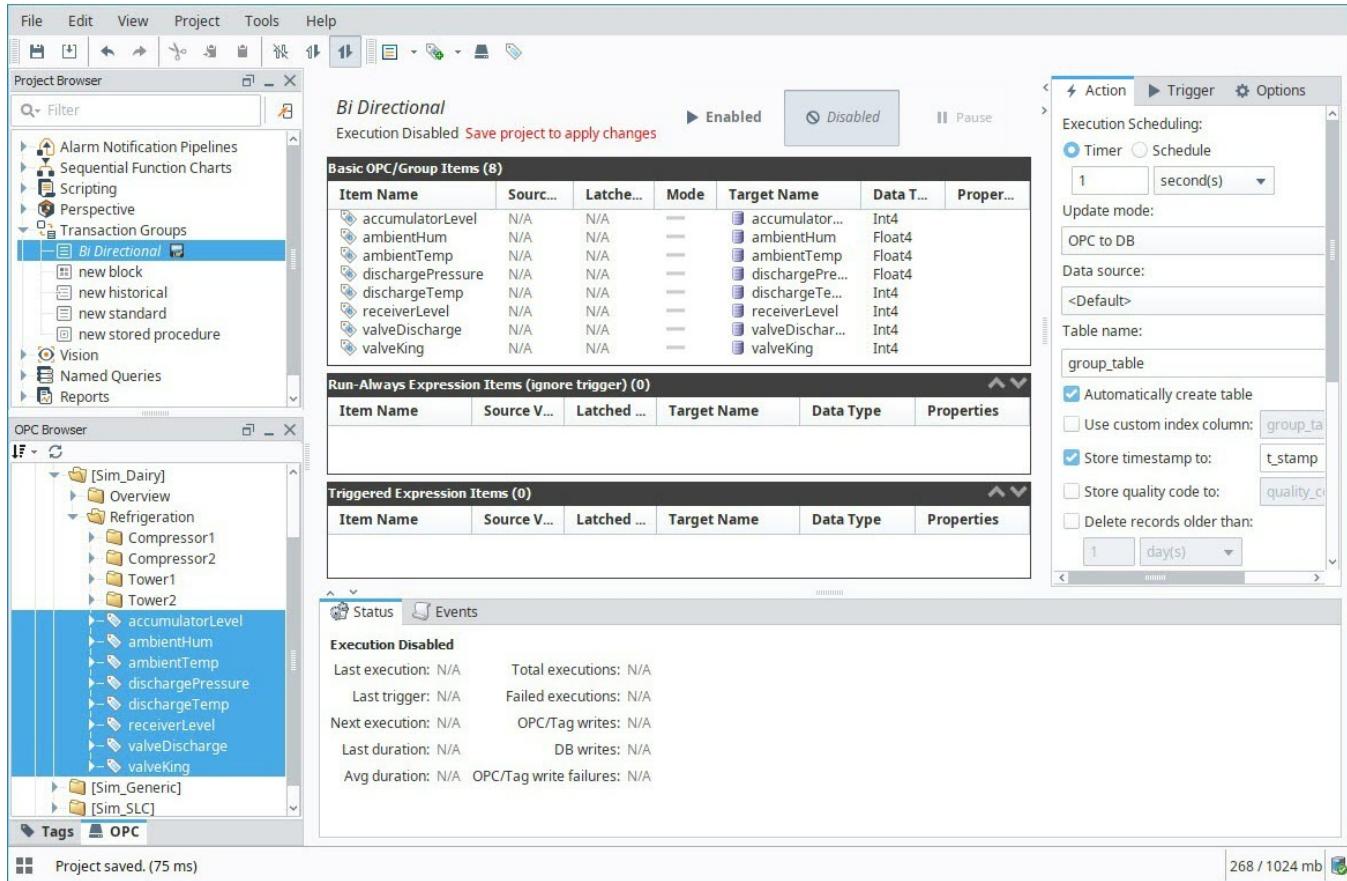
[Watch the Video](#)



Transaction Group Workspace

Transaction Groups are edited through the Ignition Designer. When a group is selected, you are presented with the transaction group workspace. The workspace is broken into several parts:

- **Title bar** - Shows the name of the currently selected group, as well as options to set it as Enabled or Disabled, and to Pause, if it's currently executing.
- **Item configuration** - Shows all of the items configured in the selected group. Many settings can be modified directly through the display, the rest by double-clicking the item or selecting **Edit** in the context menu.
- **Action / Trigger / Options tabs** - Defines how and when a group executes. Holds most of the options that apply to the group in general, such as the update rate, and which data connection it uses.
- **Status / Events tabs** - Provides information about the executing group, including the most recent messages that have been generated.



Items

Each Item (Tag) in the Transaction Group consists of several properties, but the key properties are the Source/Latched Value and Target Name.

Source and Latched Value

The Source Value will be the value of the items source. This can be something like a Tag or a direct OPC Item if writing to the database, but can also be the value pulled from the database if in DB to OPC mode. This value can change in between executions, depending on the source type. When the source is a Tag, it will update as the Tag updates, depending on how the Tag Group for the Tag is set. However, if the source is an OPC Item, it will update only when the group executes, unless the OPC subscription rate is overridden in the group.

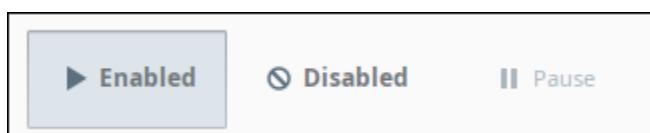
The Latched value will be the value that was written at execution. This can be the value that gets written to the database on execution in OPC to DB mode, or it can be the value that gets written to the Tag in DB to OPC mode. The value will only change on execution of the group.

Target Name

In most cases, the Target Name is a column on the database table the Transaction Group is associated with. However, it is possible to have the Target Name 'Read-only'. When set to 'Read-only' the value of the item will not be tied to any columns in the database, but is still visible from the Transaction Group and can be used as a trigger.

Enabling Group Execution

In order for groups to be evaluated, they must first be enabled. This is done by selecting **Enabled** in the group title bar, and then saving the project. The group executing can be stopped by reversing the procedure and selecting **Disabled** before saving. If you want to quickly and temporarily stop the group's evaluation, toggle the **Pause** button. This will prevent execution until the group is enabled again, or until the system is restarted.



Note: Transaction Groups exist in a project, but they execute in the global Gateway space. This means that once your groups are enabled, they will run even without a client open.

Editing Group Settings

Group settings may be modified at any time, regardless of whether or not the group is executing. Modifications will be applied when the project is saved, and the group will be started or stopped as required. Some changes such as modifying items may cause features like live values to appear to be incorrect. It is therefore important to notice the modified icon that appears next to the group, and to save often. If you would prefer to stop the group before making edits you can simply pause the group. Execution will begin again after the project is saved.

Action Settings

The action settings of a Transaction group define how often the group will be evaluated, as well as important settings that apply to the group as a whole. They are found on the **Action** tab, the first of the tabs on the right side of the Transaction Group workspace.

The Action settings vary for the different types of Transaction Groups, but a few settings are common to most of them:

Setting	Description
Execution scheduling	<p>How often the group is evaluated. For a number of reasons, the group may not execute during the evaluation. The most common reason is the trigger, but see Execution Cycle below for more possible reasons why evaluation will exit.</p> <ul style="list-style-type: none"> Timer - specifies the OPC Tag subscription rate for the OPC Tags. It can run at millisecond, second, minute, hour, or day rates. Schedule - is a specified start time on the update Rate. Set a list of time (or time ranges) that the group should run at. If the pattern specified includes a time range, at rate must be provided, and the group will execute as in timer mode during that period.
Update mode	<p>For groups that support it, sets the default for how items are compared to their targets. Options are:</p> <ul style="list-style-type: none"> OPC to DB - Only read from the OPC server and write to the database. DB to OPC - Only read from the database and write to the OPC Server. Bi-directional OPC wins - Read and Write to both the database and OPC Server. On group start, write OPC values to the database. Bi-directional DB wins - Read and Write to both the database and OPC Server. On group start, write database values to OPC items.
Data source	The database connection name the group should use. Can be Default , which will use the default connection for the project.
Table name	<p>Name of the table in the database that the group should interact with (reading or writing, depending on the Update mode and individual item Mode settings). The tables listed in this dropdown are determined by the Data source property.</p> <p>This setting allows you to type arbitrary names into it. If you type the name of a database table that doesn't exist, and the Automatically create table setting is enabled, then the group will attempt to create the database table on start.</p>
Automatically create table	If enabled, the transaction group will attempt to create a database table once the group starts running, assuming one doesn't already exist as determined by the Table name setting. If the table already exists, then nothing happens.
Use custom index	If left disabled, the group will attempt to add an index column to the database table when the group starts executing. If enabled, the group will use the column selected in the adjacent dropdown, or create a new column if you type in a column name that doesn't exist on the table (requires the Automatically create table setting to be enabled).
Store timestamp	If enabled, will attempt to store a timestamp value to the column specified in the adjacent dropdown. If you type in a column name that doesn't exist on the table, the group will attempt to create the column on start, assuming the Automatically create table setting.
Store quality code	Stores an aggregate quality for the group along with the regular data. The aggregate quality is a bit-wise AND of the qualities of the items in the group.
Delete records older than	If enabled, and the group is running, this setting will make the group delete older rows in the table. Options are minute(s), day(s), month(s), and year(s)

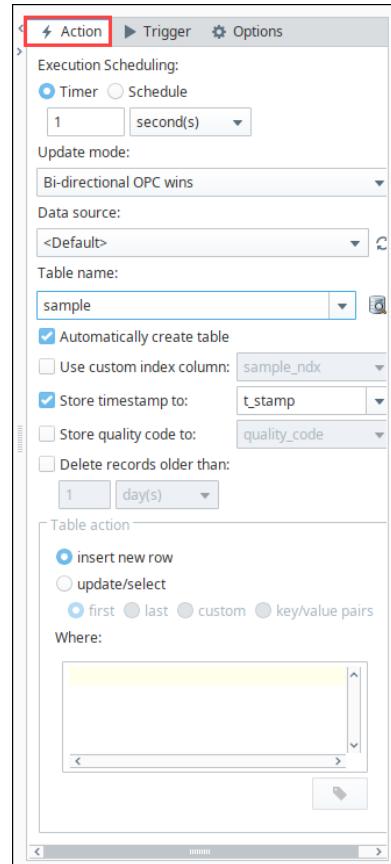
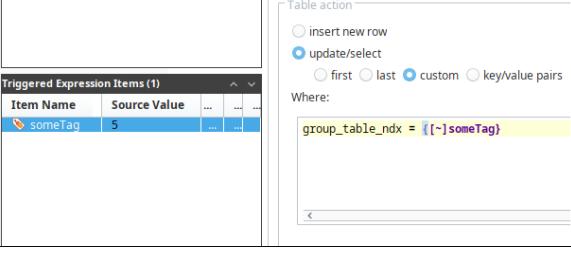
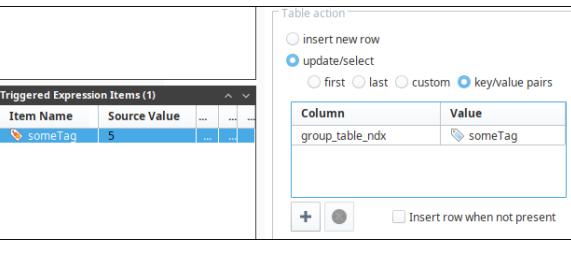


Table Action	<p>Defines which row will be targeted by the group.</p> <ul style="list-style-type: none"> • insert new row • update/select - allows you to target specific rows in the database table. Options are: <ul style="list-style-type: none"> • first row - the group always executes against the first row. • last row - the group always executes against the last row • custom - allows you to write a custom where clause to determine which row should be targeted. Uses the Where text area. The custom clause can use references to values of items in the group.  <p>key/value pairs - Provides dropdowns for both a column and a item in the group, allowing the group to target a single row in a table based on the item's value. In the image below, a value of 5 will be used in conjunction with the "group_table_ndx" column in the database table. Additional conditions can be added or removed with the Add or Delete buttons, below the table.</p> <p>Meaning, when the group executes, it will target the row where group_table_ndx has a value of 5.</p> 
--------------	--

Group Update Rate

Groups generally work on a timer. They are set to run at a certain rate. As they are running at that certain rate, they then check the rest of the settings. If the trigger conditions pass, the group is executed fully.

The Execution Schedule controls the rate at which the transaction group executes. On the Action tab of a group you selected, under Execution Scheduling, there are two options: **Timer** and **Schedule**. Timer, executes the group at a certain rate. Schedule, executes the group at specific times. When the Schedule option spans across a period of time, you must specify the rate at which the group executes during that time.

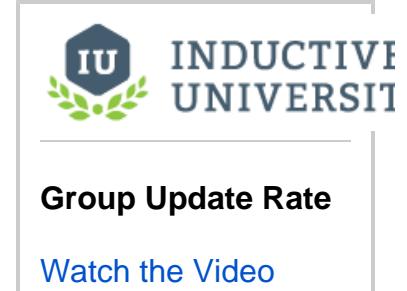
Timer

The Timer acts as the heartbeat of the transaction group and is evaluated at the provided rate. It can run at millisecond, second, minute, hour, or day rates. The Timer specifies the OPC Tag subscription rate for the OPC Tags. When a Timer is running the transaction group it first analyzes the Tags inside the **Basic OPC/Group Items** section of the transaction group. Then it looks at the trigger configuration and evaluates for Tag changes. Then it evaluates the specific trigger conditions and decides to execute on a trigger. Depending on the trigger settings, full execution may not occur, but the trigger will at least be evaluated at this rate. If the triggered condition is true, the transaction group proceeds to the **Triggered Expression Items** section of the transaction group. Only after this flow is complete, will the transaction group interact with the database, and for example, insert the Tag values into the database.

Schedule

An important difference between the Timer and the Schedule options is that the schedule option will automatically align to the specified start time on the update rate. With Schedule mode, you are providing a list of time (or time ranges) that the group should run at. If the pattern specified includes a time range, a rate must be provided, and the group will execute as in timer mode during that period.

The schedule is specified as a comma separated list of times or time ranges. You may use the following formats:



[Watch the Video](#)

- 24-hour times. Ie. "8:00, 15:00, 21:00", for execution at 8am, 3pm, and 9pm.
- 12-hour with am/pm (if not specified, "12" is considered noon): "8am, 3pm, 9pm"
- Ranges, "8am-11am, 3pm-5pm"
- Ranges that span over midnight, such as "9pm - 8am"

When using ranges, the execution times will be aligned to the start time. For example, if you specify a schedule of "9am - 5pm" with a rate of "30 minutes", the group will execute at 9, 9:30, 10, etc., regardless of when it was started. This is a useful difference compared to the Timer mode, which runs based on when the group was started. For example, if you want a group that runs every hour, on the hour, you could specify a 1 hour rate with a range of "0-24."

Execution Cycle

All of the Transaction Groups follow a similar execution cycle. The core evaluation may differ, but the general cycle is the same.

1. Timer executes, group enters execution
2. Is the group paused? Break execution.
3. Is the Gateway part of a redundant pair? If so, is it active? If not active, break execution.
Groups only execute on the active node.
4. Evaluate run-always items: OPC items, Tag references, and Expression items set to ignore the trigger (or items placed in the run always section of the Configuration window).
5. Is trigger set/active? If there is a trigger defined, but it is not active, break execution.
6. Evaluate "triggered" Expression items.
7. If applicable, read values from the database.
8. Execute a comparison between items and their targets.
9. Execute any writes to other Tags or the database that results from execution.
10. Report alerts.
11. Acknowledge the trigger, if applicable.
12. Write handshake value, if applicable.

If an error occurs at any stage besides the last stage, execution will break and the failure handshake will be written if configured. The group will attempt execution again after the next update rate period.

Caution: If the group errors due to a bad database connection, it will need to be manually restarted once the database connection is brought back.

Trigger and Handshake Settings

The trigger settings determine when a group will actually execute. They are examined each time the group evaluates (according to the update rate of the group). If they pass, the group will run and perform its action against the database.

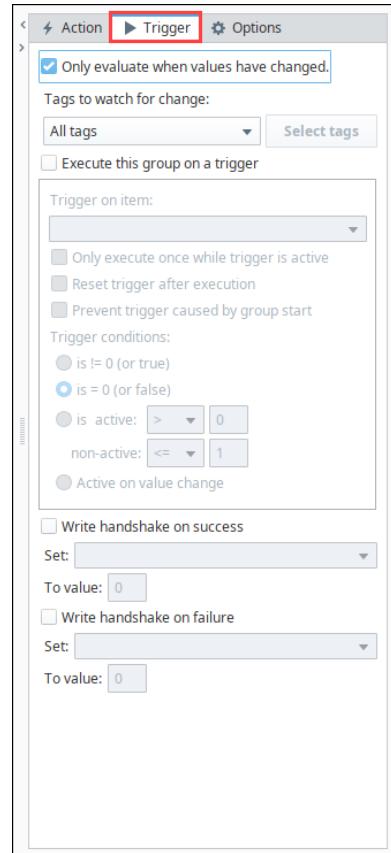
The trigger settings are the same for all group types and are found on the second tab (labeled **Trigger**), on the right side of the Transaction Group workspace.

The outcome of an execution is handled in the handshake section of the trigger section of the transaction group. When a group executes, it either completes successfully or an error prevents its execution.

The table below is a list of Trigger and Handshake settings.

Setting	Description
Only evaluate when values have changed	<p>The group will execute every time the value or values change. If the values have not changed, it will exit the evaluation. You have the option to monitor all Run-Always items in the group, or only specific ones.</p> <ul style="list-style-type: none"> • Tags to watch for change - Executes on all Tags or one or more Tags in order to monitor for value changes. Select 'all Tags' or 'Custom,' and select the Tag(s) from the dropdown.
Execute this group on a trigger	<p>Enables a trigger on a specific item in the group. The trigger item can be any Run-Always item, such as an OPC item, Tag reference, or an Expression item set to "Run-Always" mode.</p> <ul style="list-style-type: none"> • Trigger on item - select the item time you want to use as the trigger.
Only execute once	<p>The group will only execute once when the trigger goes into an active state, and will not execute again until the trigger goes inactive first. If unselected, the group will execute each time the trigger conditions evaluate to true.</p>

while trigger is active	
Reset trigger after execution	If using the ">0" or "=0" trigger modes, the trigger can be set to write an opposite value after the group has executed successfully. This is useful for relaying the execution back to the PLC.
Prevent trigger caused by group start	If selected, the group will not execute if the trigger is active on the first evaluation of the group. In the course of designing a group, it is common to stop and start it many times, and sometimes it is not desirable to have the group execute as a result of this. Selecting this option will prevent these executions, as well as executions caused by system restarts.
Trigger conditions	Set any of the following trigger conditions: <ul style="list-style-type: none"> • is !=0 (or true) • is =0 (or false) • is active or non-active, which causes the group to execute if the trigger value matches the is active condition. • Active on value change, which will cause the group to execute if the trigger changes value at all.
Write handshake on success	Set the item and the value you want to see when the group executes successfully.
Write handshake on failure	Set the item and the value you want to see when an error prevents the group execution.

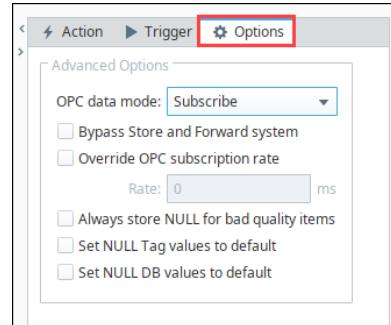


To learn more about configuring Transaction Groups with the different trigger options, refer to the [Trigger Options](#) page.

Advanced Settings

Transaction Groups offer several advanced settings that affect how execution occurs. These settings can be found under the **Options** tab for a group. The table below describes the Advanced settings.

Setting	Description	
Option	Description	
OPC Data Mode	Modifies how the group receives data from OPC.	
	Subscribe	Data points are registered with the OPC server, and data is received by the group on-change . This is the default setting and generally offers the best performance, as it reduces unnecessary data flow and allows the OPC server to optimize reads. Note: Data is received by the group asynchronously, meaning that it can arrive at any time. When the group executes, it "snapshots" the last values received and uses those during evaluation. If some values arrive after execution begins, they will not be used until the following execution cycle.
Read	Each time the group executes it will first read the values of OPC items from the server. This operation takes more time and involves more overhead than subscribed evaluation, but ensures that all values are updated together with the latest values. It is therefore commonly used with batching situations, where all of the data depends on each other and must be updated together. It's worth noting that when using an OPC item as the trigger, the item will be subscribed, and the rest of the values read when the trigger condition occurs.	
Bypass	This setting is only applicable to groups that insert rows into the database. Causes groups	



Store and Forward System	to target the database directly instead of going through the store-and-forward system. If the connection becomes unavailable, the group will report errors instead of logging data to the cache.																								
Override OPC subscription rate	Specifies the rate at which OPC items in the group will be subscribed. These items are normally subscribed at the rate of the group, but by modifying this setting it is possible to request updates at a faster or slower rate.																								
Always store NULL for bad quality items	With this option set to True, it will force the group to store a NULL value when the item has a bad quality, instead of writing the bad quality value.																								
Set NULL Tag values to default	If a NULL is read from the Tag, it will instead use a default value to write to the database, depending on the type. This can prevent errors for database columns that do not accept NULL values. The default values are the same as the table above.																								
Set NULL DB values to default	If a NULL is read from the database, it will instead use a default value to write to the Tag, depending on the type. This can prevent errors for OPC Tags that do not accept NULL values. Not available in a Historical Group. Enabling the Set DB Values to Default setting on Block Groups will clear the latched value, setting the item to a default if the corresponding database value is Null.																								
	<table border="1"> <thead> <tr> <th>Type</th> <th>Default Value</th> </tr> </thead> <tbody> <tr> <td>Byte</td> <td>0</td> </tr> <tr> <td>Short</td> <td>0</td> </tr> <tr> <td>Integer</td> <td>0</td> </tr> <tr> <td>Long</td> <td>0</td> </tr> <tr> <td>Float</td> <td>0.0</td> </tr> <tr> <td>Double</td> <td>0.0</td> </tr> <tr> <td>Boolean</td> <td>FALSE</td> </tr> <tr> <td>String</td> <td>" (Empty String)</td> </tr> <tr> <td>Date/Time</td> <td>Current Date/Time</td> </tr> <tr> <td>Dataset</td> <td>[0x0] (Empty Dataset)</td> </tr> <tr> <td>Array</td> <td>[] (Empty Array)</td> </tr> </tbody> </table>	Type	Default Value	Byte	0	Short	0	Integer	0	Long	0	Float	0.0	Double	0.0	Boolean	FALSE	String	" (Empty String)	Date/Time	Current Date/Time	Dataset	[0x0] (Empty Dataset)	Array	[] (Empty Array)
Type	Default Value																								
Byte	0																								
Short	0																								
Integer	0																								
Long	0																								
Float	0.0																								
Double	0.0																								
Boolean	FALSE																								
String	" (Empty String)																								
Date/Time	Current Date/Time																								
Dataset	[0x0] (Empty Dataset)																								
Array	[] (Empty Array)																								

Creating a Transaction Group

This example demonstrates how to configure a transaction group, specifically a Historical Group. However, the process of creating any transaction group type is very similar, especially so in the case of a standard group. The [Transaction Group Examples](#) section contains more examples.

1. Click on the **Transaction Groups** in the Project Browser to switch the Designer's workspace to the Transaction Group workspace.
2. In the Project Browser, right click on **Transaction Groups > New Transaction Group** to make a New Historical Group. Name the group 'Group.'



Basic Historical Group

[Watch the Video](#)

New Group

Name: Group



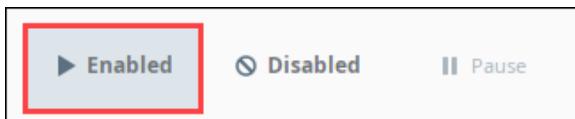
Realtime Group

[Watch the Video](#)

- Browse your OPC device and drag some OPC Tags to the **Basic OPC/Group Items** section. The group starts out 'Disabled' by default.

The screenshot shows the Ignition Designer interface. On the left is the Project Browser with various project items like 'Modbus', 'OPC UA Server', and 'Databases'. In the center is the 'Realtime Group' configuration window. It has tabs for 'Group', 'Basic OPC/Group Items (5)', 'Run Always Expression Item (0)', and 'Triggered Expression Item (0)'. The 'Group' tab shows execution status as 'Enabled' (with a red arrow pointing to it). Below the tabs are three buttons: 'Enabled' (highlighted with a red box), 'Disabled', and 'Pause'.

- Save your project.
- Click the **Enabled** button above the item tables to enable logging.



- Go to the **Action** tab and change the **Table Name**. For the example, we gave it the name "New_Test_Table".

At this step, your group only exists in the Designer.

The screenshot shows the 'Action' tab of the Realtime Group configuration. It includes sections for 'Execution Scheduling' (set to 'Timer' with a value of '1 second(s)'), 'Data source' ('<Default>'), 'Table name' ('New_Test_Table' highlighted with a red box), and several checkboxes for database options: 'Automatically create table' (checked), 'Use custom index column' (unchecked), 'Store timestamp to' (checked with 't_stamp'), 'Store quality code to' (unchecked), and 'Delete records older than' (unchecked).

- Save your project to start the group. Your group is now running and logging data to your database connection.
- To see the data, you can use the Ignition Designer's built-in **Database Query Browser**. The easiest way to do this is to click on the **Database** icon next to your group's Table Name field. The Query Browser is a convenient way to directly query your database connection without leaving the Ignition Designer. Of course, you can also use any query browser tools that came

with your database.

Database Query Browser

```
SELECT * FROM group_table_new WHERE Ramp0>0
```

Limit SELECT to: 1,000 rows

Resultset 1

group_table_new_ndx	Ramp0	Ramp1	Ramp2
1	4.649	69.896	1.93
2	4.649	69.896	1.93
3	4.649	69.896	1.93
4	5.651	75.037	1.13
5	5.651	75.037	1.13

7 rows fetched in 0.042s

Auto Refresh | Edit | Apply | Discard

MySQL

Schema History

- group_table3
- group_table_new
- history
- history_sine_tags
- hitachi_errors
- hitachi_messages

Related Topics ...

- Database Connections
- Transaction Group Examples
- Database Query Browser
- Trigger Options

In This Section ...

Types of Groups

The SQL Bridge Module provides four different types of Transaction Groups that you can use in your projects. Each of these different types of groups vary in their operation and use for data logging and database to PLC synchronization.

Historical Group

The Historical Group makes it easy to quickly log data historically to a SQL database.

General Description

The historical Group inserts records of data into a SQL database, mapping items to columns. Full support for triggering, expression items, hour & event meters and more means that you can also set up complex historical transactions. Unlike the Standard Group, the Historical Group cannot update rows, only insert. It also cannot write back to items (besides trigger resets and handshakes).

Group Settings

The settings of the Historical Group are identical to the settings in the Standard Group, but limited to inserting rows.

Typical Uses

Basic historical logging - Recording data to a SQL database gives you incredible storage and querying capabilities, and makes your process data available to any application that has DB access.

Shift tracking - Use an expression item to track the current shift based on time, and then trigger off of it to record summary values from the PLC. Use a handshake to tell the PLC to reset the values.

Standard Group

The Standard Group is called such because it's a flexible, general use group that can be adapted to a variety of situations. The data model is row based, with items mapping to columns and the data corresponding to a specific row of a table.

General Description

The Standard Group contains items, which may be mapped to the database, or used internally for features such as triggering or handshakes. Items that are mapped to the database target a specific column of a single specific row, chosen according to the group settings. Items can be mapped in a one-way fashion, or bi-directionally, in which the value of the database and the item will be synchronized.

The group may also insert new rows instead of updating a specific row. In this manner, data can be inserted for historical purposes based on a timer, with an optional trigger.

Group Settings

The Standard Group uses a timer-based execution model shared by all groups, and the normal trigger settings. Additionally, there are several settings specific to the group type:

Automatically create table - If the target table does not exist, or does not have all of the required columns, it will be created/modified on group startup. If not selected and the table doesn't match, an error will be generated on startup.

Use custom index column - If selected, you may enter any column name to hold the index. If unselected, the table index will be named <table name>_ndx.

Store timestamp to - Specifies whether or not to store a timestamp with the record, and the target column. The timestamp will be generated by the group during execution. For groups that update a row, the timestamp will only be written if any of the values in the group are also written.

Store quality code to - If selected, stores an aggregate quality for the group to the specified column. The aggregate quality is the combined quality of all of the items that write to the table. For more information about quality values, see [Data Quality](#).

Delete records older than - If selected, records in the target table will be deleted after they reach the specified age. This setting is useful for preventing tables from growing in an unbounded manner, which can cause disk space and performance problems over time.

On this page ...

- Historical Group
 - General Description
 - Group Settings
 - Typical Uses
- Standard Group
 - General Description
 - Group Settings
- Table action
 - Typical Uses
- Block Group
 - General Description
 - Typical Uses
 - Table Format
 - Row ID and Block ID
 - Group Settings
 - Table action
- Stored Procedure Group
 - Group Settings
 - Typical Uses
 - Known Issues
 - Parameters in the Stored Procedure Group

Table action

This section details how the group interacts with the table on each execution, and is not available for the Historical Group type. This means when the Timer or Schedule is active, and the Trigger condition are met. The group can insert a new row, or update the first, last or a custom record.

Insert New Row - This option will make the group insert a new record into the database every time the group executes. This is the forced behavior of the Historical Group.

Update / Select - This option will either update or select from matching rows based on the option selected below it. The **Update Mode** property above determines whether an update (OPC to DB), select (DB to OPC), or both (Bi-directional) are used when the group executes.

First - Use the first row in the table. It is not recommended to use this option unless the order of the data in the table is guaranteed.

Last - Use the last row in the table. This is commonly used when another group (or another program) is inserting new rows for us, and we always want to update the most recent record.

Custom - A custom update clause is essentially the WHERE clause of the SQL query that will be generated to read and write the group data. This usually contains a reference to a Tag in the group. IE: `column_name = {!-jitem_name}`

Key/Value Pairs - Used to inject dynamic values in order to create a WHERE clause for you. The table below this option will allow you to enter column names and link them to values (usually Tags in the group). This option also has the ability to Insert a new row with the current key/value pair if it was not found.

Typical Uses

Standard Groups can be used any time you want to work with a single row of data. This can include:

Historical logging - Set the group to insert new records, and log data historically either on a timer, or as the result of a trigger. Flexible trigger settings and handshakes make it possible to create robust transactions.

Maintain status tables - Keep a row in the database updated with the current status values. Once in the database, your process data is now available for use by any application that can access a database, dramatically opening up possibilities.

Manage recipes - Store recipe settings in the database, where you have a virtually unlimited amount of memory. Then, load them into the PLC by mapping DB-to-OPC using a custom where clause with an item binding in order to dynamically select the desired recipe.

Sync PLCs - Items in the group can be set to target other items, both for one-way and bidirectional syncing. By adding items from multiple PLCs to the group, you can set the items of one PLC to sync with the others. By creating expression items that map from one PLC item to the other, you can manipulate the value before passing it on.

Block Group

Block Groups instead allow you to store your data in a tall format. They allow you to create a unique type of item, called a **Block Item**, which represents an ordered list of values to store within a column for each execution.

General Description

A Block Group contains one or more block items. Each block item maps to a column in the group's table, and then defines any number of values (OPC or SQLTag items) that will be written vertically as rows under that column. The values may be defined in the block item in two modes. The first, List mode, lets a list of value-defining items to be entered. These value items may either be OPC items, Tag items, or static values. The second mode, Pattern mode, can be useful when OPC item paths or Tag paths contain an incrementing number. You may provide a pattern for the item's path, using the wildcard marker {?} to indicate where the number should be inserted.

Block groups are very efficient, and can be used to store massive amounts of data to the database (for example, 100 columns each with 100 row -10,000 data points- will often take only a few hundred milliseconds to write, depending on the database). They are also particularly useful for mirroring array values in the database, as each element will appear under a single column, and share the same data type.

Like the Standard Group, the Block Group can insert a new block, or update the first, last or a custom block. Additionally, the group can be set to only insert rows that have changed in the block.

In addition to block items, the group can have other OPC items, Tag references, and Expression items. These items can be used for triggers, handshakes, etc. They may also target a column to be written, and will write their single value to all rows in the block.

The block group is so named because it writes "blocks" of data to a database table, consisting of multiple rows and columns.

Typical Uses



Block Group Type

[Watch the Video](#)

Block Groups are useful in a number of situations where you need to deal with a lot of data efficiently. Mirroring/Synchronizing array values to DB - Arrays are often best stored vertically, which makes them perfect for Block Groups. Pattern mode makes configuration a breeze by allowing you to specify the array as a pattern, and set the bounds

Recipe management - Like Standard Groups, but used when set points are better stored vertically than horizontally.

Vertical history tables - Group data points by data type (integer, float, string), create a copy of the item that stores item path, and then use the insert changed rows option to create your own vertically storing historical tables. Create additional copies of the block item that refer to quality and timestamp in order to get further information about the data point.

Table Format

Due to their nature, Block Groups store records in a different format than the other groups. Consider how other Transaction Groups work. A single execution of a Standard or Historical Group would store a row that looked like the following:

table_ndx	tag1	tag2	tag3
1	10	20	30

We could take the Tags from the above example, and place them in under a single block item like so:

Block Items (1)								
Item Name	Source Value	Latched Value	Mode	Target Name	Data Type	Properties	Size	
Block Item				Tags	String		3	
[default]Tag1	10	10						
[default]Tag2	20	20						
[default]Tag3	30	30						

Note that each Tag is nested under the block item, and the block item is targeting the "Tags" column under Target name. A single execution of this group stores the records in our table as so:

table_ndx	Tags
1	10
2	20
3	30

Each additional block item would store records as a separate column.

table_ndx	Tags	More_Tags
1	10	11
2	20	22
3	30	33

Row ID and Block ID

Using the same Tag example from above, if we kept inserting new rows at every execution, our table would start to look like the following:

table_ndx	Tags
1	10
2	20

3	30
4	15
5	25
6	35

This isn't ideal, since the table doesn't have a great way to show which value came from which Tag. To help with this, Block Groups have optional `row_id` and `block_id` columns that can be enabled (see the "Store row id" and "Store block id" settings under Group Settings). If we enable both the Block ID and Row ID, our table would look like the following:

table_ndx	Tags	row_id	block_id
1	10	0	1
2	20	1	1
3	30	2	1
4	15	0	2
5	25	1	2
6	35	2	2

Block ID represents the a single execution of the group, meaning rows with the same `block_id` value were inserted together. We see `block_id` values of 1 (colored green) are part of the same execution, and rows with a `block_id` value of 2 (colored blue) are a separate execution.

Row ID is an index representing which item in the block item the row corresponds to. In our example, Tag1 is the first or top item in the block item (row index 0), Tag2 is next (row index 1), and Tag3 is last (row index 2). Now we know that any value on that table with a `row_id` of 0 came from Tag1.

Group Settings

Beyond the differences in the data, namely that the Block Group works with multiple rows instead of just 1, this group type shares many similarities with the Standard Group.

The unique settings are:

Automatically create table - If the target table does not exist, or does not have all of the required columns, it will be created/modified on group startup. If not selected and the table doesn't match, an error will be generated on startup.

Automatically create rows - If the target rows do not exist, they will be created on group execution. If not selected and the rows don't match, no records will be updated.

Use custom index column - If selected, you may enter any column name to hold the index. If unselected, the table index will be named <table name>_ndx.

Store timestamp to - Specifies whether or not to store a timestamp with the record, and the target column. The timestamp will be generated by the group during execution. For groups that update a row(s), the timestamp will only be written if any of the values in the group are also written.

Store quality code to - If selected, stores an aggregate quality for the row to the specified column. The aggregate quality is the combined quality of all of the items that write to that row. For more information about quality values, see [Data Quality](#).

Store row id - Each row will be assigned a numeric id, starting at 0. If selected, this id will also be stored with the data.

Store block id - If selected, an incremental block id will be stored along with the data. This number will be 1 greater than the previous block id in the table.

Delete records older than - If selected, records in the target table will be deleted after they reach the specified age. This setting is useful for preventing tables from growing in an unbounded manner, which can cause disk space and performance problems over time.

Table action

This section details how the group interacts with the table on each execution, and is not available for the Historical Group type. This means when the Timer or Schedule is active, and the Trigger condition are met. The group can insert a new row, or update the first, last or a custom record.

Insert New Block - If selected, each row of the block will be inserted when the group executes, even if the data has not changed.

Insert changed rows - This option will only insert the rows that have new data when the group executes. This is particularly useful for recording history for many data points on an "on change" basis, provided there is a unique id column defined. The "store row id" feature is useful for this, as well as the ability to reference the item path in an item's value property.

Update / Select - This option will either update or select from matching rows based on the option selected below it. The Update Mode property above determines whether an update (OPC to DB), select (DB to OPC), or both (Bi-directional) are used when the group executes.

First - Use the first row in the table. It is not recommended to use this option unless the order of the data in the table is guaranteed.

Last - Use the last row in the table. This is commonly used when another group (or another program) is inserting new rows for us, and we always want to update the most recent record.

Custom - Like Standard Groups, this setting allows you to target a specific section of the table, using SQL WHERE clause syntax, with the ability to bind to dynamic item values. Unlike Standard Groups, however, the WHERE clause specified should result in enough rows to cover the block. Excess rows will not be written to, but fewer rows will result in a group warning indicating that some data could not be written.

Stored Procedure Group

The Stored Procedure Group lets you quickly map values bi-directionally to the parameters of a stored procedure. It is similar to the other groups in terms of execution, triggering, and item configuration. The primary difference is that unlike the other group types, the target is not a database table, but instead a stored procedure.

Items in the group can be mapped to input (or inout) parameters of the procedure. They also can be bound to output parameters, in which case the value returned from the procedure will be written to the item. Items can be bound to both an input and output at the same time.

Parameters may be specified using either **parameter names** or **numerical index**. That is, in any location where you can specify a parameter, you can either use the name defined in the database, or a 0-indexed value specifying the parameter's place in the function call.



You cannot mix names and indices. That is, you must consistently use one or the other.

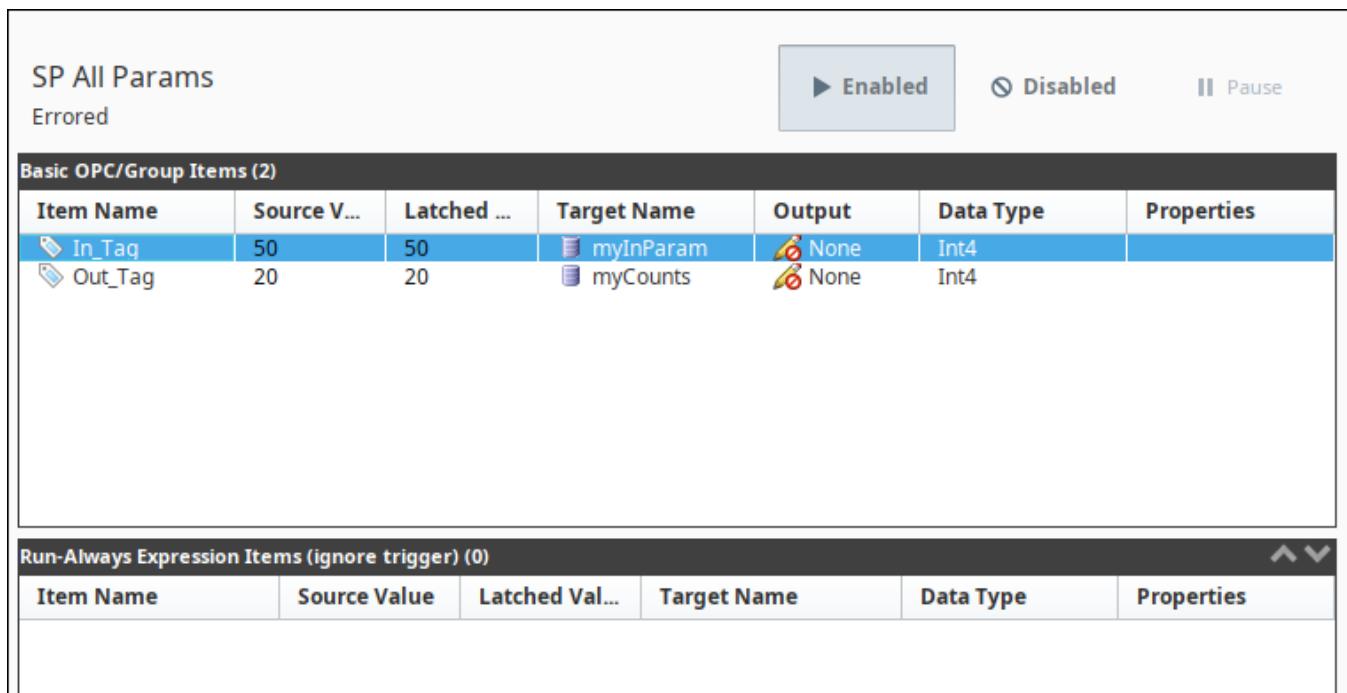


**INDUCTIVE
UNIVERSITY**

**Stored Procedure
Group Type**

[Watch the Video](#)

If using parameter names, the names should not include any particular identifying character (for example, "?" or "@", which are used by some databases to specify a parameter).



Basic OPC/Group Items (2)						
Item Name	Source V...	Latched ...	Target Name	Output	Data Type	Properties
In_Tag	50	50	myInParam	None	Int4	
Out_Tag	20	20	myCounts	None	Int4	

Run-Always Expression Items (ignore trigger) (0)					
Item Name	Source Value	Latched Val...	Target Name	Data Type	Properties

Group Settings

The Stored Procedure Group's settings look and act the same as those of the Historical Group. The primary difference, of course, is that instead of specifying a table name and column names, you'll specify a Stored Procedure and its parameters.

Store timestamp to - Specifies whether or not to store a timestamp with the record, and the target column. The timestamp will be generated by the group during execution. For groups that update a row, the timestamp will only be written if any of the values in the group are also written.

Store quality code to - If selected, stores an aggregate quality for the group to the specified column. The aggregate quality is the combined quality of all of the items that write to the table. For more information about quality values, see [Data Quality](#).

Procedure Name - The name of the Stored Procedure (SP) that you will be using. You must look into the SP definition to see what inputs and outputs are available.

Typical Uses

Call stored procedures - The Stored Procedure Group is the obvious choice when you want to bind values to a stored procedure. It can also be used to call procedures that take no parameters (though this can also be accomplished from Expression Items/SQLTags).

Replace RSSQL - The Stored Procedure Group is very popular among users switching from RSSQL, given that application's heavy use of stored procedures.

Known Issues

When using an Oracle database, you must use indexed parameters.

Parameters in the Stored Procedure Group

When using a Stored Procedure Group, parameters may be configured to each item based on the type of the parameter:

- The **Target Name** column is used for writing, so specifying an IN or INOUT parameters under this column will have the item try to write its value to the parameter
- The **Output** column is used to move the value of an OUT or INOUT parameter into an item in the group. If an item in a group is configured to reference an OUT parameters, its **Target Name** value should be set to **Read-Only**.

Related Topics ...

- [Group Update Rate](#)

Item Types

Items are the backbone of a Transaction Group. They represent a link between a source value (derived from either an OPC value or an expression) and a cell in a database table. Items generally aren't executed in a reliable order, with the exception of **Expression** items.

Expression items can be ordered using the up and down arrows located to the right of the list where the items are displayed. This can be crucial for performing complex operations that require a specific sequence. Below is a listing of each type of item.

Item Type	Description
OPC Item	Directly subscribed to an OPC server at the rate of the group. These items effectively ignore the gateway's Tag system, bypassing Tag groups and Tag providers altogether.
Expression Item	Much like an expression Tag, expression items are flexible in that their value can come from a number of different sources: specifically an expression or a database query. Expression items have two sub types: <ul style="list-style-type: none">• Run-Always expression items are evaluated every time the group executes. Meaning, they'll run their associated expression or query every time the group executes.• Triggered expression items only evaluate when the group trigger is active.
Tag Reference Item	A reference to a Tag in a Tag provider. Allows a Tag to be used in a group like any other item type, except that the Tag is evaluated by its scan class instead of by the group. For more information, see the Tag References vs. OPC Items section on this page. Tag Reference Items can reference the value on any Tag in a Tag provider, such as query Tags and memory Tags.

On this page ...

- [Tag References and OPC Items](#)
- [Expression Items](#)
 - Scope
 - Execution Order
 - Expression Type
- Run Always vs. Triggered Items
 - Changing the Evaluation State
- [SQL Queries and Expressions](#)
- [Creating a New Item](#)
- [Item Type Property Table](#)
 - [OPC Item Options](#)
 - [Tag Reference Item Options](#)
 - [Expression Item Options](#)



INDUCTIVE
UNIVERSITY

Item Types

[Watch the Video](#)

Tag References and OPC Items

It is easy to confuse the definition and purpose of Tag reference items and direct OPC items in Transaction Groups.

Tags may be referenced inside of Transaction Groups through a Tag Reference Item. Since the source of the Tag reference item exists outside of the Transaction Group, they have their own rules and configurations that determine when their value changes. Thus Tag reference items can have their value update according to their own execution (commonly, a Tag Group). Adding a Tag into a group is like creating a shortcut to that Tag. However, once in the group, the item can be used like any other item. Tag references are useful when it is necessary to have a single value in multiple groups, for example, as a trigger in order to coordinate execution.

OPC Items in groups (as well as expression items in groups), however, are completely executed by the group. They do not exist outside of the group in which they are defined. They are subscribed and evaluated according to the rate of the group.

Refer to the Properties Table at the bottom of this page to see the properties for both [Tag](#) and [OPC Items](#).



INDUCTIVE
UNIVERSITY

Tag References vs. OPC Items

[Watch the Video](#)

Expression Items

Expression Items are items not driven by a PLC. Instead they allow you to configure a static value, or use some other means to automatically set a value, such as a query. They are useful for executing comparisons, simple math, and looking up values from other database tables.

Much like OPC Items, Expression Items can have alarms configured, as well as [numeric scaling](#) applied directly to the item.

Scope



INDUCTIVE
UNIVERSITY

Expression Items

It is important to understand that an Expression Item only exists within its group, and can not be referenced by items in other Transaction Groups, Tags, and any components on a window.

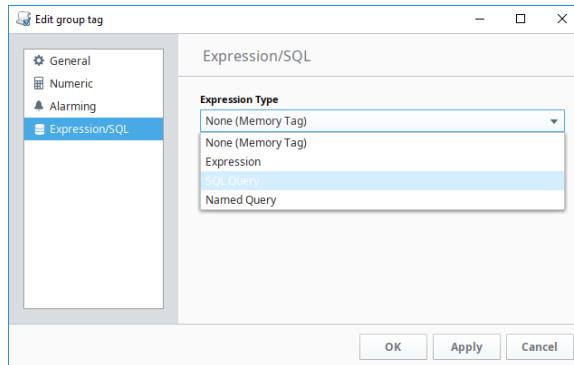
[Watch the Video](#)

Execution Order

All Expression items will evaluate in order from top to bottom. This means referencing an Expression Item above will pull the new value, but referencing an Expression Item below will give you the value from the last group execution.

Expression Type

How an Expression Item determines its value depends heavily on its type.



Expression Type	Definition
None	Behaves similar to a Memory Tag in that the value does not automatically change.
Expression	Uses Ignition's Expression Language to determine the value on the Item. The expression can reference other items in the group, as well as Tags.
SQL Query	Utilizes a SQL query to determine the item's value. Thus, a query can execute on the item and the results can be referenced by other items in the same group.
Named Query	Selecting this option will cause the value on the item to be determined by a Named Query in the same project as the Transaction Group.

Refer to the Property Table at the bottom of this page to see the [Expression Item properties](#).

Run Always vs. Triggered Items

Expression Item can be configured in two different evaluation states:

- **Triggered:** The Expression Item executes only when the Transaction Group is triggered. However if the group is **not** configured to execute on a trigger, then the item will evaluate every time the group executes (similar to how the **Run-Always** state works). This is the default evaluation state new Expression Items use.
- **Run-Always:** The Expression Items will run before the group trigger is checked, so it always executes at the group's rate. This allows your expression to always evaluate regardless of the trigger in the group. Additionally, this state allows you to use the Expression Item as the trigger for the group. We advise that you never have a Target for a Run-Always Expression item



Run Always vs. Triggered Items

[Watch the Video](#)

because it always runs.

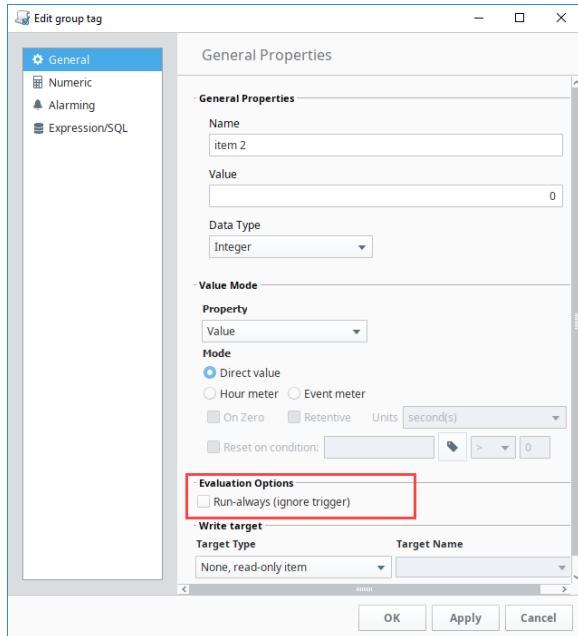
Item Name	Source ...	Latch...	Mode	Target Name	Data ...	Prop...
accumulatorLevel	N/A	N/A	—	accumul...	Int4	
ambientHum	N/A	N/A	—	ambient...	Float4	
ambientTemp	N/A	N/A	—	ambient...	Float4	
dischargePressure	N/A	N/A	—	discharg...	Float4	
dischargeTemp	N/A	N/A	—	discharg...	Int4	
receiverLevel	N/A	N/A	—	receiver...	Int4	

Item Name	Source ...	Latch...	Target Name	Data Ty...	Propert...
My Run_Always Item	N/A	N/A	Read-only	Int4	

Item Name	Source ...	Latch...	Target Name	Data Type	Properties
My Triggered Item	N/A	N/A	Read-only	Int4	

Changing the Evaluation State

Toggling between the two modes can be accomplished by dragging and dropping the Expression Item to either the **Run-Always Expression Items** table or the **Triggered Expression Items** table. Alternatively, the evaluation state can be changed by editing the Expression Item and toggling the **Run-always (ignore trigger)** checkbox



SQL Queries and Expressions

Expression items can use [SQL statements](#) and [Ignition's Expression language](#) to automatically determine the value of an Expression Item. This is useful in scenarios where you want to use a value from the database as the trigger for the Transaction Group, or aggregate several other items in the group into a single value.

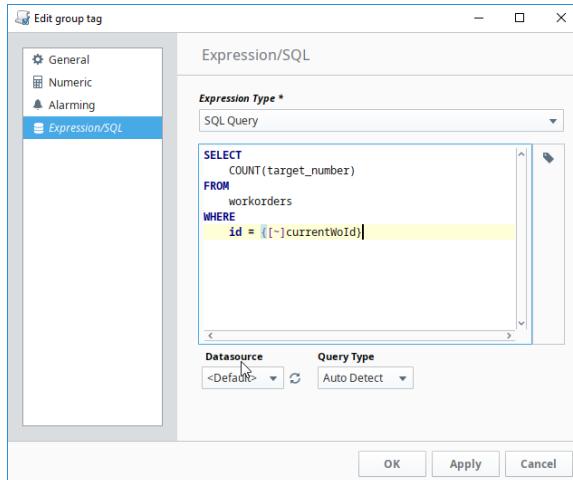
Expressions and queries on an Expression Item can reference the value of other items in the group or Tags in the system by clicking on the **Tag** icon.

There are several Expression functions available that exist only for Transaction Groups. You can find them in the **Store and Forward** and **Variables** sections of the **f(x)** function list.



SQL Query Expression Items

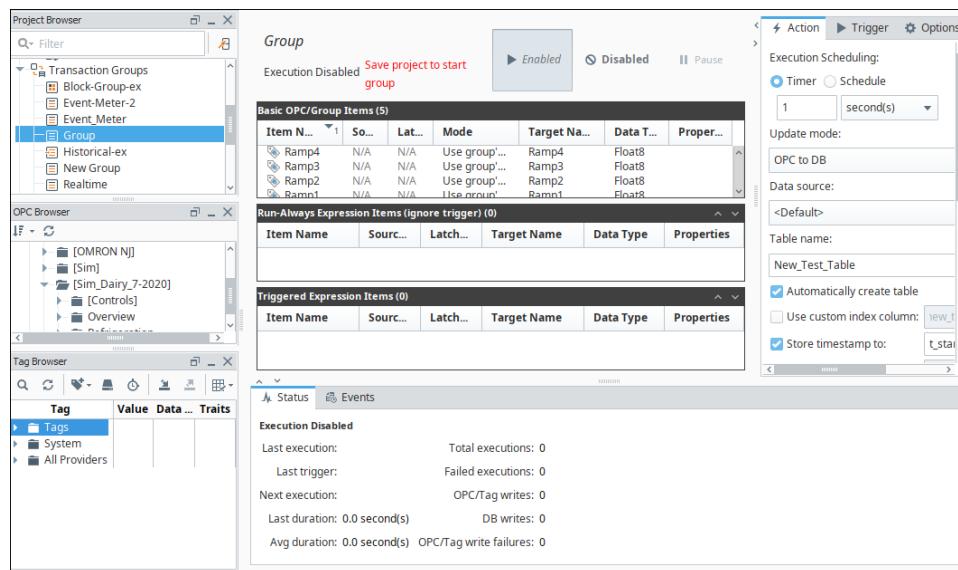
[Watch the Video](#)



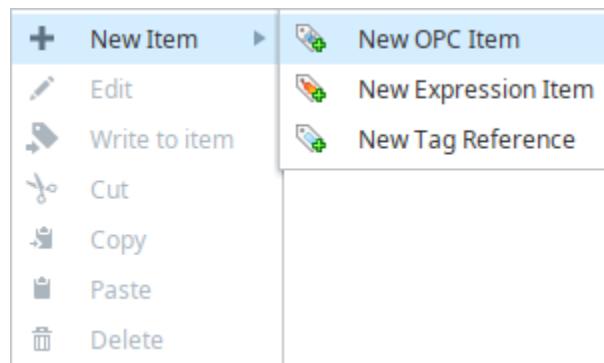
Creating a New Item

Below is an example of creating a new item. The steps can be applied to any item type.

1. In the Designer, go to Project Browser, and click on **Transaction Groups**.
The workspace now changes to the Transaction Group workspace.
2. Right-click on **Transaction Group** to create a New Transaction Group, or click on a group you have previously created.
You will now see the workspace changes to look like:



3. Right-click in the white area, and choose **New Item > New OPC Item**. The options in the popups represent the different item types. Refer to the [property table](#) on this page for more information on the various item types and their properties.



- Once you configured the item, click **OK**. Different items have different properties. A description of [item properties](#) for each type can be found on this page.

Item Type Property Table

The following tables describes the OPC, Tag and Expression Item properties.

OPC Item Options

Property	Description
General	
Name	The name of the OPC item in the group. There cannot be duplicate names within a group.
Data Type	The data type used to read values from the PLC.
OPC Properties	
OPC Server	The Selected OPC Server. This is a dropdown list showing all the OPC Servers added in the Ignition Gateway.
OPC Item Path	The OPC address assigned by the server. Dragging and dropping from the OPC Browser will automatically populate this field.
Source Data Type	Data type for the OPC item.
Value Mode	
Property	<p>Which property of the OPC item you want to use.</p> <ul style="list-style-type: none"> Value - Item value Quality - Quality code from OPC Server (192 = GOOD_DATA) Timestamp - The last time the item value changed Name - The SQLBridge Item Name property of this Item <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;"> Note: UDT Instances used in Transaction groups only support String datatypes. Other datatypes could cause Transaction Groups groups running in previous versions of Ignition to fault. </div>
Mode	<p>Options for displaying values based on the Item value.</p> <ul style="list-style-type: none"> Direct Value - Item value Hour Meter - Record the amount of time the Item value is non-zero. This accumulation will reset to zero when the item value goes to zero. The data type should be set to integer or float when using an Hour Meter regardless of the OPC Item type. <ul style="list-style-type: none"> On Zero - Use a zero value to accumulate time instead of a non-zero value Retentive - Retain the Hour Meter value when it is not accumulating. Units - The time units to display. Event Meter - Record the number or times the Item value is non-zero. The data type should be set to integer when using an Event Meter regardless of the OPC Item type. <ul style="list-style-type: none"> On Zero - Use a zero value to accumulate events instead of a non-zero value
Write Target	
Mode	<p>Changes the items directional read/write option.</p> <ul style="list-style-type: none"> Use group's mode - Inherit the Update Mode from the Item's Group. OPC to DB - Only read from the OPC server and write to the database. DB to OPC - Only read from the database and write to the OPC Server. Bi-directional OPC wins - Read and Write to both the database and OPC Server. On group start, write OPC Server values to the database. Bi-directional DB wins - Read and Write to both the database and OPC Server. On group start, write database values to the OPC Server.
Target Type	<p>This is the selection for what the Item will write to when the group executes.</p> <ul style="list-style-type: none"> None, read-only item - Do not write this value to the database.

	<ul style="list-style-type: none"> • Database field - Write the Item value to the specified column in the database table. This list will populate with all the column names from the Group's target table after the first time the group is run.
Target Name	The name of the column in the database that this Item will write to when the group executes. The Target Name list will populate with all the column names from the Group's target table if the Target Type is Database field.
Alarming	The Alarming settings for the OPC items. See Alarming Properties for a full explanation.

Tag Reference Item Options

General	
Name	The name of the OPC item in the group. There cannot be duplicate names within a group.
Tag Path	The path to the tag being referenced. This value is not editable except by clicking the Insert Tag button. There cannot be duplicate names within a group.
Data Type	The data type to write to into the database if this item is not read-only.
Value Mode	
Property	<p>Which property of the Tag you want to use.</p> <ul style="list-style-type: none"> • Value - Item value • Quality - Quality code of the Tag (192 = GOOD_DATA) • Timestamp - The last time the item value changed • Name - The SQLBridge Item Name property of this Item.
Mode	<p>Options for displaying values based on the Item value.</p> <ul style="list-style-type: none"> • Direct Value - Item value • Hour Meter - Record the amount of time the Item value is non-zero. This accumulation will reset to zero when the item value goes to zero. The data type should be set to integer or float when using an Hour Meter regardless of the OPC Item type. On Zero - Use a zero value to accumulate time instead of a non-zero value Retentive - Retain the Hour Meter value when it is not accumulating. Units - The time units to display. • Event Meter - Record the number or times the Item value is non-zero. The data type should be set to integer when using an Event Meter regardless of the OPC Item type. On Zero - Use a zero value to accumulate events instead of a non-zero value
Write Target	
Mode	<p>Changes the items directional read/write option. This is only editable when the target Type is set to Database field.</p> <ul style="list-style-type: none"> • Use group's mode - Inherit the Update Mode from the Item's Group. • OPC to DB - Only read from the OPC server and write to the database. • DB to OPC - Only read from the database and write to the OPC Server. • Bi-directional OPC wins - Read and Write to both the database and OPC Server. On group start, write OPC Server values to the database. • Bi-directional DB wins - Read and Write to both the database and OPC Server. On group start, write database values to the database.
Target Type	<p>This is the selection for what the Item will write to when the group executes.</p> <ul style="list-style-type: none"> • None, read-only item - Do not write this value to the database. • Database field - Write the Item value to the specified column in the database table.
Target Name	The name of the column in the database that this Item will write to when the group executes. The Target Name list will populate with all the column names from the Group's target table if the Target Type is Database field.

Expression Item Options

General	
Name	The name of the OPC item in the group. There cannot be duplicate names within a group.
Value	The static value of this Expression item. This will be overwritten by an Expression/SQL binding.
Data	The data type values are stored as.

Type	
Value Mode	
Property	<p>Which property of the OPC item you want to use.</p> <ul style="list-style-type: none"> • Value - Item value • Quality - Quality code of the expression/SQL Query (192 = GOOD_DATA) • Timestamp - The last time the item value changed. • Name - The SQLBridge Item Name property of this Item.
Mode	<p>Options for displaying values based on the Item value.</p> <ul style="list-style-type: none"> • Direct Value - Item value • Hour Meter - Record the amount of time the Item value is non-zero. This accumulation will reset to zero when the item value goes to zero. The data type should be set to integer or float when using an Hour Meter regardless of the OPC Item type. On Zero - Use a zero value to accumulate time instead of a non-zero value Retentive - Retain the Hour Meter value when it is not accumulating. Units - The time units to display. • Event Meter - Record the number or times the Item value is non-zero. The data type should be set to integer when using an Event Meter regardless of the OPC Item type. On Zero - Use a zero value to accumulate events instead of a non-zero value
Evaluation Mode	Run-always (ignore Trigger) - When selected, this causes the group to evaluate at each group interval, before the trigger state is evaluated.
Write Target	<p>Target type - This is the selection for what the Item will write to when the group executes.</p> <ul style="list-style-type: none"> • None, read-only item - Do not write this value to the database. • Database field - Write the Item value to the specified column in the database table. • Other Tag - Write the Expression Item's value back to an OPC item or Tag Reference.
Target Name	The name of the column in the database that this Item will write to when the group executes. The Target Name list will populate with all the OPC Item and Tag Reference names from this Group, or the column names from the Group's target table depending on the Target Type selected.
Numeric	These are the Numeric properties for Expression Items. For a full description, see Tag Scaling Properties .
Alarming	These are the Alarming settings for the OPC items. See Alarming Properties for a full explanation.
Expression	These are the Expression/SQL Query options for Expression Items. See Expression/SQL Properties for a full explanation.

Hour and Event Meters

Hour meter and **Event meter** refer to the **Value Mode** option settings on Tags in Transaction Groups. The Value mode drives values that are used to create values that determine how long a value was true. While the selected Value Mode for most transactions is **Direct**, however, the **Hour meter** mode accumulates value for the duration of a condition, and the **Event meter** accumulates count in response to the condition.

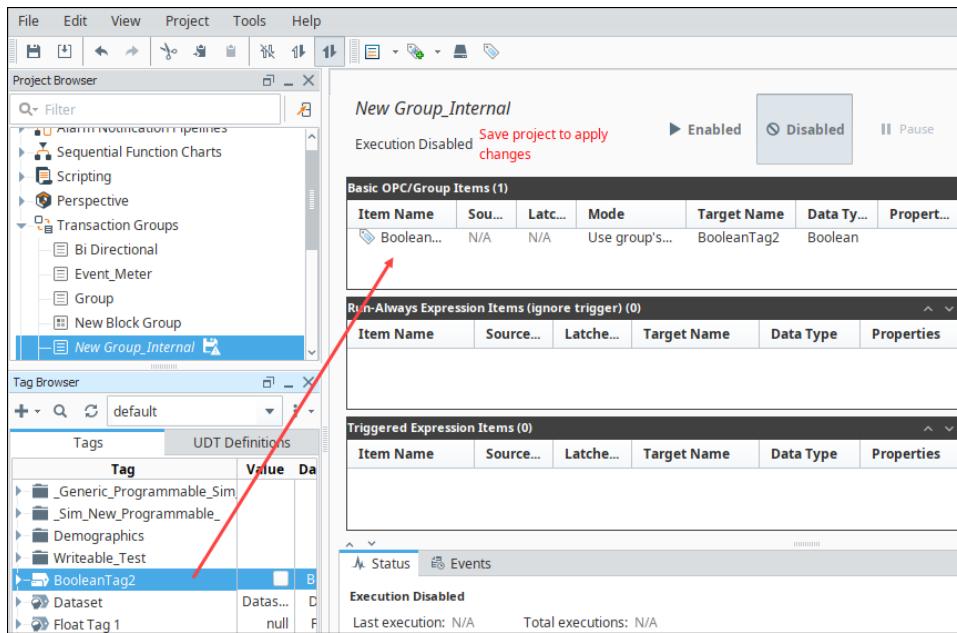
Hour Meter

It is common to write to a Tag during the time when a Tag's value is true. An hour meter simplifies this effort. Hour meters can be meters that accumulate the millisecond, second, minute, hour, or day.

The Inductive University logo is displayed, featuring a blue hexagon with 'IU' and green laurel wreaths. To its right, the text 'INDUCTIVE UNIVERSITY' is written in a serif font. Below this, the section title 'Hour and Event Meters' is centered. At the bottom right of the section, there is a blue rectangular button with the text 'Watch the Video'.

Count the Duration of a Tag Being True

1. From the Tag Browser, drag a boolean Tag into the **Basic OPC/Groups Items** area of a **Standard Transaction Group**.



2. From OPC Browser or Tag Browser, drag a memory tag or OPC tag (must be a numeric data type) into the **Basic OPC/Groups Items** portion of the **Standard Transaction Group**.
3. Right-click on the boolean tag in the Transaction Group and select **Edit** to edit it. The **Edit group tag** window is displayed.

New Group_Internal
Unable to Start Save project to apply changes

Enabled Disabled Pause

Basic OPC/Group Items (2)

Item Name	Source V...	Latched ...	Mode	Target Na...	Data Type	Properties
_Sim/_Ramp/Ramp0	N/A	N/A	—	Ramp0	Float8	
_Sim/_Boolean Exam...	N/A	N/A	—	Boolean_E...	Boolean	

Run-Always Expression Items (0)

Item Name	So...	a...	Target Name	Data Type	Properties

Triggered Expression Items (0)

Item Name	Source Val...	Latched Va...	Target Name	Data Type	Properties

+ New Item Edit Write to item Cut Copy Paste Delete

4. In the **Edit group tag** window, for **Value Mode**, select **Hour meter**.
5. In the **Edit group tag** window, for **Target Type** select **Other tag** from the dropdown menu, and in **Target Name** enter the name of the memory tag as the target, and click **OK**.

Edit group tag

Tag Item

General

Name: _Sim/_Boolean Example Tag

Tag Path: [default]_Sim/_ReadOnly/ReadOnlyBoolean

Data Type: Boolean

Value Mode

Property: Value

Mode:

Direct value Hour meter Event meter

On Zero Retentive Units: second(s)

Reset on condition: [] > [] 0

Write target

Mode: Use group's mode

Target Type: Other tag

Target Name: _Sim/_Ramp/Ramp0

OK Apply Cancel

6. In the **Basic OPC/Groups Items** area where the tags are located, go to the **Target Name** column, left-click on each tag to get the dropdown menu, and set the following:

- a. For the boolean tag, select the memory tag from the dropdown which is set previously as the target tag to write the hour meter to.

Item Name	Source V...	Latched ...	Mode	Target Na...	Data Type	Properties
_Sim-/Ramp/Ramp0	N/A	N/A	---	Ramp0	Float8	
_Sim-/Boolean Exam...	N/A	N/A	---	n-/Ramp/Ramp0	Boolean	

- b. For the memory Tag, select Read-only from the dropdown.

Item Name	Source V...	Latched ...	Mode	Target Na...	Data Type	Properties
_Sim-/Ramp/Ramp0	N/A	N/A	---	Ramp0	Float8	
_Sim-/Boolean Exam...	N/A	N/A	---		Boolean	

7. Click **Enabled** at the top of the page, and do a **File > Save** to start the group.

8. Make the boolean Tag true to start the Hour meter.

Event Meter

Another common scenario is to count the number of times an event occurred. For example, where there is boolean Tag and you want to count the number of cycles the boolean Tag has experienced.

Count in Response to a Tag being True

- From OPC Browser or Tag Browser, drag a boolean Tag into the **Basic OPC/Groups Items** area of a **Standard Transaction Group**.
- From OPC Browser or Tag Browser, drag a memory Tag or OPC Tag (must be a numeric data type) into the **Basic OPC/Groups Items** portion of the **Standard Transaction Group**.

The screenshot shows the configuration interface for the 'Event-Meter-2' transaction group. At the top, there are three status buttons: 'Enabled' (green), 'Disabled' (grey), and 'Pause' (yellow). Below these are three sections:

- Basic OPC/Group Items (2)**: A table with columns: Item Name, Source..., Latched ..., M..., Target ..., Data Type, Properties. It contains two rows:

_Sim_New_Programmable/_Boolean1	N/A	N/A		Boolean1	Boolean
_Sim_New_Programmable/_Events	N/A	N/A		Events	Int4
- Run-Always Expression Items (ignore trigger) (0)**: An empty table with columns: Item Name, Source Value, Latched Va..., Target Name, Data Type, Properties.
- Triggered Expression Items (0)**: An empty table with columns: Item Name, Source Value, Latched Va..., Target Name, Data Type, Properties.

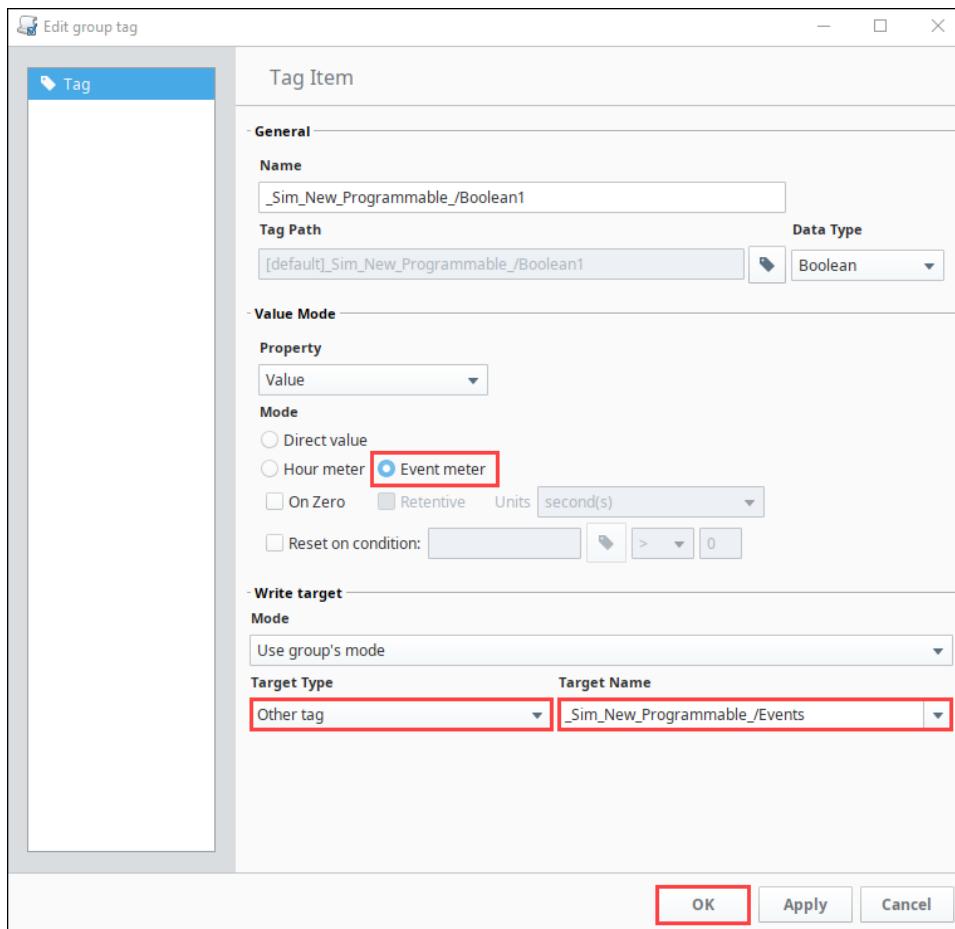
3. Right-click on the boolean Tag in the Transaction Group and select **Edit** to edit it.
4. In the **Edit group tag** window, set the following:

Value Mode: **Event meter**

Target Type: **Other Tag**

Target Name: **_Sim_New_Programmable/_Events** (or name of the Tag you are using)

5. Click **OK**.



6. In the **Basic OPC/Groups Items** area where the Tags are located, go to the **Target Name** column. Left-click on each Tag to get the dropdown menu, and set the following:

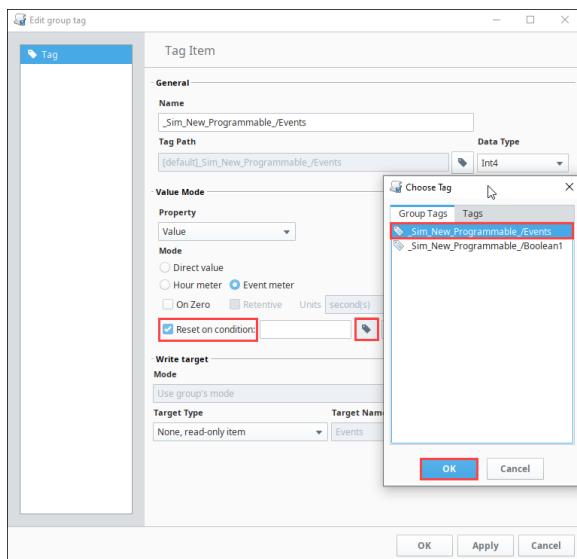
- For the boolean Tag, select the memory Tag from the dropdown which is set previously as the target Tag to write the hour meter to.
- For the memory Tag, select Read-only from the dropdown.

- Click **Enabled** at the top of the page, and do a **File > Save All** to start the group.
- Make the boolean Tag true to start the Event meter. You'll see the Event Tag update in the Tag Browser.

Reset an Hour or Event Meter Based on a Condition

You can set the hour or event meter based on a condition.

- In the Basic OPC/Group Items section, right-click and Edit a Tag that is serving as the Hour or Event meter. The Edit group tag window is displayed.
- In the **Value Mode** area, select the **Reset on condition** check box.
- Click the **Tag** icon to display the **Choose Tag** window, and select a Group Tag from the popup window.
- Click **OK**.

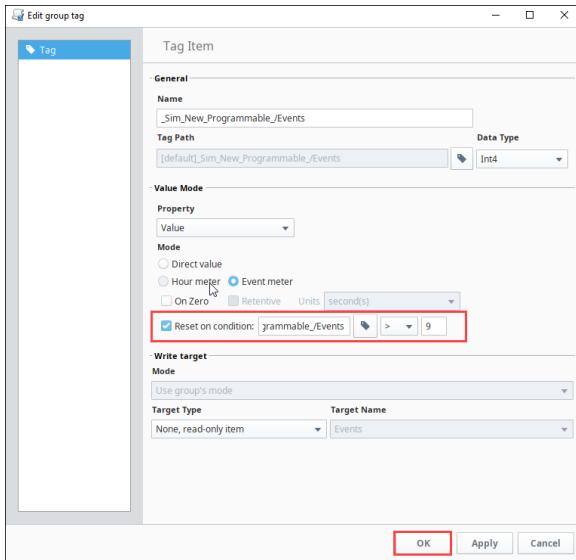



**INDUCTIVE
UNIVERSITY**

Resetting Hour and Event Meters

[Watch the Video](#)

- Next to the **Tag** icon, choose the **operator** sign (for example **>**), and enter a number. In the example we entered **9**.



6. Click **OK**. The target Tag will now reset in response to the condition (after nine occurrences in our example).

Next...

- [Trigger Options](#)

Transaction Group Examples

Transaction Groups

There are four basic types of Transaction Groups that can be used in Ignition:

- **Standard:** The heart of bi-directional data storage and management
- **Historical:** Simple historical trending
- **Block:** Efficient large scale data storage
- **Stored Procedure:** Interact with existing protected data systems

This Section has examples for each type of group and shows the different ways that you can use them. For a more complete understanding of how the parts of each group works, see [Understanding Transaction Groups](#).

On this page ...

- [Transaction Groups](#)
- [Standard Group](#)
- [Historical Group](#)
- [Block Group](#)
- [OPC to OPC Interaction](#)

Standard Group

The Standard Group is the most flexible group. It is commonly used as a bi-directional sync between your PLCs and databases. In addition to this, it can also be used to push data in either direction. This means the Standard Group can be used to store historical data, [add to/update existing tables](#), create [recipe management tools](#), and more.

The screenshot shows the Ignition configuration interface for a Standard Group named "Bi Directional". The top bar displays the group name, status (Enabled), and mode (Disabled). Below the header are three tabs: "Basic OPC/Group Items (8)", "Run-Always Expression Items (ignore trigger) (0)", and "Triggered Expression Items (0)". The first tab contains a table with columns: Item Name, Source..., Lat..., Mode, Target Na..., Data T..., Proper... . The table lists eight items: accumul..., ambient..., ambient..., discharge..., discharge..., receiver..., and valveDis... . The second and third tabs are currently empty. To the right of the tabs is a configuration panel titled "Action" which includes sections for "Execution Scheduling" (set to Timer, 1 second(s)), "Update mode" (Bi-directional OPC wins), "Data source" (<Default>), "Table name" (group_table), and various checkboxes for table management like "Automatically create table" and "Store timestamp to".

Historical Group

The Historical Group is the most straightforward and simplest to use. It will take OPC data and store it as history in a database.

New Historical Group

Execution Disabled [Save project to apply changes](#)

► Enabled Disabled || Pause

Basic OPC/Group Items (10)

Item Name	Source ...	Latche...	Target Name	Data Type	Properties
Realistic0	N/A	N/A	Realistic0	Float8	
Realistic1	N/A	N/A	Realistic1	Float8	
Realistic2	N/A	N/A	Realistic2	Float8	
Realistic3	N/A	N/A	Realistic3	Float8	

Run-Always Expression Items (ignore trigger) (0)

Item Name	Source ...	Latche...	Target Name	Data Type	Properties
-----------	------------	-----------	-------------	-----------	------------

Triggered Expression Items (0)

Item Name	Source ...	Latche...	Target Name	Data Type	Properties
-----------	------------	-----------	-------------	-----------	------------

Action **Trigger** **Options**

Execution Scheduling:

Timer Schedule

1 second(s)

Data source: <Default>

Table name: group_table

Automatically create table

Use custom index column: group_table_ndx

Store timestamp to: t_stamp

Store quality code to: quality_code

Delete records older than: 1 day(s)

Block Group

The Block Group is used to efficiently store large amounts of data in blocks or chunks of similar data in the database. This is very useful if you have many devices with the same Tags in them.

New Block Group

Execution Disabled [Save project to apply changes](#)

► Enabled Disabled || Pause

Item View **Block View**

Block Items (3)

Item Name	So...	Lat...	Mode	Targe...	Data ...	Pr...	Size
Item_Ramp0	Use g...		Ramp0	Float8			10
Item_Realistic0	Use g...		Realis...	Float8			10
Item_Sine0	Use g...		Sine0	String			10

Basic OPC/Group Items (0)

Item Name	Source...	Latche...	Target Name	Data Type	Properties
-----------	-----------	-----------	-------------	-----------	------------

Run-Always Expression Items (ignore trigger) (0)

Item Name	Source...	Latche...	Target Name	Data Type	Properties
-----------	-----------	-----------	-------------	-----------	------------

Triggered Expression Items (0)

Item Name	Source...	Latche...	Target Name	Data Type	Properties
-----------	-----------	-----------	-------------	-----------	------------

Action **Trigger** **Options**

Execution Scheduling:

Timer Schedule

1 second(s)

Update mode: OPC to DB

Data source: <Default>

Table name: group_table

Automatically create table

Automatically create rows

Use custom index column: group_table_ndx

Store timestamp to: t_stamp

Store quality code to: quality_code

Store row id to: row_id

Store block id to: block_id

Delete records older than: 1 day(s)

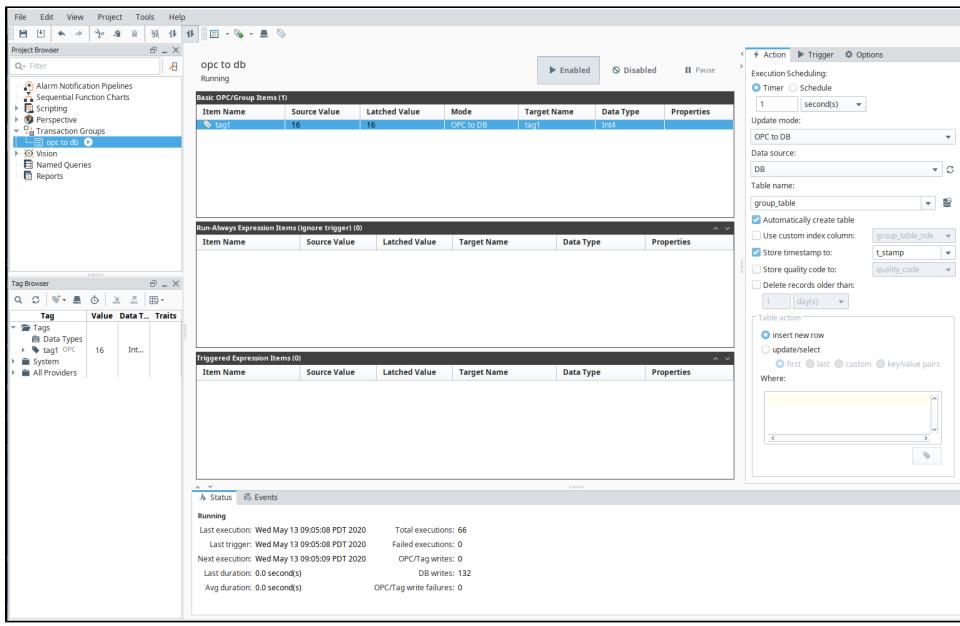
Table action:

insert new block

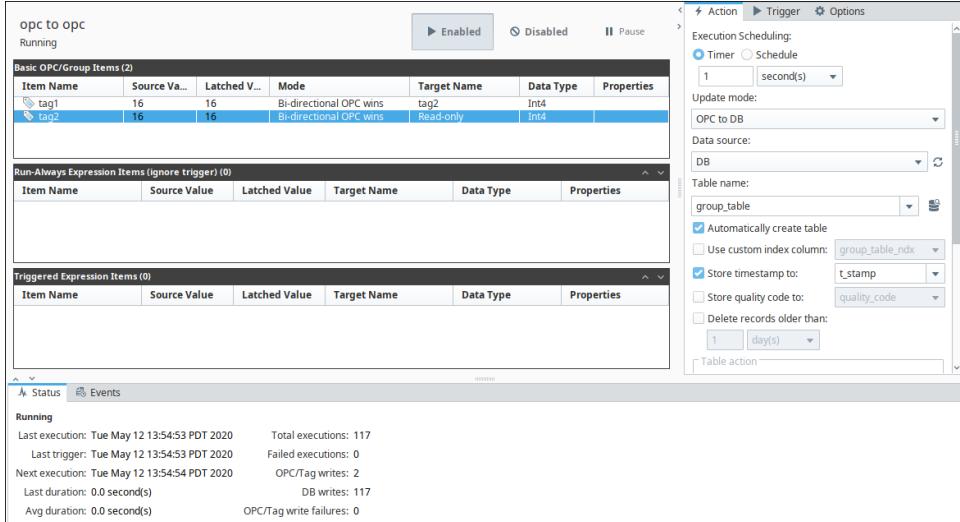
OPC to OPC Interaction

It is possible to configure your Standard Transaction Group to be able to get information from one OPC data point to another. This is useful in the event that you have Tags coming from one PLC and you need the Tag information to be sent to another PLC on your plant floor.

1. Create a **Standard Transaction Group** and from your Tag Browser, drag two Tags into your Transaction Group's Basic OPC/Group Items section. For this example, the Tags will be called 'tag1' and 'tag2' and they will be coming from two different PLC's.
2. Set the Mode on 'tag1' to '**Bi-directional OPC Wins**' and set its Target Name to be 'tag2'.
3. Set the Mode on 'tag2' to be '**Bi-directional OPC Wins**' and set its Target Name to 'Read-only.' (The Mode on 'tag2' is not as important here as it is a Read-only item, but we set it to '**Bi-directional OPC Wins**' anyway.) Your configuration should match what is shown below:



What this will do is make sure that every 1 second, the value from 'tag1' will be written to 'tag2' as below:



Related Topics ...

- [Understanding Transaction Groups](#)

In This Section ...

Block Group

The Block group is a type of Transaction Group that stores data vertically. Whereas, a Standard group stores the information horizontally in a single row. Block groups share many of the same features as the Standard group. They can be bidirectional, insert into a database, or simply update the database. All the rows in a Block group are associated with a single database transaction therefore the process of writing to the database is very efficient.

On this page ...

- [Create a Block Group](#)
 - [DB to OPC Mode with Custom Where Clause](#)
 - [Next...](#)



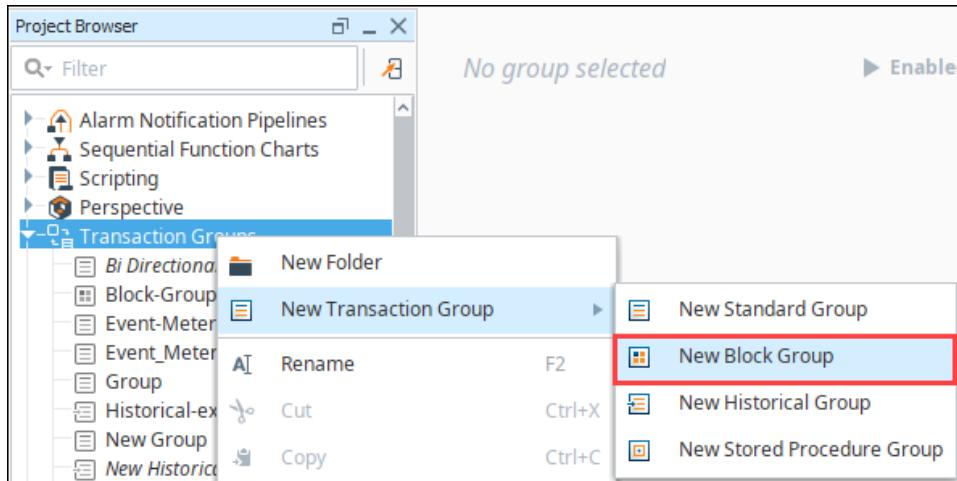
INDUCTIVE
UNIVERSITY

Block Group Type

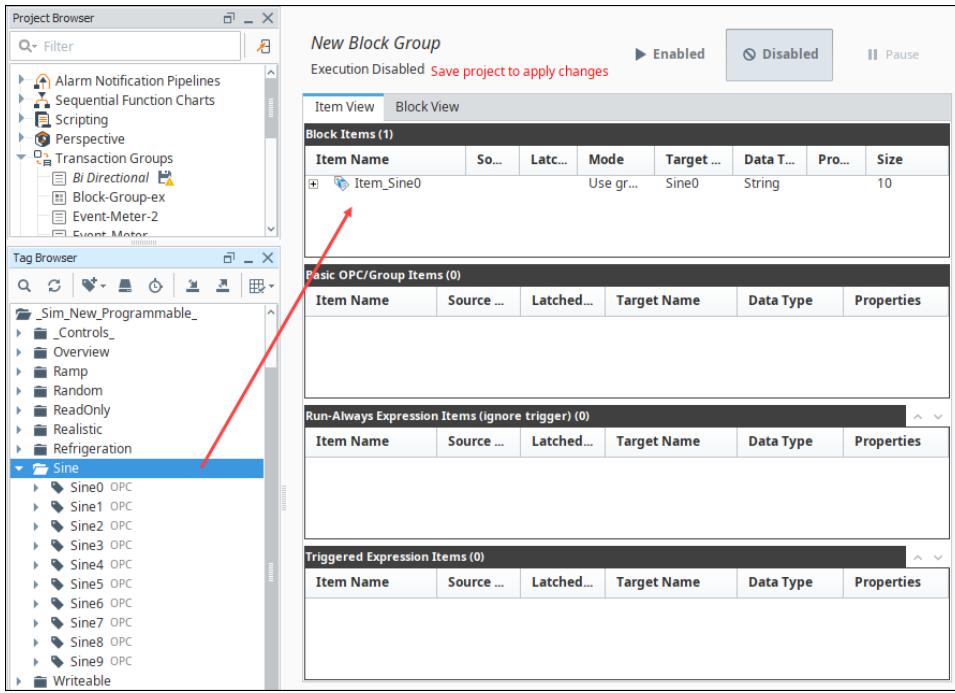
[Watch the Video](#)

Create a Block Group

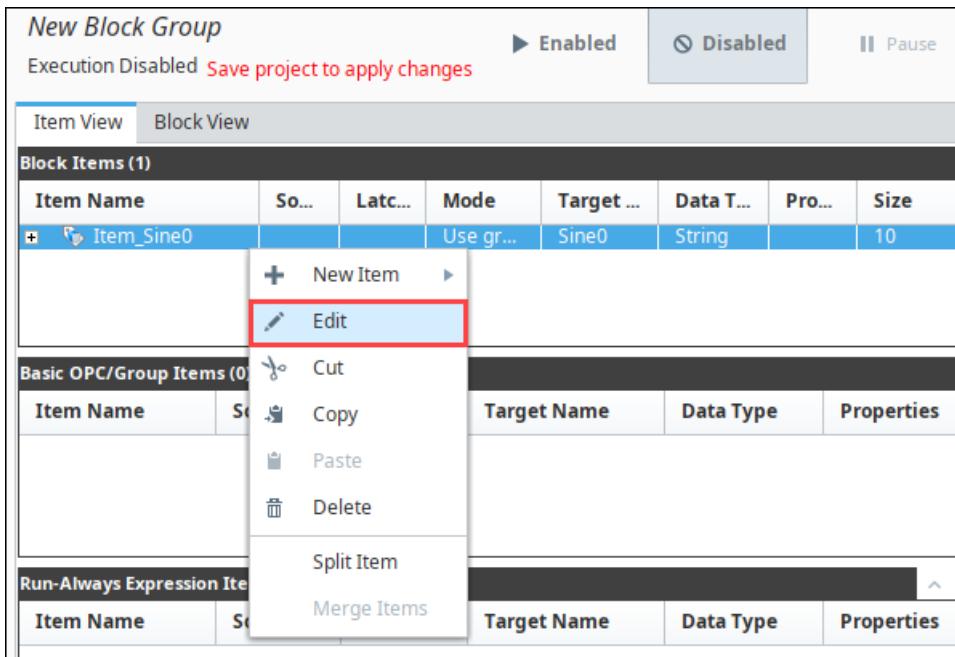
1. In the Project Browser, right-click on Transaction Groups and select **New Transaction Group > New Block Group**.



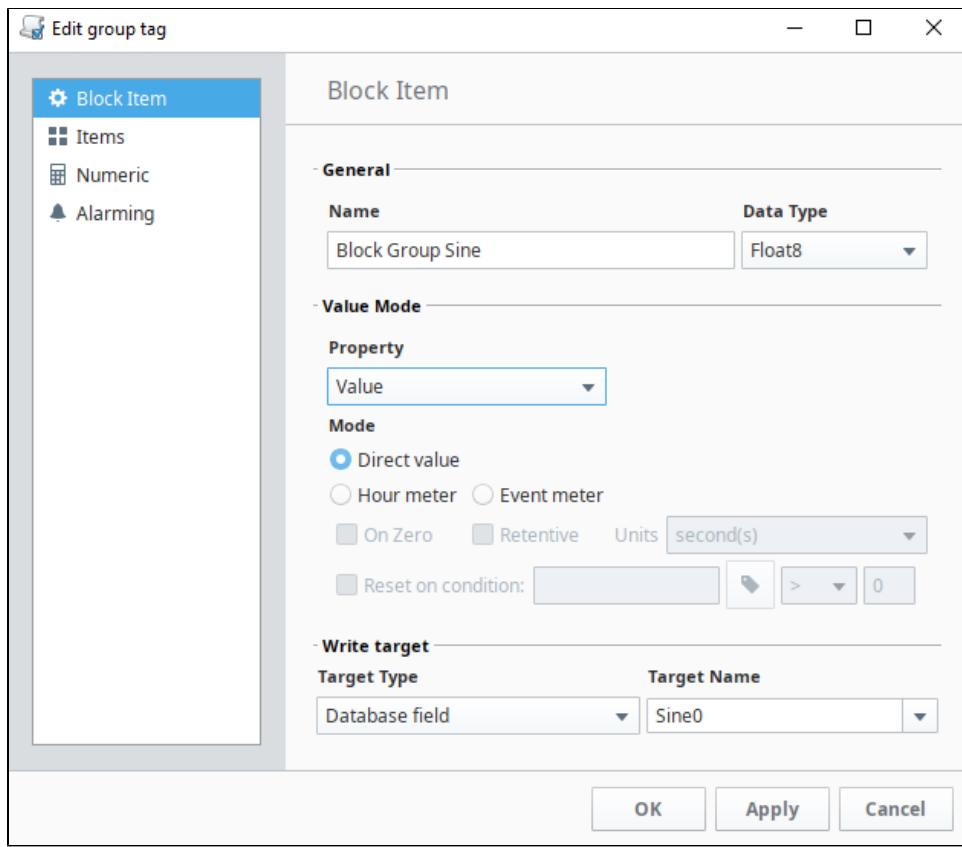
2. Give the group a name and click **Create Group**.
3. Drag a Tag folder into the **Block Items** section of the new Transaction Group.



4. Select the item in the Block group, right-click and select **Edit**.



5. Change the **Name**, enter the **Target Name** to anything appropriate. Click **OK**.



6. Configure the remainder of the group settings under the **Action** tab.

Action **Trigger** **Options**

Execution Scheduling:

Timer Schedule
1 second(s)

Update mode:

OPC to DB

Data source:

<Default>

Table name:

group_table

Automatically create table
 Automatically create rows
 Use custom index column: group_table_ndx
 Store timestamp to: t_stamp
 Store quality code to: quality_code
 Store row id to: row_id
 Store block id to: block_id
 Delete records older than:
1 day(s)

Table action

insert new block
 insert changed rows
 update/select
 first last custom

Where:

7. Select the group, and click **Enabled**.
8. **Save** the project to start the group.

DB to OPC Mode with Custom Where Clause

Like the Standard Group, block groups can be configured to retrieve records from the database, writing back to an OPC address or Tag. When using a custom WHERE clause, you can write the WHERE statement in such a way that multiple rows are returned, which would then update multiple items, which in turn write back to OPC addresses. We could then add a dynamic OPC value as a "lookup" that would determine which set of rows to return.

This is a great way to retrieve multiple datapoints that are stored in a tall format on a database table, ideally when you're looking to retrieve multiple sequential rows. For example a table with the following content, a single block item targeting the "itemValue" column, and a "lookup" Tag or OPC item that the group will use in the WHERE clause.

Table structure

table_ndx	itemValue
1	1
2	20
3	300
4	4,000
5	50,000
6	600,000
7	7,000,000

Our block item

The screenshot shows a configuration interface for a block item. At the top, there is a status bar with 'Block' and two radio buttons: 'Enabled' (selected) and 'Disabled'. Below this is a navigation bar with 'Item View' (selected) and 'Block View'. The main area is titled 'Block Items (1)' and contains a table with the following columns: Item Name, Source..., Latche..., M..., and Target Name. A single row is shown, labeled 'itemValue'. The 'Source...' column shows '[default]Block Group/itemValue1', '[default]Block Group/itemValue2', and '[default]Block Group/itemValue3'. The 'Target Name' column shows 'itemValue'. There is also a small icon of a gear next to the 'itemValue' label.

Our Tags, including "lookup"

Tag	Value	Data Type	Traits
Tags			
Data Types			
_Generic_Simulator_			
Alarms			
Block Group			
itemValue1 Memory	0	Integer	
itemValue2 Memory	0	Integer	
itemValue3 Memory	0	Integer	
lookup Memory	0	Integer	

1. Set the "Update mode" for the group to "**DB to OPC**."
2. Set the Table action (under the "Action" tab) to "**update/select**."
3. Select the "**custom**" radio button.
4. Under the "**Where:**" text area, click the Tag icon, and select the "**lookup**" Tag, which adds a reference to the Tag like this: {[default]Block Group/lookup}
5. Write the rest of the our condition. In this case, we'll say we want results from our table starting a value greater than our lookup value. Using the table specified above, we could write the following condition:

```
null_table_test_ndx > {[default]Block Group/lookup}
```

6. **Enable** the group, and **save** the project.

When the group is running, with an initial lookup value of 0, the group automatically grab table_ndx of 1, and write a value of 1 (from the first row) to itemValue1, a value of 20 (from the second row) to itemValue2, and so on.

Block Group			
itemValue1 Memory	1	Integer	
itemValue2 Memory	20	Integer	
itemValue3 Memory	300	Integer	
lookup Memory	0	Integer	

If we set the value on lookup to 3, that means the first row in the result set will be row 4, setting itemValue1 to 4000, itemValue2 to 50,000, and so on.

Block Group			
itemValue1 Memory	4,000	Integer	
itemValue2 Memory	50,000	Integer	
itemValue3 Memory	600,000	Integer	
lookup Memory	3	Integer	

If you set the value of lookup to 6, then that will set the value on itemValue1 to row 7's value (7,000,000), but you'll notice the other Tags are retaining a value, which is notable since those items don't have a corresponding value to retrieve.

Values on our Tags

Block Group			
itemValue1 Memory	7,000,000	Integer	
itemValue2 Memory	50,000	Integer	
itemValue3 Memory	600,000	Integer	
lookup Memory	6	Integer	

Items in the group

Block Items (1)								
Item Name	Source Val...	Latched V...	Mo...	Target Name	Da...
itemValue				itemValue	Int4			3
[default]Block Group/itemValue1	7000000	7000000						
[default]Block Group/itemValue2	50000	50000						
[default]Block Group/itemValue3	600000	600000						

This is expected. By default, when a Block Group is configured like this, and some items can't receive updated values as a result of the dynamic WHERE clause not returning enough rows, the items will retain their previous latched value: that is to say, the group will not automatically clear or reset the values on the other items. Refer to [Set NULL DB Values to Default](#).

Next...

- [Recipe Group](#)

Recipe Group

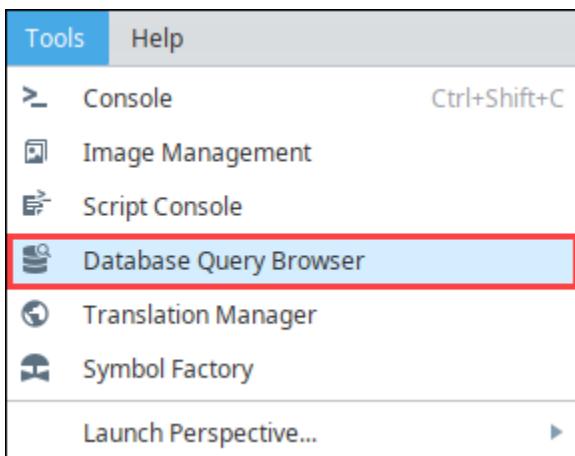
You can use Transaction Groups to create a recipe management system which will pull recipe information from the database and push it to the PLC when requested. With this system, the Transaction Group is what queries the database rather than writing scripts to handle it all.

Before the Transaction Group

Before we make the Transaction Group, we first need to make sure we have a table set up in our database that holds recipes. If you already have this, then you can skip to the next step on making the Transaction Group.

We will make a table in our database that will hold our recipes. Our recipes will be simple, containing a name, unique id, and two setpoints, so we will need a column for each of those values.

1. Verify the Designer's **Comm Mode** is set to Read/Write, and open up the [Database Query Browser](#).



2. Execute the query below in the Database Query Browser to create the table we'll use in this example:

```
CREATE TABLE recipes(
    id INT PRIMARY KEY,
    recipe_name VARCHAR(50),
    setpoint1 FLOAT,
    setpoint2 FLOAT)
```

Note: This query was designed for an MSSQL database. If you are connected to a different database, the syntax on the CREATE statement may differ. Check your database's documentation for more details.

3. Next we need to put some data into the table by using an `insert` statement. Execute the below query to insert a new record into our recipes table:

```
INSERT INTO recipes (id, recipe_name, setpoint1, setpoint2)
VALUES (1, 'Recipe 1', 10, 0)
```

You can rerun this query as many times as you want, incrementing the id to give you a new unique id, changing the name, and providing different setpoints. Your table might look something like the one below.

id	recipe_name	setpoint1	setpoint2
1	The First Recipe	34.7	54.1
2	The Wrong Recipe	12.8	42.3



Recipe Group

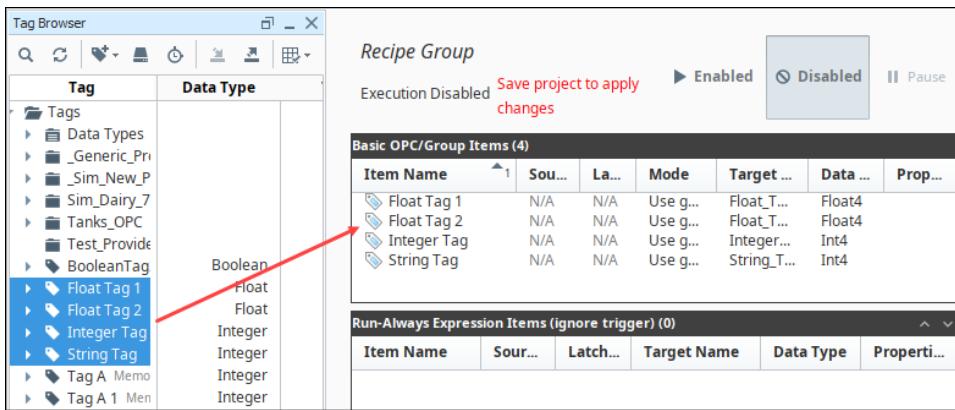
[Watch the Video](#)

3	The Best Recipe	65.7	95.1
4	The Other Recipe	49.8	112.2

Create the Transaction Group

Now that we have a recipe table in the database that is populated with some records, we can create the **Transaction Group** that will load a recipe from the table into our Tags. We will be using the recipes table that we put together previously, but if you already had a table, you can use that here instead.

1. Create a new **Standard Transaction Group**.
2. We have four columns in our database table, so we will need four Tags to use in the Transaction Group: an integer, string, and two floats for the id, name, and setpoints respectively. Add the four Tags to the Transaction Group.



3. Set the Table Name to '**recipes**', the table that we created earlier.
4. We then need to ensure that our Tags will be receiving the proper values from the database.
 - a. Set the Target Names for each of the Tags: the string to '**recipe_name**', the floats to '**setpoint1**' and '**setpoint2**'
 - b. Set integer to '**Read Only**'. We don't need to set the integer to the id column, because we will not pull the id from the database, but rather use the id as a trigger and in the where clause.

Item Name	Source ...	Latche...	Mode	Target Name	Data Type	Properties
Integer Tag	N/A	N/A	—	Read-only	Int4	
String Tag	N/A	N/A	—	recipe_name	String	
Float Tag 1	N/A	N/A	—	setpoint1	Float4	
Float Tag 2	N/A	N/A	—	setpoint2	Float4	

5. Now we can finish setting up the rest of the Transaction Group. Set the Update mode to **DB to OPC**.
6. Set the **Table Action** to **Update>Select** using **Key/Value Pairs** with the **Column** set to **id**, and the **Value** set to the **Integer Tag** you are using.

Action Trigger Options

Execution Scheduling:

Timer Schedule

1 second(s)

Update mode:

DB to OPC

Data source:

<Default>

Table name:

recipes

Automatically create table

Use custom index column: recipes_ndx

Store timestamp to: t_stamp

Store quality code to: quality_code

Delete records older than: 1 day(s)

Table action

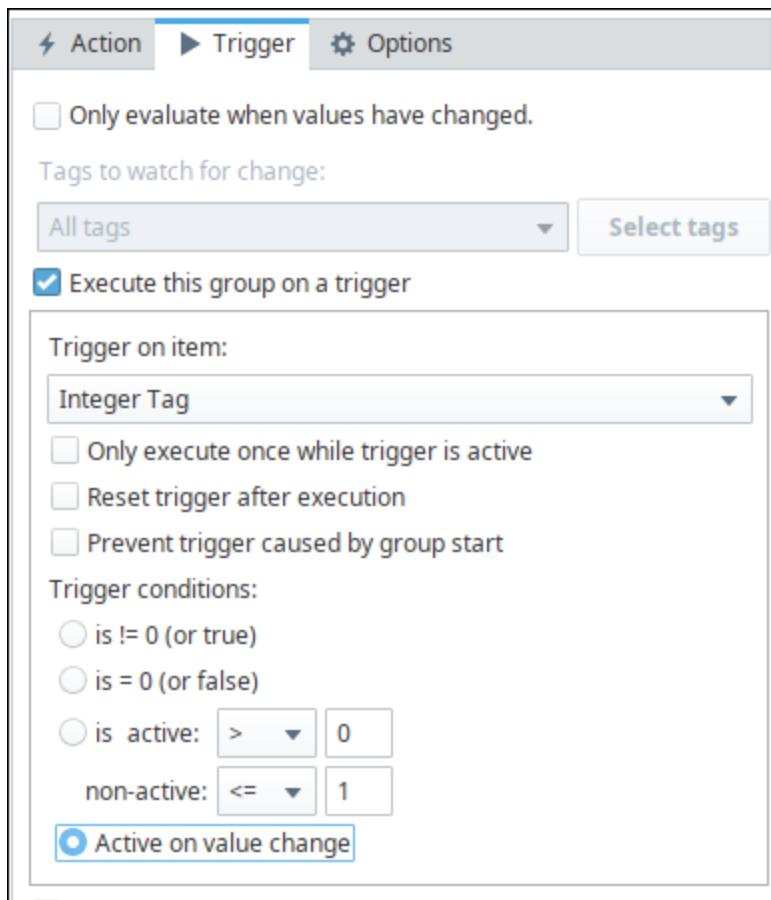
insert new row

update/select

first last custom key/value pairs

Column	Value
id	Integer Tag

7. Set the Update Rate to **1 second**. We want to query the values out of the database as soon as we ask for them, so we need the group to update quickly. However, we don't want the group to actually query the database every second, so we will need to set up the trigger.
8. Go to the Trigger tab, and select **Execute this group on a trigger**. Trigger on the item the int Tag that is being used for the id. Specify the Trigger condition as **Active on value change**.



9. Finally, **Enable** the Transaction Group and **save** the project to get it started. The Transaction Group will now pull the recipe out of the database where the id matches the value of the int Tag. The trigger also prevents it from running all the time, instead running only when the int Tag value changes.
10. To test it out, simply change the value of id Tag to an id of one of the recipes in the recipes table.

Update or Insert Group

You can update a row or insert a new row into the database when a key pair combination does not exist. This eliminates the need to have a database that has every possible option considered in its original design. Because of the **insert row when not present** setting, the group will insert a new record whenever the designated ID doesn't exist. Afterwards, it will update the rows in the table that are associated with the key/value references as shown in this example.



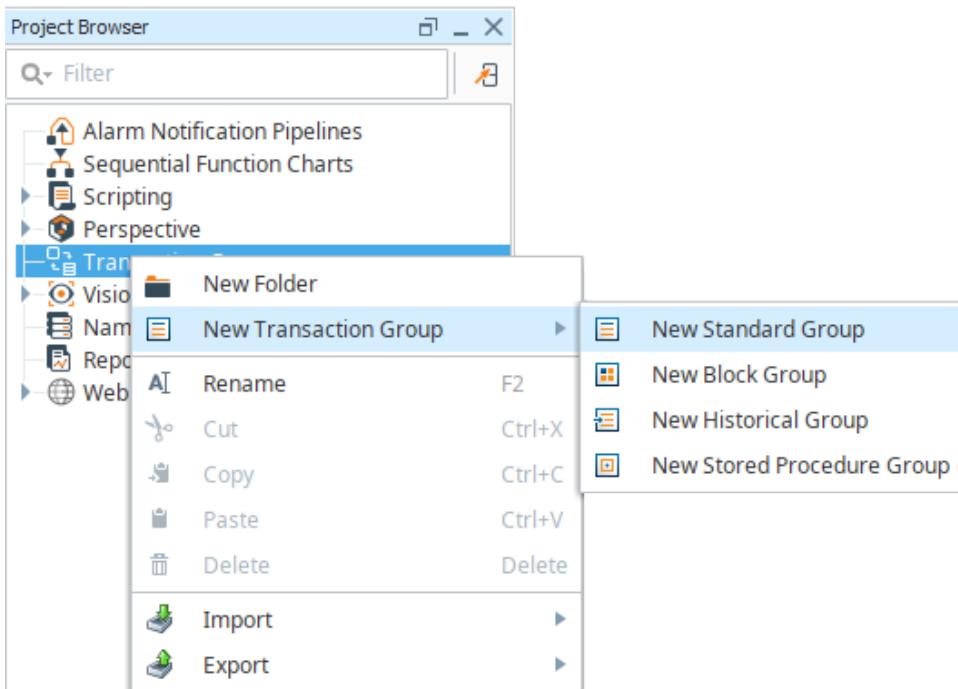
INDUCTIVE
UNIVERSITY

**Update or Insert
Group**

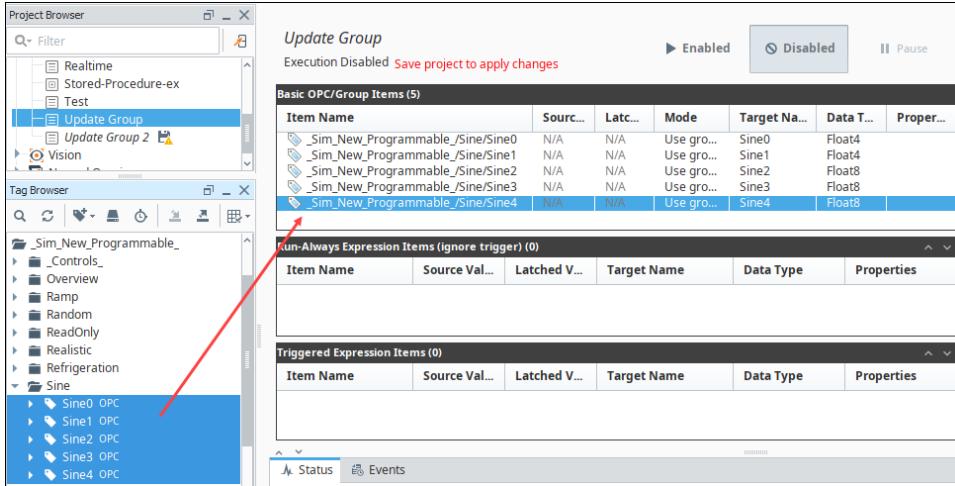
[Watch the Video](#)

Update or Insert a New Row into the Database

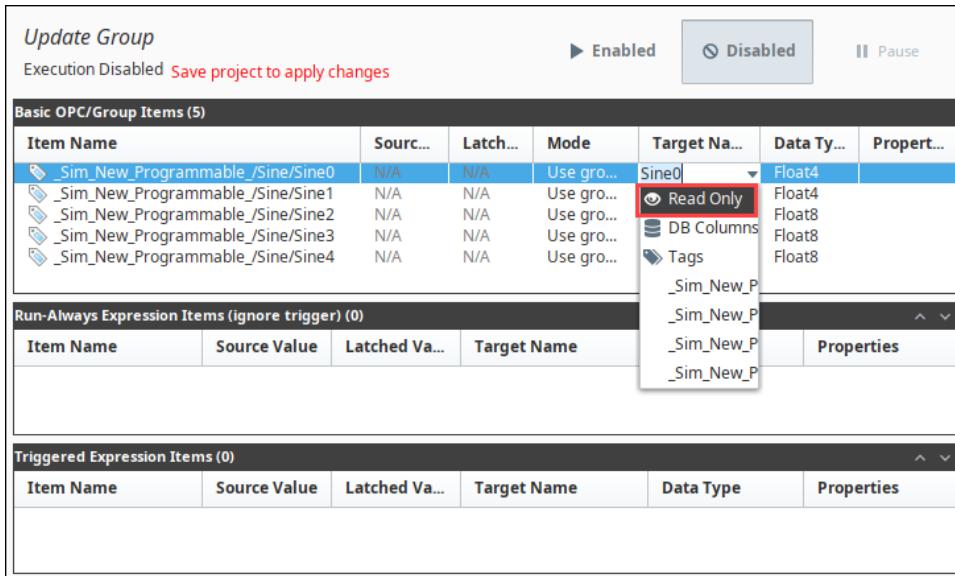
1. In the Project Browser, right-click on Transaction Groups and select **New Standard Group**.



2. Give the group a name and click **Create Group**.
3. Drag a group of Tags into the groups **Basic OPC/Group Items** section.



4. Change one of the Tags to be read-only by selecting **Read Only** from the Tag's **Target Name** column.



5. In the group's **Action** tab, in the **Table action** area, select the **update/select** radio button and the **key/value pairs** radio button.

Action **Trigger** **Options**

Execution Scheduling:

Timer Schedule
 1 second(s)

Update mode:

OPC to DB

Data source:

<Default>

Table name:

group_table

Automatically create table

Use custom index column: group_table_ndx

Store timestamp to: t_stamp

Store quality code to: quality_code

Delete records older than:
 1 day(s)

Table action

insert new row
 update/select
 first last custom key/value pairs

Column	Value
	null

Add **Remove** Insert row when not present

6. Click the **Add** icon.
 - a. For the **Column** select the database table ID column.
 - b. For the **Value** column, select the read-only Tag.
 - c. Select **Insert row when not present** check box at the bottom of the **Table action** area.
7. Select the group, and click **Enabled**.
8. Save the project to start the group.

Next...

Trigger Options

It is often useful to execute a group only when a certain condition is met or as a bit turns on or off. Triggers allow Transaction Groups to run based on values changing in various ways.

Execute on Value Change

A group can execute when the group's Tags have changed, or when a particular Tag within the group has changed. In either case, the Transaction Group will execute every time the value or values change.

1. In the **Trigger** tab, select at the very top the **Only evaluate when values have changed** checkbox.
Now the group will execute if any of the Tags change.
2. To execute when only one Tag changes, from the **Tags to watch for change** dropdown, select **Custom**, click on **Select Tags**, select the Tag from the pop-up window, and click **OK**. You can select more than one Tag at a time in order to monitor more than one Tag for value changes.
3. From the **Trigger on item** dropdown, select the appropriate Tag to execute the Tag on a trigger.
4. Select the **Active on value change** radio button.
5. **Save** the Project to start the Transaction Group.

On this page ...

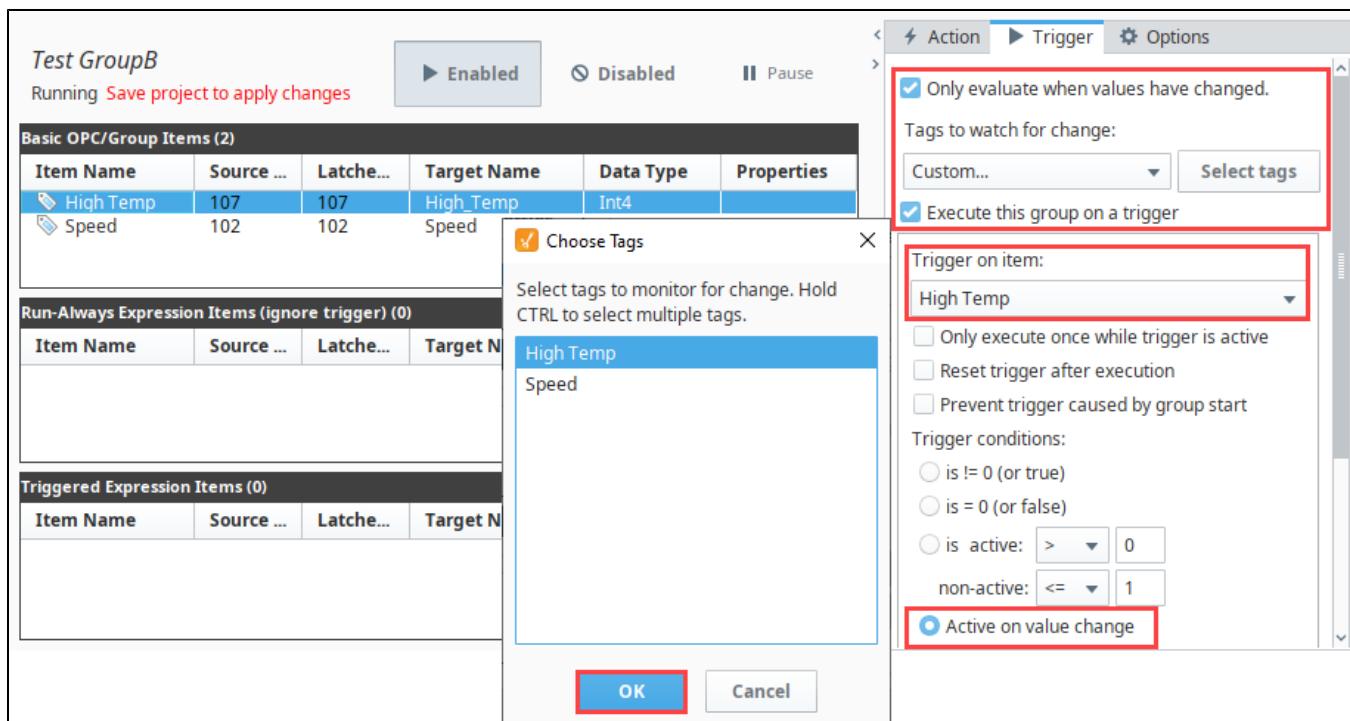
- Execute on Value Change
- Execute while Condition Is True
- Execute on a Rising Edge
- Reset Trigger
- Handshakes
- Next...



INDUCTIVE
UNIVERSITY

Trigger – On Value Change

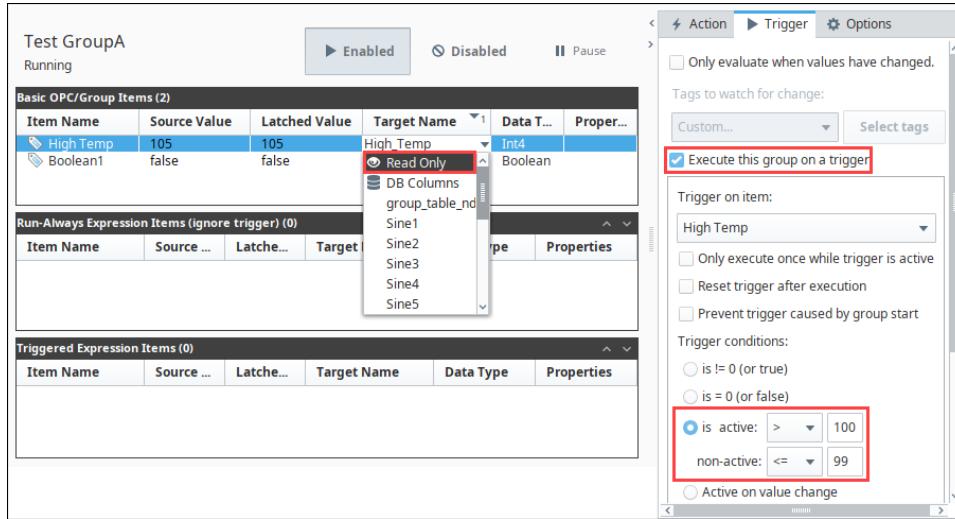
[Watch the Video](#)



Execute while Condition Is True

Groups can execute while a condition is true resulting in the Transaction Group continuing to execute for the duration of this condition.

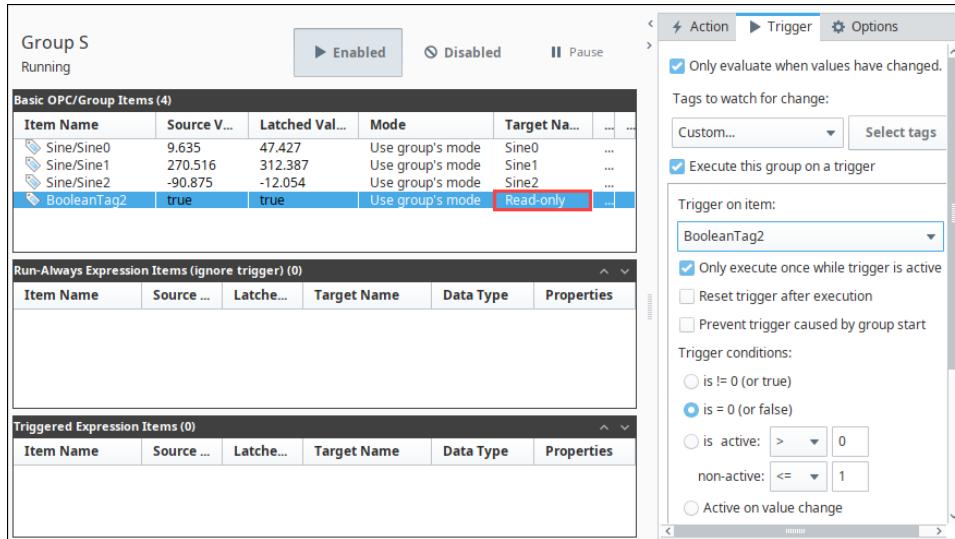
1. Create a Transaction Group, and drag a numeric or boolean Tag into the **Basic OPC/Group Items** section.
2. From the **Target Name** column dropdown, select **Read Only**.
3. Go to the **Trigger** tab, and select the **Execute this group on a trigger** checkbox.
4. In the **Trigger conditions** area, set the trigger conditions which will determine under what condition the group executes.
5. **Save** the Project to start the Transaction Group.



Execute on a Rising Edge

Groups can execute when the trigger becomes True. This is known as a **rising edge trigger** and it will only execute once and will not execute again until the trigger repeats the same cycle.

1. Create a standard Transaction Group with any number of Tags as long as one of them is a boolean Tag that will serve as the trigger for the group.
2. Set the **Write Target** for the boolean Tag to **Read-only** by selecting read-only from its drop down in the Target Name column.
3. Go to the **Trigger** tab and select the check box to **Execute this group on a trigger**. Select the boolean Tag from the drop down menu and select to have the group only **execute once while the trigger is active**.
4. **Save** the Project to start the Transaction Group.

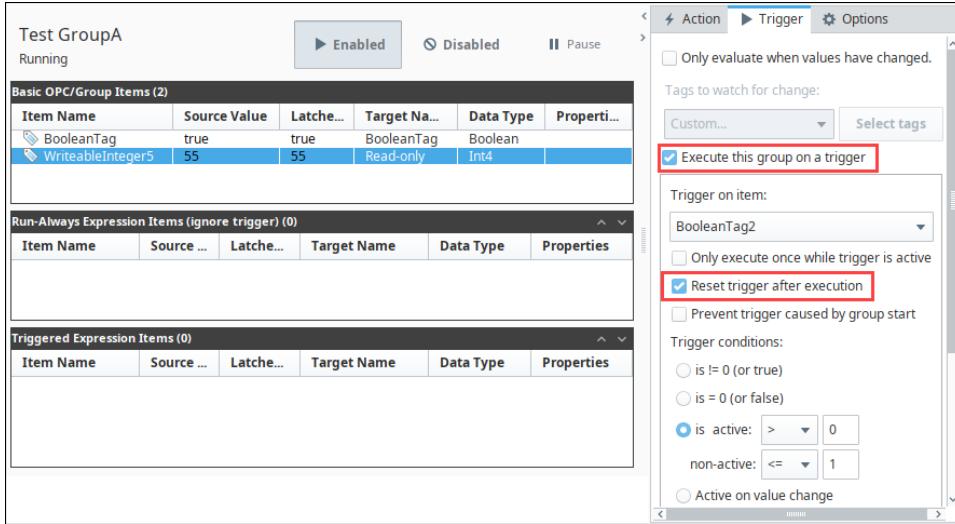


Reset Trigger

Resetting a trigger after execution of a triggered Transaction Group will result in the Transaction Group writing once to the targets followed by writing back to the trigger to reset it.

To reset the trigger after execution:

1. Create a Transaction Group with a boolean Tag. The write target for this Tag should be read only.
2. Select the **Trigger** tab and select the **Execute this group on a trigger** check box.
3. Select the **Reset trigger after execution** check box.
4. **Save** the Project to start the Transaction Group.

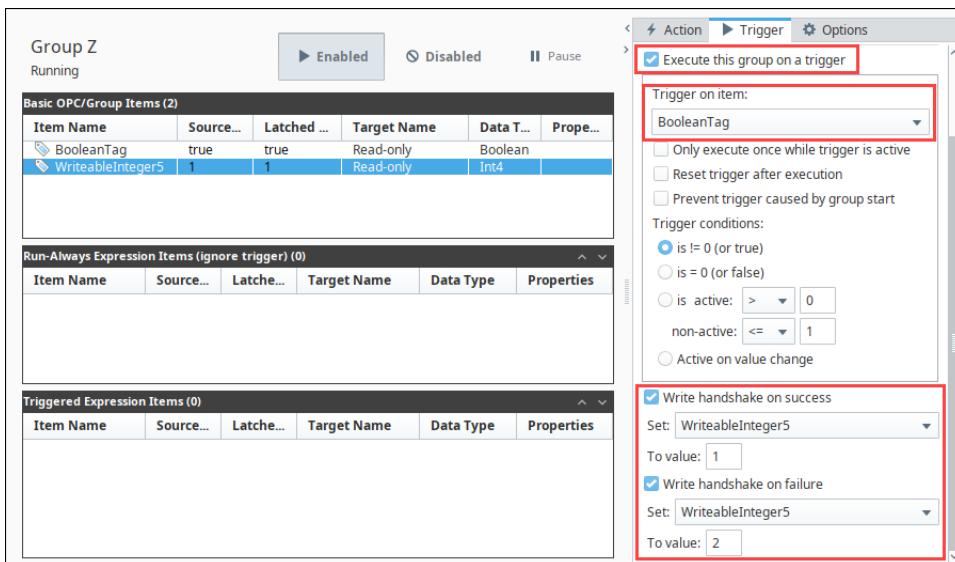


Handshakes

When a group executes, it either completes successfully or an error prevents its execution. The outcome of an execution can be handled in the handshake section of the trigger section of the Transaction Group. When a group executes successfully or fails to execute, the handshake can write a value back to a Tag to alert the user that the group executed successfully or unsuccessfully.

To set handshake values for alerting the user:

1. Create a Transaction Group with a boolean Tag and a numeric Tag.
2. Set the boolean and the numeric Tag to read only.
3. Go to the **Trigger** tab and choose to **Execute this group on a trigger**.
4. Select the boolean Tag as the trigger in the **Trigger on item** drop down, and select the appropriate execution conditions.
5. In the bottom section, select **Write handshake on success**, select the numeric Tag to write to, and choose a number that signifies success.
6. Likewise, in the bottom section, select **Write handshake on failure**, select the numeric Tag to write to, and choose a number that signifies failure.
7. **Save** the Project to start the Transaction Group.



Next...

- Understanding Transaction Groups
- Transaction Group Examples

Transaction Group Update Modes

Transaction Groups are generally used to store OPC data into a database. Transaction Group Update Modes give users additional flexibility as to whether data should flow from an OPC server to a database or from a database to an OPC server. Additionally, it is possible to configure data to be synchronized between a database and an OPC server via Bi-directional Update Modes.

All update modes do not work for all Transaction Group types. For example, Historical Transaction Groups can only insert data to a database table and not update it. In addition, Historical Transaction Groups also cannot write back to OPC items so Bi-directional Update Mode will not be an option for users using the Historical Transaction Group type.

The different Update Modes:

- OPC to DB - Only read from the [OPC server](#) and write to the [database](#).
- DB to OPC - Only read from the [database](#) and write to the [OPC Server](#).
- Bi-directional [OPC](#) wins - Read and Write to both the [database](#) and [OPC Server](#). On group start, write [OPC values](#) to the [database](#).
- Bi-directional DB wins - Read and Write to both the [database](#) and [OPC Server](#). On group start, write [database](#) values to [OPC items](#).

On this page ...

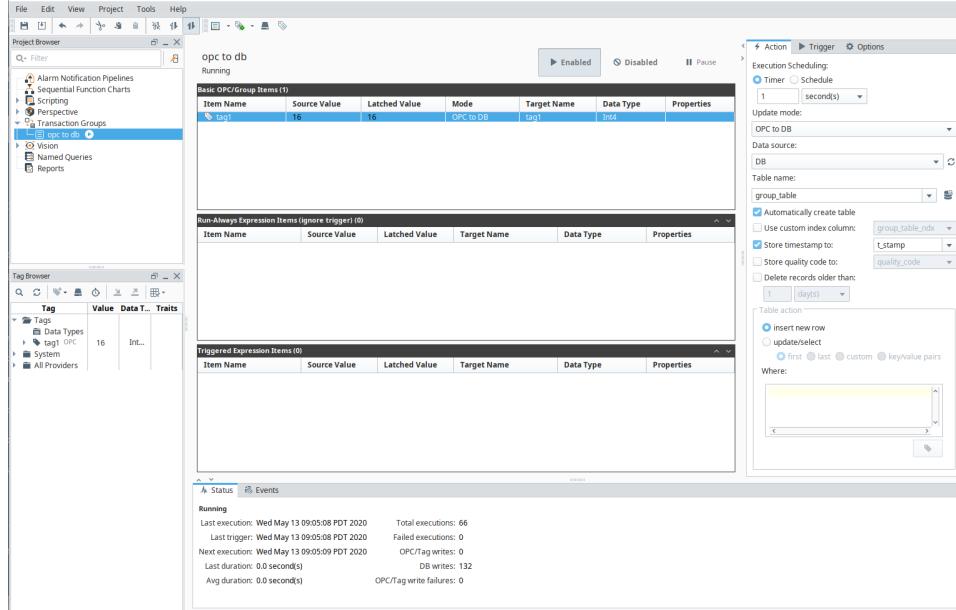
- [OPC to DB](#)
- [DB to OPC](#)
- [Bi-directional OPC Wins](#)
- [Bi-directional DB Wins](#)

OPC to DB

The **OPC to DB** Update Mode allows a Transaction Group to store OPC data to a Database Ignition that it has a connection to as shown in the following example.

1. Create a [Standard Transaction Group](#) and from your Tag Browser, drag a single tag into your Transaction Group's Basic OPC/Group Items section. For this example, the tag is called 'tag1.'
2. Set the Update Mode on 'tag1' to **OPC to DB** and set its **Target Name** to be 'tag1.' The Target Name will correlate to the name of the column in your database table where 'tag1' will be stored.

Your Transaction Group will look like the screenshot below:



This configuration will allow for tag1's value to be stored into a database table called '**group_table**' every 1 second to a column called 'tag1'.

We can see this working through the [Database Query Browser](#) as shown below:

The screenshot shows the Database Query Browser interface. In the top-left pane, there is a code editor containing the SQL query: `SELECT * FROM group_table order by t_Stamp desc`. Below the code editor is a checkbox labeled "Limit SELECT to: 1000 rows". To the right of the code editor is a large button labeled "Execute". In the bottom-right corner of the code editor area, there is a small progress bar icon.

The main area of the window is titled "Resultset 1" and displays a table with three columns: "group_table...", "tag1", and "t_stamp". The table contains 24 rows of data, ordered by "t_stamp" in descending order. The data is as follows:

group_table...	tag1	t_stamp
24	16	2020-05-13 09:04:26
23	16	2020-05-13 09:04:25
22	16	2020-05-13 09:04:24
21	16	2020-05-13 09:04:23
20	16	2020-05-13 09:04:22
19	16	2020-05-13 09:04:21
18	16	2020-05-13 09:04:20
17	16	2020-05-13 09:04:19
16	16	2020-05-13 09:04:18
15	16	2020-05-13 09:04:17
14	16	2020-05-13 09:04:16
13	16	2020-05-13 09:04:15

At the bottom of the result set, it says "24 rows fetched in 0.001s". Below the result set are four buttons: "Auto Refresh", "Edit", "Apply", and "Discard".

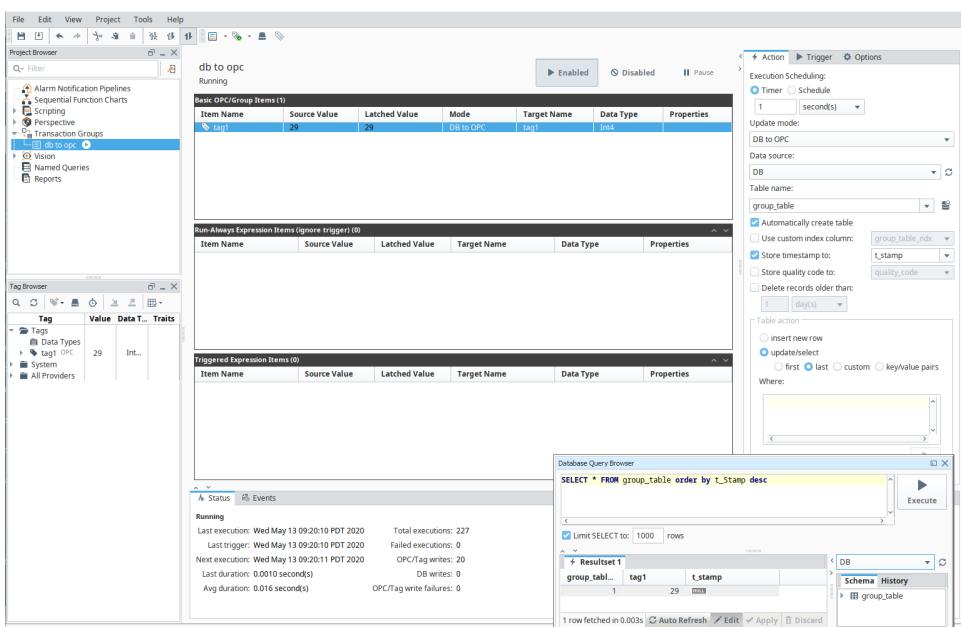
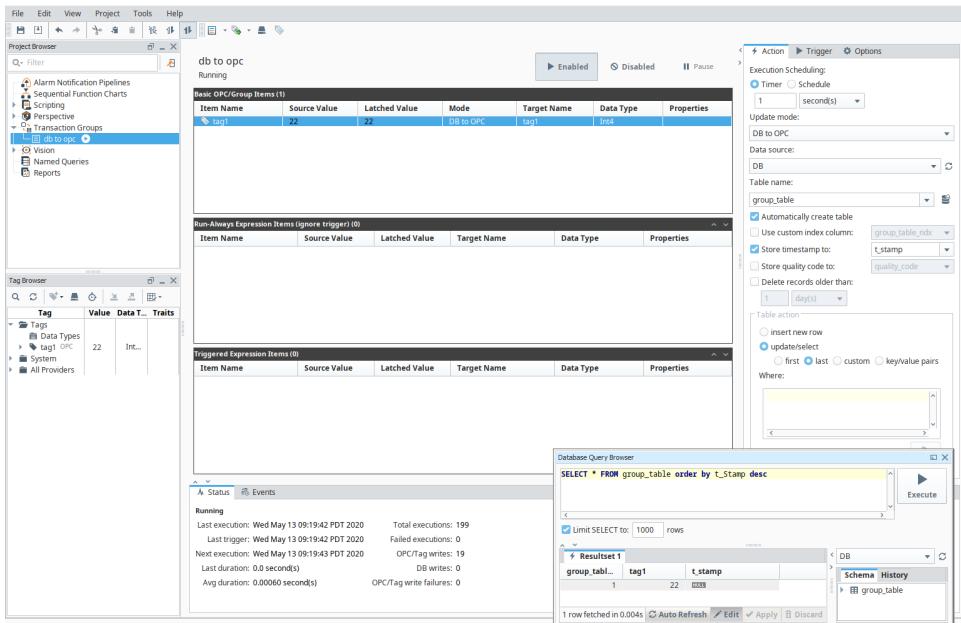
In the top-right corner of the browser window, there is a dropdown menu labeled "DB" with options "Schema" and "History". The "Schema" option is currently selected. On the left side of the window, there is a sidebar titled "DB" which lists the table "group_table".

DB to OPC

DB to OPC Update Mode allows you to write data from your Database to an OPC tag. This can be done by configuring the following:

1. Create a [Standard Transaction Group](#) and from your Tag Browser, drag a single tag into your Transaction Group's Basic OPC/Group Items section. For this example, the tag will be called 'tag1.'
2. Set the mode of the Transaction Group to **DB to OPC** and set the Mode for tag1 to **DB to OPC**.
3. Set the Transaction Groups Table Action to '[update/select](#)' and check the '**last**' option. What this will do is ensure that we do not have a new value inserted to the database. What we will have instead is a single row of data in the table group_table where the value of the 'tag1' column will control tag1's OPC value.

From the screenshots below, we can see that when the value in the [Database Query Browser](#) for column 'tag1' is 22, the value for 'tag1' is also 22. When we change the value on column 'tag1' to 29, we see tag1's value change to 29 as well.



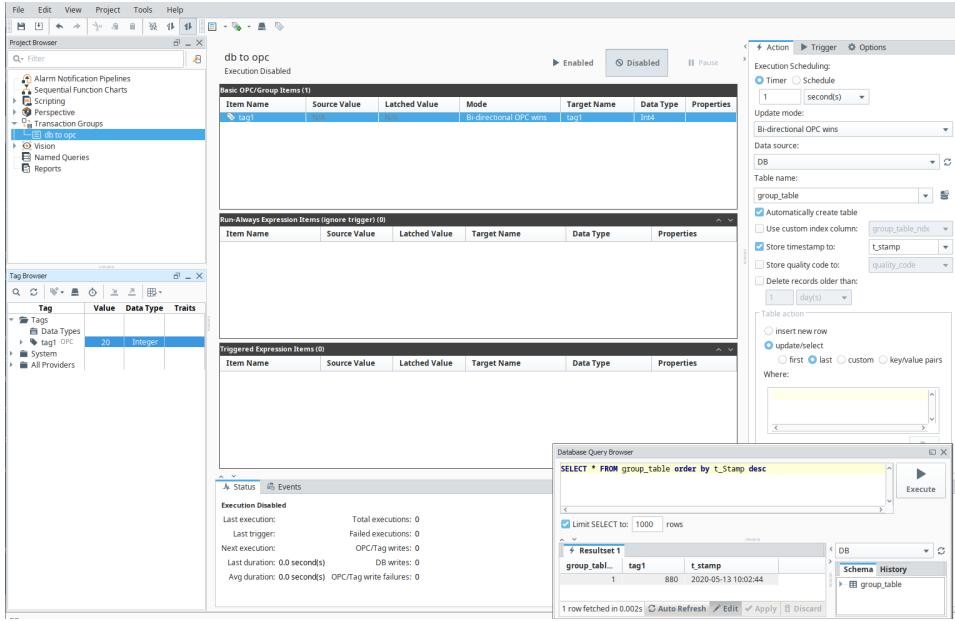
Bi-directional OPC Wins

Bi-directional OPC wins means that Ignition will Read and Write to both the database and OPC Server. However, on initial group start, if the OPC and database values are different, the OPC value will win and the Transaction Group will write OPC values to the database.

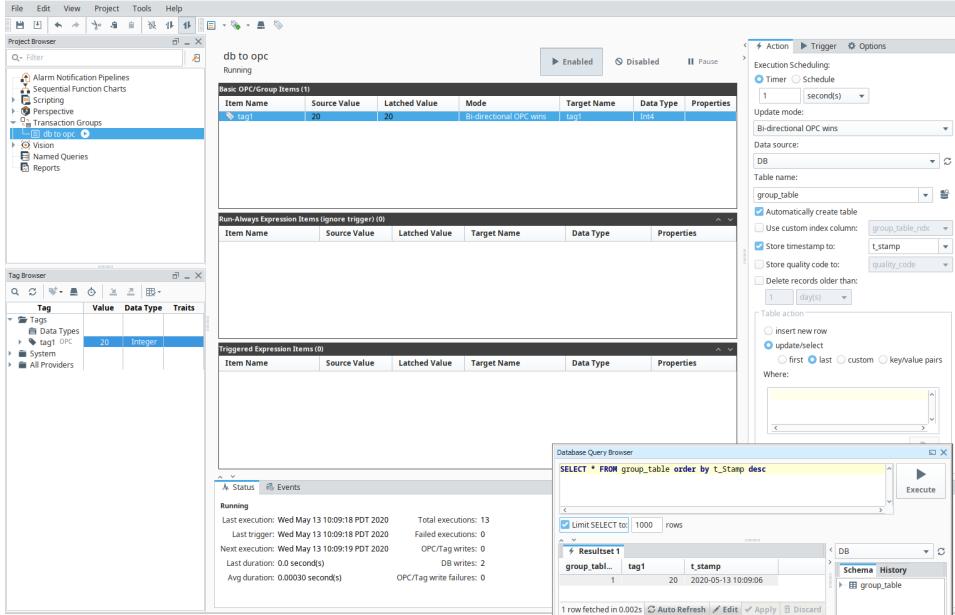
1. Create a [Standard Transaction Group](#) and from your Tag Browser, drag a single tag into your Transaction Group's Basic OPC/Group Items section. For this example, the tag will be called 'tag1'.
2. Set the mode of the Transaction Group to '**Bi-directional OPC wins**' and set the Mode for tag1 to '**Bi-directional OPC wins**'.
3. Set the Transaction Groups Table Action to '**update/select**' and check the '**last**' option. What this will do is ensure that we do not have a new value inserted to the database. What we will have instead is a single row of data in the table group_table where the value of the 'tag1' column will control tag1's OPC value and similarly, 'tag1's OPC value will control the value of the 'tag1' column database side.

What you will have at this point is a bi-directionally controlled Transaction Group where any change to tag1's value will be reflected on the database and any change database side for the 'tag1' column value will be reflected on your 'tag1' tag.

In the event that the OPC and database values do not match on Transaction Group start, the OPC value will win and it will be written to the database. This can be observed below:



Notice how the Transaction Group is disabled, 'tag1' value is 20, and the 'tag1' column value is 880. When the group is enabled, since the OPC and database values are different, the Update Mode being 'Bi-directional OPC wins' means the 'tag1' column value will be set to 20 when the Transaction Group starts.



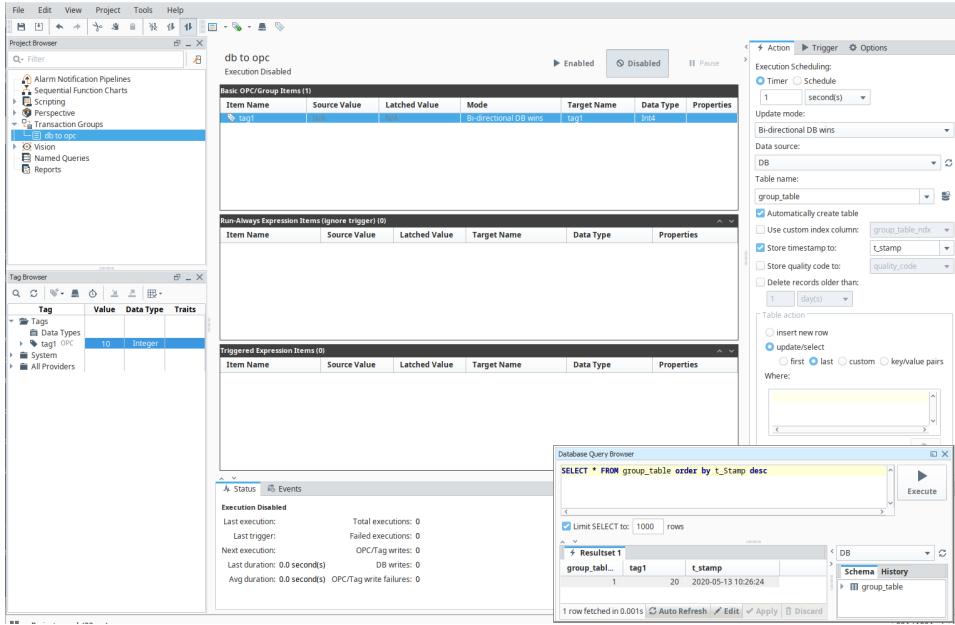
Bi-directional DB Wins

Bi-directional DB wins means that Ignition will Read and Write to both the database and OPC Server. However, on initial group start, if the OPC and database values are different, the database value will win and the Transaction Group will write database data to your OPC data points.

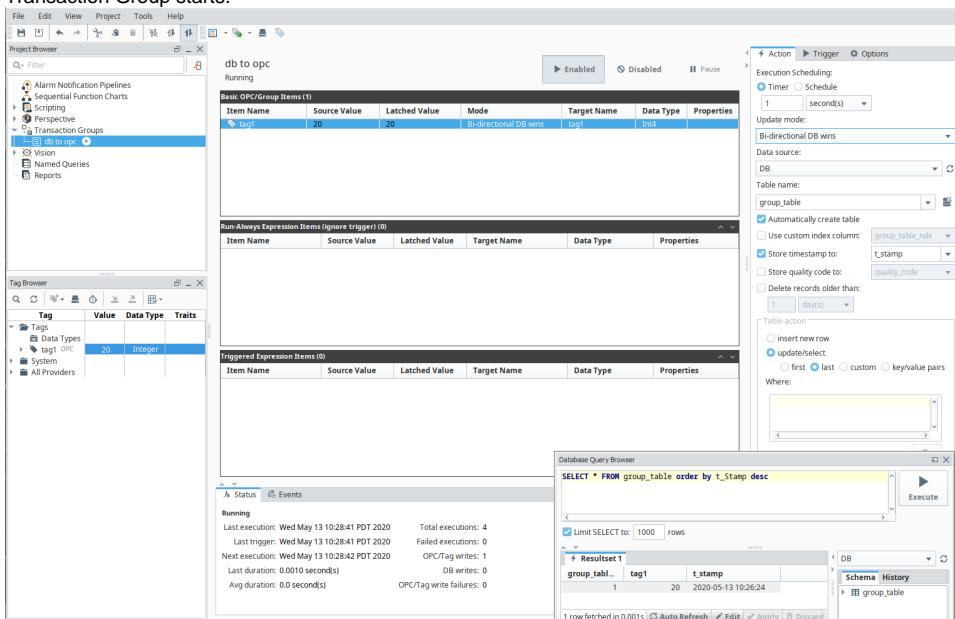
1. Create a [Standard Transaction Group](#) and from your Tag Browser, drag a single tag into your Transaction Group's Basic OPC/Group Items section. For this example, the tag will be called 'tag1'.
2. Set the mode of the Transaction Group to '**Bi-directional DB wins**' and set the Mode for 'tag1' to '**Bi-directional DB wins**'.
3. Set the Transaction Groups Table Action to '**update/select**' and check the '**last**' option. What this will do is ensure that we do not have a new value inserted to the database. What we will have instead is a single row of data in the table group_table where the value of the 'tag1' column will control tag1's OPC value and similarly, tag1's OPC value will control the value of the 'tag1' column database side.

What you will have at this point is a bi-directionally controlled Transaction Group where any change to tag1's value will be reflected on the database and any change database side for the 'tag1' column value will be reflected on your 'tag1' tag.

In the event that the OPC and database values do not match on Transaction Group start, the database value will win and it will be written to the OPC data point. This can be observed below:



Notice how the Transaction Group is disabled, 'tag1' value is 10, and the 'tag1' column value is 20. When I enable the group, since the OPC and database values are different, the Update Mode being 'Bi-directional DB wins' means the tag1 tag value will be set to 20 when the Transaction Group starts.



Related Topics ...

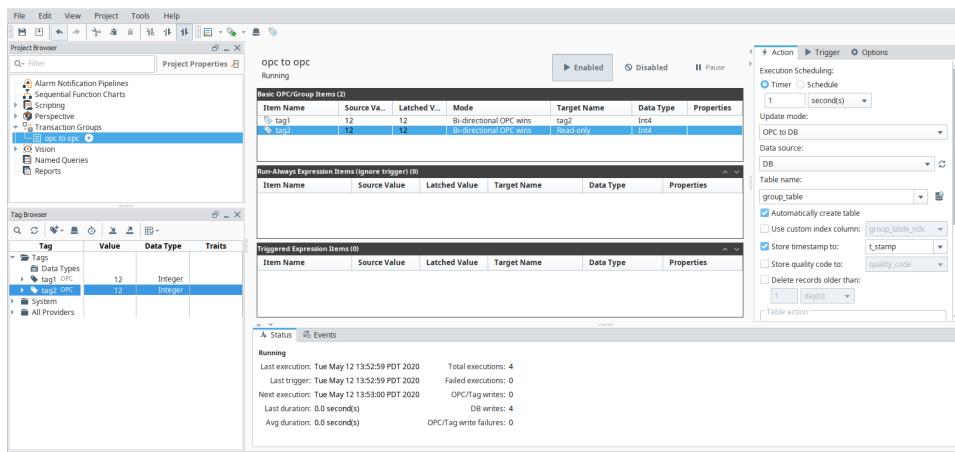
- Understanding Transaction Groups
- Types of Groups

OPC to OPC Transaction Group

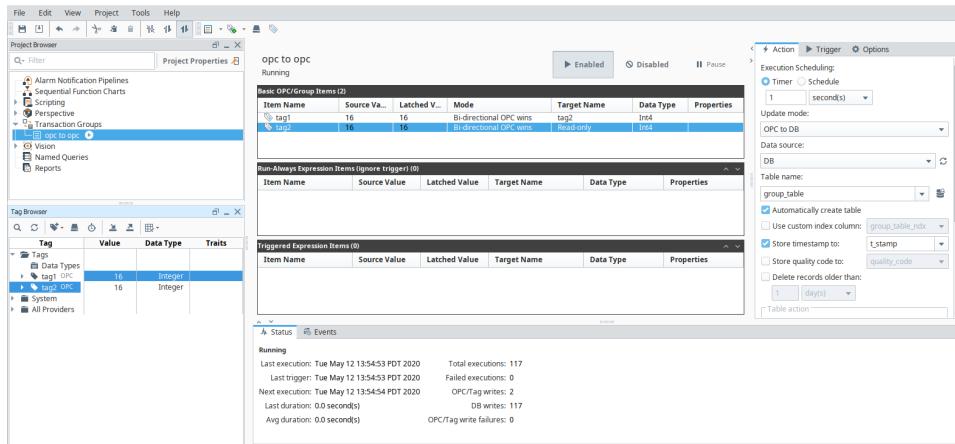
Configuring Transaction Group for OPC to OPC Interaction

Transaction Groups are generally used to channel OPC data to a database or vice-versa. It is also possible to configure your Standard Transaction Group to be able to get information from one OPC data point to another. This is useful in the event that you have tags coming from one PLC and you need the tag information to be sent to another PLC on your plant floor.

1. Create a **Standard Transaction Group** and from your Tag browser.
2. Drag two Tags into your Transaction Group's Basic OPC/Group Items section. For this example, the Tags will be called tag1 and tag2 and they will be coming from two different PLCs.
3. Set the mode on tag1 to **Bi-directional OPC Wins** and set its Target Name to be **tag2**.
4. Set the mode on tag2 to be **Bi-directional OPC Wins** and set its Target Name to **Read-only**. The mode on tag2 is not as important here as it is a read-only item, but we set it to Bi-directional OPC Wins anyway. Your configuration should match what is shown below:



What this will do is make sure that every 1 second, the value from tag1 will be written to tag2 as below:



Reporting

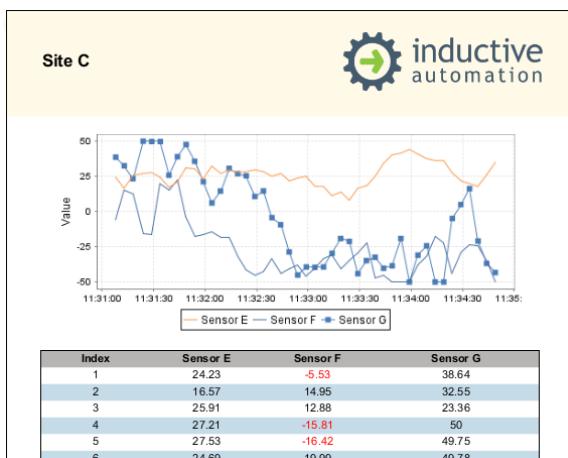
The Ignition Reporting Module makes creating professional reports easy with a rich library of tools including: images, graphs, tables, and basic shape tools. The Reporting Module enables you to create your own custom reports on the fly or generate them based on a schedule. Data is introduced through Ignition, providing access to any SQL database.

The Report Scheduler allows automatic report generation and automated distribution. Access to live reports is available through the web-based Ignition runtime (a Java application), providing authenticated users access from anywhere. Access is based on networking standards that your IT department can support. Reports are printer friendly and can easily be exported to a variety of formats including PDF. Here are some common uses of dynamic reports:

- Production management
- Inventory tracking
- Efficiency monitoring
- Historical trending
- Downtime tracking
- Quality assurance
- Data analysis

On this page ...

- Features
 - Intuitive Report Design
 - Powerful Components
 - Reporting Module Components for the Vision Module
 - Scheduled Report Execution
 - Supports Multiple File Formats
 - Scripting in the Reporting Module
 - Trial Mode Functionality
 - Legacy Reports



Features

Here are some of the other innovative features of the Reporting Module:

- Report designer Interface
- Scheduled report generation
- Powerful data collection utilities
- Drag-and-drop query builder
- Table and chart components
- 2D barcode generation
- Familiar property editing
- Flexible report distribution to file, email, FTP and more
- Scripting capabilities

Intuitive Report Design

The [report designer interface](#) was designed with the same look and feel as the other systems in Ignition. Once you install the Reporting Module, you can open the Ignition Designer and see a Reports section in the Project Browser. Reports are a Project-Level resource. They can be created independently or set to be viewed in a Vision window. With a right-click on the **Reports** node in the Project Browser and the option to create a new report is a click away.

From the first informative panel to full previews of the report you are creating, the Reports workflow significantly reduces the time it takes to design, edit, and distribute reports. The Report workflow is understandable to the new developer, yet powerful enough for a seasoned analyst.



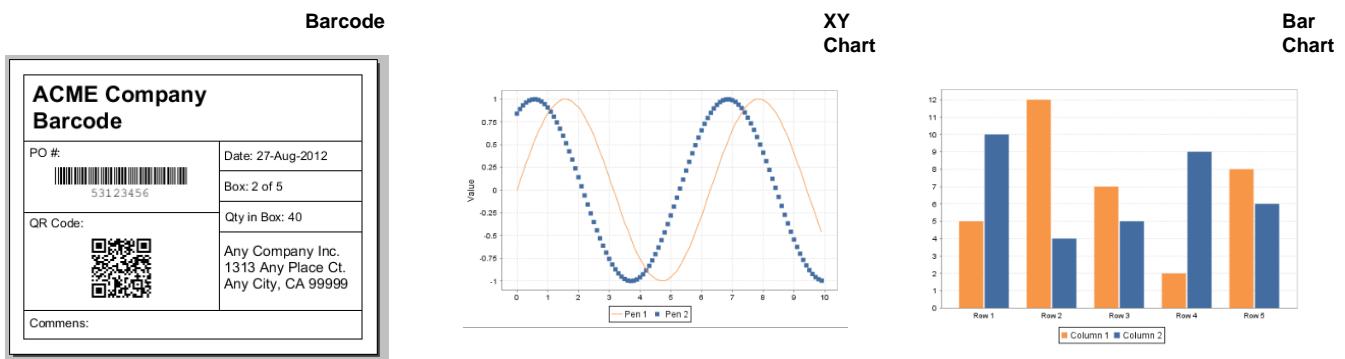
Reporting Interface

[Watch the Video](#)

The screenshot shows the Ignition Report Designer interface. The top navigation bar includes 'Report Overview', 'Data', 'Design', 'Preview', and 'Schedule'. The left sidebar contains a 'Project Browser' with categories like 'Multi-State Indicator', 'Tab Strip', 'Tank 3', 'Tree View', 'Popup', 'Templates', 'Named Queries', and 'Reports'. Below it is a 'Key Browser' with 'Show Calculations' checked. The main workspace displays a report with a bar chart, a line chart, and a table. The right sidebar lists 'Components' (Table, CrossTab, Simple Table, Labels, Barcode, Image), 'Graphs & Charts' (Timeseries Chart, XY Chart, Bar Chart, Pie Chart), and 'Shapes' (Text, Line, Rectangle, Ellipse, Star, Polygon, Pencil). A 'Property Inspector' panel is open on the left, showing settings for a selected component.

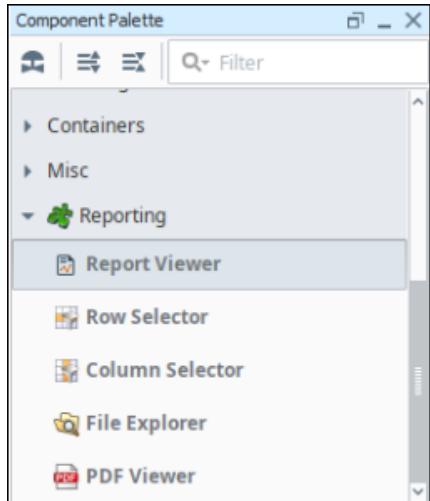
Powerful Components

The variety of [report design components](#) include tables, crosstab tables, XY charts, barcodes, pie charts, bar charts and more. [Report tables](#) can dynamically add pages to account for varying amounts of data, or change appearance based on certain values. [Report charts](#) can provide visual representation of comparisons and trends in data.



Reporting Module Components for the Vision Module

The Reporting Module provides several components that can be used with the Vision Module. The Report Viewer component allows reports to be viewed directly from the client. The Row Selector and Column Selector components let clients manipulate datasets graphically, while the File Explorer and PDF File Viewer components allow clients to access files outside of Ignition.



Scheduled Report Execution

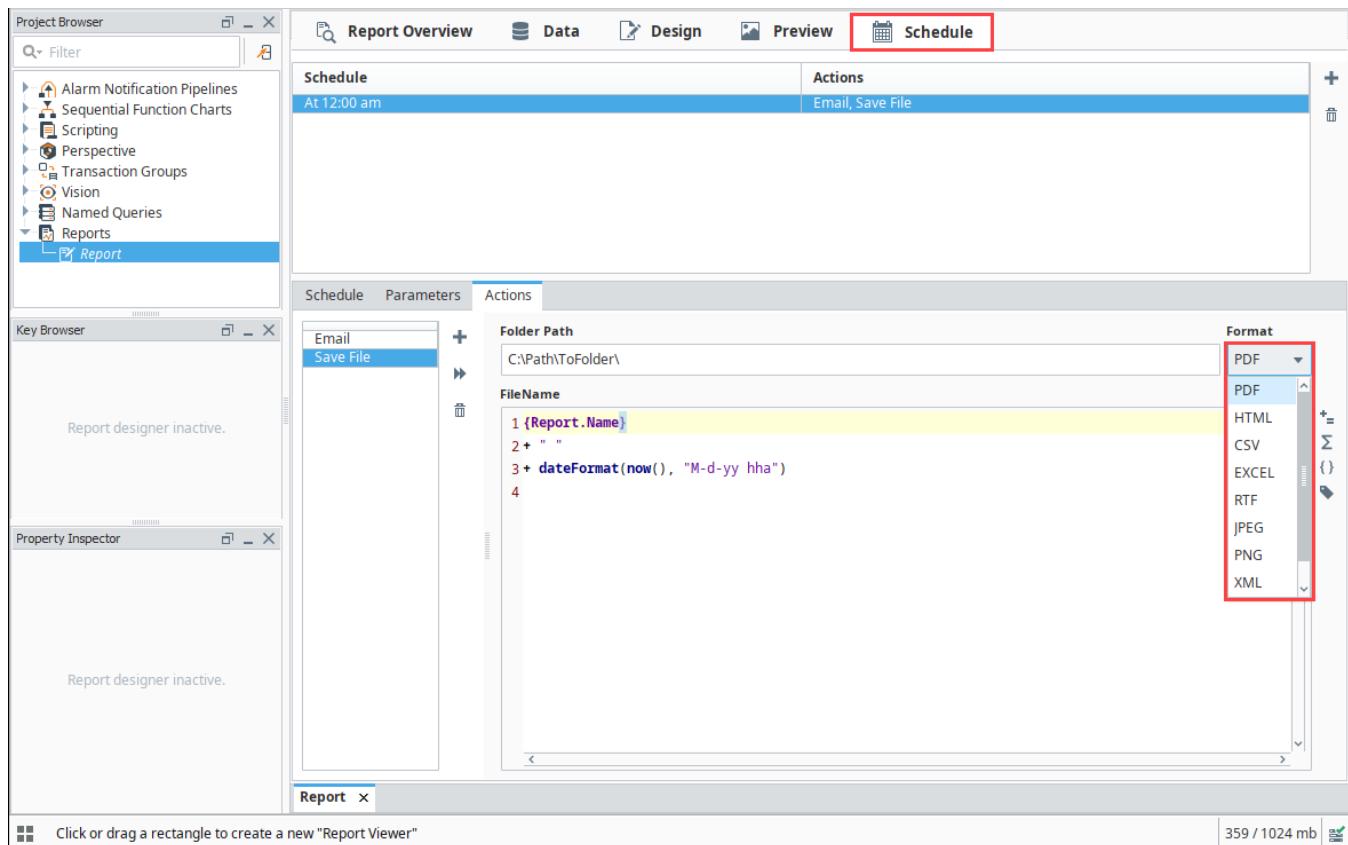
The [scheduling system](#) allows you to set the date, time, and frequency for executing a report. You can also set options for distribution of the report, including email, print, save, and more.

The screenshot shows the Ignition Report Designer interface with the 'Schedule' tab selected in the top navigation bar. The 'Schedule' table lists a single entry: 'At 12:00 am' under the 'Actions' column. Below the table, the 'Actions' configuration panel is visible, showing the following details:

- Email** action selected.
- From Address:** reports@ignition.com
- Subject:** 1{Report.Name}
- Attachment Filenamne:** 1{Report.Name}
2 + " - "
3 + dateFormat(now(), "M-d-yy ha")
4 + ".pdf"
- Body:** 1|Report attached|
- Mail Server:** <Select One>
- Format:** PDF
- Retries:** 0
- Address Source:** Email Addresses
- Recipient and ReplyTo Emails:** Address: Method

Supports Multiple File Formats

Reports can be generated in the following file formats: CSV, HTML, JPEG, PDF, PNG, RTF, XML, and EXCEL.



Scripting in the Reporting Module

The Reporting Module allows you to extend the existing functionality through scripting. You can customize reports with one of the following:

- **Script Data Source:** The Script Data Source type allows you to add to or modify your data set with Python code. For example, you can alter other Data Sources, combine results, and do complex calculations.
- **Scheduling:** The Run Script action in the Report Scheduling system enables you to create a script to do exactly what you want with a generated report.
- **Scripting Functions:** The Reporting Module adds additional functions to the list of `system.report.*` functions available throughout Ignition. You can use these functions to execute and/or distribute reports on demand.

Trial Mode Functionality

Like other systems in Ignition, the Reporting Module has full functionality in trial mode. There is no limit on the number of reports or how you can view or distribute them. Reports created in trial mode will have a watermark on each page.

Legacy Reports

To take advantage of Ignition's powerful platform, the Reporting Module was updated in version 7.8. You may view reports created by an older Reporting Module version in your project. If you need to modify an existing report, you'll still have access to the same customizer that you always had, double-clicking on them to open up their editor. They will continue to work as they always have without any modification. To learn more about converting reports created before Ignition version 7.8, refer to the section on [Converting Legacy Reports](#).

We recommend that you convert your older reports to the latest version to take advantage of the many features of the Reporting Module.

[In This Section ...](#)

Report Designer Interface

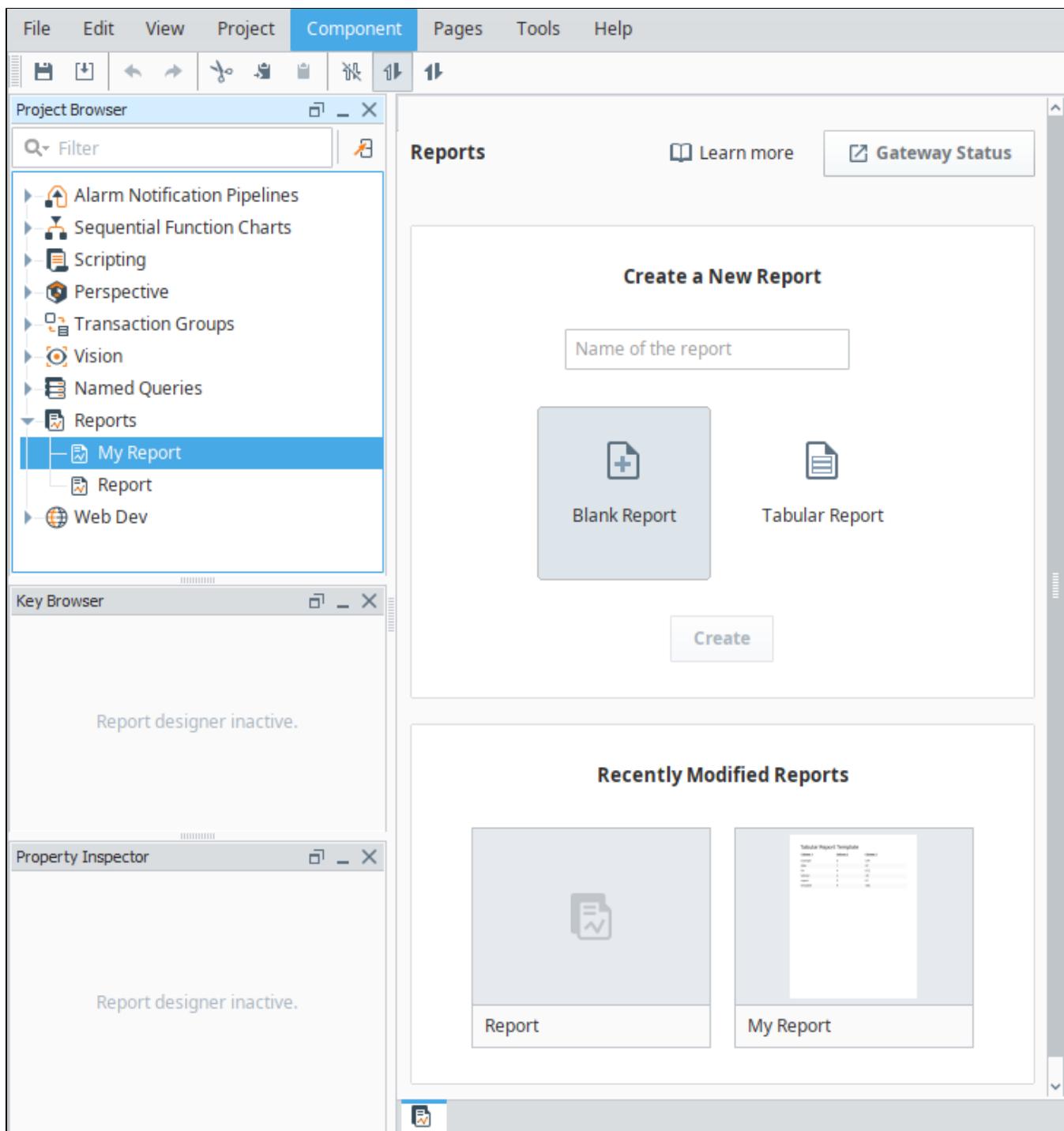
Built from the ground-up to be intuitive and familiar, the Reporting workspace has a logical workflow which makes it easy for anyone to create meaningful reports. Before we talk about the workflow, let's first mention where you create and find your Reports. Reports are located in the Project Browser of the Project Area under Reports. Any report that is created in your project will be found here.

The Reports Welcome tab presents you with two types of reports to help get you started: a Blank report and a Tabular report. The Blank report lets you design your report from the ground up. The Tabular report is a template with some sample data and a report design. This is helpful if you want to present the data in your report in tabular format.

To create a report, select a report type, enter a report name, and click 'create' and you are well on your way. It will even show you the most recently modified reports along with the thumbnail of the report. The Reports Welcome tab provides a quick way to create a new report and update an existing report.

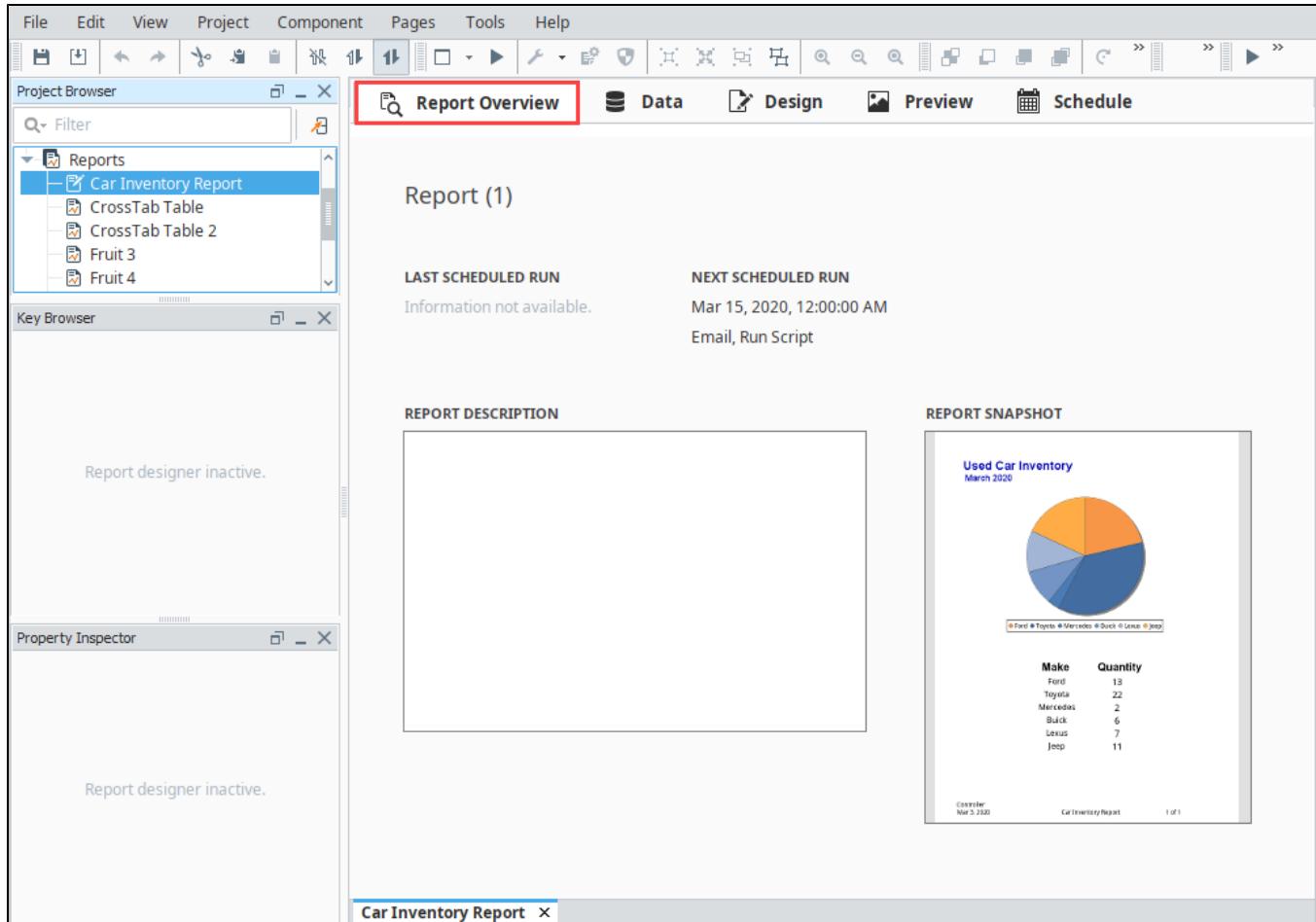
On this page ...

- [Report Overview Tab](#)
- [Data Tab](#)
- [Design Tab](#)
- [Preview Tab](#)
- [Schedule Tab](#)
- [Key Browser](#)
- [Property Inspector](#)



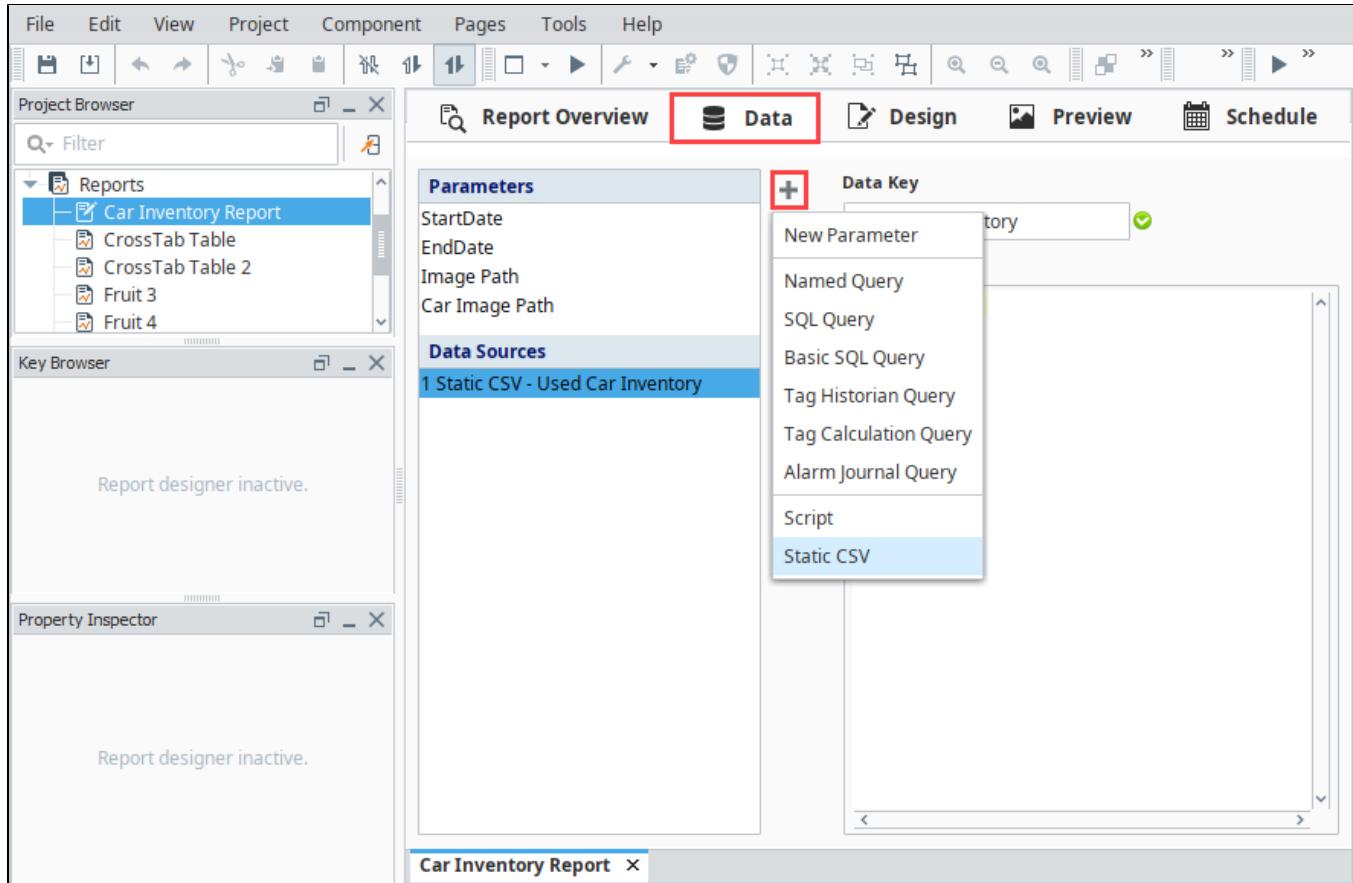
Report Overview Tab

The Report workspace has five tabs which make up the workflow of a report: Report Overview, Data, Design, Preview, and Schedule tabs. Once you create a report, the first step in the report workflow process is the Report Overview. The Report Overview tab provides valuable information at a glance about your report. There is space to add notes, which is a good place to provide background information and context about the report. The Overview will also show a thumbnail of the most recent report, its last execution time, and the next scheduled execution. The Report Snapshot is generated each time you visit the Preview tab.



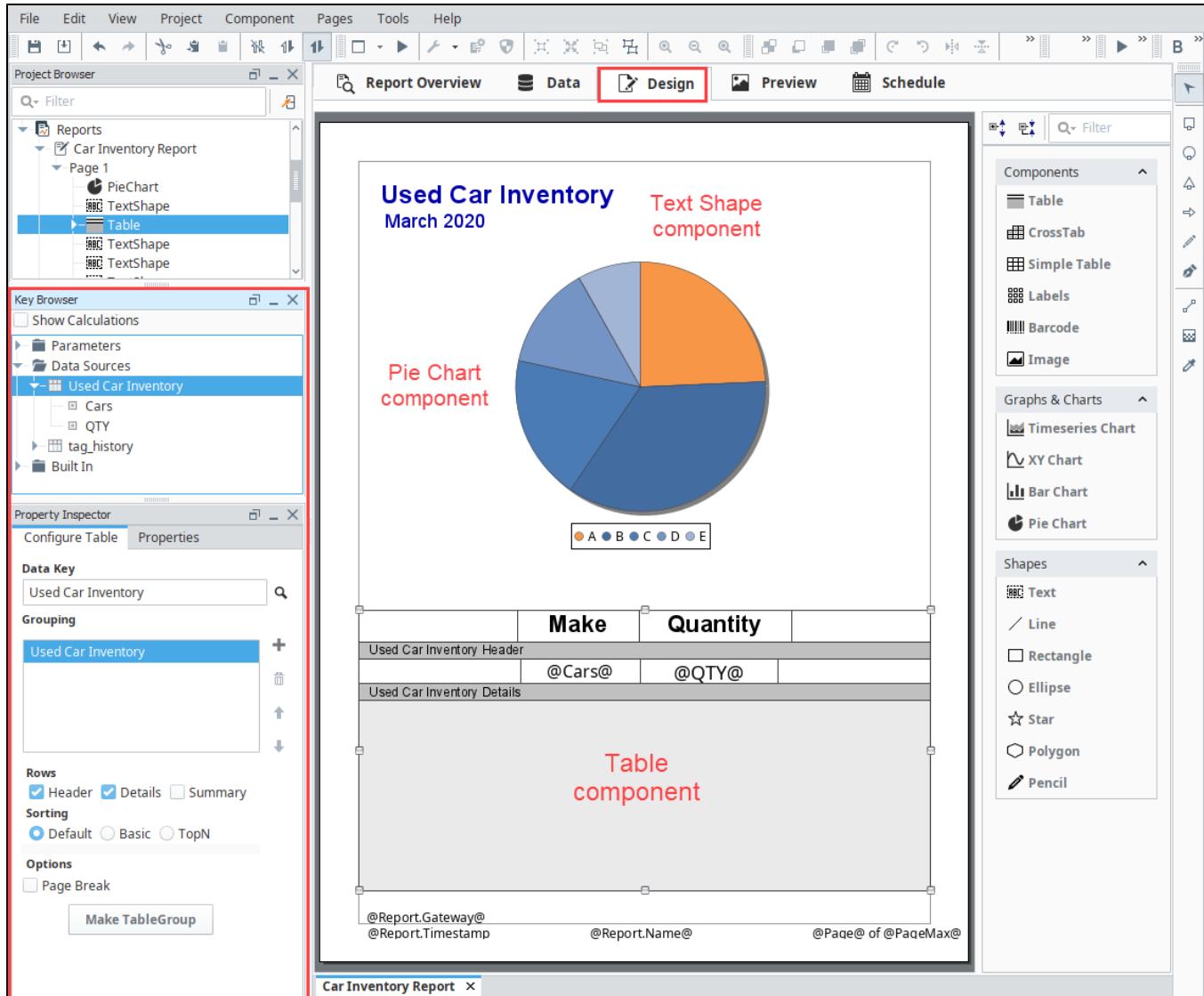
Data Tab

The [Data tab](#) is the first stop for configuring a new report. You'll notice the list on the left that has labels for Parameters and Data Sources, as well as some buttons. On this page you can generate data that you want to make available on your report. Click the **Add**  icon to see a list of built-in Data Sources and Parameters. Selecting any one of them will open the configuration screen for that Data Source type and enable you to start configuring the data for your report.



Design Tab

If you've ever designed a Perspective View or a Vision window, the **Design tab** should make you feel right at home. Starting from the top-left – the structure of a Report is represented in the **Project Browser** tree, allowing easy visualization of the tree of elements on your report, but don't stop there! Grab a component off the Reporting Palette on the far right and add it to the page. For complex components, there are configuration tabs in the **Property Inspector** to supplement the Property Inspector table that all the design objects have. In place of the Tag Browser, there is a **Key Browser** that gives you an easy way to add the Data Sources and Parameters that you configured in the Data tab to the Report.



Preview Tab

The Preview tab, while not one of our three design steps, is a huge help while building your report. This tab provides quick visual feedback on your report. Not only does it provide you an instant example of what your report looks like, but it also gives you the ability to view the actual data (on the right) that is being sent to your report. Having a snapshot of your data can be incredibly helpful when trying to figure out why your report doesn't appear as you expect it to. Whether you have an error in your query, or a bad Data Key in a component, you'll be able to figure out the structure of the data your report is getting and quickly find ways to solve any problems.

The screenshot shows a report designer application with the following components:

- Project Browser:** Shows 'Named Queries' and 'Reports'. 'Car Inventory Report' is selected.
- Key Browser:** Shows 'Report designer inactive.'
- Property Inspector:** Shows 'Report designer inactive.'
- Preview Tab:** Active, showing the report output. It includes a pie chart titled "Used Car Inventory" for March 2020, a table of car inventory by make and quantity, and footer information: Controller Mar 3, 2020, Car Inventory Report, 1 of 1.
- Schedule Tab:** Not active in the screenshot.
- Code View:** XML code for the report, highlighted with line numbers 1 through 40.

```

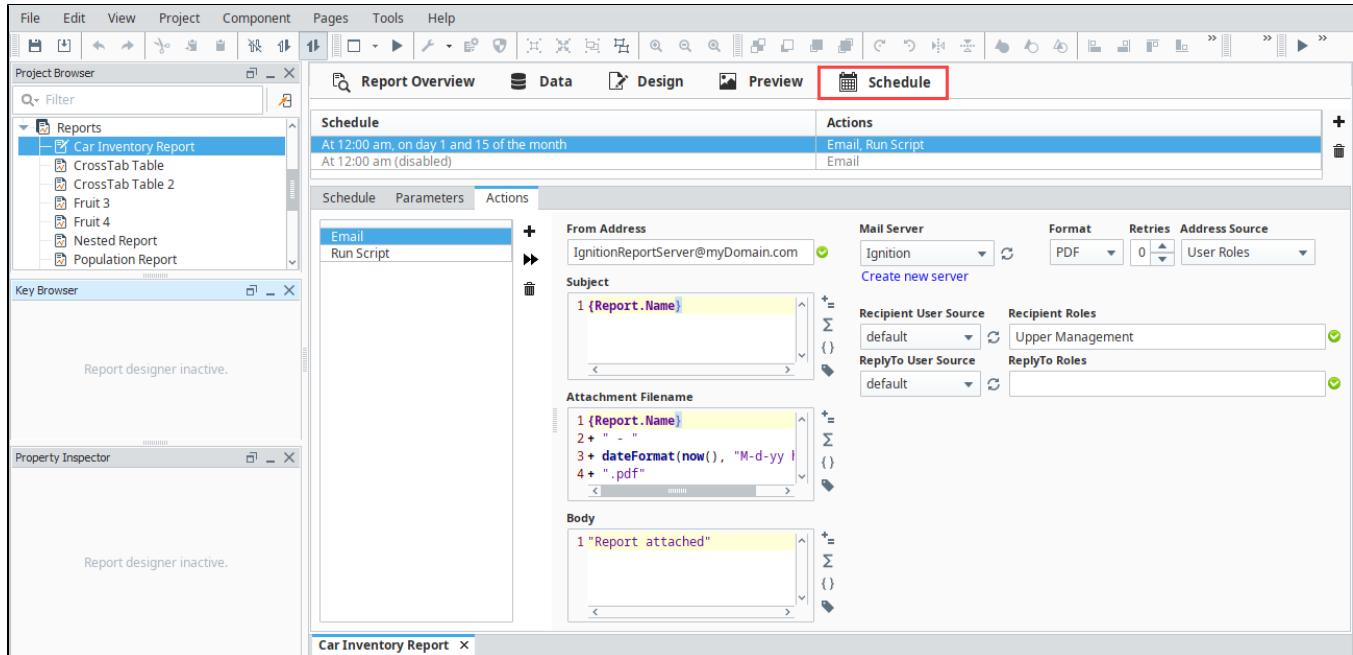
1 <?xml version="1.0" ?>
2 <sample-data>
3   <!--This is the raw data used to create the
4   <Car Image Path>http://i.ndtvimg.com/i/2016
5   <EndDate>2020-03-03 13:35:47.228</EndDate>
6   <Image Path>http://i.stack.imgur.com/WCveg.
7   <Report>
8     <Gateway>Controller</Gateway>
9     <Name>Car Inventory Report</Name>
10    <Path>Car Inventory Report</Path>
11    <Timestamp>2020-03-03 13:35:47.226</Timestamp>
12  </Report>
13  <StartDate>2020-03-03 05:35:47.227</StartDate>
14  <Used Car Inventory>
15    <row-0>
16      <Cars>Ford</Cars>
17      <QTY>13.0</QTY>
18    </row-0>
19    <row-1>
20      <Cars>Toyota</Cars>
21      <QTY>22.0</QTY>
22    </row-1>
23    <row-2>
24      <Cars>Mercedes</Cars>
25      <QTY>2.0</QTY>
26    </row-2>
27    <row-3>
28      <Cars>Buick</Cars>
29      <QTY>6.0</QTY>
30    </row-3>
31    <row-4>
32      <Cars>Lexus</Cars>
33      <QTY>7.0</QTY>
34    </row-4>
35    <row-5>
36      <Cars>Jeep</Cars>
37      <QTY>11.0</QTY>
38    </row-5>
39  </Used Car Inventory>
40  <tag_history>

```

Schedule Tab

[Scheduling a report](#) to run is built-in and easy to use. The Schedule tab is an incredibly powerful way to automate the execution and delivery of your reports. Whether you have extremely long running queries that you don't want tying up a window, need to deliver to multiple file servers via FTP, or want to simply email to an existing roster, this tab provides you the means to do it. It's broken up into two major areas. On top, you find a simple table of schedules that have been established. To add a schedule, click the **Add +** icon. With a schedule added, you can now choose when and how that scheduled Action will occur. Our schedules use the common Cron format, but don't worry if you aren't familiar with it. The preset options and combo-box configurability allow you to easily create just about any schedule you can imagine.

Once you selected your schedule, you can add parameters that may affect your Actions. Actions are incredibly powerful and allow you to choose how you want to distribute or act on your finished reports. Whether saving a file locally, posting to an FTP server, or emailing, you can get your report where it needs to be automatically. If our normal distribution Actions somehow fall short, you can choose to act on the finished report by triggering a script! We've also given you a convenient way of immediately executing the Action.



Key Browser

The Key Browser contains three very important elements: [Data Keys](#), [Built-in Keys](#), and the [Show Calculations](#) property. Data Keys are used to pull values from your data sources that you configure in the Data tab and display them in your report. They act as placeholders for your data and resolve to values when your report is generated. Built-In Keys are utility type functions that are commonly used when generating a report, such as adding a report name, date, page numbers, etc. The [Show Calculations](#) property adds several aggregates to each data key allowing you to display the total of a key. They are typically used in a summary row of a Table component.

Property Inspector

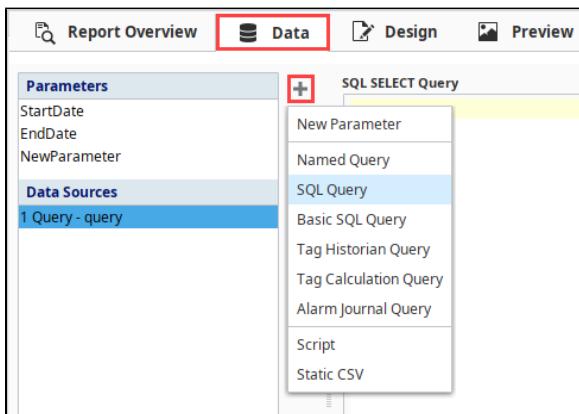
The Property Inspector is similar to the Property Editor in the Designer. It displays the properties of the selected component, but also includes configuration tabs for a number of the more complex report design objects such as the Pie Chart, Barcode, Tables, etc. The configuration tabs are specific to the selected Report component, and will only be used for the [Report Design Components](#).

Related Topics ...

- [Report Data](#)
- [Report Design](#)
- [Report Schedules](#)
- [Data Keys](#)

Report Data

The most critical part of any report is data. In the [Reporting Module](#), all data is collected as either a Parameter or a Data Source, and it is all configured in the Data tab of the report. Clicking on the **Add**  icon allows you to add new sources of data, bringing up a menu so you can choose the type of data to add. If a Parameter or Data Source is highlighted, clicking the **trashcan**  icon will delete that particular source of data.



On this page ...

- Order Matters
- Parameters
- Data Sources



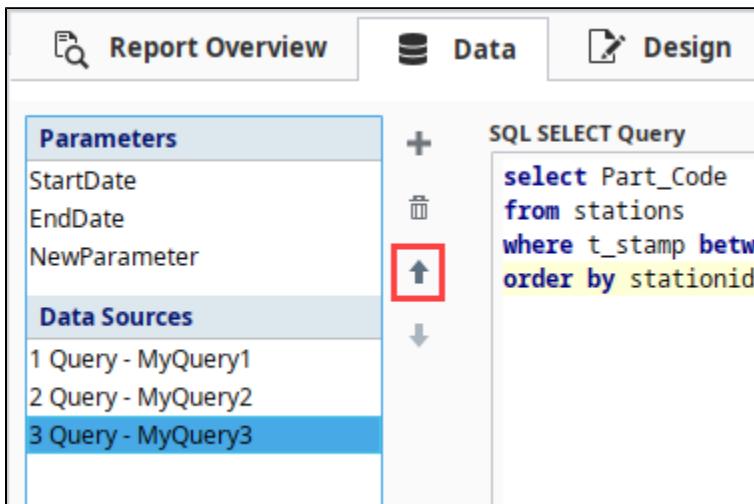
Report Data Tab

[Watch the Video](#)

Order Matters

The order of the Parameters and Data Sources is important. In some cases, a parameter or data source may have the capability to reference the results of another type, such as a Parameter referencing the value of another Parameter. Parameters and Data Sources may *only* reference other types of data that are listed vertically above them in the list. In short, the bottom-most Data Source may reference all other parameters and data sources, while the top-most Parameter may not reference any other parameter or data source.

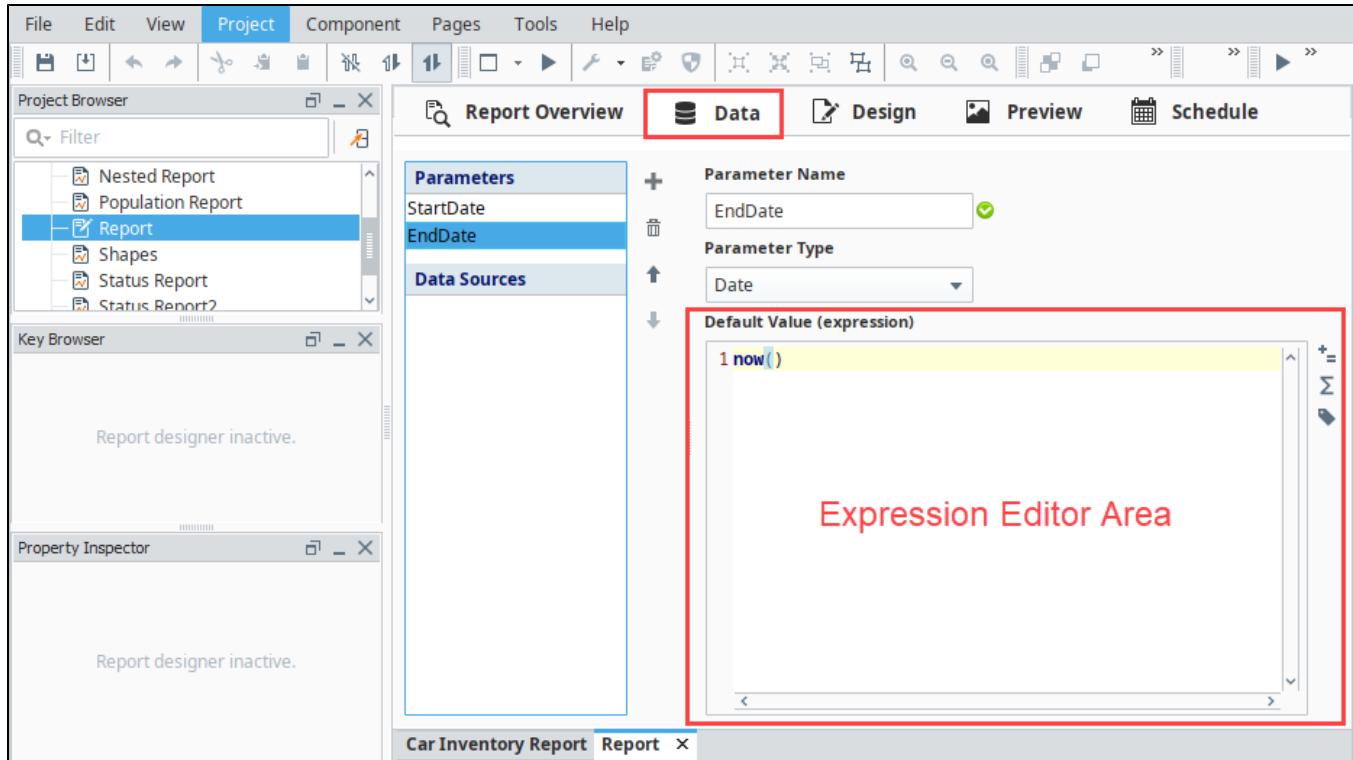
To reorder parameters, use the **up**  and **down**  arrow icons.



Parameters

[Parameters](#) are the way you get dynamic data into your charts. These properties are exposed on the [Report Viewer](#) component and can be bound to other components to allow your users to select what report data they want to see. You can use these to set up dates, time ranges, area selectors, titles, and anything else you want in your report.

By default, StartDate and EndDate properties are automatically created for you because they are almost always used to filter report data. You can, of course, delete them if you don't need them.



Data Sources

Data Sources are the primary means of getting data out of where it lives and into a report. There are eight Data Sources included in the Reporting Module: six Query types, a Script data source and Static CSV. Each data source offers a unique method of collecting (or amending in the case of *Script Source*) data. Aside from the *Script* data source, all will require you to specify a *Data Key*. The *Data Key* is a unique identifier that represents the top level of the data collected through this source – think of it as the label for the data source, or as the parent node of your source's collected 'data tree'. When your report is generated, the data collected under this identifier is passed to the reporting engine, which uses this identifier to appropriately place your data in the final report.

Here is a complete list of all the data source types:

- **Named Query:** A pre-configured query that runs as a prepared statement. If your Gateway is already using Named Queries to display data, then you can easily add those queries to your report.
- **SQL Query Data Source:** A straightforward query that allows parameters to be inserted with question marks (?) and has a graphical Query Builder.
- **Basic SQL Query:** A simplified version of the SQL Query that supports references directly in the query using the brace characters { }.
- **Tag Historian Query:** The same Tag History query builder you are familiar with from property bindings.
- **Tag Calculation Query:** Similar to the Tag Historian query, but this one allows calculations to be performed on the resulting data (min, average, duration on, count off, etc.).
- **Alarm Journal Query:** The same alarm journal query builder you are familiar with from the Functions property binding.
- **Script:** A blank script that you can use to create a dataset in any way you like.
- **Static CSV:** Static CSV data can be copied and pasted directly as an easy way to test your reports.



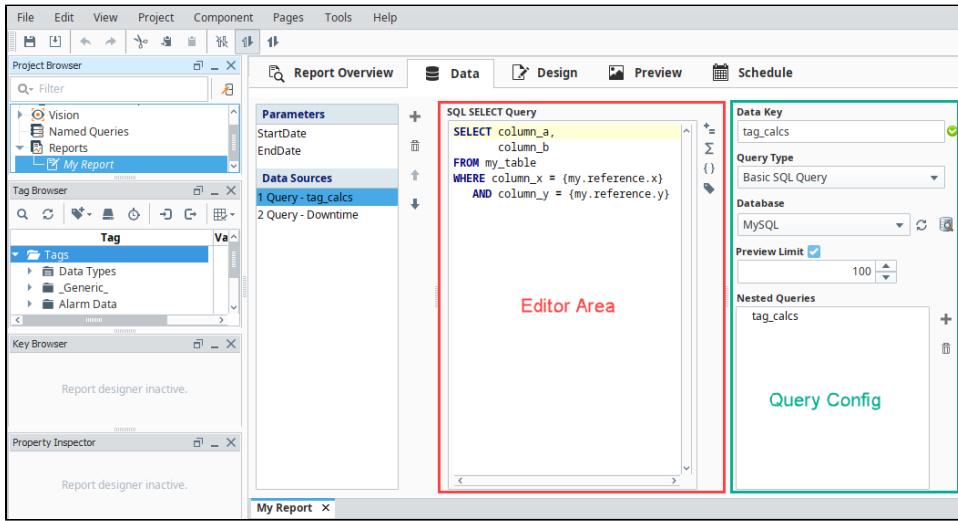
Developer Tip

Give your Data Keys readable names that say something about the table or data they represent. This will make it easier to build meaningful reports.

The Query data source types share some common features in the GUI, and they all return what is essentially a dataset. If you take a look at the image below, you'll see we outlined two main areas of the Basic SQL Query data source type. By default, the largest part of any Query panel is taken up by the central Editor Area. This space is tailored for each query type. On the right side of the Data panel, we have a "Query Configuration" area. In the screenshot, you'll see there are options for the *Data Key*, *Query Type*, *Database*, *Preview Limit* and *Nested Queries*. Of these, *Data Key*, *Query Type* and *Nested Queries* are shared among all Queries, while *Preview Limits* are conveniently available in SQL query types. Here is some brief information on each type:

1. **Data Key** - The identifier we will reference when we design our reports. The Data Key needs to be unique, can not contain special characters (spaces, underscores, dashes are allowed), and must start with a letter. We added a convenient validator to help detect name collisions.
2. **Query Type** - Gives you a convenient place to change the type of query.

3. **Nested Queries** - All of the query-based Data Source types can have [nested queries](#) in them. That is, for each row of returned data, a second query can be run based on the results in that row. You can have multiple subqueries for each row, and nest queries as deep as you want. This is particularly useful when you have data organized into runs and want to see historical data for each run.
4. **Preview Limit** - Allows you to choose to limit the number of rows returned from a query when a report is being generated in the Preview Panel, and when data is sampled for the purposes of generating Data Keys in the Designer. If supported by your database, this makes it much easier to craft and preview deep query structures without the overhead and wait-times of long-running queries.



Related Topics ...

- [Report Parameters](#)
- [Named Query Data Source](#)
- [SQL Query Data Source](#)
- [Data Keys](#)

In This Section ...

Report Parameters

As we covered in the overview, the Data Panel is pretty simple at first glance. When creating a new report, there is an empty list of Data Sources as well as pre-made parameters. **Parameters** are Ignition Expressions. These parameters resolve at report execution time and provide a convenient way to dynamically specify content for your report. These expressions are expected to resolve to specific types.

The Parameter Types available in the ComboBox will be familiar to anyone who has used Expressions, and include:

1. Date
2. String
3. Long
4. Double
5. Boolean
6. Dataset

Adding Parameters function similar to custom properties on Perspective views and Vision windows and their components. Their value can be shown directly on your report, or be referenced by data sources and other parameters. Parameters are given a default value when first created, but this value can be overridden once the report runs.

On this page ...

- [How Parameters Are Used](#)
- [Default Starting Parameters](#)
- [Creating New Parameters](#)



INDUCTIVE
UNIVERSIT

Parameters

[Watch the Video](#)

How Parameters Are Used

Parameters share some similarities to Custom Properties in Perspective and Vision components in that they allow an Ignition user to specify dynamic data to be used in reports at runtime. For example, a manufacturer may want to generate reports for the active production lines every Monday morning, but you don't know ahead of time which production lines will be running. We can create a parameter called `activeProductionLines` and use an expression or Tag value to determine which lines were active, and use the parameter in a SQL Query datasource to only include data from active lines.

Parameters can be declared with an empty (null) value. In this event, it can be fed values from the Vision Report Viewer component, or in the Report Scheduling panel. The Report Viewer can use parameters to bind to data through Ignition Bindings. If a parameter is supplied a default value in the Report Data panel (we call these *default parameters*), that value will be automatically be supplied to the Report Viewer component. If a parameter has no default, the Report Viewer and Preview Panel will notify the user that a parameter is undefined. It is important to note that both Scheduled Report and Vision component parameters take precedence over default parameters, so any default parameter can easily be overridden while also providing a baseline value.



Note that default values for parameters are evaluated in the Gateway, not by clients. As such, the expressions have a Gateway scope.

Default Starting Parameters

When you first create a report, you start with two parameters: **StartDate** and **EndDate**. EndDate will have a default value of `'now()'`, while StartDate will have a default value of `'dateArithmetic(now(), -8, "hr")'`. Notice how we changed the default value of the StartDate to be a difference of 10 days instead of 8 hours by modifying the Default Value expression. You can also delete both StartDate and EndDate parameters if you do not need date parameters in your report.

The screenshot shows the 'Report Overview' interface with the 'Data' tab selected. On the left, there's a sidebar with 'Parameters' and 'Data Sources' sections. Under 'Parameters', 'StartDate' is selected and highlighted in blue. To the right, the 'Parameter Name' field contains 'StartDate', 'Parameter Type' is set to 'Date', and the 'Default Value (expression)' field contains the expression `1 dateArithmetic(now(), -10, "days")`. A green checkmark icon is present next to the parameter name.

Creating New Parameters

New parameters may be added by clicking the Add icon.

Once created, a default value should be given to the parameter. In addition to expressions, you can also have your default value reference a Tag using the Tag icon on the right of the expression area.

This screenshot is similar to the one above, but the 'StringParameter' is now selected in the 'Parameters' list. The 'Parameter Name' field contains 'StringParameter', 'Parameter Type' is set to 'String', and the 'Default Value (expression)' field contains the expression `1 {String Tag}`. The 'String Tag' part is highlighted in blue, indicating it's a tag reference.

You can also use a literal value. Normal Expression languages syntax applies, so **dates** and **strings** must be wrapped in quotation marks, but **numerical** values can be written without quotes.

The screenshot shows the 'Report Overview' interface with the 'Data' tab selected. On the left, a sidebar lists parameters: StartDate, EndDate, StringParameter, and IntegerParameter, with IntegerParameter currently selected. The main panel displays the configuration for IntegerParameter: Parameter Name is set to 'IntegerParameter' (with a green checkmark), Parameter Type is 'Long', and the Default Value (expression) is '1 987654321'. There are also icons for adding, deleting, and reordering parameters.

Finally, you can reference other parameters. In the example below, **BoolParameter** is referencing the value of **IntegerParameter**.

It is important to note that when referencing other parameters, you must type the name of the parameter exactly within a set of curly braces {}, including capitalization. Also, you can only reference parameters that are above the current parameter. Thus, while **BoolParameter** can reference the **IntegerParameter**, the **IntegerParameter** may not reference **BoolParameter**.

The screenshot shows the 'Report Overview' interface with the 'Data' tab selected. On the left, a sidebar lists parameters: StartDate, EndDate, StringParameter, IntegerParameter, and BoolParameter, with BoolParameter currently selected. The main panel displays the configuration for BoolParameter: Parameter Name is set to 'BoolParameter' (with a green checkmark), Parameter Type is 'Boolean', and the Default Value (expression) is '1 if ({IntegerParameter} = 1000, 1, 0)'. There are also icons for adding, deleting, and reordering parameters.

Related Topics ...

- [Basic SQL Query](#)

Named Query Data Source

Using the Named Query Data Source

Using the Named Query datasource is simple to configure provided you already have a [Named Query](#) created, and works in the same way that other Named Query bindings work. The Named Query report data source will execute the selected Named Query on the Gateway, and can use the [Report Parameters](#) as its parameters.

This data source supports [Nested Queries](#).

The screenshot shows the 'Report Overview' interface with the 'Data' tab selected. In the 'Path' field, 'FirstNamedQuery' is entered. Below it, under 'Parameters', there is a table with one row: Type 'Value', Name 'BayNum', Data Type 'Int2', and Value '(BayNumber)'. The 'Query' section contains the following SQL:

```
SELECT *  
FROM storage  
WHERE bay_num = :BayNum
```

On this page ...

- [Using the Named Query Data Source](#)
- [Configuring a Named Query Data Source](#)

Configuring a Named Query Data Source

First select a Named Query by clicking the **Selection** icon next to the Path field.

The screenshot shows the 'Design' interface. In the 'Path' field, there is a red box around the 'Selection' icon. A dropdown menu is open, showing 'Named Queries' and 'FirstNamedQuery', with 'FirstNamedQuery' also having a red box around it. At the bottom right of the interface is a 'Select' button.

Named Queries in Reports

[Watch the Video](#)

In the resulting list, select one of the configured named queries. The Named Query parameters will appear, along with a preview of the query.

The screenshot shows the 'Data' interface. In the 'Path' field, 'FirstNamedQuery' is entered. Below it, under 'Parameters', there is a table with one row: Type 'Value', Name 'BayNum', Data Type 'Int2', and Value '(BayNumber)'. The 'Query' section contains the following SQL:

```
SELECT *  
FROM storage  
WHERE bay_num = :BayNum
```

One-by-one, **select** each parameter, and then click the **Selection** icon next to the **Parameters** table. This provides a popup of all report parameters that are available on the report. Select a Report parameter that will provide a value to the Named Query parameter. Once every parameter has a value, check the results in the Preview Panel.

Related Topics ...

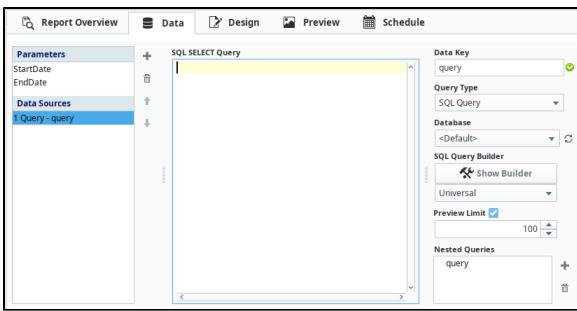
- [Named Queries](#)
- [Tag Calculation Query](#)
- [Report Charts](#)

SQL Query Data Source

Using the SQL Query Datasource

The SQL Query Data Source allows you to craft parameterized queries that run as a prepared statement. As Prepared Statements, these queries are more resistant to SQL injection offering additional security over basic queries.

The SQL Query type looks very similar to the Basic SQL Query type. It has a large text area where you can enter in a SQL select query. On the right, you have the option to rename the query, choose what database to run this query against, and add in a [Nested Query](#).



On this page ...

- [Using the SQL Query Datasource](#)
- [Parameters in SQL Query](#)
- [Crafting Queries with the Query Builder](#)
 - [Using the Builder](#)



SQL Query

[Watch the Video](#)

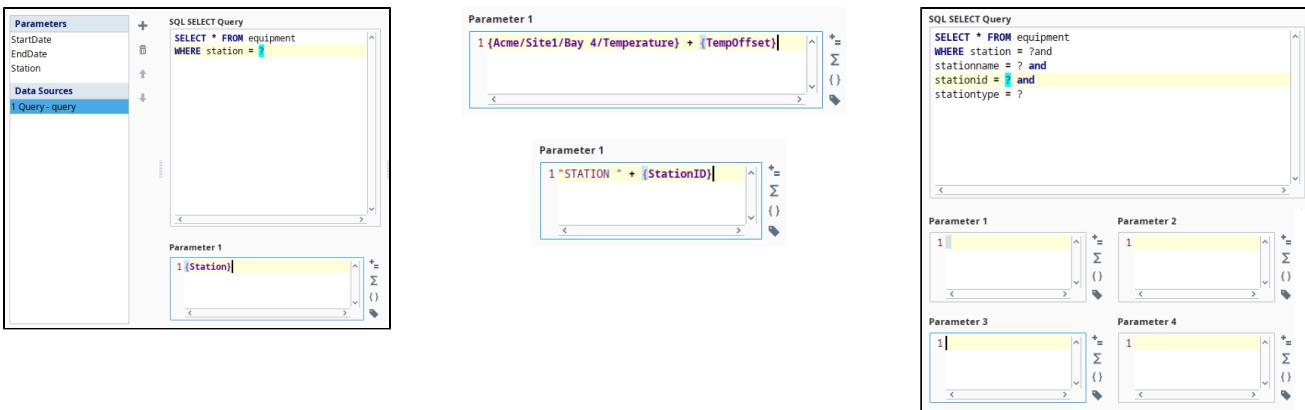
Parameters in SQL Query

Because the SQL Query Data Source runs as a prepared statement, passing parameters into this query type works a little differently.

Instead of placing a parameter within {} characters like the Basic SQL Query, we place a ? where we would like to pass in a parameter. Doing this will actually create a new text area below the query area for your parameter. This smaller text area is where you can pass a parameter into the query. You can pass in Tag values or Report Parameters, and since the parameter area allows expressions, you can use expressions to create any value you like from a combination of Tags and Report Parameters.

This is the preferred way to pass dates into your queries. You can use the {StartDate} and {EndDate} default Parameters directly in the Parameter fields below.

You can add as many of these to a query as needed, with new parameter areas popping up underneath as they get added. To help keep track of what parameter corresponds to which "?", when entering in a value into the parameter area, its associated "?" will highlight to show you which parameter you are currently working on.



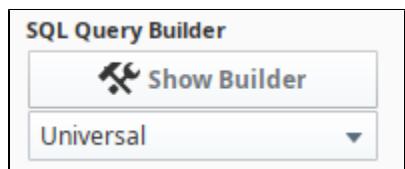
Crafting Queries with the Query Builder

The SQL Query data source includes the powerful SQL Query Builder tool. The Query Builder is a powerful Drag-and-Drop query building GUI that allows you to make complex queries from your connected databases. While a basic understanding of SQL helps make the most of the Query Builder

tool, most people will have no problem creating effective queries after a brief tutorial. The Query Builder is a third party tool that we brought into the Reporting Module. We go into detail on how to use it on the [Query Builder](#) page, but you can also check out the [Active Query Builder's documentation](#) for additional information.

Using the Builder

To activate the Query Builder in your **SQL Query** datasource type, start by selecting the SQL Syntax version from the drop down menu beneath the SQL Query Builder button. If your Database type isn't available (or you aren't sure), you can get most of the general functionality by selecting the **Universal** option. Then push the SQL Query Builder button to show the [Query Builder](#).



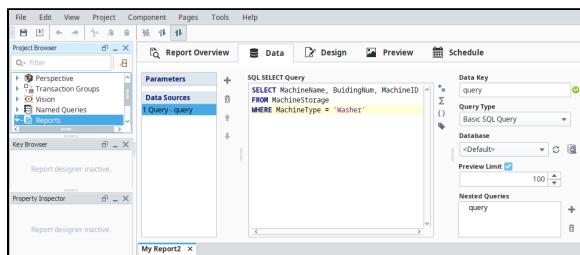
Related Topics ...

- [Tag Historian Query](#)
- [Report Parameters](#)

Basic SQL Query

This query type is the common type typically seen through much of Ignition before version 7.8. You can write queries which include Tag path references, expressions, or report parameters which resolve at run time.

You can enter in a static SQL query like in the image below. The query will return rows that have a MachineType of 'Washer'.



On this page ...

- Report Parameters in a Basic SQL Query
 - Working with Dates
 - Create New Formatted Parameters

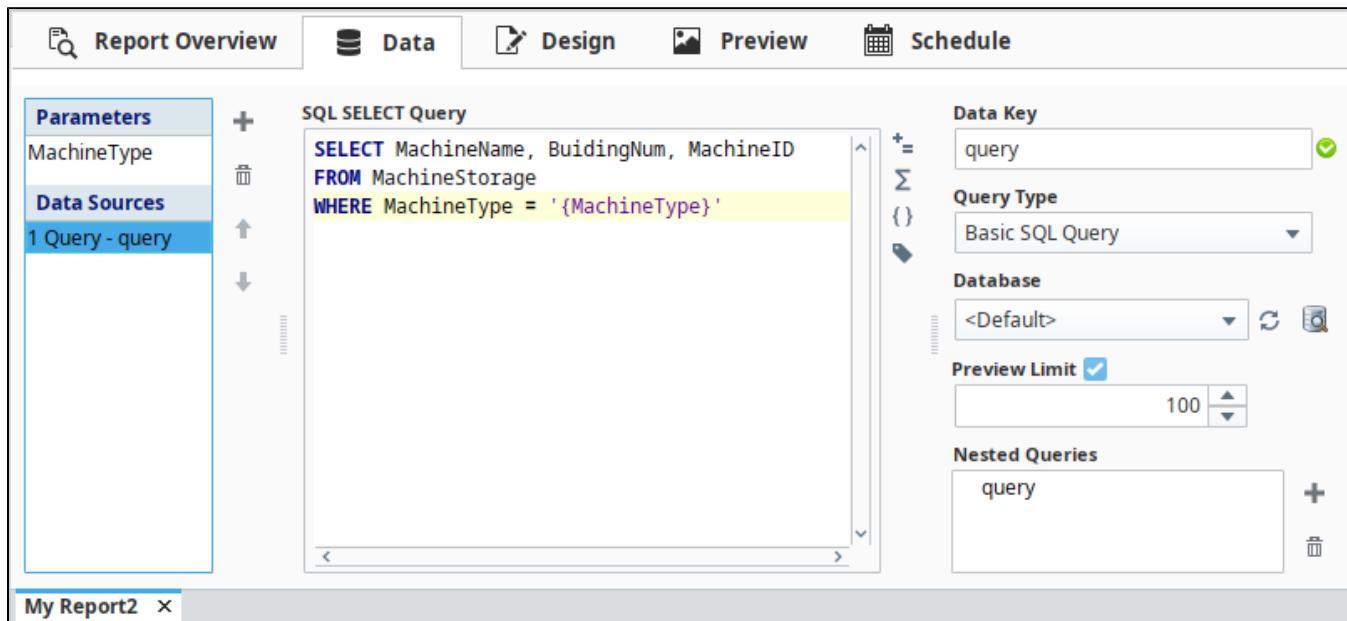


Basic SQL Query

[Watch the Video](#)

Report Parameters in a Basic SQL Query

Queries can also be made dynamic using things like report [parameters](#). To insert a report parameter, click the **Parameters** icon to the right of the query area and select your parameter. This allows for a more dynamic query, since new values can be passed into the parameter at runtime, giving the ability to change the type of machine this query is looking for.



Parameters are inserted directly into a report. This means that the datatype of parameter will affect how it should be referenced in the query. For example, since the parameter MachineType is a string, it will need a single or double quotes around it.

String Parameters

```
WHERE MachineType = '{MachineType}'
```

Since an integer does not need quotes around it, if your parameter is a Long, Double, or a Boolean, you can directly place the parameter in your query, without the quotes.

SQL - Long, Double, and Bool Parameters

```
WHERE MachineNum = {MachineNumber}
```

Working with Dates

If your parameter is a date object, then special consideration must be made.

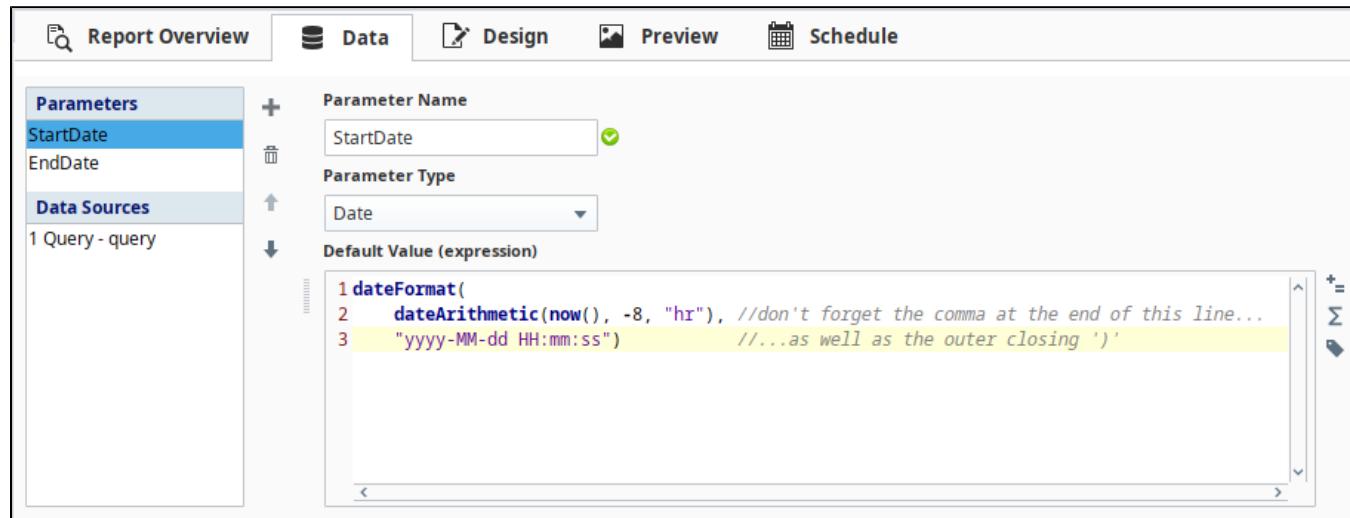
The query will not accept a date object directly, it must first be converted to a string by putting quotes around it. However, database generally prefer datetime objects in very specific formats, such as **yyyy-MM-dd HH:mm:ss**. This means we need to reformat the date on any report parameters we want to pass to the Basic SQL Query. There are two main approaches to this:

Reformat the Date Parameters

By utilizing the expression language's [dateFormat\(\)](#) function, we can simply specify the format of the date.

SQL - Reformatted Date Parameter

```
dateFormat(  
    dateArithmetic(now(), -8, "hr"), //don't forget the comma at the end of this line...  
    "yyyy-MM-dd HH:mm:ss") //...as well as the outer closing ''
```



Create New Formatted Parameters

In some cases, you may wish to leave the original "raw" Date parameter alone, and create a display-friendly version as a string.

To do this, simply make a parameter with type string and use the [dateFormat\(\)](#) expression on a date. In the image below, you can see that the **StartDa**
te parameter is used in a new **StartString** parameter. Additionally, an **EndString** parameter has been created that is using the **EndDate** parameter. This way, we can bind a calendar component directly to the Start and EndDate parameters and all the formatting will be done automatically in the report.

The screenshot shows the 'Report Overview' interface with the 'Data' tab selected. On the left, a sidebar lists parameters: StartDate, EndDate, StartString (selected), and EndString. Below this is a 'Data Sources' section with '1 Query - query'. The main panel displays the configuration for 'StartString': Parameter Name is 'StartString', Type is 'String', and the Default Value (expression) is `dateFormat({StartDate}, 'yyyy-MM-dd HH:mm:ss')`.

This format was used with a MySQL database, so your database may take a different format. Refer to your database's documentation for suggested date formats. Once the new string parameters have been created, we can then reference them in the Basic SQL Query just like a normal string.

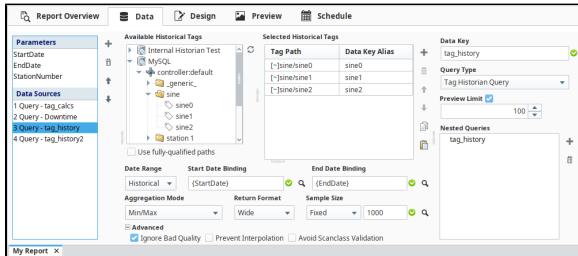
The screenshot shows the 'Report Overview' interface with the 'Data' tab selected. On the left, a sidebar lists parameters: StartDate, EndDate, StartString, and EndString. Below this is a 'Data Sources' section with '1 Query - query'. The main panel displays a 'SQL SELECT Query' configuration: the query is `SELECT * FROM group_table WHERE t_stamp BETWEEN '{StartString}' AND '{EndString}'`. To the right, configuration options include: Data Key 'query' (checked), Query Type 'Basic SQL Query', Database '' (with a refresh icon), Preview Limit set to 100, and Nested Queries 'query'.

Related Topics ...

- [SQL Query Data Source](#)
- [Report Tables](#)
- [Report Charts](#)

Tag Historian Query

The Tag Historian Query provides a simple way to query data from Tag Historians. In the Tag Historian Query you can collect data from Historical Tags for specific date ranges, apply aggregates, and specify the sample size. It functions the same as a [Tag History Binding](#) and uses the same interface, so check out that page for more information on the individual properties.



On this page ...

- [Using Parameters in a Tag History Query](#)



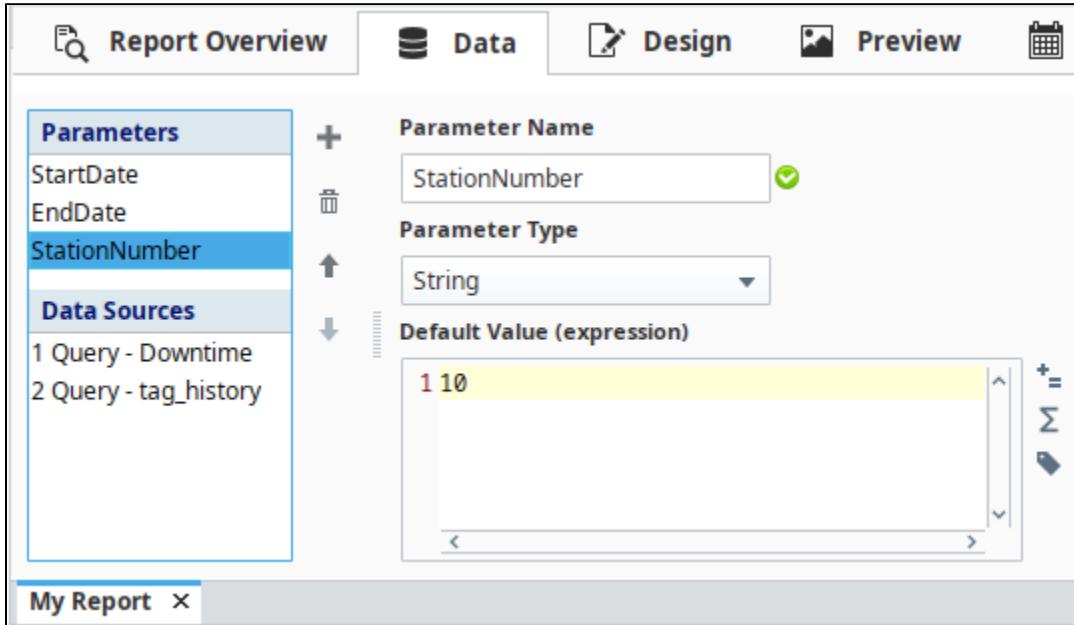
Tag Historian Query

[Watch the Video](#)

Using Parameters in a Tag History Query

The only difference between a Tag History Binding and the Tag History Query is when using indirection. In the Tag History Query, indirection is inserted directly into the Tag Path in the Selected Historical Tags section.

Assume we have a parameter named **StationNumber**:



Indirection is typically done using a parameter which like the rest of reporting, is the name of the parameter enclosed in { }. In the example below, we substituted the parameter 'StationNumber' into my Tag Path for the number of the station. Note that you need to manually type out the name of the parameter, including exact spelling and capitalization.

Assuming base Tag Paths like the following:

```
[~]station 1/paint pressure  
[~]station 1/paint fill level
```

we can replace the "1" with a reference to **StationNumber**:

```
[~]station {StationNumber}/paint pressure,  
[~]station {StationNumber}/paint fill level
```

The screenshot shows the 'Report Overview' interface with the 'Data' tab selected. On the left, under 'Parameters', are 'StartDate', 'EndDate', and 'StationNumber'. Under 'Data Sources', '4 Query - tag_history2' is selected. The central area displays 'Available Historical Tags' and 'Selected Historical Tags'. The 'Selected Historical Tags' table has two rows:

Tag Path	Data Key Alias
[~]station {StationNumber}/paint fill level	paint fill level
[~]station {StationNumber}/paint pressure	paint pressure

On the right, the 'Data Key' is set to 'tag_history2', 'Query Type' is 'Tag Historian Query', and 'Preview Limit' is 100. A 'Nested Queries' section shows 'tag_history2'. At the bottom, there are date range and aggregation mode settings.

Related Topics ...

- [Tag Calculation Query](#)
- [Report Charts](#)

Tag Calculation Query

Tag Calculations are executed by the Tag Historian, providing multiple calculated values for each Tag path. While the Tag Historian Query returns a history of values and only uses the aggregates to perform calculations on small chunks of data in the range of time, the Tag Calculation Query will aggregate, or run calculations, on the entire range to produce a single, calculated value per Tag Path.

Assuming the following configuration:

The screenshot shows the configuration of a Tag Calculation Query. In the 'Parameters' section, 'StartDate' and 'EndDate' are defined. Under 'Data Sources', a query named 'tag_calcs' is selected. The 'Available Historical Tags' tree includes 'Internal Historian Test', 'MySQL', and 'controllerdefault'. The 'Selected Historical Tags' list contains three tag paths: 'sine0', 'sine1', and 'sine2', each with a 'Data Key Alias' of 'sine0', 'sine1', and 'sine2' respectively. The 'Data Key' is set to 'tag_calcs'. The 'Query Type' is set to 'Tag Calculation Query'. A 'Nested Queries' section shows a query named 'tag_calcs'. The 'Calculations' section is expanded, showing several options: 'Time-weighted Average' (selected), 'MinMax', 'Closest Value', 'Basic Average', 'Sum', and 'Minimum'. The 'Date Range' is set to 'Historical' with 'Start Date Binding' at '(StartDate)' and 'End Date Binding' at '(EndDate)'. At the bottom left is a 'My Report' button.

On this page ...

- [Tag Historian vs Tag Calculation](#)
- [The 'Min/Max' Exception](#)



Tag Calculation Query

[Watch the Video](#)

We could show the query results on a table and view the following in the Preview Panel:

Average	Sum	Tag Path	Timestamp
0.28	1601.21	sine0	Mar 10, 2020
165.56	1260330.57	sine1	Mar 10, 2020
1.07	6024.47	sine2	Mar 10, 2020

Tag Historian vs Tag Calculation

When deciding between a Tag Calculation Query and a Tag Historian Query, the main difference between the two is as follows:

- The Tag Calculation Query will always return a single row for each Tag Path, with multiple columns for each aggregate/calculation, except in the case of Min/Max (see below).
- Multiple calculations may be called on each Tag Path. This is not possible with a single Tag Historian Query.

The 'Min/Max' Exception

Since Min/Max returns two values (the minimum and maximum value over the range), this calculation will generate two rows per Tag Path. This extra row will appear in the underlying data, even if the MinMax key isn't used in a table or chart, as it is part of the underlying data.

If we added Min/Max to our example above, the preview panel would look like the following:

Average	Sum	Tag Path	Timestamp	MinMax
0.09	351.01	sine0	Mar 10, 2020	-50
165.71	1224266.37	sine1	Mar 10, 2020	321
0.12	379.52	sine2	Mar 10, 2020	100
0.09	351.01	sine0	Mar 10, 2020	50
165.71	1224266.37	sine1	Mar 10, 2020	10
0.12	379.52	sine2	Mar 10, 2020	-100

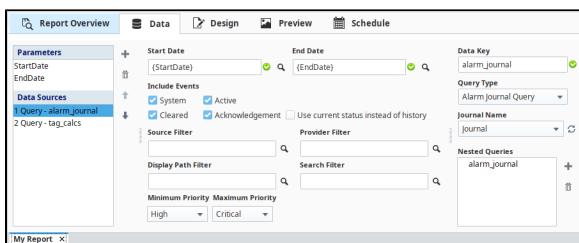
Note: The other calculations (Average and Sum) are simply duplicated for both MinMax rows.

Related Topics ...

- [Alarm Journal Query](#)
- [SQL Bridge Module](#)

Alarm Journal Query

The Alarm Journal Query data source is a simple way to access Alarming data within a report. It can pull from both the alarm journal data in the database as well as the current live events that would show up in the Alarm Status Table.



On this page ...

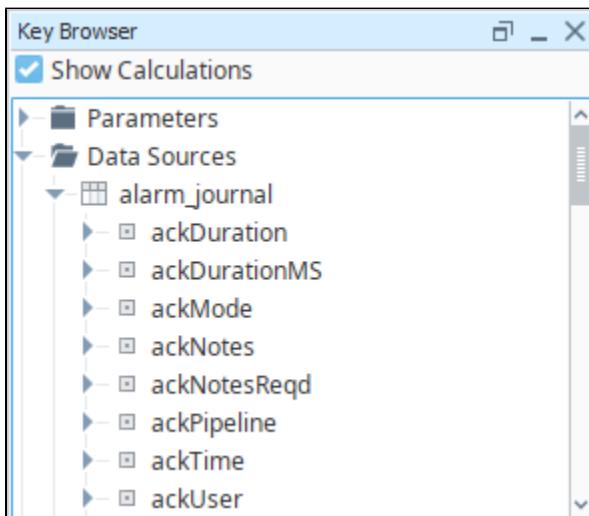
- Properties



Alarm Journal Query

[Watch the Video](#)

Once created, the new Data Source will expose many alarming related keys in the Design Panel's Key Browser.



Properties

The Alarm Journal Query has a few properties that can be configured to filter the types of events you see in the returned dataset.

Property	Description
Start Date and End Date	The time range from which to pull Alarms from. Start Date being the furthest date in the past, while End Date is the most recent date.
Include Events	Allows you to select which type of events will be returned. The options are: <ul style="list-style-type: none">• System• Cleared• Active• Acknowledgment There is also a Use current status instead of history checkbox. As the text implies, this will use the current alarm status instead of querying from an Alarm Journal profile. Enabling this checkbox will Checking this will cause the data source to ignore the Journal Name property on the right.

Source Filter	Will filter the alarms based on the source path. Multiple filters can be specified by separating them with a comma, and they accept the wildcard (*) symbol. Works in the same way that the filters on the Alarm Status Table component work.
Provider Filter	Will filter the alarms based on the Tag Provider that they originate from. Can specify multiple filters by separating them with a comma.
Display Path Filter	Will filter the alarms based on the display path. Multiple filters can be specified by separating them with a comma, and they accept the wildcard (*) symbol. Works in the same way that the filters on the Alarm Status Table component work.
Search Filter	Will filter alarms by searching for the given string in both the display path and source path. Accepts comma separated paths.
Minimum Priority and Maximum Priority	The minimum and maximum priority of alarms that will be returned.
Journal Name	Located on the right side, this property allows you to specify the Journal that will be used for the query.

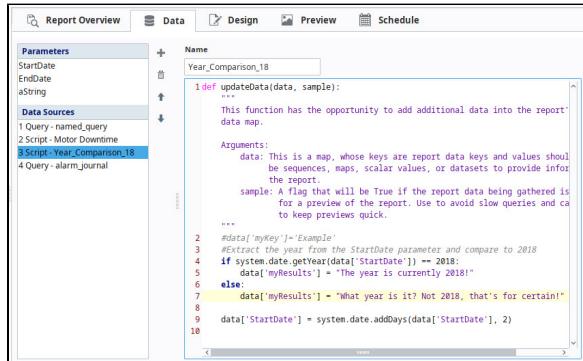
Related Topics ...

- [Static CSV](#)
- [Report Tables](#)

Scripting Data Source

Script Data Source

The Script data source allows you to use [scripting](#) to add additional data into a report, or modify existing data. With this data source, you can pull in data from the database and then modify it using a script before pushing it out as a data key for use in the report.



The screenshot shows the 'Report Overview' interface with the 'Data' tab selected. In the 'Parameters' section, there are three parameters: 'StartDate', 'EndDate', and 'aString'. In the 'Data Sources' section, there are four entries: '1 Query - named_query', '2 Script - Motor Downtime', '3 Script - Year_Comparison_18', and '4 Query - alarm_journal'. The '3 Script - Year_Comparison_18' entry is highlighted. The details pane shows the script code:

```
1 def updateData(data, sample):
    """
    This function has the opportunity to add additional data into the report's data map.

    Arguments:
        data: This is a map, whose keys are report data keys and values should be sequences, maps, scalar values, or datasets to provide info for the report.
        sample: A flag that will be True if the report data being gathered is for a preview of the report. Use to avoid slow queries and cache to keep previews quick.

    ...
    #data['myKey'] = 'Example'
    #Extract the year from the StartDate parameter and compare to 2018
    if system.date.getYear(data['startDate']) == 2018:
        data['myResults'] = "The year is currently 2018!"
    else:
        data['myResults'] = "What year is it? Not 2018, that's for certain!"
    ...
    9
    10 data['StartDate'] = system.date.addDays(data['StartDate'], 2)
```

On this page ...

- [Script Data Source](#)
- [Accessing Parameters and Data Sources](#)
 - [Creating a New Key](#)
 - [Parameters](#)
 - [Data Sources](#)
 - [Nested Queries](#)



Scripting Data Source

[Watch the Video](#)

Accessing Parameters and Data Sources

One of the main uses of the Scripting Data Source is to allow for data access and manipulation as the report is being generated, allowing you to replace the original results, or add new additional content.

The syntax for creating or referencing a data key in the Script datasource is described below:

Pseudocode - Data Key Syntax

```
data['keyName']
```

Order Matters

As mentioned on the [Report Data](#) page, the order of Data Sources determines which parameters and data sources they may reference. Because of this, it is **highly recommended** that Scripting Data Sources are placed at the bottom of the Data Sources list.

Creating a New Key

To create a new key that the report can use:

Python - Creating a New Key

```
data['newKey'] = "This key was generated on the fly!"
```

The type of data assigned to the key determines where it appears in the Design Panel.

- Simple data types, like strings and integers, will appear as **Parameters**.
- Datasets will appear as **Data Sources**.

Report Data

The screenshot shows the 'Report Overview' tab selected. On the left, there's a sidebar with 'Parameters' and 'Data Sources' sections. Under 'Parameters', 'StartDate', 'EndDate', and 'aString' are listed. 'aString' is highlighted with a blue selection bar. Under 'Data Sources', four items are listed: '1 Query - named_query', '2 Script - Motor Downtime', '3 Script - Year_Comparison_18', and '4 Query - alarm_journal'. On the right, a detailed view of 'aString' is shown: Parameter Name is 'aString', Type is 'String', and Default Value (expression) is '1 "I'm a string!"' with a note '2' below it.

Key Browser

The screenshot shows the 'Key Browser' window. It has a 'Show Calculations' checkbox. Below it, a tree view shows 'Parameters' containing 'StartDate', 'aString', and 'EndDate'. It also shows 'Data Sources' containing 'alarm_journal' and 'myResults', and a 'Built In' section.

String Data Sources can be given a name using the Name field.

The screenshot shows the 'Report Overview' tab selected. On the left, 'aString' is highlighted. On the right, a data source named 'Motor Downtime' is selected. The 'Name' field contains 'Motor Downtime' with a red border. Below it, a code snippet starts with '1 def updateData(data, samp...'. A note says 'This function has the data map.' and 'Arguments:' followed by 'data: This is a ma... be sequences'.

Parameters

Where the 'keyName' is the name of your data key. Thus, reading the value of a parameter, such as the initial StartDate parameter, can be accomplished by using `system.date.getYear()` and an if-statement.

Python - Accessing a Parameter's Value

```
# Extract the year from the StartDate parameter and compare to 2017.  
if system.date.getYear(data['StartDate']) == 2017:  
    # Do work here if the year lines up.
```

Of course, we can write back to the key and override its value:

Python - Overriding the Default StartDate Parameter

```
# This line would override the default StartDate parameter by add adding a day.  
data['StartDate'] = system.date.addDays(data['StartDate'], 1)
```

Additionally, this allows you to expand the Accessing a Parameter's Value example above by creating a new key:

Python - Accessing a Parameter's Value

```
# Extract the year from the StartDate parameter and compare to 2017.  
if system.date.getYear(data['StartDate']) == 2017:  
    # Create a new key and assign it one value if our condition is true...  
    data['myResults'] = "The year is currently 2017!"  
else:  
    # ...or assign a different value if the condition is false. This way we can always assume the key  
    'myResults' exists.  
    data['myResults'] = "What year is it? Not 2017, that's for certain!"
```

Data Sources

Static CSVs

While uncommon, Static CSVs may be accessed in a Scripting Data Source. The syntax is similar to working with Parameters.

Python - Static CSV example

```
# Take a Static CSV data source, and replicate its contents in a new key.  
data['static_data'] = data['Area Data']
```

Query-Based Data Sources

Reading the contents of a query-based Data Source, such as a [SQL Query Data Source](#) or [Tag Historian Query](#), requires the `getCoreResults()` function, which returns the results in a [standard dataset](#):

Python - Accessing a Query-Based Data Source's Value

```
# Query-based Data Sources are slightly different than parameters, so we must use getCoreResults() to  
extract the data.  
rawResults = data['keyName'].getCoreResults()  
  
# getCoreResults() returns a dataset, so we can utilize getValueAt() and rowCount within our scripts.  
resultsCount = data['keyName'].getCoreResults().rowCount
```

When working with data sources, it is unusual to attempt to write back, since datasets are immutable. Instead, the preferred approach is to create a new key with the modified results.

Say we have a query data source named "**Area Data**" which contains four columns: **month**, **north_area**, **south_area**, and **t_stamp**. If we need to build a new data source without the **t_stamp** column, we can use the following code:

Python - Building a New Data Source

```
# build a header and initialize a pydataset
header = ['month', 'north_area', 'south_area']
filteredDataset = []

# get the results from the Area Data data source
rawDataset = data['Area Data'].getCoreResults()

# build the new pydataset out of only some of the Area Data's data keys
for row in range(rawDataset.rowCount):
    valCategory = rawDataset.getValueAt(row, 'month')
    valNorthArea = rawDataset.getValueAt(row, 'north_area')
    valSouthArea = rawDataset.getValueAt(row, 'south_area')
    filteredDataset.append([valCategory, valNorthArea, valSouthArea])

# convert the pydataset to a standard dataset
filteredDataset = system.dataset.toDataSet(header, filteredDataset)

# create a new data source with the filtered results
data['updated Area Data'] = filteredDataset
```

The screenshot shows the Report Overview interface with the 'Data' tab selected. On the left, there's a sidebar with 'Parameters' (StartDate, EndDate, aString) and 'Data Sources' (1 Query - named_query, 2 Script - Motor Downtime, 3 Script - Year_Comparison, 4 Script - filtered_dataset). The 'filtered_dataset' script is selected.

```

Name: filtered_dataset

1 def updateData(data, sample):
    """
    This function has the opportunity to add additional data into the report's
    data map.

    Arguments:
        data: This is a map, whose keys are report data keys and values should
              be sequences, maps, scalar values, or datasets to provide information
              to the report.
        sample: A flag that will be True if the report data being gathered is
                for a preview of the report. Use to avoid slow queries and calculate
                to keep previews quick.
    """

2
3     # build a header and initialize a pydataset
4     header = ['month', 'north_area', 'south_area']
5     filteredDataset = []
6
7     # get the results from the Area Data data source
8     rawDataset = data['Area Data'].getCoreResults()
9
10    # build the new pydataset out of only some of the Area Data's data keys
11    for row in range(rawDataset.rowCount):
12        valCategory = rawDataset.getValueAt(row, 'month')
13        valNorthArea = rawDataset.getValueAt(row, 'north_area')
14        valSouthArea = rawDataset.getValueAt(row, 'south_area')
15        filteredDataset.append([valCategory, valNorthArea, valSouthArea])
16
17    # convert the pydataset to a standard dataset
18    filteredDataset = system.dataset.toDataSet(header, filteredDataset)
19
20    # create a new data source with the filtered results
21    data['updated Area Data'] = filteredDataset
22

```

Nested Queries

What if our 'Area Data' query has a nested query called 'Area Details' that we would like to manipulate in a script? This is useful when using [Table Groups](#).

Python - Script data source for nested query

```

nested = data['Area Data'].getNestedQueryResults()      # Gets results from our parent query
subQuery = nested['Area Details']          # Gets the subquery we want -- there can be more than one
header = ['productName', 'cost', 'triple']
alteredDataset = []
for child in subQuery:
    children = child.getCoreResults()      # children is a dataset
    for row in range(children.rowCount):
        valProductName = children.getValueAt(row, 'productName')
        valCost = children.getValueAt(row, 'cost')
        valTimesThree = None
        if valCost != None:
            valTimesThree = 3 * valCost
        alteredDataset.append([valProductName, valCost, valTimesThree])

# convert the pydataset to a standard dataset

```

```
alteredDataset = system.dataset.toDataSet(header, alteredDataset)

# create a new data source with the altered results
data['Updated Area Details'] = alteredDataset
```

Static CSV

The Static CSV data source allows you to quickly craft a data source to use in your report. This data source is ideal when you need some test data to begin creating a new report. All values are brought in as strings, which means that you don't need quotation marks around any of the string values.

As the name implies, this Data Source is strictly static, and does not have any built-in means to update or import new values in. In these scenarios, you may want to look into the [Scripting Data Source](#) to import CSV directly into the report, or first store the data into a SQL database and retrieve the results with a [SQL Query Data Source](#) data source.

Format

The first line of the Static CSV will be a list of columns separated by commas. Each subsequent line after that will then be a row in the dataset, with each value separated by a comma as well. See the text below for an example dataset that has three columns and thirteen rows. You can double click the block to copy its text.

CSV - Creating a Custom Static CSV

```
Equipment, Time, Site
Motor, 15, Site A
Motor, 23, Site A
Conveyor Line, 148, Site B
Pallet Wrapper, 58, Site A
Motor, 96, Site C
Conveyor Line, 23, Site B
Palletizer, 40, Site B
Conveyor Line, 56, Site A
Pallet Wrapper, 45, Site C
Motor, 43, Site C
Conveyor Line, 87, Site D
Motor, 23, Site D
```

On this page ...

- [Format](#)



Static CSV

[Watch the Video](#)

A screenshot of the Report Overview interface. The top navigation bar includes Report Overview, Data, Design, Preview, and Schedule. The left sidebar shows Parameters (StartDate, EndDate) and Data Sources (1 Static CSV - static_data). The main panel displays the "Data" section under "static_data", showing a list of 13 rows of data. The last row, "Motor, 23, Site D", is highlighted with a yellow background.

Related Topics ...

- [Nested Queries](#)

Nested Queries

What Are Nested Queries?

The simple definition is that a *Nested Query* uses the results of a previously executed query to collect data. The general structure of a Nested Data Source is one in which you have a Parent query and child queries. Each child query can then use the columns from the parent query as parameters using {columnName}. For example, if a parent query had a column called machineID, the child query can then use {machineID} as a parameter, and the child query will run for each row of the parent query, using the different values of machineID for each run. Those well-versed in SQL are probably thinking that this sounds like a JOIN and in fact, there are some similarities. There are also some major differences which allow Nested Data to be both easier and more powerful:

- Nesting relationships are not restricted to data in a single schema, database or even source! Nesting is easy to configure across tables, between different databases, or even with sources like the Tag Historian!
- Writing queries for nested query sources can be far simpler and easier to maintain than writing complex JOIN operations.
- Nested structures allow more control in how data is collected, allowing data structures and relationships that are more expressive.

On this page ...

- [What Are Nested Queries?](#)
- [How Nesting Works](#)
- [Special Considerations](#)
- [Nested Query Example - Equipment Downtime](#)



INDUCTIVE
UNIVERSITY

Nested Queries

[Watch the Video](#)

How Nesting Works

Let's use a simple data relationship to help illustrate how nesting occurs. Imagine we have data collected from two unrelated sources that look like the ones seen in this table.

Codes		Frequency		
CodePK	Code	FrequencyID	CodeID	FreqValue
1	ZG	1	3	11
2	GB	2	1	41
3	DC	3	2	13
		4	3	26
		5	1	13
		6	3	32
		7	1	11

We want to create a data source connecting all these things for reporting using nesting. The trick is to identify where the two datasources connect. You may notice in the data above The CodePK column in the Codes datasource matches up with the CodeID column of the Frequency datasource. This is where we will connect the two datasources. We will make our Codes datasource the parent and the Frequency the child. The queries would look something like below.

Pseudocode - Select Statement Examples

```
-- Parent Query
SELECT CodePK, Code FROM Codes
```

```
-- In the Child Query, accessed by clicking on the Frequency leaf of the Nested Queries Tree
-- This assumes the value of Parameter 1 would be equal to {CodePK}
SELECT FrequencyID, FreqValue FROM Frequency WHERE CodeID = ?
```

The screenshot shows the SSIS Data Editor interface. The top navigation bar includes tabs for Data, Design, Preview, and Schedule. The Data tab is selected.

SQL SELECT Query:

```
SELECT
    FrequencyID
    ,FreqValue
FROM
    Frequency
Where
    CodeID = ?
```

Data Key:

- Frequency

Query Type:

- SQL Query

Database:

- <Default>

SQL Query Builder:

- Show Builder
- Universal

Preview Limit: 100

Nested Queries:

- Codes
 - Frequency

What this means for our resulting data is that the parent query is called first, and a set of results is returned to the parent query, named Codes. After this data has been retrieved, the Child query will execute, once for each row of the parent, substituting the value of CodePK into the child query where we have the {CodePK} reference.

The resulting data will have a structure like this:

Pseudocode - Code Frequency Structure

```
Codes
|-Row 1
| |_ CodePK - 1
| |_ Code - ZG
|     | Frequency
|         |-Row 1
|             |_ FrequencyID - 2
|             |_ FreqValue - 41
|         |-Row 2
|             |_ FrequencyID - 5
|             |_ FreqValue - 13
|         |-Row 3
|             |_ FrequencyID - 7
|             |_ FreqValue - 11
|-Row 2
| |_ CodePK - 2
| |_ Code - GB
|     | Frequency
```

```

|           | -Row 1
|           |   |_ FrequencyID - 3
|           |   |_ FreqValue - 13
|
|-Row 3
|   |_ CodePK - 3
|   |_ Code    - DC
|       Frequency
|           | -Row 1
|           |   |_ FrequencyID - 1
|           |   |_ FreqValue - 11
|           | -Row 2
|           |   |_ FrequencyID - 4
|           |   |_ FreqValue - 26
|           | -Row 3
|           |   |_ FrequencyID - 6
|           |   |_ FreqValue - 32

```

Each row returns the CodePK and Code from the parent query, but also the results of the child query, that apply to the CodeID we get from the parent PK.

Now, in our Report Design, we will have access to our datasource `Frequency` that linked these two sets of data together through a shared value.

Special Considerations

Nested Queries are powerful and easy to use, but users should be aware of runtime implications. Imagine the scenario above, where we have two sets of data, each with 5000+ rows. When our child query executes, each row of its query is going to require a lookup from the parent. For most common sets of data and database sizes, this won't be an issue, but it's possible to imagine that instead of just one child query, we have a dozen.

In addition, some of those children *also* have many children. It's very easy to see in this scenario how exponential growth occurs and our system performance may suffer. Most report designers will limit query sizing as oversized data structures are simply not as easy to work with. However, if you feel an urge to generate massive complex trees of million line queries, you may be waiting a while.

Nested Query Example - Equipment Downtime

Here in this example, we have two database tables: an equipment table, which contains all of the equipment we have, as well as a downtime table, which contains a list of every downtime event. The tables are put together like the tables below.

Equipment_Table			Downtime_Table			
id	Name	Description	id	Equipment_id	Cause	Minutes_Down
25489	Conveyor Line	Transfers product	1	25489	Backup	22
55684	Labeler	Makes labels	2	55684	Out of labels	25
88456	Palletizer	Makes pallets	3	55684	Stuck labels	15
626145	Filler	Fills tanks	4	88456	Misalignment	38
			5	626145	Overflow	50
			6	25489	Scheduled Maintenance	12
			7	626145	Scheduled Maintenance	40
			8	88456	Misalignment	55

1. Create two tables in the database that are similar to the tables listed above.
2. Open a report, and navigate to the **Data Panel**.
3. Click the **Plus**  icon and add a **SQL Query** to the data sources.
4. Change the **Data Key** property to **Equipment**.
5. Type the query below into the query area. Here we are pulling out three columns of our equipment table, and giving each column a more descriptive name. Pay close attention to the spelling of the aliased names (names following the **AS** keywords), as we will have to reference **EquipmentIDNumber** later in our subquery.

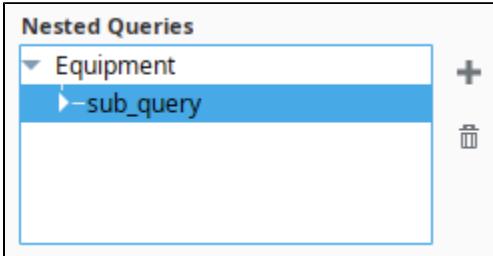
SQL - Equipment Table Query

```

SELECT id AS EquipmentIDNumber,
       Name AS EquipmentName,
       Description AS EquipmentDescription
FROM Equipment_Table

```

6. On the right of the Nested Query section, click the Plus  icon to create a sub_query.



7. Select the sub_query, and rename the Data Key property to rename it. In this example, we will use the name **EquipDowntime**.
 8. Type the following query into the query area.

SQL - Downtime Table Query

```

SELECT cause AS DowntimeCause,
       minutes_down AS DowntimeMinutes
FROM Downtime_Table
WHERE Equipment_id = ?

```

Here we pull in our downtime table, but we only need the cause and minutes_down, since we are already grabbing the Equipment_id from the first query. The WHERE clause is where we link this query to the parent with the equipment ids.

9. In the **Parameter 1** field, type the following:

Expression Language - Referencing a Parameter

```
{EquipmentIDNumber}
```

Note that we are directly referencing one of the aliased column names from the **Equipment** query.

10. That is all it takes to make a Nested Query! We now have two separate tables being called and linked together by the equipment id.
 11. To check that your queries worked, go to the Preview tab and look at the XML data that comes up. You should have a data set inside (indented) each row of Equipment data.

If you'd like, you can continue with this example and either use this data [in a table group](#), or in a [nested chart](#).

Related Topics ...

- [Table Groups](#)
- [Charts Inside of Tables](#)

Report Design

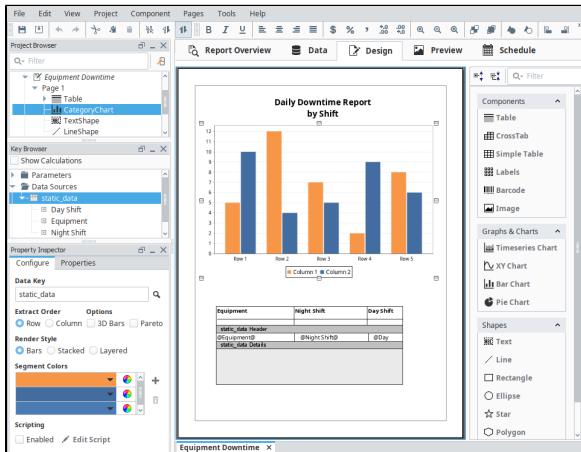
Report Designer Interface

The Report Workspace's Design tab lets you design reports with the same intuitive feel, familiar layout, and drawing tools that you get when designing windows or views in Ignition. Just like designing Vision windows and Perspective views, you can design reports using the same familiar drag-and-drop method and choose from a library of graphs, charts, tables, and images.

Looking at the screenshot of the Report Designer, you'll notice that an open Report Resource in the **Project Browser** can be expanded to provide information at a glance. This tree lets you visualize the relationship and hierarchy of Report Design elements on the page. Find an element in the tree by selecting items on the Design Panel, or find an element on the page by selecting it from the tree. As you do so, you'll notice that the bottom left of the default Report Designer will change to provide configuration panels and/or property tables depending on the selected item.

Just below the Project Browser is the **Key Browser**, which provides the **Data Keys** we use to reference data in our report. The Key Browser is home to all your Data Sources, **Parameters**, as well as a number of built-in calculation keys to speed you through the report design process. When you configure a data source and switch to the Design tab, a sample of all your queries is run on the Gateway to get information about the structure of your data. Columns in your data are represented as children in the Key Browser tree.

The right side of the panel is where the component palette lives, providing a number of powerful components, charts, and shapes which are used to build the layout and visualization of your report.



On this page ...

- [Report Designer Interface](#)
- [Report Design Components](#)
 - [Report Objects](#)
 - [Reporting Components](#)
 - [Reporting Charts](#)
 - [Shapes](#)
- [Selection and Alignment](#)
 - [Super Selection](#)
 - [Multiple Selection](#)
 - [Resizing and Moving Objects](#)
 - [Alignment](#)
 - [Shift Drag](#)

**INDUCTIVE
UNIVERSITY**

Report Design Tab

[Watch the Video](#)

Report Design Components

The Design Panel has a Report Design Palette with a host of components, charts, and shapes that help you design and create meaningful, informative, and professional reports. **Tables** are used to collect and store varying amounts of data. Sophisticated charts can be created from the data collected in a table. **Images** and shapes can add that extra touch of class and polish making a report stand out.

Just like designing in Vision windows and Perspective views, Report Design components have properties which are editable in the Property Inspector which is located on the lower left side of the Design Panel. Below is a brief overview of the report design tools you can choose from for building your reports.



Some property changes don't take effect in the Design panel

Note that modifying some component properties, such as the **Font** or **Fill Color**, will immediately cause a visual change in the **Design Panel**.

However, other properties, such as the **Date Format**, **Number Format**, and **Overflow Behavior** properties on a Text Shape component will not cause an immediate change. The properties described here wait until the report executes. Because of this, you will only see the results of these changes in the **Preview Panel**.

It is highly recommended that you review your report in the Preview Panel often. This provides an opportunity to visualize what the resulting report will look like.

Report Objects

Report Objects are the actual report and page components, and each have their own properties. They have a host of settings and properties that allow you to customize the appearance of your reports based on your requirements.

- **Report Object** - Allows you to set the report properties for paper size, dimensions, orientation, margins, and much more.
- **Page Object** - Allows you to add and remove pages in your report, as well as configuring additional properties that can add that creative touch to your reports.

Reporting Components

The Reporting Components are the foundation for creating useful reports. There are variety of different tables that you can choose from based on the type of data you're collecting. There are even Label and Barcode components you can use to create shipping and product labels. Here is a brief description of all the report design components.

- **Table** - Allows you to display tabular data in a variety of ways using the Table properties. Tables are configurable and highly flexible giving Reporting users the ability to flexibly layout and organize tabular data. They also support advanced grouping and pagination.
- **CrossTab** - Similar to a Table component, the CrossTab component is commonly used to summarize the relationship between two categories of data by showing summaries of cross sections of the data source. The CrossTab component creates a dynamic pivot table from your data.
- **Simple Table** - A basic grid table that dynamically creates new rows and columns for rows returned by the Data Keys on the component.
- **Labels** - Used to print out a host of different types of labels such as mailing labels, name tags, and more. You can create a set of labels using preset or custom sizes.
- **Barcodes** - Allows reports to contain dynamically generated barcodes. The following formats are supported: Code 128, Code 39, QR Codes, PDF 417, Aztec, Data Matrix, to name a few. Refer to **Barcode** in the Appendix section to see all the supported formats.
- **Images** - Can be embedded into reports. They can be dropped onto reports, pulled as binary data from a data source, or even a URL.

Reporting Charts

Report charts allow you to display your data in a variety of graphical ways. There are four types of charts so you can show the data any way you like. Below is a brief description of the different types of charts you can choose from to build your reports.

- **Timeseries Chart** - A type of line chart whose domain, or X-Axis, represents a timestamp or datetime. The Timeseries Chart is a great way to display Tag History.
- **XY Chart** - Similar to the Timeseries chart with the main difference being the domain, or X-Axis which is a numeric value and not a date.
- **Bar Chart** - A very easy chart to use that provides a bar representation of any numeric values, and because it has many properties, it can greatly impact how the data is presented. It is useful for displaying series with relative values.
- **Pie Chart** - Displays items as pie wedges. It represents relative quantities as wedges of a circle.

Shapes

You may have seen some of these shapes when designing your Vision windows and Perspective views. The shapes you see in the Report Design Palette are shape tools. They are not dragged from the design palette on to your report, instead you select a shape tool and begin to draw your shape on to your report. You create its size, height, and shape. Some of the shape tools allow you to create various shapes as well as edit them. Just like components, once a shape is created, you can change its fill color, stroke color, stroke style, and more. Each shape has its own properties. Here is a list and brief description of the available shapes in the Report Design Palette.

- **Text Shape** - Creates a text area for static or data key bound content.
- **Line** - Creates a straight line. It can run north-south, east-west, or diagonally.
- **Rectangle** - Creates a square or a rectangle. Once a rectangle or square is created, you can use the handles to change its height and width.
- **Ellipse** - Creates circles and ellipses. Once a ellipse or circle is created, you can use the handles to change its size.
- **Star** - Creates a star or a polygon. Once the star is created you can use the handles to resize it.
- **Polygon** - Creates custom polygon shapes by drawing and connecting the lines. Once all the lines are connected to form the polygon shape, you can use the handles to change its size. Double click on any of the individual lines to change the size and shape of the selected line.
- **Pencil** - Draws freehand shapes with smooth paths.

To learn more about report design components, refer to the [Report Design Tools](#) section, and to the Appendix for details on each component.

Selection and Alignment

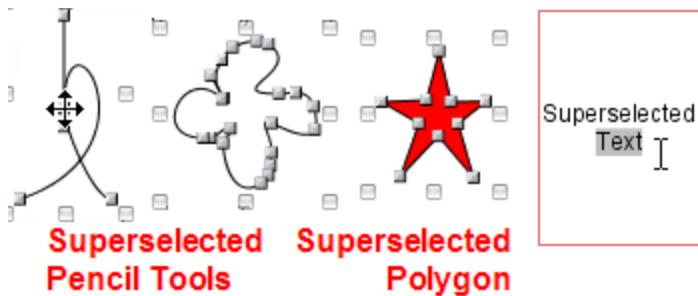


Selecting components is done with the selection tool (), which is the default tool in the Report Designer whenever another tool is not activated.

Reporting has a "deeper" selection model than the Ignition Designer. Simple object selection is done by single clicking an object, and is typically used to move an object around or resize it. "Selecting deep" is done by double-clicking on the object to get into the report hierarchy. For instance, if you group two rectangles together, you can select the individual rectangles by double clicking into the group. Visualizing selection is simplified by viewing the Project Brower to see which node in the Report Resource tree is selected.

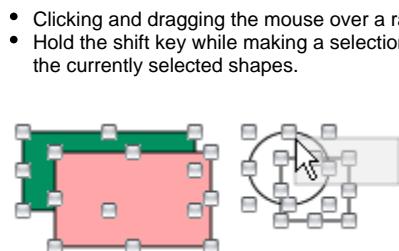
Super Selection

Super Selection refers to an editing state that some shapes go into when double clicked. Text is the most common of these. When a text box is selected you can move and resize it. When it's super-selected, you can place the text cursor or select a range of characters and insert or delete text. The polygon and pencil are two other basic tools that support Super Selection.



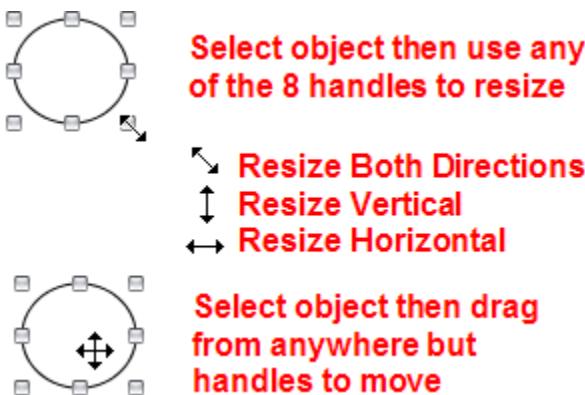
Multiple Selection

Multiple Selection of components can be done two ways:



Resizing and Moving Objects

To resize an object, first select it with a single click, then left or right click, and drag one of the 8 resizing handles, and drag one of the 8 resizing handles to resize. To move an object, left or right click on the shape, and drag it anywhere on the report. Both resizing and moving operations support shift-dragging.



Alignment

Alignment is accomplished by selecting multiple objects, then choosing from any of the **Make** commands in the **Component Menu** to align objects. You can align objects in rows by their top, center or bottom border, and align columns by their sides or center. You can make objects the same size (width and height), and equally space rows and columns horizontally and vertically.

In report design, z-order defines relative order of objects when they overlap. Select the object and click **Bring to front** or **Send to back** in the Component Menu to reorder the objects.

Shift Drag

Holding the shift key while you drag shapes around your report will constrain movement to horizontal, vertical, or 45 degrees.

Related Topics ...

- [Tutorial: The Report Workflow](#)

- Report Charts
- Report Tables

In This Section ...

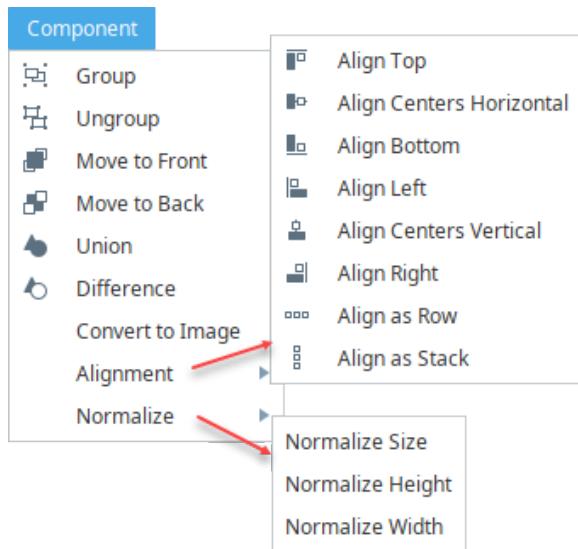
Report Design Tools

Object layout is an important aspect in creating professional reports. The Reporting Module provides a host of intuitive tools and functions to help you design and edit reports. Ignition Reporting uses a WYSIWYG (what you see is what you get) approach.

Report Menus

The Ignition Designer menubar provides quick access to many common Reporting Design functions. The Component and Pages menu items are added to the menubar when the Report Design Panel is open. Many of the menus, such as [File](#), [Edit](#), and [Tools](#) are similar to other areas of the Designer. Reporting-specific Toolbar and Menu functions are described below.

Component Menu



The Component menu allows you to modify the layout of objects in a report. To perform any of the Component menu functions, you must first select an object or multiple objects before you're able to perform the function.

On this page ...

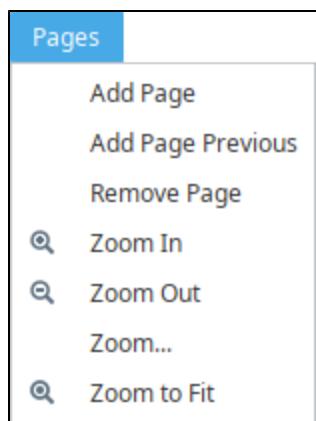
- [Report Menus](#)
 - [Component Menu](#)
 - [Pages Menu](#)
- [Report Design Toolbar](#)
- [Shape Tools](#)
 - [Text Shape](#)
- [Headers and Footers](#)

Menu Item	Function
Group/Ungroup	Grouping allows you to merged selected shapes and makes them behave as one with respect to: selection, moving, and resizing. To "drill down" to individual objects, superselect the grouped object. Ungroup separates grouped shapes.
Move to Front /Move to Back	All shapes have an order on the page that determines what is drawn on top when two shapes overlap. These options allow you to move a selected shape either forward or backward. This is also known as the Z Order.
Union/Difference Paths	Takes multiple overlapping shapes (such as a rectangle and an oval) and combines them into a single shape using the combined paths. A powerful tool to construct complex shapes.
Convert to Image	Converts the report page to an image.
Align Selected Items - Make Row Top/Center /Bottom	Quickly align several shapes in a row, either by their top, center, or bottom border. This is particularly useful when shapes are of different heights.
Align Selected Items - Make Column Left /Center/Right	Same as above, but for columns, aligning their sides or center.
Align as a Row /Stack (column)	Equalizes the distance between shapes horizontally or vertically.

Normalize

Make several shapes the same width, height or both.

Pages Menu



The Pages menu allows you to add or remove pages in the report. Adding and removing pages in the Design Panel is easy. Under the **Page menu**, click on either **Add Page** to add a page after the currently selected page, or **Add Page Previous** to add a page before the currently selected page. To remove a page, open the page you want to remove and select **Remove Page**.

Menu Item	Function
Add Page	Adds a page to the current open document, after the currently selected page.
Add Page Previous	Adds a page to the current open document, before the currently selected page.
Remove Page	Removes the currently selected/visible page in the current open document.
Zoom In	Increase the size of the report in view.
Zoom Out	Decrease the size of the report in view.
Zoom	Enter the percentage to zoom to.
Zoom to Fit	Have the report fill the available space.

Report Design Toolbar



The Toolbar provides a variety of functions to assist in editing while designing a report. Here is a complete list of editing functions and their descriptions in the table below.

Toolbar Item	Function
B	Toggles bold of the selected text.
I	Toggles <i>italic</i> the selected text.
U	Toggles <u>underline</u> of the selected text.
L	Aligns the selected text left.
C	Centers selected text.

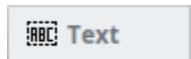
	Aligns the selected text right.
	Justifies the selected text.
	Adds a \$ to the data key.
	Adds a % to the data key.
	Forces a decimal place on an integer value in a text shape.
	Add the number of allowed decimal digits.
	Subtract the number of allowed decimal digits.
	Increase the size of the report in view.
	Decrease the size of the report in view.
	Have the report fill the available space.
	Move the selected components to the back of the Z-order.
	Move the selected components to the front of the Z-order.
	Union: alters the first selected shape to become the union of all selected shapes.
	Difference: alters the first selected shape such that the other shapes are subtracted from it.
	Align the left edge of a group of components.
	Align the right edge of a group of components.
	Align the top edge of a group of components.
	Align the bottom edge of a group of components.
	Align two or more components centerpoints' horizontally.
	Align two or more components centerpoints' vertically.
	Align three or more components in a row.
	Align three or more components in a stack.
	Combine the selected components in a group.

	
	Break apart the selected group into its child components.

Shape Tools

You may have seen some of these same shapes tools when designing your Vision windows or Perspective views. The shapes you see in the Report Design Palette are shape tools: Line, Rectangle, Ellipse, Star, Polygon, and Pencil. Shape tools allow you to create various shapes as well as edit them to create other shapes. They behave a little differently from the typical drag and drop function that you perform on a Vision window. They are not dragged from the design palette on to your report, instead you select a shape tool on the design palette to make the tool active and begin to draw your shape on to your report. You create its size, height, and shape. You'll also notice the Property Inspector will change to reflect that specific tool's properties once the tool is active. Just like components, once a shape is created, you can change its fill color, stroke color, stroke style, and more. Each shape has its own properties. To learn more about shapes, refer to the section on [Report Design](#).

Text Shape



The [Text Shape](#) is worth mentioning here because it is a fundamental component used to create text within a report. The Text Shape is used for report titles, customized page headers and footers, and of course, any additional text you want to add to your report. Simply drag a Text component on to report and begin typing. You can move the Text component around, expand it, or shrink it by using the handles when the component is selected. You can even change the font type, size, and color. The Text Shape also has a lot of properties associated with it, including Data Key Format Properties for Date and Number formats by choosing a format from the list of available templates.

Headers and Footers

Creating headers and footers is just like creating any other set of objects in your report. There is no explicit header or footer functions, or specific area on the report where they belong. Typically, headers and footers are placed at the top and bottom of the first page respectively, using the Text component or the [Built-In Keys](#).

When designing your report, the key is sizing and positioning of components within your page around your header and footers. If you have a lot of data in your data table, your report will automatically create additional pages. Each new page that the table creates will have the same header and footers.

You can use the Text component to customize your header and footers or you can use the Built-In Keys from the Key Browser. The Built-in Keys allow you to drag and drop keys to your header and footer areas such as Path, Timestamp, Date, Page number, and more. So place your header and footers where you want them on the first page, and the Report Designer will automatically place them on all additional pages.

Related Topics ...

- [Data Keys](#)

Stroke and Fill Properties for Reports

As you probably learned working with Perspective and Vision components, each component has its own unique properties. The [Report Design components](#) are no different. The Report Designer has some of the same components like tables, bar charts, graphs, and various shape tools, but their properties are different from the Perspective and Vision components because they are designed to go in reports. For example, Stroke and Fill properties control the appearance of the reporting components so you can change stroke style or border, create a dashed line around the shape, change the width of the lines and the color, and more. Stroke Style and Stroke properties can be particularly useful when using a Table in a report. You can make objects in a table and add an outline or set any edges of they outline to show to make your report stand out.

Stroke and Fill Properties

Stroke and Fill properties are commonly used by many reporting components. All of the components, charts, graphs, and shapes in the Report Design Palette use the same Stroke and Fill properties. These properties affect how the components look in a report. There are five basic Stroke and Fill properties:

- **Fill** - if this property is marked true, the shape will fill its space with color.
- **Fill Color** - if the Fill property is selected, the color you select will fill the shape.
- **Opacity** - defines how opaque the color is in the shape, between 0 and 1.
- **Stroke Style** - defines what type of stroke or border to use.
- **Stroke** - An expandable list of properties for the chosen stroke style.



The Fill, Fill Color, and Opacity properties are pretty easy to understand, so we are going to focus on Stroke Styles and Stroke properties in the following sections so you can take full advantage of how to use these properties to make professional looking reports.

Stroke Styles

Stroke Style defines what style of stroke or border to use. Primarily, the Stroke Style controls the type of line drawn around the component (table, chart, shape, etc) that you are using. There are four pre-configured Stroke Style templates you can choose from.

- **Hidden** - no stroke or border
- **Shape Outline** - a solid line outline
- **Border** - border or rectangle shape
- **Double** - two parallel lines

Each of these Stroke Styles come with their own set of properties. In the Design panel, select a component, and choose any one of the style templates from the Stroke Style property in the Properties tab. You can use the Stroke Style as is, or double click on any of the rows or cells to change its individual properties. For example, you might want to change the thickness of the lines or create a dashed line around the border of a component. This is easy to do: select the component, row, or cell, and change the Width under the Stroke property (thickness of the lines is specified in pixels). Then view your report with the Preview tab. You can keep repeating this step until you get the desired result.

Stroke Properties

Each Stroke shares some properties with other strokes, but have their own unique properties. The only exception is that the **Hidden** Stroke Style does not have any properties. **Note:** If you switch between stroke styles your values will be overwritten by defaults. This is not true when switching to the Hidden type.

The properties for each Stroke Style are shown below.

Shape Outline Stroke Properties

Shape Outline is a solid line outline around a component or shape. It has the following properties that you can use as is, or change.

Property	Description
Color	Color to use for the stroke.

On this page ...

- [Stroke and Fill Properties](#)
- [Stroke Styles](#)
- [Stroke Properties](#)
 - [Shape Outline Stroke Properties](#)
 - [Border Style Stroke Properties](#)
 - [Double Style Stroke Properties](#)

Dash Pattern	A "Dash Pattern" string to specify the number of pixels on and off. For example "5,10"
Width	Width of the shape in pixels.

Stroke Style - Shape Outline Example

This example shows a report with a Shape Outline around the Table component at the bottom of the image. Select the whole table to set stroke and fill for it. The property values used to create the outline around the data are shown in the properties list in the table below.

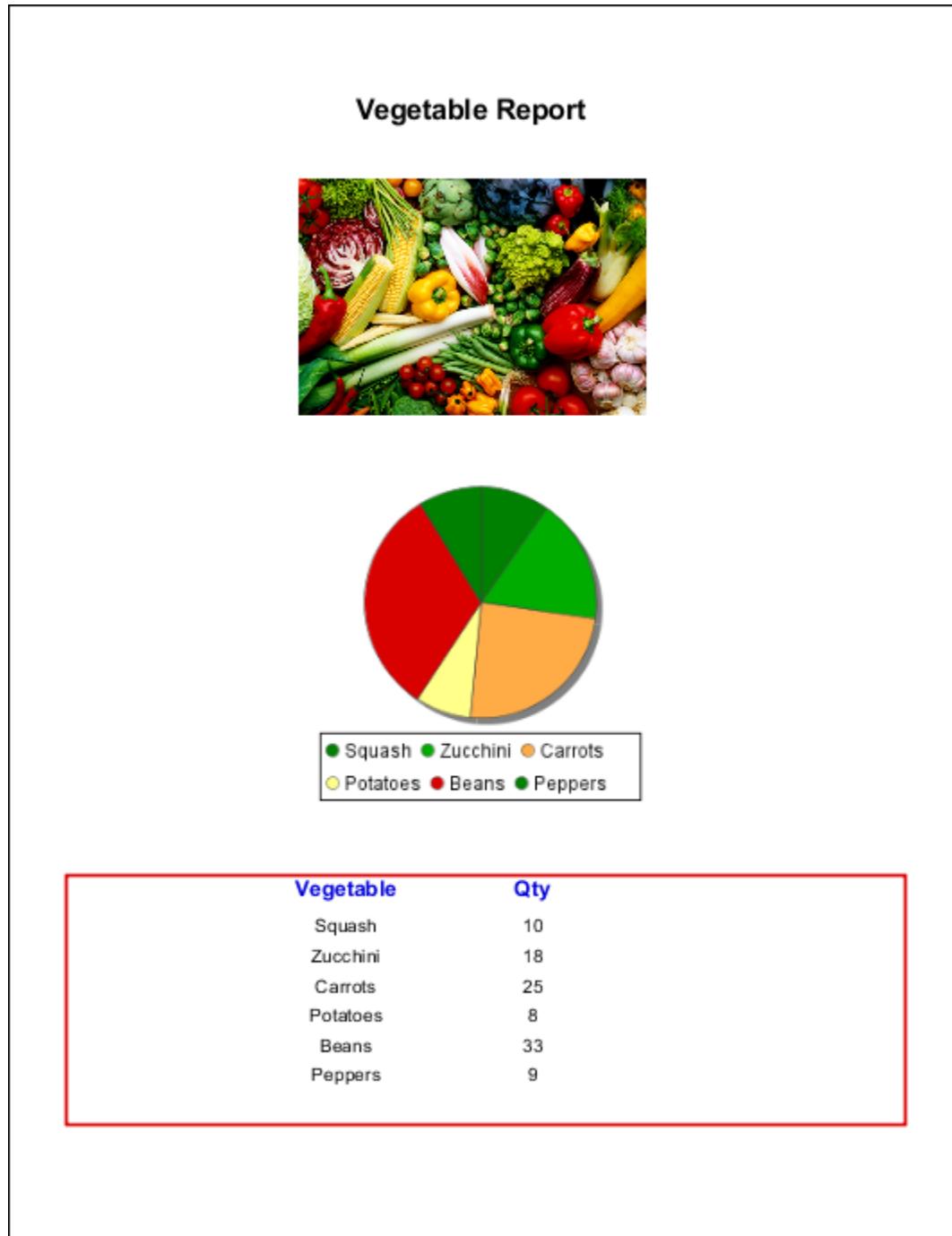
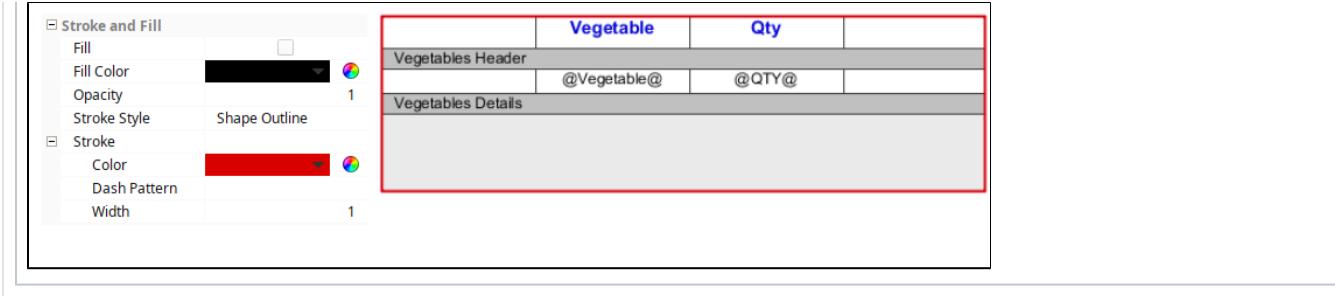


Table Stroke and Fill Properties



Border Style Stroke Properties

The **Border** Stroke Style is similar to the Shape Outline, but it gives you the option to disable the any of the lines around the border; bottom; left; right; or top.

Property	Description
Color	Color to use for the stroke.
Dash Pattern	A "Dash Pattern" string to specify the number of pixels on and off. For example "5,10"
Border Bottom	If true, show the border on the bottom side of the shape.
Border Left	If true, show the border on the left side of the shape.
Border Right	If true, show the border on the right side of the shape.
Border Top	If true, show the border on the top side of the shape.
Width	Width of the stroke in pixels.

Stroke Style - Border Example

This example used the Border Stroke Style for the Equipment Summary cell at the bottom of a Table. (Total: @total.Downtime@Minutes). Three images appear inside the outline because they are in the Equipment Header ([unstructured](#)) row. The property values are shown for the Table, Row, and Cell in the property lists below.

Equipment



Labeler

	Out of labels	Duration (Minutes)	Date
	50		Jan 20, 2017
	21		Feb 12, 2017
		71 Minutes	
	Misalignment	Duration (Minutes)	Date
	33		Jan 20, 2017
		33 Minutes	

Filler

	Overflow	Duration (Minutes)	Date
	22		Feb 20, 2017
	30		Feb 28, 2017
		52 Minutes	
	Maintenance	Duration (Minutes)	Date
	19		Feb 25, 2017
		19 Minutes	

Conveyor Line

	Backup	Duration (Minutes)	Date
	45		Jan 21, 2017
		45 Minutes	
	Maintenance	Duration (Minutes)	Date
	47		Feb 13, 2017
		47 Minutes	

Total: 267 Minutes

Equipment Table in the Designer

Equipment

Equipment Header Standard ▾

@Equipment@

Equipment Details Standard ▾

@Cause@	Duration (Minutes)	Date
---------	--------------------	------

Cause Details Standard ▾

	@Downtime@	@T_stamp@
--	------------	-----------

Data Details Standard ▾

@total.Downtime@ Minutes	
--------------------------	--

Data Summary Standard ▾

	Total: @total.Downtime@ Minutes	
--	------------------------------------	--

Equipment Summary Standard ▾

Table Property Values

Table Stroke and Fill Properties	Property	Value																								
Stroke and Fill <table> <tr> <td>Fill</td> <td><input type="checkbox"/></td> </tr> <tr> <td>Fill Color</td> <td>█</td> </tr> <tr> <td>Opacity</td> <td>1</td> </tr> <tr> <td>Stroke Style</td> <td>Border</td> </tr> <tr> <td>Stroke</td> <td></td> </tr> <tr> <td>Color</td> <td>█</td> </tr> <tr> <td>Dash Pattern</td> <td></td> </tr> <tr> <td>Border Bottom</td> <td><input checked="" type="checkbox"/></td> </tr> <tr> <td>Border Left</td> <td><input checked="" type="checkbox"/></td> </tr> <tr> <td>Border Right</td> <td><input checked="" type="checkbox"/></td> </tr> <tr> <td>Border Top</td> <td><input checked="" type="checkbox"/></td> </tr> <tr> <td>Width</td> <td>1</td> </tr> </table>	Fill	<input type="checkbox"/>	Fill Color	█	Opacity	1	Stroke Style	Border	Stroke		Color	█	Dash Pattern		Border Bottom	<input checked="" type="checkbox"/>	Border Left	<input checked="" type="checkbox"/>	Border Right	<input checked="" type="checkbox"/>	Border Top	<input checked="" type="checkbox"/>	Width	1	Stroke Style	Border
Fill	<input type="checkbox"/>																									
Fill Color	█																									
Opacity	1																									
Stroke Style	Border																									
Stroke																										
Color	█																									
Dash Pattern																										
Border Bottom	<input checked="" type="checkbox"/>																									
Border Left	<input checked="" type="checkbox"/>																									
Border Right	<input checked="" type="checkbox"/>																									
Border Top	<input checked="" type="checkbox"/>																									
Width	1																									
	Stroke	<ul style="list-style-type: none"> • Border Bottom • Border Left • Border Right • Border Top 																								

Data Summary Cell

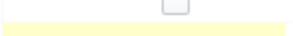
Shows the Total Downtime Minutes for each downtime cause for each piece of equipment.

Cell Text Properties for Data Summary	Property	Value
	Text Color	Red, Bold

Text Properties		
Text	@total.Downtime@Minutes	
Text Color		
Character Spacing		0
Coalesce Newlines	<input type="checkbox"/>	
Font	Arial Bold 12.0 (Arial)	
Horizontal Alignment	Left	
Line Spacing		1
Margin	1; 2; 0; 2	
Overflow Behavior	Grow row	
Underlined	<input type="checkbox"/>	
Vertical Alignment	Top	

Equipment Summary Cell

Shows the Total Downtime Minutes for all causes and for all equipment.

Cell Stroke and Fill Properties for Equipment Summary		Property	Value
Stroke and Fill		Stroke Style	Border
Fill	<input type="checkbox"/>	Fill	Enabled
Fill Color		Fill Color	Yellow
Opacity		Width	2 pixels
Stroke Style	Border		
Stroke			
Color			
Dash Pattern			
Border Bottom	<input checked="" type="checkbox"/>		
Border Left	<input checked="" type="checkbox"/>		
Border Right	<input checked="" type="checkbox"/>		
Border Top	<input checked="" type="checkbox"/>		
Width	2		

Double Style Stroke Properties

By default, the **Double** Stroke Style consists of two parallel lines around the component. You can choose to disable any of the lines at the bottom, left, right, or top. You can also change the width of both lines and the separation between these lines, if desired.

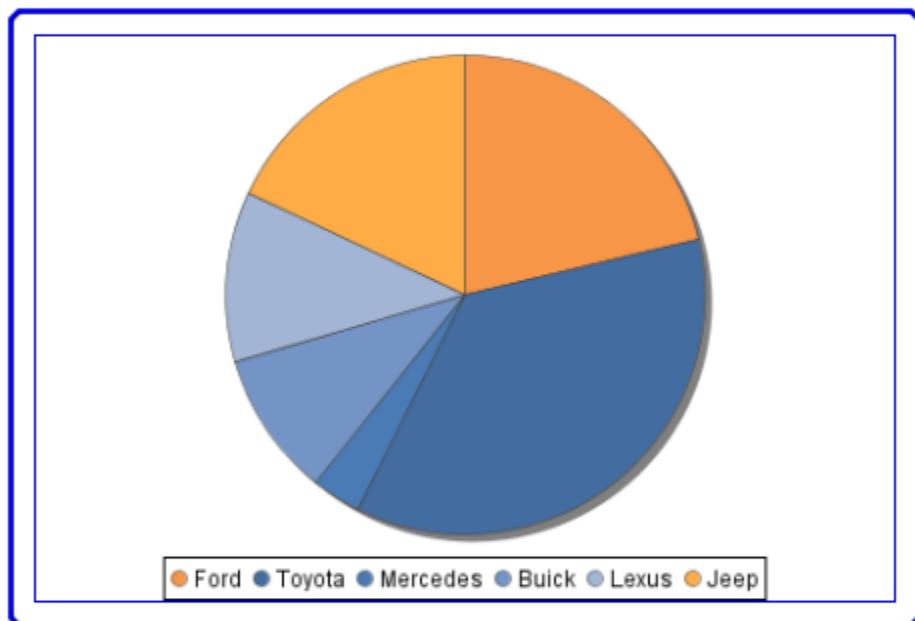
Property	Description
Color	Color to use for the stroke.
Inner Width	Width of the inner border in pixels.
Outer Width	Width of the outer border in pixels.
Position	Position of borders relative to the bounds of the shape. <ul style="list-style-type: none"> • Outer on path - places the outer border on the edge of the component and the inner border inside the component. • Inner on path - places the inner border on the edge of the component and the outer border outside the component. • Centered about path - centers the whole border (based on inner+outer+gap widths) on the shape edge. • Gap on path - centers on the gap on the shape edge.

Stroke Style - Double Example

Here is an example of a Pie Chart with a Double border used in a report. The property values for the Border Stroke Style are listed in the properties list table below.

Used Car Inventory Report

August 8, 2017



Pie Chart Stroke and Fill Properties	Property	Value
	Stroke Style	Double

Opacity	1
Stroke Style	Double
Stroke	
Color	
Inner Width	1
Outer Width	3
Position	Inner on Path
Separation	10

Stroke Color	Blue
Inner Width	1
Outer Width	3
Position	Inner on Path
Separation	10

Related Topics ...

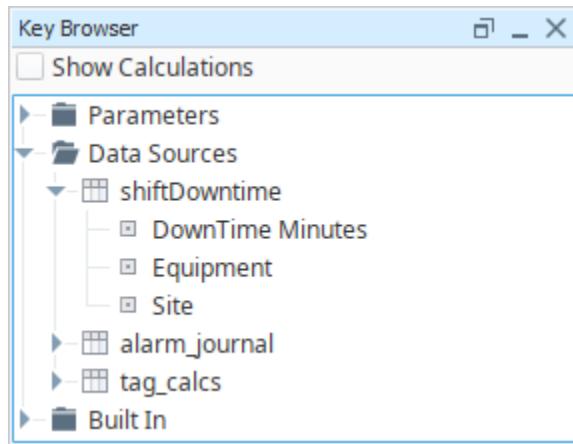
- [Report Design](#)
- [Report Design Tools](#)

Data Keys

Data Keys

In the Reporting Module, we use **Data Keys** to pull values from data sources and show them on the report. In simple terms, Data Keys are placeholders for your data. The simplest reference to data is a simple Data Key. At report generation time, these keys resolve to the values (or sets of values) provided by the data source. Additionally, Data Keys may be used as expressions, which are referred to Keychain Expressions.

As you add Parameters and Data Sources to the [Data section](#) of your report, they will appear in the Key Browser's **Parameters** and **Datasources** folders.



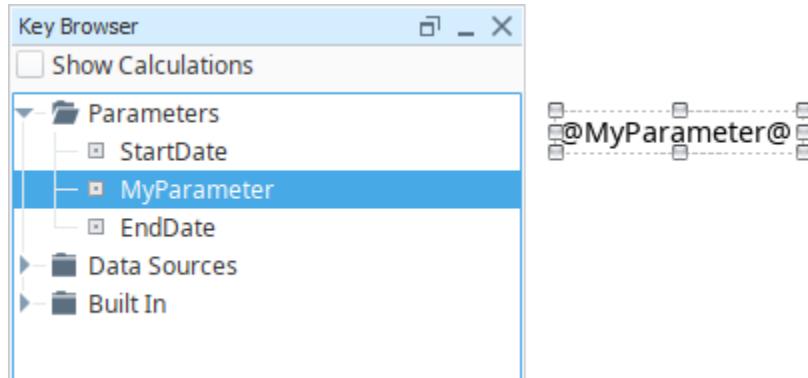
On this page ...

- [Data Keys](#)
 - [Data Keys on the Report](#)
 - [Escaping the @ Character](#)
- [Built-In Keys](#)
 - [Built-In Data Key Description](#)
 - [Show Calculations Property](#)
 - [Dynamic Data Keys](#)
 - [Configuring a Dynamic Data Key](#)
- [Data Key Usage](#)
 - [Data Keys as Paths](#)
 - [Array Index of Data](#)
 - [Colors in Expressions](#)

Data Keys on the Report

Data Keys are enclosed in the "@" character when utilized by components in the report. They may be typed manually, or dragged directly from the Key Browser.

Keys that contain a single value will create a [Text Shape](#) when dragged onto the report.



Keys that represent datasets will create a [Table component](#), and configure the Data Key property to use the key.

The screenshot shows two interface windows side-by-side. The window on the left is titled 'Key Browser' and contains a tree view of keys. The 'shiftDowntime' key is expanded, showing its sub-keys: 'DownTime Minutes', 'Equipment', and 'Site'. Below these are collapsed keys: 'alarm_journal' and 'tag_calcs', followed by the 'Built In' folder. A red arrow points from the 'shiftDowntime' key in the Key Browser to the 'Data Key' input field in the window on the right. The window on the right is titled 'Property Inspector' and has tabs for 'Configure Table' and 'Properties'. The 'Configure Table' tab is selected. Under the 'Data Key' section, the 'shiftDowntime' key is selected and highlighted in blue. Below it is a grouping section containing a single item, also labeled 'shiftDowntime', with a '+' icon to its left.

Escaping the @ Character

In some cases, your report may need to contain multiple strings containing the "@" character. Since these characters are used to denote keys, multiple instances of these characters may lead to undesirable behavior. You can escape this key lookup with "@@". For example, if a Text Shape needed to contain multiple email addresses, they could be typed in the following way:

```
user1@company.com
user2@company.com
```

When the report generates, the double "@@" characters will be replaced with a single "@" character.

Built-In Keys

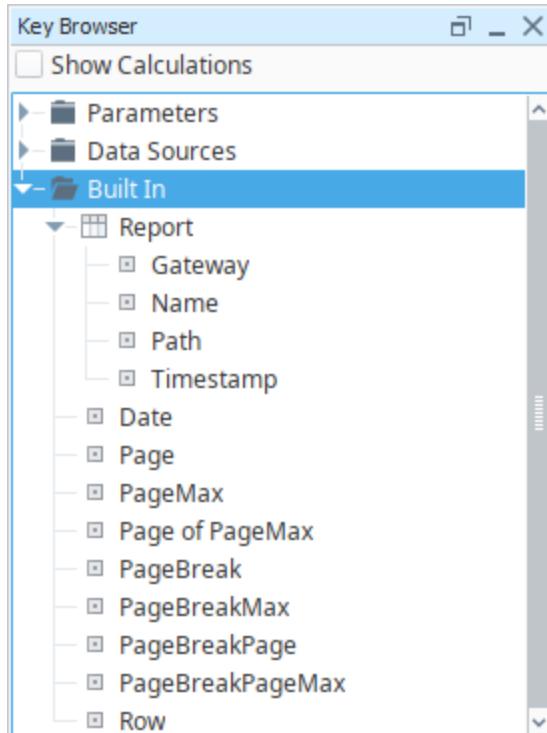
Built-In Keys provide a lot of useful information on your report at a glance. The Built-In keys are found in the Key Browser. Expand the Built-In folder and you'll see all the default keys, including a Report folder. The keys in the Report folder are specifically related to the report: Gateway name that the report is located in, report name, folder path from the Project Browser to the report, and the Timestamp of the Gateway. The other Data Keys are related to information you may want to add to a report like the date you are viewing or printing the report, page number, number of total pages, and more.

Here's a screenshot of the Key Browser showing all the default Built-In Keys that can be used on a report.



Built-in Keys

[Watch the Video](#)



The tables below show the Built-In Report Data Keys and Built-in Data Keys along with a brief description of each key.

Built-In Data Key Description

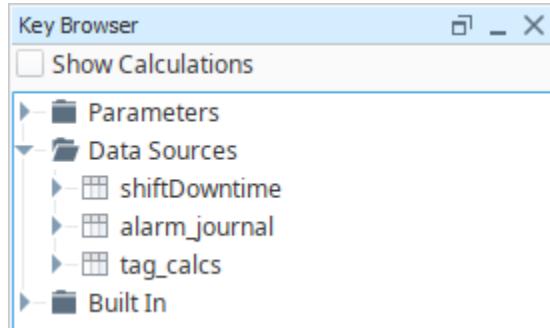
Key	Description											
Report	This key has multiple sub-keys that provide meta-data about the report.											
	<table border="1"> <thead> <tr> <th>Key</th><th>Description</th></tr> </thead> <tbody> <tr> <td>Gateway</td><td>Name of the Report's Ignition Gateway</td></tr> <tr> <td>Name</td><td>Name of the Report</td></tr> <tr> <td>Path</td><td>Path to the Report in the Project Browser</td></tr> <tr> <td>Timestamp</td><td>The Gateway's current timestamp</td></tr> </tbody> </table>		Key	Description	Gateway	Name of the Report's Ignition Gateway	Name	Name of the Report	Path	Path to the Report in the Project Browser	Timestamp	The Gateway's current timestamp
Key	Description											
Gateway	Name of the Report's Ignition Gateway											
Name	Name of the Report											
Path	Path to the Report in the Project Browser											
Timestamp	The Gateway's current timestamp											
Date	The current date/time											
Page	The current page											
PageMax	The total number of pages in the generated report											
Page of PageMax	Shows current page number and the total number of pages in the report											
PageBreak	The number of explicit page breaks encountered											
PageBreakMax	The total number of explicit page breaks in generated report											
PageBreakPage	The number of pages since last explicit page break											
PageBreakPageMax	The total number of pages in current explicit page break											
Row	Shows the current row number. Must be used in a table											

Show Calculations Property

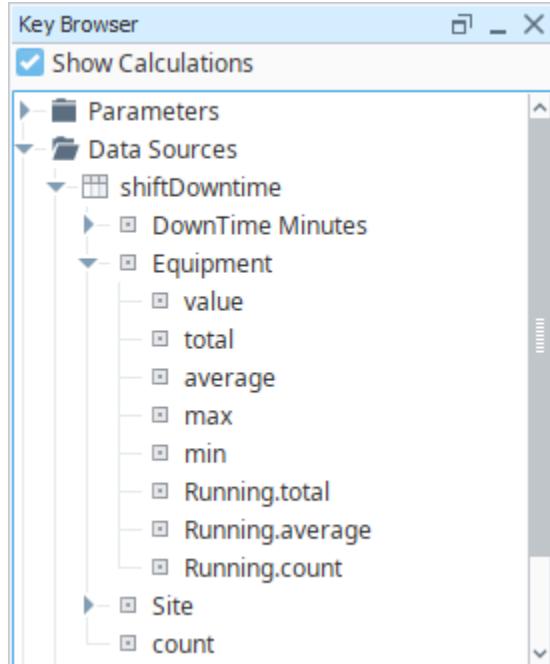
In the Key Browser, the **Show Calculations** property will add several aggregates to each key. These allow your reports to easily display things like the total of a key. These calculations are typically used in the [summary row](#) on the Table component.

Once Show Calculations is enabled, the Key Browser will refresh, and each key will be expandable. Expanding a key will show the available calculations.

Show Calculations Disabled



Show Calculations Enabled



Calculation Keys work like any other key: they may be dragged onto the report, and utilized in [Keychain Expressions](#).

Dynamic Data Keys

Normally, Data Keys may only be used to display the value of a key, such as the Text property on a [Text Shape](#) component. However, they can not be used in the same manner to modify other properties on a report component. Instead, you can utilize Dynamic Data Keys.

Dynamic Data Keys allow you to use the value of a Data Key on a non-string property. With Dynamic Data Keys, you can modify properties on report components, such as the background color or width, based on the value of a key. This is very similar to the binding system used by components in the [Vision Module](#).



Use Dynamic Data Key

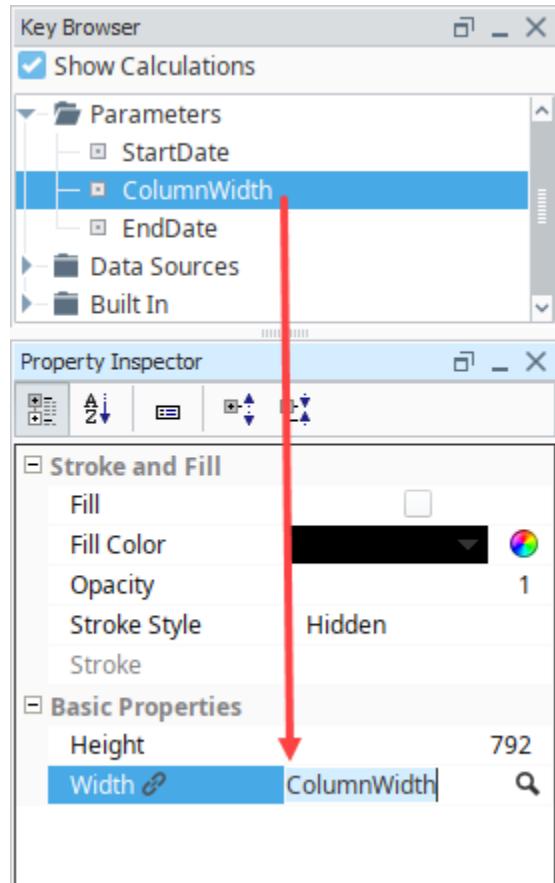
[Watch the Video](#)

Configuring a Dynamic Data Key

There are two ways to configure a dynamic data key. Note that the syntax of keys differs in Dynamic Data Keys: the "@" are omitted, as demonstrated below.

Drag-and-Drop

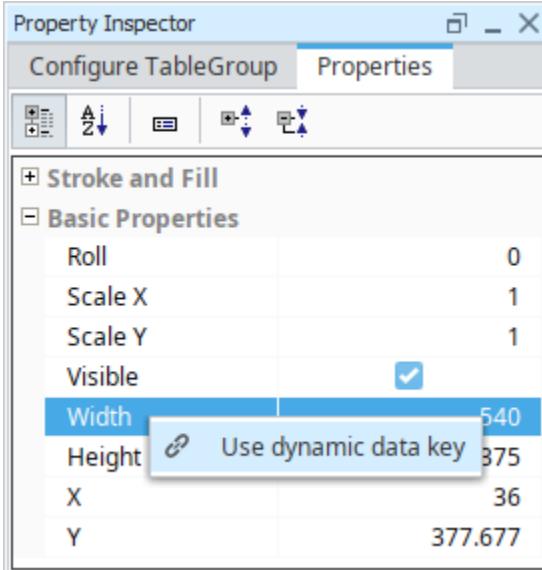
The easiest approach is by simply dragging a data key from the **Key Browser** directly to a property on a report component.



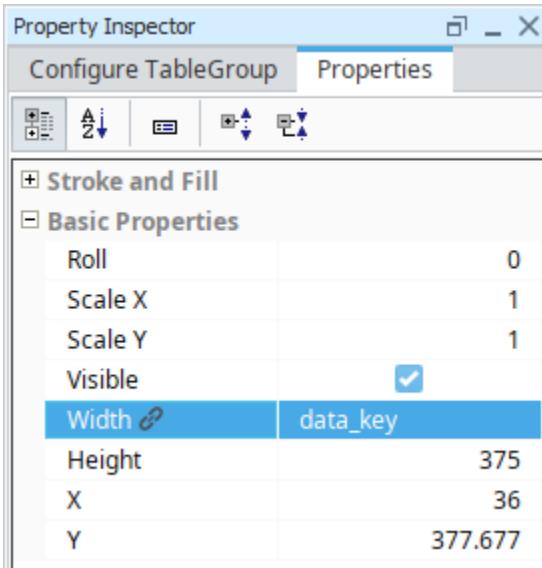
Right-Click

1. With a report component selected, look in the **Property Inspector**, and left click on the name of a property you wish to place the dynamic data key on.
2. Once selected, you can right-click on the property name and a menu will appear.

- Click on the **Use dynamic data key** menu item.



- This will place a dynamic data key on the property. An icon of a link () will appear next to the property name, and a default key will be applied to the property.
- Next we will want to override the default value with one of the keys from the Key Browser. Simply left click on the value field and a magnifying glass () icon will appear.



- Click on the icon. From here a popup of available keys will appear. Select the key you wish to use, and then click OK.

Data Key Usage

Data Keys as Paths

Data Keys are *relative*, and use 'dot notation' to reference children. Meaning, if we have a nested data structure, we can use Data Key paths (also known as *Keychains*) to reference the nested data. In the key browser image below, we have a nested data source called *Downtime*. *Downtime* contains a number of columns, and then contains a reference to additional data called *runInfo*. If we wanted to access the highlighted operator data, we could use the keychain dot notation in the Designer - `@Downtime.runInfo.operator@`. Nested data sources are outside the scope of this page, but you can learn about Data source nesting in the [Nested Queries](#) section.

Array Index of Data

You can reference an individual object in a list using standard array indexing syntax (brackets) like this: `@dataSource[0].columnName@`, where "dataSource" is a data source that contains a child data key named `columnName`. Assuming a data source with the values listed below, we can retrieve the value of "Second Row" by specifying index 1 and the column `stringValue`: `@static_data[1].stringValue@`

static_data Example

```
indexColumn, stringValue
0, "First Row"
1, "Second Row"
2, "Third Row"
4, "Fourth Row"
```

The screenshot shows the Keychain application interface. At the top, there are three tabs: 'Data' (disabled), 'Design' (highlighted with a red box), and 'Preview'. Below the tabs, there is a tree view of data structures. Under the root node, there is a node labeled `@static_data[1].stringValue@`. In the 'Preview' tab, which is also highlighted with a red box, the value 'Second Row' is displayed.

Colors in Expressions

Colors may be references in Keychain Expressions in several ways.

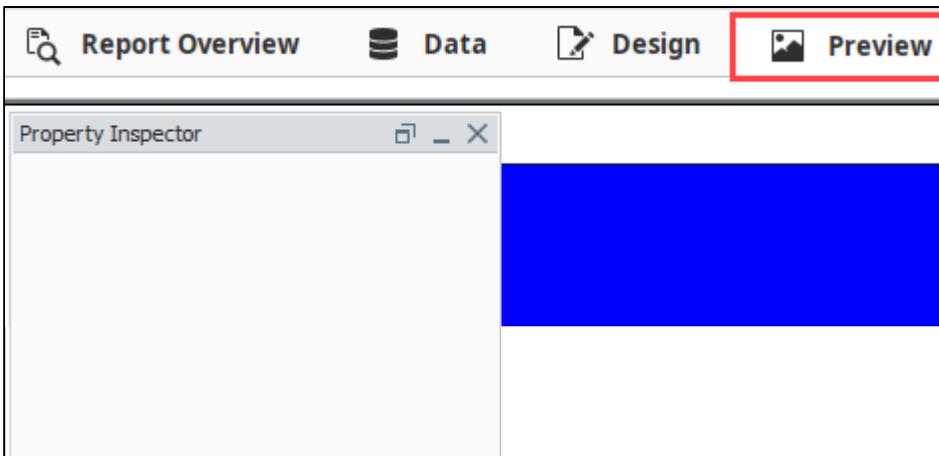
Colors in Hexadecimal

First, hexadecimal case-insensitive color codes may be used. The code must be wrapped in quotation marks to be evaluated correctly. Note that the color change will only appear when the report is executed. The easiest way to test the expression is to switch to the Preview Panel.

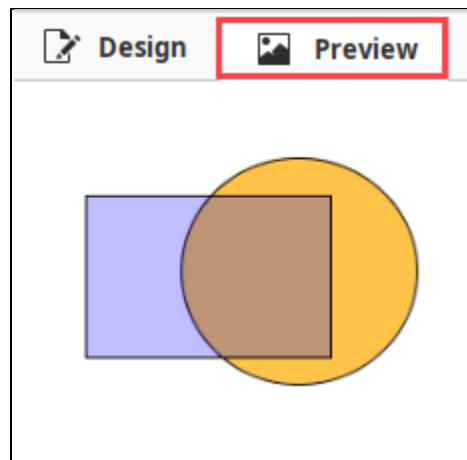
Below we see a Blue hexadecimal code of "0000FF" is used on the Fill Color of a Rectangle. The Fill Color on the Rectangle was originally set to White. Because the expression will not evaluate in the Design Panel, the Rectangle will appear as a White color.

However, switching over to the Design Panel will generate the report, and evaluate the expression. This in turn returns a Blue Fill Color.

The screenshot shows the Keychain application interface with four tabs at the top: 'Report Overview' (disabled), 'Data' (disabled), 'Design' (highlighted with a red box), and 'Preview'. Below the tabs, there is a 'Property Inspector' panel. In the 'Stroke and Fill' section, the 'Fill' checkbox is checked, and the 'Fill Color' field is set to "0000FF".



The 7th and 8th digits may be added to specify an alpha channel, or the opacity of the color: **00** is fully transparent, while **FF** is fully opaque. Below, we see a similar rectangle overlapping an ellipse, but with a code of "**0000FF40**". This represents ~25% opacity, so objects behind the rectangle will be visible, and the fill color will only be slightly opaque.



Parameters as Colors

You may also leverage Report Parameters to specify colors. This typically involves creating a parameter with a string datatype, and using the `color` expression function. Once created, you can simply create a dynamic data key reference on the property by dragging the parameter from the **Key Browser** onto the property in the **Property Inspector**. This way, you can have several components use the same color, and modify the color in a single location.

A screenshot of the Report Overview interface. The 'Data' tab is highlighted with a red border. On the left, there is a sidebar with sections for 'Parameters' and 'Data Sources'. Under 'Parameters', there is a list of parameters: 'StartDate', 'EndDate', and 'colorParameter'. The 'colorParameter' item is selected and highlighted with a blue background. On the right, there is a configuration panel for the 'colorParameter'. It includes fields for 'Parameter Name' (set to 'colorParameter'), 'Parameter Type' (set to 'String'), and 'Default Value (expression)' (set to '`1 color(0, 250, 0)`', which is highlighted with a yellow background). There are also up and down arrows for reordering the parameters.

The screenshot shows two interface windows. The top window is 'Key Browser' with a tree view containing 'Parameters' (expanded), 'StartDate', 'colorParameter' (selected and highlighted in blue), and 'EndDate'. The bottom window is 'Property Inspector' with tabs 'A Z' and 'Basic Properties' (selected). Under 'Stroke and Fill', the 'Fill Color' field is set to 'colorParameter'. Other settings include 'Fill' checked, 'Opacity' at 1, 'Stroke Style' as 'Hidden', and 'Stroke'.

Strings as Colors

Additionally, case-insensitive string color names may be used to return a color. Again, the value must be wrapped in quotation marks.

The screenshot shows the 'Property Inspector' window with the 'Basic Properties' tab selected. Under 'Stroke and Fill', the 'Fill Color' field is set to the string value "'orange'". Other settings include 'Fill' checked, 'Opacity' at 1, 'Stroke Style' as 'Hidden', and 'Stroke'.

The following string values may be used:

String Value Color Reference

Value	Example	Value	Example	Value	Example	Value	Example	Value	Example
"beige"		"gold"		"lavender"		"pink"		"tan"	
"black"		"goldenRod"		"lightGray"		"plum"		"teal"	
"blue"		"gray"		"lime"		"powderBlue"		"violet"	
"brown"		"green"		"magenta"		"purple"		"white"	
"crimson"		"hotPink"		"maroon"		"salmon"		"yellow"	

"cyan"		"indigo"		"navy"		"silver"		"clear"	Zero opacity. Similar to disabling the Fill property.
"darkGray"		"ivory"		"olive"		"skyBlue"			
"fuchsia"		"khaki"		"orange"		"red"			

In This Section ...

Keychain Expressions

It's possible to perform calculations on data keys. This section documents the various operators and functions that are available.

Keychain Expressions

Keychains have their own expression language that is largely similar in syntax to Java. To separate this language from others in Ignition, we refer to it as a Keychain Expression.

Keychain Expressions are configured by simply utilizing any operators or functions within the "@" characters. Assuming a key named "myKey" with a value of 10, we can multiply its value by 10 with the following expression:

```
//Expression  
@myKey * 10@  
  
//Output  
100
```

Again, note that the "*" operator and multiplier are enclosed in the "@" characters. Of course, we can also use a different key as the multiplier. Assuming we have another key named "myMultiplier" that has a value of 5:

```
//Expression  
@myKey * myMultiplier@  
  
//Output  
50
```

Any characters outside of the "@" characters are not part of the expression, so you can easily add prefixes and suffixes with static text.

```
//Expression  
Total: @myKey * myMultiplier@ gal  
  
//Output  
Total: 50 gal
```

Additionally, you can utilize separate Keychain Expressions in the same TextShape. Note that "myUnits" (which has a string value of "gal") is enclosed in a separate set of "@" characters, since it is a separate expression:

```
//Expression  
Total: @myKey * myMultiplier@ @myUnits@  
  
//Output  
Total: 50 gal
```

Dynamic Data Key Expressions

Keychain Expressions may also be used with Dynamic Data Keys. The syntax and operators work exactly the same, except there is no need to type the "@" characters. Because of this, the entire field is treated as a single keychain expression.

On this page ...

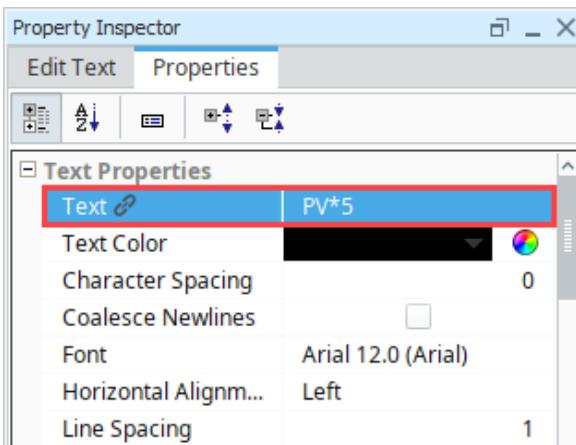
- [Keychain Expressions](#)
 - [Dynamic Data Key Expressions](#)
- [Conditional Keychain Example](#)
- [Operators and Functions](#)
 - [Operators](#)
 - [Math Functions](#)
 - [String Functions](#)



INDUCTIVE
UNIVERSITY

Key Calculations

[Watch the Video](#)



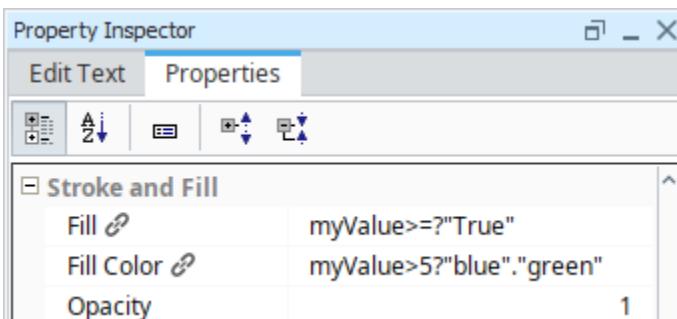
Conditional Keychain Example

The following example demonstrates an if-statement using a Dynamic Data Key. This allows us to highlight different values or ranges contextually, making important values stand out.

Assuming a key named "myValue" has been created, and contains a numerical value, we can use the following syntax:

```
#If the value of the "myValue" key is greater than 5, a blue color will be returned. Otherwise, a green
color will be used.
@myValue>5?"blue":"green"@
```

In the image below, the Fill property is also using a dynamic data key, so the Fill Color will be disabled if myValue is less than 1, Blue if myValue is between 1 and 4, and Green if greater or equal to 5. Note that in the Property inspector, the @ symbols are not needed.



To add more color-value pairs, we simply add more if statements to the end of the expression with a colon:

```
#If the value of "myValue" will determine one of multiple colors:
#Greater than 10 will return Red
#greater than 5 (but not greater than 10) will return Blue
#anything else will return Green.
@myValue>10?"red":myValue>5?"blue":"green"@
```

Operators and Functions

Operators

The following operators may be used in a Keychain expression.

Operator	Function	Example
Parenthesis	(expr) Nested	Any portion of a Key Chain can be enclosed with parenthesis to guarantee precedence.

	expressions	
Multiplicative	*, /, % Multiply, divide, modulo	These are the most common and intuitive operators. You might want to display @quantity*price@ in an invoice line-item or calculate a percent like this @profit/revenue*100@.
Additive	+, - Add, subtract	See multiplicative above
Relational	>, <, >=, <= Greater-than, less-than, greater/less-than-equal	These are most useful for conditionals: @amount>=0? "Credit" : "Debit"@ or @name=="this"? "that" : name@
Equality	==, != Equal, not-equal	See Relational above
Logical	AND &&	These operators make it possible to test multiple conditions: @revenue>100 && budget<50? "Winner!"@ or @name=="Jack" name=="Sam"? "Good Name!"@.
Logical	OR	See and above
Conditional	? : If/then - with form "expr? true_expr : false_expr"	Provides IF/THEN/ELSE expressions. Note: a false expression is optional. 'null' will be evaluated to false and non-null as true. You can provide null substitutions like this: @name? name : "(None provided)"@. You can also nest conditionals for more conditions. For example, @age>=21?"Adult":(age>12?"Teen":"Child")@.
Assignments	=, +=	For the brave, you can create temporary variables for use in a report. Most of the functionality you might use this for is covered in more intuitive ways (such as the Running key), but it is possible to define a variable in a header row: @revTotal=0@ and update it in details rows @revTotal+=revenue@.

Math Functions

The following functions return floats.

Menu Item	Function
floor(float)	Round input down to the nearest whole number.
ceil(float)	Round input up to the nearest whole number.
round(float)	Round input to the nearest whole number.
abs(float)	Returns the absolute value of the input (if number < 0 return number * -1).
min(float, float)	Returns the input number with the least value.
max(float, float)	Returns the input number with the greatest value.
pow(float, float)	Returns first number to the second number power.

String Functions

The following functions return strings.

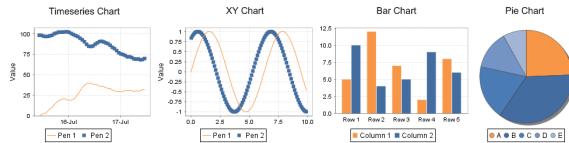
Menu Item	Function
startsWith(String, String)	Returns true if the first string starts with the second.
endsWith(String, String)	Returns true if the first string ends with the second.
substring(String, int start)	Returns a substring of String beginning at position start.
join(List aList, String aKeyChain, String aDelimiter)	Used to display an individual attribute of individual objects as a single String. Suppose you have a list of movies and want to show their titles in a comma separated list: @join(getMovies, "getTitle", ",")@
substring(Object aString, int start, int end)	Obtain a subset of a given string. This could be useful if you wanted to restrict a text field to a certain number of chars:@substring(title, 0, 10)@

Report Charts

Report Charts

Report charts allow you to display your data in a graphical way, just like the charts in the rest of Ignition. Charts can be driven by any [data source](#) in a report, or even [embedded into table rows](#) using nested data sets. There are four types of charts so you can show the data any way you like.

- **Timeseries Chart:** a simple chart that plots values against a timestamp X axis. Great for showing historical trending.
- **XY Chart:** similar to the Timeseries chart, this chart plots lines on X-Y axes, but is configured to show numeric categories on the X axis.
- **Bar Chart:** a bar chart that uses text categories on the X axis. Can be configured as a Pareto chart.
- **Pie Chart:** a basic pie chart that uses text categories. Percentages are automatically calculated.



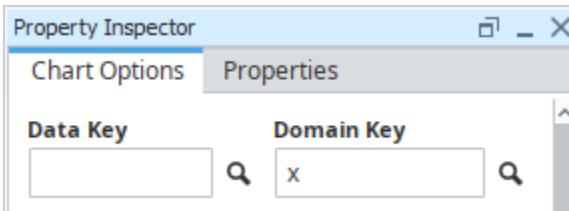
On this page ...

- [Report Charts](#)
- [The Data Key Property](#)
- [Timeseries Chart](#)
 - [Usage](#)
- [XY Chart](#)
 - [Usage](#)
- [Bar Chart](#)
 - [Usage](#)
- [Pie Chart](#)
 - [Usage](#)
- [Chart Scripting](#)

The Data Key Property

Similar to Tables in reporting, all charts must **first** be assigned a Data Key. This assignment configures the chart to look at specific keys in a data source, and prevents any name collisions with other data keys: if multiple data sources in your report return a column named "id", the chart wouldn't know which column you were referring to without this initial assignment. Like all keys in the Report manual, make sure your Domain and Range keys are just the column name. If your keys look like "query.column" then you will not see data in your chart.

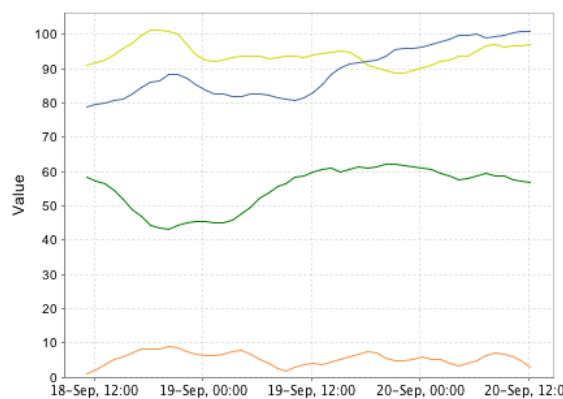
Assignment can be achieved by drag-and-dropping a key into the Data Key property, or by clicking the Key Search button.



Once assigned, you can start adding pens or bars to your chart.

Timeseries Chart

The [Timeseries Chart](#) is a type of XY chart whose domain or X-Axis represents time series data and range can be one or more pens. The Timeseries Chart is a great way to display data visually from Tag History, or similar time related data sources.



Timeseries Chart

[Watch the Video](#)

Usage

To use the chart, drag the component from the Report Palette to your report. Type or drag a data key from the Key Browser into the **Data Key** field of the Chart Options tab in the Property Inspector. Select a time series domain (for example a `t_series` column of your query). The Y axis (value) can be modified in the Chart Options tab, but the X axis (time range) is based on the data in the 'Data Key' that powers the chart dataset.

To add pens to your query, simply click the button on the Chart Options tab next to the Pens table in the Property Inspector. You can either double click a pen, or select a pen and click the button on the right side, to navigate to the Pen Configuration area. Click the to return to the Chart Options tab.

Chart Options area

The Chart Options area shows the following configuration:

- Pens:** A table with columns Range Key, Pen Name, and Preview. It contains three rows: sine0 (orange wavy line), sine1 (blue wavy line), and sine2 (light blue wavy line).
- Axes:** A table with columns Name, Label, Auto?, Low, and High. It contains one row: Default Axis (Value, Auto? checked, Low: 0, High: 100).
- Scripting:** A section with an Enabled checkbox and an Edit Script button.

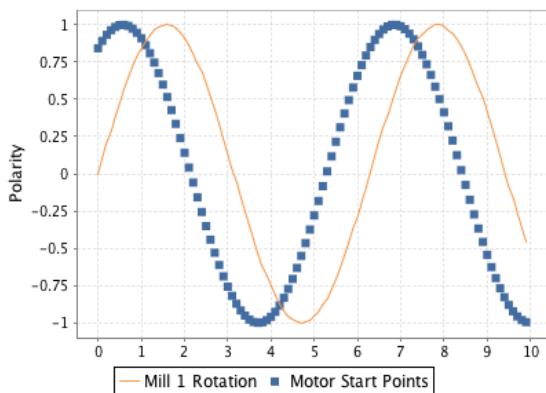
Pen Configuration area

The Pen Configuration area shows the following settings for the sine0 pen:

- General:** Data Key: sine0, Pen Name: sine0, Axis: Default Axis.
- Style:** Color: orange, Style: Line, Dash Pattern: solid, Line Weight: 1, Shape: square, Fill Shape?: true.
- Preview:** A small preview window showing the orange wavy line.

XY Chart

The **XY Chart** generates an X vs Y plot of your data. XY Charts can have multiple pens and axes per data source, and each is easily configurable in the Chart Options tab for the component.



**INDUCTIVE
UNIVERSITY**

XY Chart

[Watch the Video](#)

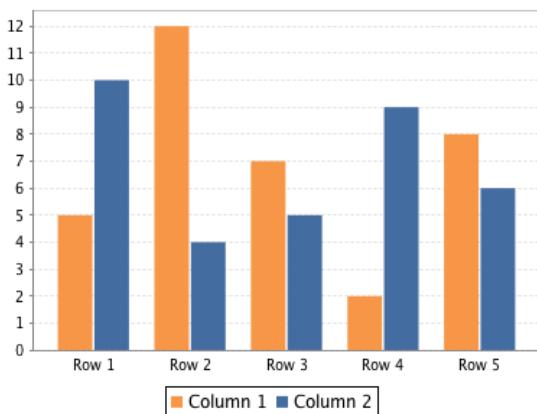
Usage

To use the chart, drag the component from the Report Palette to your report. Type or drag a data key from the Key Browser into the **Data Key** field of the Chart Options tab. Select Domain Key to use for the X axis.

To add pens to your query, simply click the button on the Chart Options tab next to the Pens table in the Property Inspector. You can either double click, or select a pen and click the button on the right, to navigate to the Pen Configuration area. Click the to return to the Chart Options tab. The setup and Configuration of this chart and its pens works similarly to the Timeseries Chart, the difference being that instead of using a date datatype for the domain, a different datatype is needed.

Bar Chart

The **Bar Chart** component can be used to add bar charts to a report. It uses text categories on the X axis, and can also be configured as a Pareto chart.



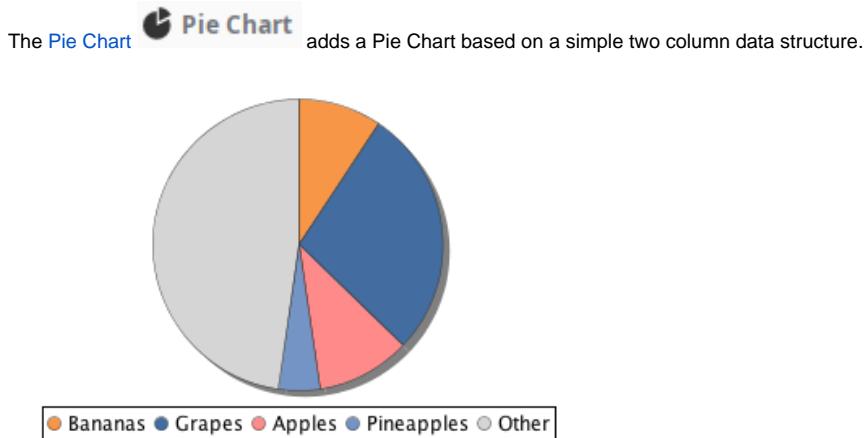
Bar Chart

[Watch the Video](#)

Usage

To create a Bar Chart, simply drag the component from the Report Palette and drop it onto your report. Bar Charts are quite easy to use and have a large number of customization options. Configuring a Bar Chart requires a data source whose first column generally contains the name or identifier the bar represents, and the following one or more columns represent some numerical value to be plotted. A Bar Chart example can be found in the [Report Workflow Tutorial](#).

Pie Chart

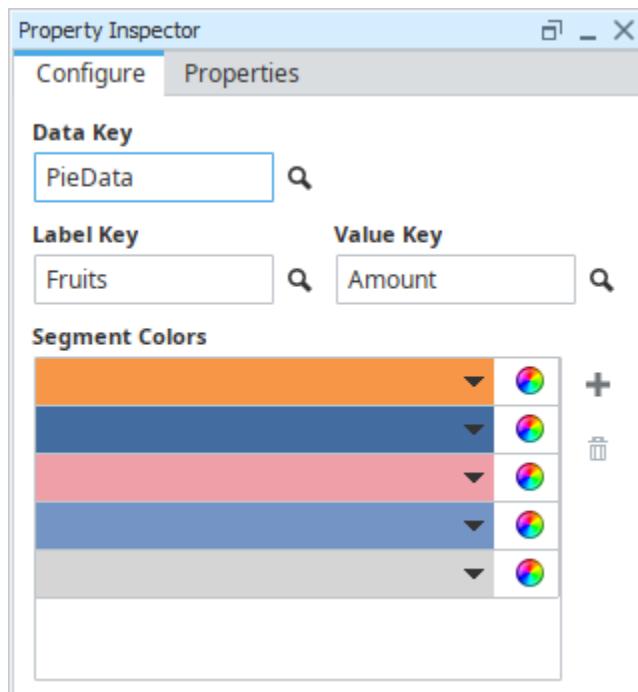


Pie Chart

[Watch the Video](#)

Usage

To create a new Pie Chart, drag the pie chart palette item from the Report Design Palette onto your Report Page or Parent Shape. A pie chart has a simple configuration consisting of label values (generally Strings) and numeric quantity values. Color segments can be added or removed by clicking the button on the Configure tab.

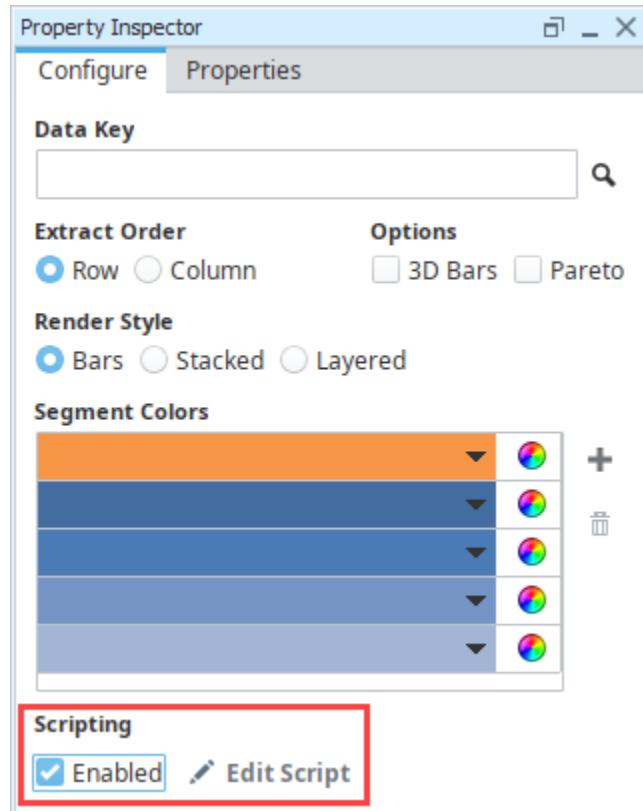


The Pie Chart in the images above was created using a simple data source which had just two columns, the first which represented our Label, and the second being a column of values.

```
Fruits, Amount
"Bananas", 52
"Grapes", 154
"Apples", 58
"Pineapples", 25
"Other", 265
```

Chart Scripting

The look and feel of the Timeseries, XY, and Bar charts may be modified through scripting. Scripting may be accessed on these charts by selecting them, and then clicking the **Edit Script** icon in the Property Inspector. Note that the **Enabled** property must be checked for the script to become active.



Clicking on **Edit Script** will provide access to **configureChart**, which allows you to make modifications to the chart right before the report is rendered. The charts are simply JFreeCharts, so [reading through the JFreeChart API](#) would be useful here.

Related Topics ...

- [Report Tables](#)
- [Tutorial: The Report Workflow](#)

In This Section ...

Getting Started with Report Charts

Report charts have a ton of functions and features that it's hard to know where to begin. You'll be amazed how simple a standard chart is to create. To start with, there are several chart types to select from, but the chart type you choose depends on the type of data you have and how that data might best be graphically displayed. Next, charts are driven by specific data keys in a data source. This page provides a quick start to getting you well on your way to creating report charts that are easy to configure and look stunning in your reports. Let's get started!

Creating a Report Chart

One of the most common charts used for historical trending is a [Timeseries Chart](#). The Timeseries Chart plots values against a timestamp X axis.

Adding a chart to a report involves a couple of steps. In order, they are:

1. The chart needs data, so we must create a data source.
2. Once a data source exists, we can create the chart and assign the data source.
3. Apply any additional chart configuration, such as adding an additional axis.

Now, let's create a Timeseries Chart using Tag History data.

On this page ...

- [Creating a Report Chart](#)
 - [Creating a Data Source](#)
 - [Adding a Chart](#)
 - [Adding an Axis](#)



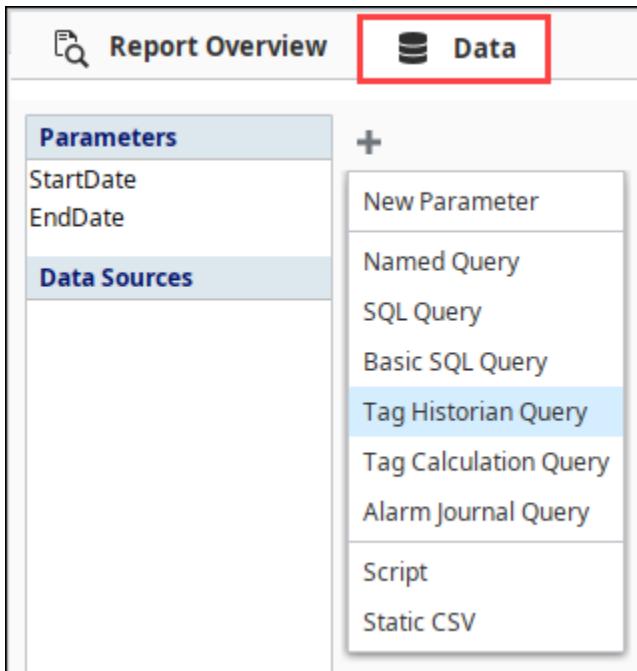
Before you start this example...

Ignition comes with one internal Tag Provider so you can get started right away, but first you need to connect to a device. The [Programmable Device Simulator](#) is a great starting point if your Gateway is not currently connected to a PLC. This example assumes you named your device 'generic' are using the Generic Program device tags, but any tags could be used instead.

You also need a [database connection](#) that the historian can use to store Historical Tag data. This example assumes [Tag History has been enabled](#) on the **Sine** tags.

Creating a Data Source

1. In the **Data** tab, under Parameters and Data Sources, click the plus button to add a data source. Select **Tag Historian Query**.



2. Under the **Available Historical Tags** column, open the folder structure until you locate your Tags. As mentioned above, we are using the Programmable Device Simulator and loaded the Generic program. so we will select the 'generic' folder in this example. Here you'll find a bunch of Tags you can use immediately. Expand the **Sine** folder.

3. Select a few of the Tags (i.e. sine0, sine1, sine2) and drag them to the **Selected Historical Tags** column under **Tag Path**. Take a look at the properties down at the bottom. Set the **Data Range** property to **Historical**, and the **Aggregation Mode** to **Time-weighted Average**.

The screenshot shows the 'Data' tab of the Report Overview interface. On the left, there are sections for 'Parameters' (StartDate, EndDate) and 'Data Sources' (1 Query - tag_history). The 'Available Historical Tags' tree view shows a MySQL connection with a controller:default schema, which contains a _generic_ folder and a sine folder. Inside the sine folder are tags sine0, sine1, sine2, station 1, and station 2. A checkbox 'Use fully-qualified paths' is unchecked. To the right, the 'Selected Historical Tags' table lists three tags: [-]sine/sine0, [-]sine/sine1, and [-]sine/sine2, each with a data key alias sine0, sine1, and sine2 respectively. Below the table are sections for 'Date Range' (set to 'Historical'), 'Start Date Binding' (set to {StartDate}), 'End Date Binding' (set to {EndDate}), 'Aggregation Mode' (set to 'Time-weighted Average'), 'Return Format' (set to 'Wide'), 'Sample Size' (set to 'Fixed' with value 1000), and an 'Advanced' section with checkboxes for 'Ignore Bad Quality', 'Prevent Interpolation', and 'Avoid Scanclass Validation'. The 'Aggregation Mode' and 'Advanced' sections are highlighted with red boxes.

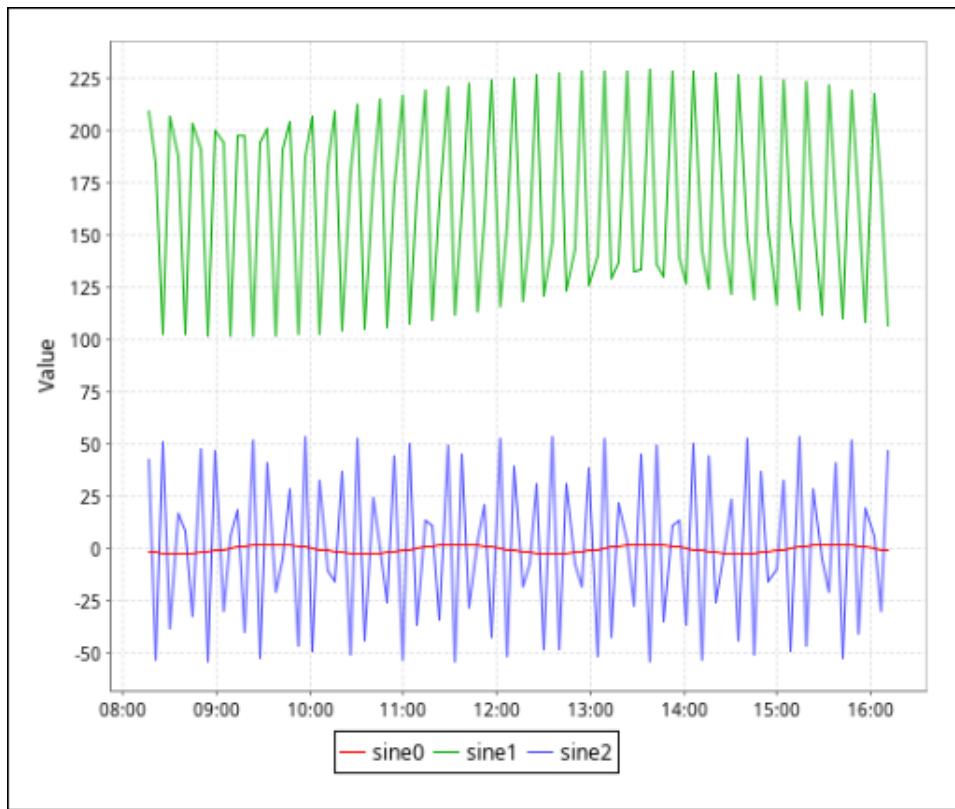
Adding a Chart

1. Now switch over to the **Design** panel. Drag a **Timeseries Chart** from the Report Design Palette into the report and expand it.
2. From the **Key Browser**, expand the **Datasources** folder, and drag your datasource (i.e., tag_history) down to the **Data Key** under the **Chart Options** tab of the Property Inspector.
3. Select **Test1** and **Test2** pens and delete them by selecting each pen individually, and then clicking the **Trash Can** icon .
4. Once the Data Key is correct, you can assign pens to your chart. Drag in your **Keys** (i.e., sine0, sine1, sine2) to the **Pens** property table. You'll immediately notice your chart is populated with fake data, this is just a preview and does not represent the data returned by your data source.

The screenshot shows the 'Design' tab of the Report Overview interface. On the left, the 'Key Browser' shows a 'Parameters' section and a 'Data Sources' section where 'tag_history' is selected. The 'Property Inspector' shows the 'Chart Options' tab with 'Data Key' set to 'tag_history' and 'Domain Key' set to 't_stamp'. The 'Pens' table lists three pens: sine0, sine1, and sine2, each associated with its respective sine tag. To the right, a timeseries chart displays three lines representing sine0 (orange), sine1 (green), and sine2 (blue) over a time period from March 30 to April 1. The chart has a y-axis labeled 'Value' ranging from 10 to 110 and an x-axis showing dates from 30-Mar, 12:00 to 1-Apr, 00:00. The chart area is highlighted with a red box.

5. You can modify a selected pen by clicking the **Edit** button or double clicking the pen to open the **Edit Pen** tab. Here you can change the name of the pen, change the pen style, color, and more. Let's change the color of the pens to make them really stand out.
 - a. Lets start with the sine0 pen.
 - i. Set the **Color** to be red.
 - ii. To return to the **Chart Options** tab, click the double arrow button.
 - b. Repeat the above steps for the remaining two pens, making sine1 green, and sine2 dark blue.

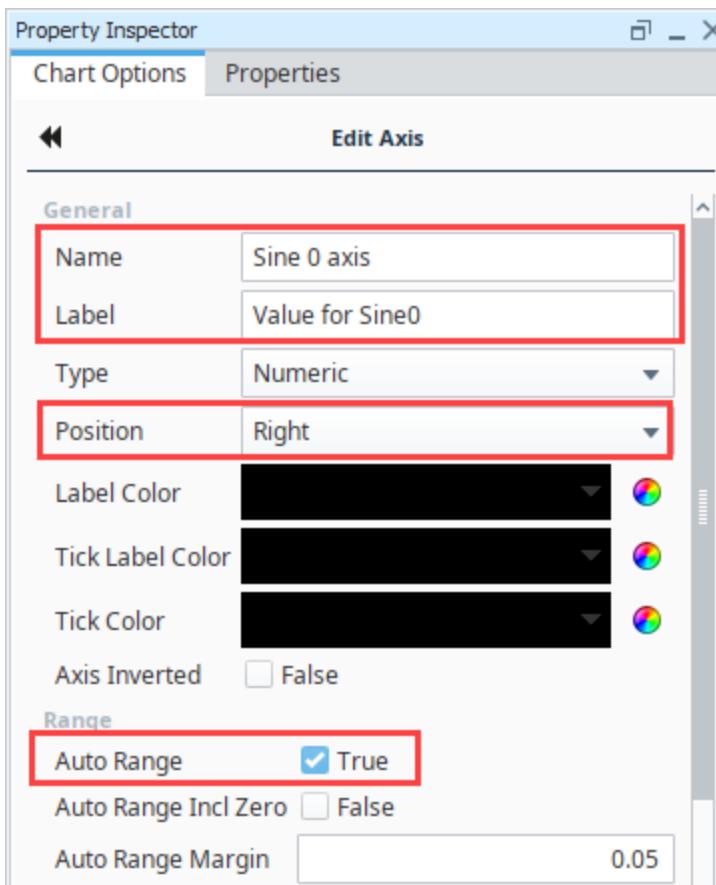
6. Go to the **Preview** tab to view the Timeseries Chart in the report.



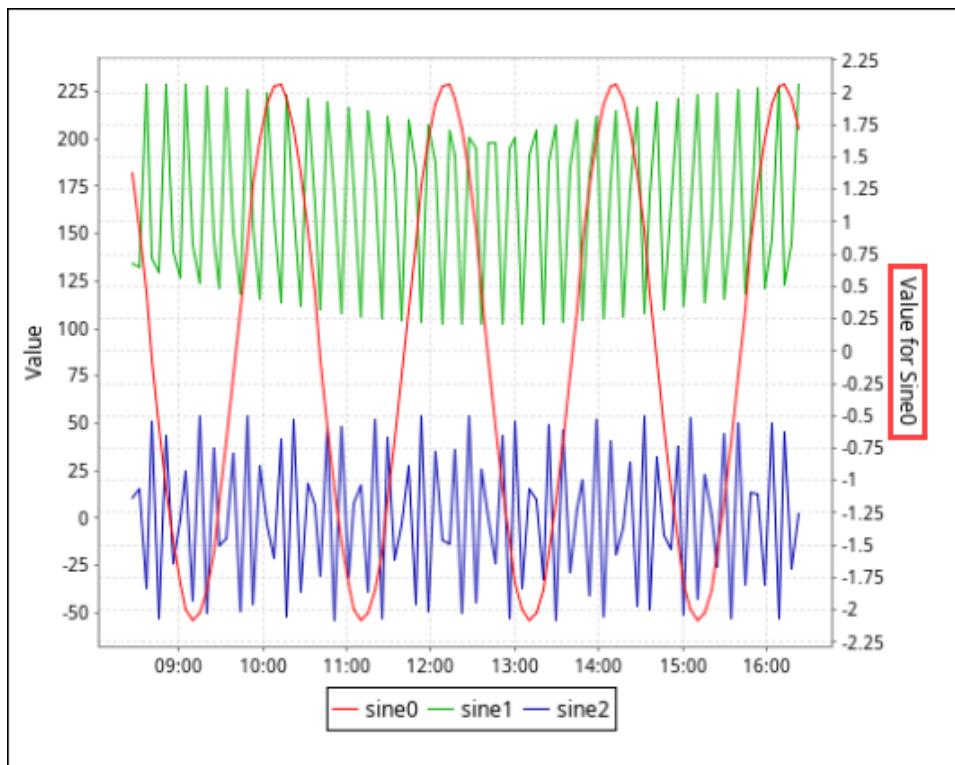
Adding an Axis

Unfortunately, it is a little difficult to see the small changes in our sine0 pen because its range is much smaller than the other pens. Let's fix this by adding a new axis.

1. Go back to the Design tab and select the chart. Click the Plus button next to the Axes table to add a new axis, then click the **Edit** button below it to edit our new Axis.
 - a. Give it a name of **Sine 0 axis**
 - b. Give it a Label of **Value for Sine0**
 - c. Change the position to right so that it shows up on the right side of the chart.
 - d. We can leave **Auto Range** turned on, so that it will automatically pick the appropriate min and max for the sine0 pen.
 - e. Return to the **Chart Options** tab by clicking the double arrow button.



2. We now need to tell our sine0 pen to use that new axis. Select the sine0 pen from the pens list and click the **Edit**  button. Change the Axis of the pen to be the pen we just created.
3. Navigate to the **Preview** tab to view our finished chart.



Related Topics ...

- [Tag Properties](#)
- [Report Charts](#)

Report Tables

Tables are a major part of Ignition Reporting. Tables are objects that display data in a structured, repetitive format giving users the ability to flexibly layout and organize tabular data in a variety of ways. Their complexity can range from trivially simple to complicated, but fundamentally they do the same thing - take the data given to them and repeat it in a structured way for each row of the data provided. The Reporting engine will automatically create new pages to fit all data within the Table's boundaries while applying formatting preferences each step along the way such as font, size, layout and alignment. Combine that feature with powerful data manipulation and expressive layout tools, and you get an object that often forms the basis of your reports.

Dec 20, 2005 17:55	labeler	50 minutes	Out of labels
Dec 22, 2005 11:55	filler	15 minutes	Scheduled maintenance
Jan 02, 2006 22:55	palletizer	10 minutes	Misalignment
Jan 03, 2006 02:55	conveyor line	25 minutes	backup
Feb 12, 2006 06:13	labeler	10 minutes	<NA>
Feb 12, 2006 12:01	labeler	3 minutes	Out of labels
Feb 12, 2006 14:01	palletizer	17 minutes	Misalignment
Feb 12, 2006 16:23	conveyor line	23 minutes	Scheduled maintenance
Feb 12, 2006 20:04	filler	33 minutes	Overflow
Feb 12, 2006 20:13	labeler	21 minutes	Stuck labels
Feb 12, 2006 20:25	filler	20 minutes	Overflow
Feb 12, 2006 20:36	conveyor line	30 minutes	Scheduled maintenance

On this page ...

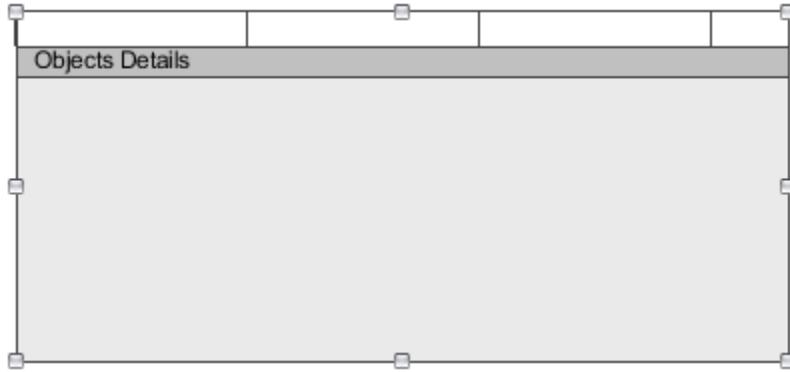
- [The Table Component](#)
 - [Anatomy of a Table](#)
 - [Table Configuration](#)
- [Report Table Features](#)
- [Other Table Components](#)

Let's start by looking at the various parts of the Table component and move into more features.

The Table Component



The Table component can be created by dragging it from the Report Design Palette, or by simply dragging a Data Source from the Key Browser to a page in your report. A new Table with no bound Data Key is fairly simple looking, but hides a wealth of functionality. When you create a table, by default, you will see a basic table like the image below on the report page. It says "Object Details" because it has not yet been set up to use a Data Source.

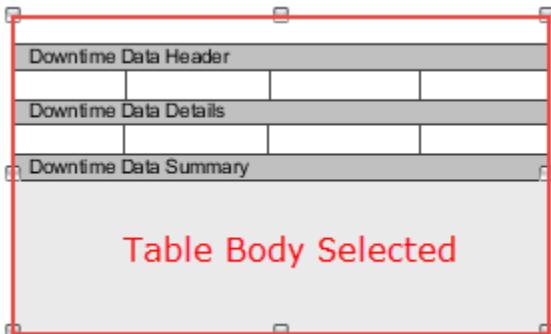


Once you have your dataset, you're probably thinking about how to display that data in a report. To best understand how to use a table to illustrate your data in a report, you need to have a basic understanding of how the report table works.

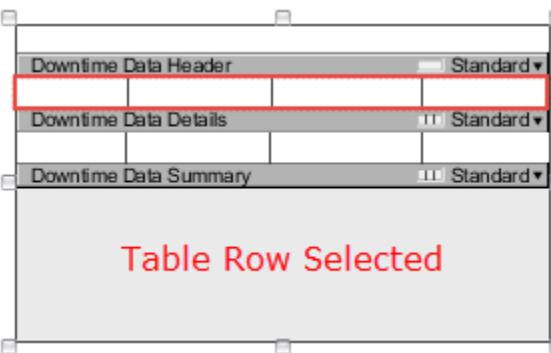
Anatomy of a Table

A table has multiple sections, each with its own properties. By default, a Table consists of three sections:

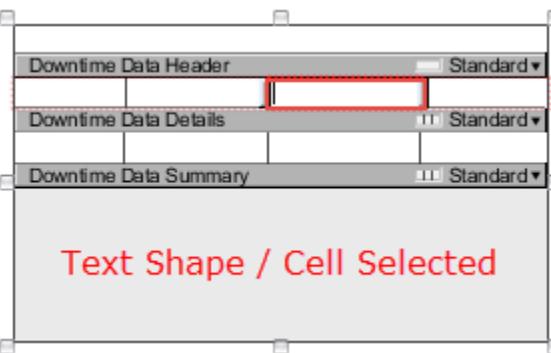
- **Table Body** - Grey area at the bottom of the table template. You can stretch and shrink the table boundaries to position and size a table.



- **Table Rows** - There are three types of table rows: Header, Details, and Summary. More information on these row types can be found on the [Table Rows page](#).



- **Text Shapes / Cells** - are available only in a structured table row. In Reporting, Text Shapes are commonly referred to as cells. Can select multiple cells with alt or shift.

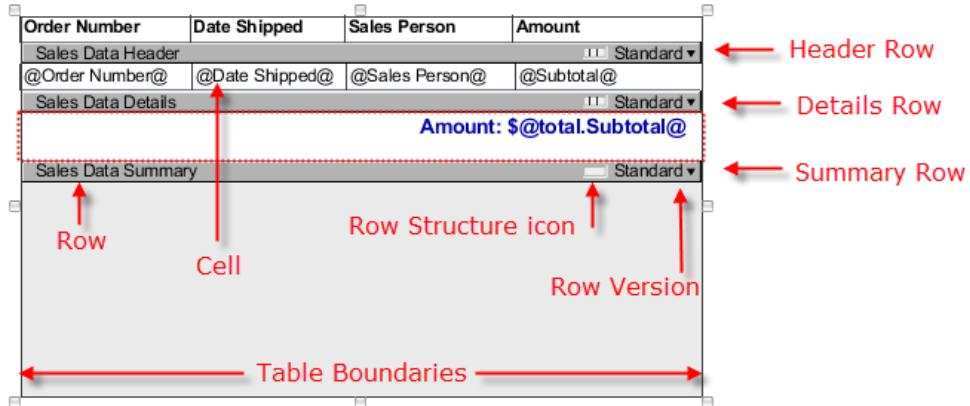


Text Shapes or Cells

If you're looking in the Project Browser tree, you'll notice a node called 'Text Shape.' In Reporting, the Text Shape is also called a 'cell.' For the sake of clarity, Text Shapes will be referred to as 'cells,' unless we are referring to the Text Shape component.

The following example shows the basic anatomy of a table with data populated in the Header, Details, and Summary rows. Once you're finished designing your report, go to the Preview panel and check your results. You can continue to navigate between the Design and the Preview panels

making changes until you get your report just right!



Each section (i.e., Table, Rows, and Cells), when selected, has its own unique properties (shown below), along with some common basic properties that appear in the **Properties** tab of the Property Inspector. Refer to the [Table](#) page in the Appendix for a complete list of all Table properties.

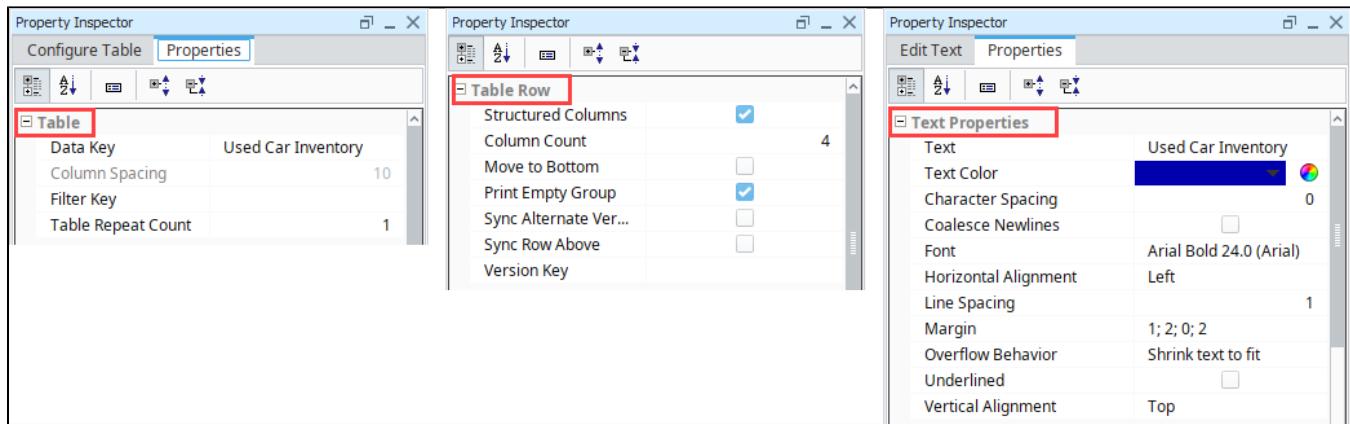
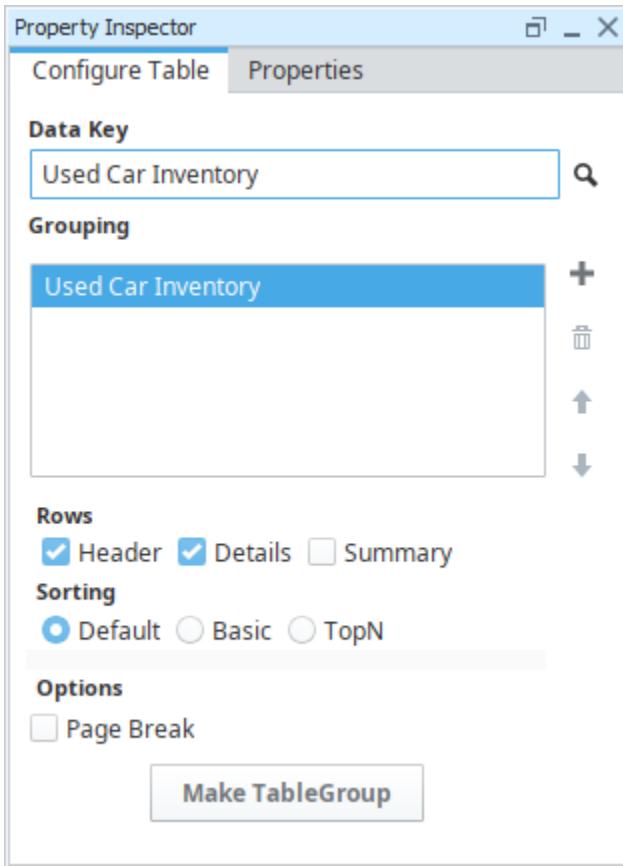


Table Configuration

When you first drag a table into the Design panel of your report, the **Configure Table** tab will appear in the Property Inspector. You will see that it has a default **Data Key** of "Objects", which matches the **Details** row in the table. The first step when configuring your table is to drag your **Datasource** from the Key Browser to the Data Key field in the Configure Table tab. Changing the Data Key will also change the label on the Details row making it easier to determine what you are looking at when editing multiple tables or [Groupings](#). Depending on how you want to present your data, you may also want to add Header and Summary Rows.



Once you add a Data Key to your table, you can start designing. There are many tools built into the Table interface that you can use to help you layout and organize the data in your table. Now all you need to do is decide how you want to organize and display the data in your report.

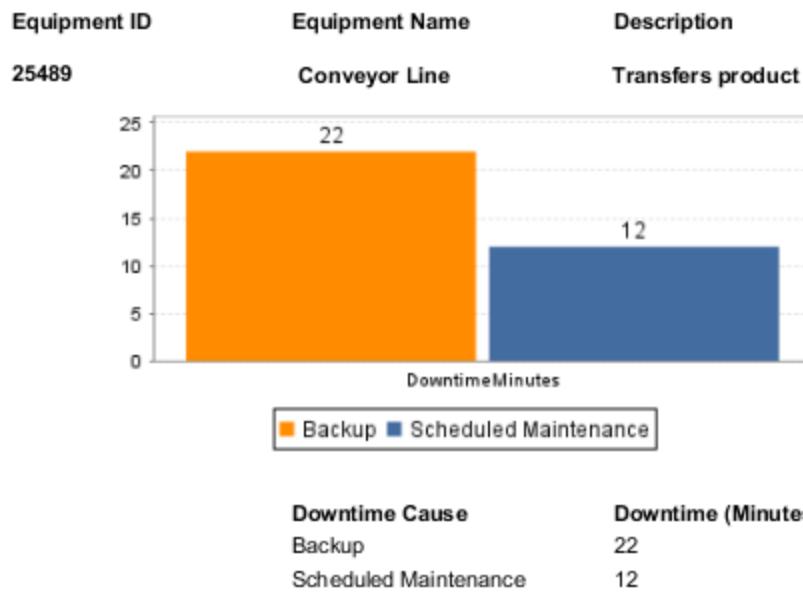
If you have a lot of tabular data, you probably want to use a Standard row. You might start by simply dragging your Data Keys from the Key Browser to columns in your Details row. Then, you might give each Header column a label. If you have a bunch of numbers that you want to add together, find out the 'min', 'max,' you can create a Summary row, enable the [Show Calculations](#) property, and choose from the list of calculation keys. Lastly, you might want to make some cosmetic changes to the font size, font color, or changing the Header row from [structured to unstructured](#) so you can add an image, chart, text shapes, etc.

Report Table Features

Tables have a lot of built-in features that give reporting users the ability to create simple tables and increasingly more complicated tables. The table features are quite powerful, not only do they allow you to organize your tabular data, they also provide sophisticated data manipulation and layout tools. There are three types of table rows: [Header](#), [Details](#), and [Summary rows](#) that make up the body of a table. You can break tables down by data keys that share a common value using Dataset Grouping. The different types of rows can be independently enabled for each level of grouping, each group having its own Header, Details, and Summary rows. Additionally, the keys from '[Show Calculations](#)' and other [keychain functions](#) are supported for any level of grouping.

By default, when you initially create a table all the row types are [structured](#). A structured row has a variable number of columns and allows you to organize your text based table data, whereby, an unstructured row is highly customizable allowing you to place images, text shapes, and charts within the table row. Unstructured rows are perfect for placing charts anywhere within the row of the table. An unstructured Details row typically works best when used in conjunction with [Grouping Data Inside of Tables](#) or [Nested Queries](#) so that each row has a chart with data from that group or query.

Equipment Downtime Details



Another feature is [Table Row Versioning](#) which gives you the option of displaying rows with a different format to make them stand out, such as creating an alternate background row color, making the first row different, or even creating your own custom row version. You have have a choice of using any of the Built-in row versioning options or creating a custom version.

Candy Bar Name	Inventory	Cost per Candy	Value of Candy Supply
Pay Day	22	\$ 2.10	\$ 46.20
Baby Ruth	44	\$ 2.40	\$ 105.60
Milky Way	16	\$ 1.90	\$ 30.40
KitKat	28	\$ 1.85	\$ 51.80
Butterfinger	7	\$ 2.30	\$ 16.10
Snickers	19	\$ 2.00	\$ 38.00
Twix	13	\$ 2.05	\$ 26.65

Report Tables also support [Table Grouping](#) which is an easy way to add multiple Data Sources to an existing table in a report using Peer Tables and Child Tables. Peer Table groups allow the second table to begin exactly where the first table ends. Child Table groups allow you to nest one table inside another. What's really nice about Table Groups and Nested Data Sources is that you can create Summary Tables for categories of items or drill-down charts all in one report .

Once you get familiar with all the report table features, you can easily design professional tables for your reports.

Other Table Components

There are a few [other table components](#) in the Report Module, Simple Table and CrossTab Table, and they both behave a little differently. Selecting the right table depends on the type of data you have and how you want to display it.

The [Simple Table](#) is similar to a basic table that dynamically creates new rows and columns for rows returned by the Data Keys on the component. With a Simple Table, you can very quickly add a table inside a report.

The [CrossTab Table](#) is commonly used to summarize the relationship between two groupings of data by showing summaries of cross sections of the datasource. The CrossTab Table has lots of repetitious data, a datasource that provides at least two columns of data which are repetitious compared to the number of rows, and one or more columns that represent a value that requires a calculation. This table will stretch both its height and width to accommodate the underlying dataset.

Related Topics ...

- [Report Design](#)

- Report Design Tools
- Table

In This Section ...

Getting Started with the Report Table

The [Report Tables](#) is another Reporting component, like report charts, that has a lot of features and functions that help you create meaningful reports. Tables are a major part of Reporting and simple to create. This page will give you a great head start for creating your own standard report tables using the built-in design tools.

Creating a Report Table

Creating a table is a simple process, but the order of how you create a table is important. The first thing you need before creating your report table is to create a [datasource](#). Your data source can be a query, script, or CSV file. Next, add a table and configure the data in the table that you want to display. Lastly, you can use the built-in design tools to enhance your reports. You can add images to your reports, change row types, show mathematical calculations, change the font style, size, or color of the text, and much more.

Let's get started!

This example creates a standard chart that tracks the downtime minutes for each production line. We'll configure the data in the table, do a calculation, add an image, and make a few property changes so you get a sampling of the power of report tables.

Adding a Datasource



Datasource

This example uses a CSV file as its datasource. You can use the one shown in the code block below, or create your own.

1. In the **Data** panel, click the plus button to add a datasource, and select **StaticCSV** from the resulting list of options. From the code block below, copy the text and paste it into the **Data** field. This is the data that we will use in this example.

Downtime Data

```
Production Line, Downtime
Line A, 75
Line B, 92
Line C, 43
Line D, 54
Line E, 66
Line F, 80
Line G, 40
Line H, 88
```

On this page ...

- [Creating a Report Table](#)
 - [Adding a Datasource](#)
 - [Table Configuration](#)
 - [Finishing Touches](#)
- [Preview a Report](#)

Parameters

- StartDate
- EndDate

Data Sources

- 1 Static CSV - Downtime Data

Data Key

Downtime Data

Data

Production Line, Downtime	
Line A	75
Line B	92
Line C	43
Line D	54
Line E	66
Line F	80
Line G	40
Line H	88

Table Configuration

1. Go to the **Design** panel, and drag a **Table** component into the page.
2. From the **Key Browser**, expand the **Datasources** folder, and drag your data source (i.e., Downtime Data) into the **Data Key** field of the **Configure Table** tab of the Property Inspector.
3. Add **Header** and **Summary** rows by marking the checkboxes.

Key Browser

- Show Calculations
- Parameters
- Data Sources
 - Downtime Data
 - Downtime
 - Production Line
- Built In

Property Inspector

Configure Table Properties

Data Key

Downtime Data

Grouping

Downtime Data

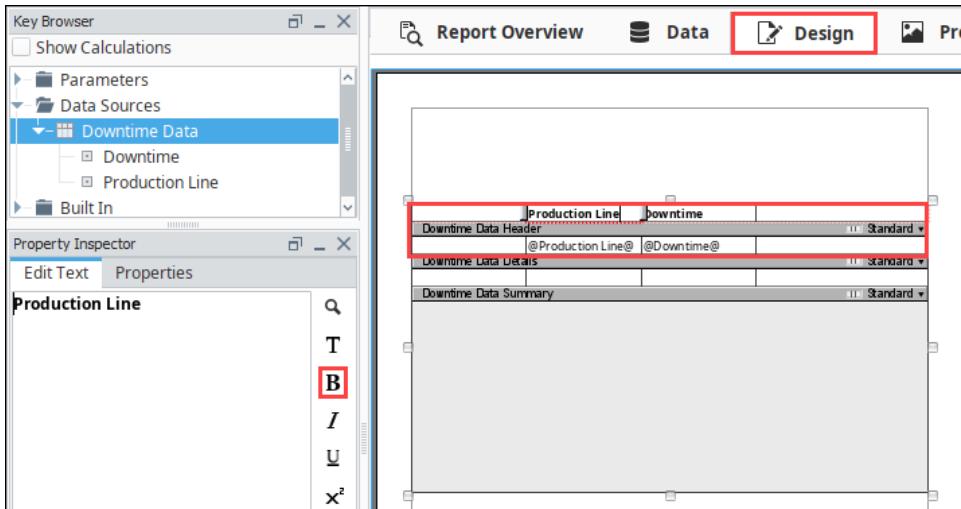
Rows

Header Details Summary

Sorting

Default Basic TopN

4. Now, let's configure the data in the Table. With the Table still selected, drag each of the data keys to a cell on the table above the Data Details row: (i.e., Production Line, and Downtime).
5. Next, let's add column headers by selecting a cell above one of our details cells, and enter the appropriate column header name (i.e., Production Line, and Downtime [Minutes]). You can change the font style, size, and color using the functions on the right side of the **Edit Text** Tab. Bold the headers using the **B** button.



6. Go to the **Preview** panel to view your report with the data. It's very common to make formatting changes once you see how your data looks in the report.

Production Line	Downtime
Line A	75
Line B	92
Line C	43
Line D	54
Line E	66
Line F	80
Line G	40
Line H	88

Finishing Touches

1. Back in the **Design** panel, let's make a few changes. Select the "@Production Line@" in the Details row and click on the **Properties Tab** to see all the cell properties. Let's change the **Horizontal Alignment** property from left to **Center** so the data falls nicely under the header. Repeat this for "@Downtime@."

Checking your table in the report

At any point during the table configuration process you can go to the **Preview** panel and see how your table looks with all the data. Configuring the data and designing the table layout is an iterative process. You can go back and forth between the **Design** and **Preview** panels as many times as you want making and viewing changes until you get the results you want.

2. Let's add the total downtime by using the 'total' calculation. In the **Key Browser**, set **Show Calculations** to 'true.'
3. The Data Summary row is a good place to show the total number of downtime minutes. Expand the **Downtime** data key and you'll see a list of calculations you can use. Drag the 'total' key to a cell in the Data Summary row.
4. You can even give the total number of downtime minutes a title, make it blue, and bold it (i.e., Total Downtime). If the cell is too small to show the title, you can make the cell larger by dragging the cell border either to the left or right. You can also select the Data Summary row, go to the **Properties tab** and change the **Column Count** (i.e., 2), as shown in the screenshot below.

5. Click the **Preview** tab to check your work.

The screenshot shows the Report Designer interface. On the left is the 'Property Inspector' panel, which is currently displaying settings for a 'Table Row'. A red box highlights the 'Column Count' field, which is set to '2'. On the right is the 'Report Overview' window, showing a report structure with three main sections: 'Downtime Data Header', 'Downtime Data Details', and 'Downtime Data Summary'. The 'Design' tab is selected at the top of the window.

Preview a Report

The **Preview** panel lets you validate that all your data is organized and formatted, and you're completely satisfied with how your data is presented in the report. Once you're satisfied, you can [schedule the report](#) to be run and delivered automatically.

The screenshot shows the 'Preview' panel of the Report Designer. The report title is 'Weekly Production Downtime' and the date is 'April 2020'. Below the title is a table with two columns: 'Production Line' and 'Downtime'. The data rows are:

Production Line	Downtime
Line A	75
Line B	92
Line C	43
Line D	54
Line E	66
Line F	80
Line G	40
Line H	88

Total Downtime: 538

Related Topics ...

- [CrossTab and Simple Tables](#)
- [Nested Queries](#)

Table Rows

Row Types

Rows are an important fundamental aspect of tables. There are three types of table rows: **Header**, **Details**, and **Summary**. The different types of rows can be independently enabled for each level of **Grouping**, and for each table in a **Table Group**. Adding or removing the Header, Details, or Summary rows is as simple as selecting the Row boxes in the Configure Table tab of the Property Inspector. A row can also be **structured or unstructured**. A structured row having a variable number of columns, and unstructured row having no columns so you can add images, charts, data, and text shapes anywhere in the row. Additionally, **Table Row Versioning** gives you the option of conditionally displaying rows with a different format to make them stand out.

To learn more about Tables, go to [Report Tables](#) and [Table](#) in the Appendix.

Header Row

The Header Row allows for a single row to be placed before the Details rows. This is commonly used to create a header for the Table. In many cases where one header is used, the other header could be used equivalently in its place. An interesting feature of the header row is the **Reprint When Wrapped** property which allows the header row to be reprinted on a new page when its data crosses a page boundary. To disable this feature, uncheck the **Print When Wrapped** property in the Properties tab.

Automatic Text Resizing

If text in a Header Row is so long that it will result in truncation or overflow, the size of the text will automatically resize to accommodate.

Detail Rows

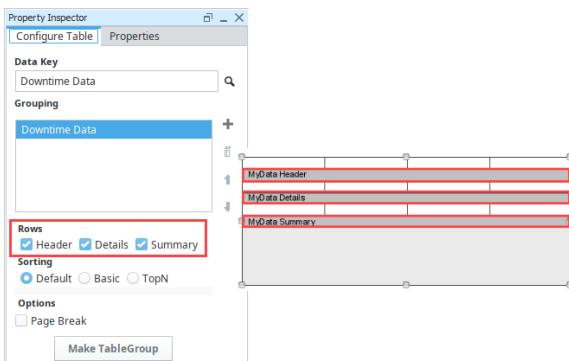
The Detail Rows typically represent the majority of the data on a table or commonly referred to as the "middle" rows. Once you configure the Detail rows with your datasource, the table displays your data in a structured, repetitious format. You can customize the layout and organize the data to determine how you want your data to look on the report. You can even disable Detail Rows in unusual situations such as only displaying aggregate summaries.

Summary Row

The Summary Row is like the Header row only it prints at the bottom of the table, and is typically used to display aggregates for keys. They are typically used in conjunction with some of the [Show Calculation Keys](#), such as count, total, running total, etc.

Configuring Header, Details, and Summary Rows

Adding Header, Details, and Summary Rows is super easy. Once a Table component is in your Design Panel and your Datasource is populated in the Data Key field, check the **Header** and **Summary** boxes. The Header, Details, and Summary rows will be added to your report. The **Details** box will be checked by default. To remove the Header, Details, or Summary rows, uncheck the applicable rows.



Structured vs Unstructured Rows

Structured Rows behave like a row in a spreadsheet: a series of columns are horizontally adjacent to each other. In the case of a row, the columns are Text Shapes. A structured row can have a variable number of Text Shapes. Structured rows provide more control over the layout, wrapping, and

On this page ...

- [Row Types](#)
 - [Header Row](#)
 - [Detail Rows](#)
 - [Summary Row](#)
 - [Configuring Header, Details, and Summary Rows](#)
- [Structured vs Unstructured Rows](#)
- [Report Data Configuration](#)
 - [Sorting](#)
 - [Filtering](#)



Table Rows

[Watch the Video](#)

organization of text based table data. Since structured rows only allow for **Text Shapes**, they can not contain components like images, barcodes, charts, and text shapes.

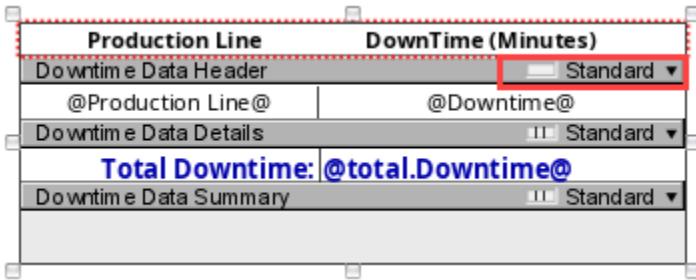
To view the properties of the Details row (or any row), super-select the row by clicking on the Details row (dark bar in the table), or clicking the **Details** node in the Project Browser tree. You can confirm that a row is structured by looking at the **Structured Columns** property in the Properties Tab, or by looking at the **Row Structure** icon  next to the **Standard** row version label. You can also change the number of columns in a row, using the **Column Count** property.

You can tell when a row (any row) is selected by looking at the Project Browser tree, or by looking at the red outline of the rows in a table. A row's content is above its respective dark row bar.

Unstructured Rows are functionally very similar to structured rows, but they do not offer the same column based text constraints. Instead, Unstructured Rows allow you to place any other report component inside of the row. Unstructured rows are highly customizable rows allowing you to place data, text shapes, images, or charts anywhere you want them within the row.

There are two ways to make a row unstructured once your row is selected:

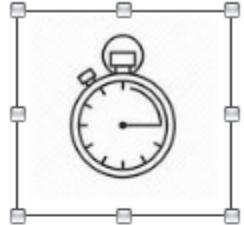
- In the Property Tab of the Property Inspector, set the **Structured Columns** property to 'false.'
- In your Table, click on the **Row Structure icon**  on the right side of your row. Click the icon to make the row unstructured . You can toggle the Row Structure icon to switch between a structured row and unstructured row.



The screenshot shows a report table with four rows. The first row, 'Downtime Data Header', has a red border and is highlighted with a red outline, indicating it is selected. To the right of this row, there is a 'Row Structure' icon (a small square with a vertical line). A red box highlights this icon. The second row, 'Downtime Data Details', contains the text '@Production Line@' and '@Downtime@'. The third row, 'Total Downtime: @total.Downtime@', contains the text 'Total Downtime: @total.Downtime@'. The fourth row, 'Downtime Data Summary', contains the text 'Downtime Data Summary'. Each row has a 'Standard' dropdown menu icon to its right.

Now that the Header row is unstructured, drag the Header row down the page making room to add any component(s) from the Report Design Palette, and place it anywhere in the row. In this example, you can see a **Text Shape** was added for the report title, an image was added, and a **Line Shape** was drawn to separate the image from the report data.

Production Line Downtime Report



Production Line Downtime (Minutes)

Downtime Data Header		Standard ▾
@Production Line@	@Downtime@	
Downtime Data Details		Standard ▾
Total Downtime: @total.Downtime@		
Downtime Data Summary		Standard ▾

Here is what the report looks like in the **Preview** panel.

Production Line Downtime Report



Production Line	Downtime (Minutes)
Line A	75
Line B	92
Line C	43
Line D	54
Line E	66
Line F	80
Line G	40
Line H	88
Total Downtime: 538	

Report Data Configuration

In addition to configuring rows, you can also configure how you want your data to appear in a report using the **Sort** and **Filter** functions in the Table component.

Sorting

Sorting orders your data by a single data key or list of data keys. There are three types of Sorting in Tables.

- **Default** - data is sorted based on the order in which it is retrieved.
- **Basic** - takes a list of data keys and sorts by the first one. If the sort results in a tie, the tie will be resolved by the next data key in the list, and so on.
- **TopN** - uses a single key path, with a **Count** value that allows a limit to the number of rows that are processed.

Basic and **TopN** sorts can be configured for either ascending (↗) or descending (↘) sorts. They can also utilize aggregate (calculation) keys.

The **TopN** sort option, **Include Others** **Include "Others"** if selected, will include all values outside of the specified **Count** range by compressing them into a single row.

Sort and Filter Examples

The CSV dataset containing all the data and the table used for the following Sort and Filter examples are shown below.

Dataset	Table in the Design Panel
Download	View

Code Block | language = js | title = Dataset

Default Sort Example

The data for the **Default** sort is retrieved directly as-is from the dataset. No sort order is applied.

Order Number	Date Shipped	Subtotal	Sales Person
10248	"Jul 16, 2017"	440	"Buchanan, Steven"
10260	"Jul 18, 2017"	517.80	"Smith, Steven"
10249	"Jul 17, 2017"	624.30	"Howard, Rodney"
10250	"Jul 16, 2017"	1444.20	"Johnson, Grace"
10255	"Jul 18, 2017"	2018.25	"Jain, Kala"
10253	"Jul 16, 2017"	101.10	"Wilson, George"
10251	"Jul 16, 2017"	1259.40	"Lawrence, Sara"
10254	"Jul 16, 2017"	843.60	"Buchanan, Steven"
10259	"Jul 16, 2017"	1050.75	"Smith, Steven"

Basic Sort Example

This table was sorted by the **Order Number** data key in **ascending** order. To sort in descending order, right click on the **Order Number** and select **Sort Descending**.

Order Number	Date Shipped	Subtotal	Sales Person
10255	"Jul 18, 2017"	2018.25	"Jain, Kala"
10253	"Jul 16, 2017"	101.10	"Wilson, George"
10251	"Jul 16, 2017"	1259.40	"Lawrence, Sara"
10254	"Jul 16, 2017"	843.60	"Buchanan, Steven"
10259	"Jul 16, 2017"	1050.75	"Smith, Steven"
10256	"Jul 16, 2017"	2114.30	"Johnson, Grace"

TopN Sort Example

This table was sorted by the **Subtotal** data key in **ascending** order with a **Count** value of **10** rows. To sort in descending order, click the **Ascending icon** next to the Data Key to switch to **Sort Descending**.

Order Number	Date Shipped	Subtotal	Sales Person
10255	"Jul 18, 2017"	2018.25	"Jain, Kala"
10253	"Jul 16, 2017"	101.10	"Wilson, George"
10251	"Jul 16, 2017"	1259.40	"Lawrence, Sara"
10254	"Jul 16, 2017"	843.60	"Buchanan, Steven"
10259	"Jul 16, 2017"	1050.75	"Smith, Steven"
10256	"Jul 16, 2017"	2114.30	"Johnson, Grace"
10250	"Jul 16, 2017"	1444.20	"Johnson, Grace"
10249	"Jul 17, 2017"	624.30	"Howard, Rodney"
10248	"Jul 16, 2017"	440	"Buchanan, Steven"
10260	"Jul 18, 2017"	517.80	"Smith, Steven"

Configure Table

TopN Sort Results

Filtering

Filtering gives the option of processing data based on an expression. The **Filter Key** property can be found in the Property Inspector under the **Properties** tab.

Notes

- Once you enter your expression in the **Filter Key** field, hit return to commit your expression.
- If the expression resolves to **'false'**, the row will be skipped.

Filtering Example

This example is sorted in ascending order using the **Subtotal** data key, and filtered for any '**Subtotal>1000**.' To alternate to descending order, click the **Ascending icon** to toggle to a descending sort.

Configure Table & Property Tab

Filtering Results

Related Topics ...

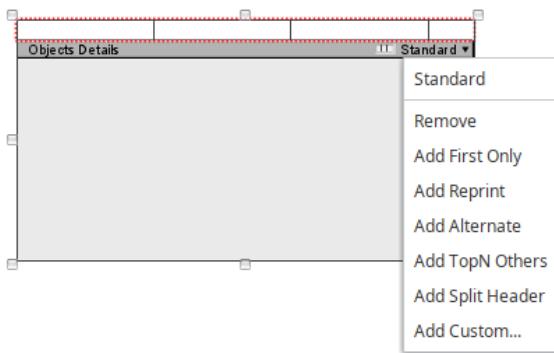
- [Table](#)
- [Grouping Data Inside of Tables](#)
- [Table Groups](#)
- [Report Tables](#)

Table Row Versioning

Table Row Versioning

Table Row Versioning allows you to conditionally display rows of data in different formats. It is used to make certain data standout or to make a report more legible. It allows you to do things like create alternate background row colors, or make the first row different, or show negative valued rows differently. You will see a list of your existing row versions (only one named "Standard" appears first in the list) followed by options to add new row versions. When creating new row versions, the currently selected row will be duplicated as the new row version. From here, you can modify your row in any way. You can make it similar to the Standard row or completely different. Once you added a few row versions, you can swap between them by selecting one from this list.

To use Row Versioning, click on the word "**Standard**" on the far right of a row. Row versions are either **Built-in** or **Custom**, and may be specified with a version key expression. They are applicable to the [Header](#) and [Summary Rows](#), but are most often added to the [Detail Row](#).



On this page ...

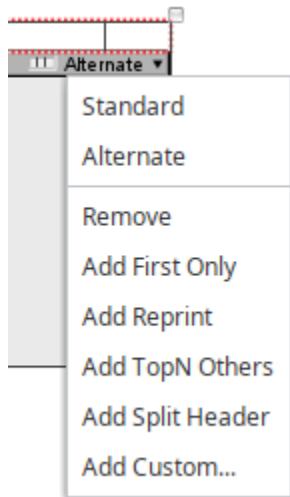
- Table Row Versioning
 - Using Row Versions
 - Sync Alternate Versions
 - Custom Row Version



Table Row Versioning

[Watch the Video](#)

Once a Row Version has been added, it becomes selectable in the dropdown. The word "**Standard**" also gets replaced with whatever Row Version have currently selected to edit. In this case, an Alternate Row Version was added, and it is currently selected.



Note: It is important to always be aware of what Row Version you are currently working on by checking the name of the Row Version on the right of the Table row. After navigating away from the Design panel, the Table will automatically display the standard Row Version on return to the Design panel.

Here is a list of the Row Version types and their descriptions.

Row Versions	Description
Standard	Default row version. When adding a new Row Version, it will start off as a copy of the Standard Row Version.

Remove	Removes the currently selected Row Version. The Standard row cannot be removed.
First Only	Applies only to the first instance of the row. Good for showing header information without using an upper level Detail row. This is not the same as a Header row.
Reprint	Applies to every page after the first. Good for one time headers or (continued) indications to save space.
Alternate	Applies to every other row starting with the second row. Good for changing the background color on alternate rows.
TopN Others	Applies to count number of rows in a TopN sort. Using "Include Others" will then distinguish between TopN and non-TopN rows.
Split Header	Applies to Headers that have been split due to excessive height. Good for providing "Continued" type indicators.
Custom	Create your own Row Version. When this option is selected, you must enter a label for the new row version. Instead of being used automatically like with the other Row Versions, all custom Row Versions are driven by the Version Key Property.

Using Row Versions

Once a new Row Version has been added to the table row, it is easy to configure each row version to appear visually distinct.

When using a Table to display a large number of rows, an Alternate Row Version can be added to alternate between two fill colors, which in turn improves readability. We start off by adding a table to the report, assign a data source to the table and add some data keys to the text shapes in the Details row. In this example, we will alternate between white and orange, so we will leave the standard row's **Fill** property disabled to use the default color of our paper.

The screenshot shows a table with four columns: Candy Bar Name, Inventory, Cost per Candy, and Value of Candy Supply. The first row is a header labeled 'CandyData Header'. Below it are two data rows, both labeled 'CandyData Details'. The second and third rows are highlighted with a red dotted border, indicating they are selected for editing. A floating panel titled 'Stroke and Fill' is open, showing the following settings:

- Fill: A small square preview showing black.
- Fill Color: A color picker set to black.
- Opacity: Set to 1.
- Stroke Style: Set to Hidden.
- Stroke: Not applicable as the fill is black.

An Alternate row was added to the details row of the table. On the Alternate row, we can enable the **Fill** property and set **Fill Color** to an orange background.

The screenshot shows the same table structure as the previous image, but the second data row ('CandyData Details') is now filled with orange, while the first and third rows remain white. A floating panel titled 'Stroke and Fill' is open, showing the following settings for the orange row:

- Fill: A small square preview showing orange.
- Fill Color: A color picker set to orange.
- Opacity: Set to 1.
- Stroke Style: Set to Hidden.
- Stroke: Not applicable as the fill is orange.

Taking a look at a Preview of the report, we can see that the Alternate row with the orange color is automatically used on every other row starting with the second row.

Candy Bar Name	Inventory	Cost per Candy	Value of Candy Supply
Pay Day	22	\$ 2.10	\$ 46.20
Baby Ruth	44	\$ 2.40	\$ 105.60
Milky Way	16	\$ 1.90	\$ 30.40
KitKat	28	\$ 1.85	\$ 51.80
Butterfinger	7	\$ 2.30	\$ 16.10
Snickers	19	\$ 2.00	\$ 38.00
Twix	13	\$ 2.05	\$ 26.65

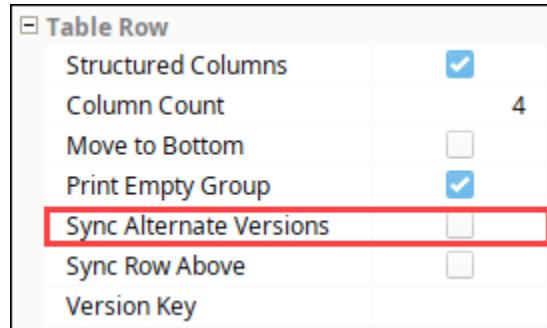
Sync Alternate Versions

Once a new row version has been created, it is not linked to the standard row in any way, so configuration changes made to one row will not be automatically applied to other row versions.

The common case where this behavior becomes a problem is when the width of one or more Text Shapes in a row are modified. Here we see a row with an Alternate version using a different **Fill Color**. After the Alternate row version was created, the second and third Text Shapes on the Standard version has their **Width** increased, causing the text inside to shift. However, the Text Shapes on the Alternate version will not automatically resize themselves by default, so the Preview Panel shows offset columns on the Alternate version.

Motor	Site A	15
Motor	Site A	23
Conveyor Line	Site B	148
Pallet Wrapper	Site A	58
Motor	Site C	96

Instead of manually adjusting the Text Shapes on the Alternate version, we can toggle the **Sync Alternate Versions** property on the Standard row. Assuming both columns have the same value for **Column Count**, enabling **Sync Alternate Versions** will automatically resize the widths of Text Shapes on the alternate row version to match the widths of the standard version Text Shapes.



Heading back to the Preview Panel, we see that the Text Shapes on both rows are now synchronized.

Motor	Site A	15
Motor	Site A	23
Conveyor Line	Site B	148
Pallet Wrapper	Site A	58
Motor	Site C	96

Custom Row Version

Custom row versions are ideal when the built-in Row Versions don't fit your needs. Custom versions are identified by a string-based name, and will be used when the Version Key property is a string that matches the Row Version name. If that string equals the name of a Row Version, that Row

Version will be used. An invalid string will default back to normal built-in Row Version behavior. The Version Key property also accepts Data Keys used in expressions, using the same syntax that [Keychain Expressions](#) use. For example, if you wanted to highlight red in all of the rows where the Inventory is less than 20, you can use a simple expression of:

```
Inventory<20?"Row3"
```

Where "Row3" is the name of my custom Row Version enclosed in quotes. This can help me keep track of low inventory, excessive downtime, or anything that may be important to highlight.

Candy Bar Name	Inventory	Cost per Candy	Value of Candy Supply
Pay Day	22	\$ 2.10	\$ 46.20
Baby Ruth	44	\$ 2.40	\$ 105.60
Milky Way	16	\$ 1.90	\$ 30.40
KitKat	28	\$ 1.85	\$ 51.80
Butterfinger	7	\$ 2.30	\$ 16.10
Snickers	19	\$ 2.00	\$ 38.00
Twix	13	\$ 2.05	\$ 26.65

Finally, it is actually possible to use many different Row Versions in conjunction with each other. You can setup multiple custom rows and have a more complex expression that helps decide when a particular Row Version gets used. You can also use multiple built-in rows, and even a combination of built-in rows and custom rows. In the event of a conflict between the custom row and a built in row, the custom row will take precedence. For example, when combining my custom Row3 with the alternate rows, you can see that the row Snickers would be the next alternate row in the table, but since it has less than 20 inventory, it uses the custom Row3 instead.

Candy Bar Name	Inventory	Cost per Candy	Value of Candy Supply
Pay Day	22	\$ 2.10	\$ 46.20
Baby Ruth	44	\$ 2.40	\$ 105.60
Milky Way	16	\$ 1.90	\$ 30.40
KitKat	28	\$ 1.85	\$ 51.80
Butterfinger	7	\$ 2.30	\$ 16.10
Snickers	19	\$ 2.00	\$ 38.00
Twix	13	\$ 2.05	\$ 26.65

You can also reference multiple row versions in the same conditional expression:

```
Inventory<20?"Row3":Inventory<40?"Row4":"Row5"
```

Related Topics ...

- [Table](#)
- [Grouping Data Inside of Tables](#)
- [Table Groups](#)
- [Report Tables](#)

Charts Inside of Tables

Adding Charts Inside of Tables

Adding charts inside of tables provides a lot of flexibility designing reports as well as organizing and displaying data in a report. The most common way of adding charts inside tables is using an unstructured row and placing the chart inside the row. Unstructured rows are highly customizable allowing you to place charts anywhere within the row of the table.

There are two common locations to place a chart inside a table:

- Using a Header followed by a chart component.
- Using a Details Row followed by a chart component.

It is very common to add a chart in an unstructured Header row because then the chart will be at the top of the first page of a report. An unstructured Details row typically works best when used in conjunction with [Grouping Data Inside of Tables](#) or [Nested Queries](#) so that each row has a chart with data from that group or query.

On this page ...

- Adding Charts Inside of Tables
 - Adding a Chart Inside a Header Row
 - Adding a Chart Inside a Details Row
- Using a Chart in a Table with Nested Queries
 - The Tables
 - Queries
 - Configuring the Chart
 - Example Images



Charts Inside of Tables

[Watch the Video](#)

Adding a Chart Inside a Header Row

This example shows several pieces of hardware including how many pieces were produced, and how many pieces were shipped. A Bar Chart was added after the Header row in the table.

1. In the **Data** panel, create a Static CSV data source (i.e., Hardware Data). The dataset is shown below.

Hardware Dataset

Hardware, Produced, Shipped
Nails, 95, 85
Screws, 80, 60
Bolts, 50, 47

2. In the **Design** panel, drag a Table component on to your report. Drag your **Datasource** (i.e., Hardware Data) from the **Key Browser** to the **Data Key** field, and click the **Header** box in the **Configure Tab**.
3. Next, drag each **Data Key** (i.e., Hardware, Produced, and Shipped) to a column in your table in Data Details row.

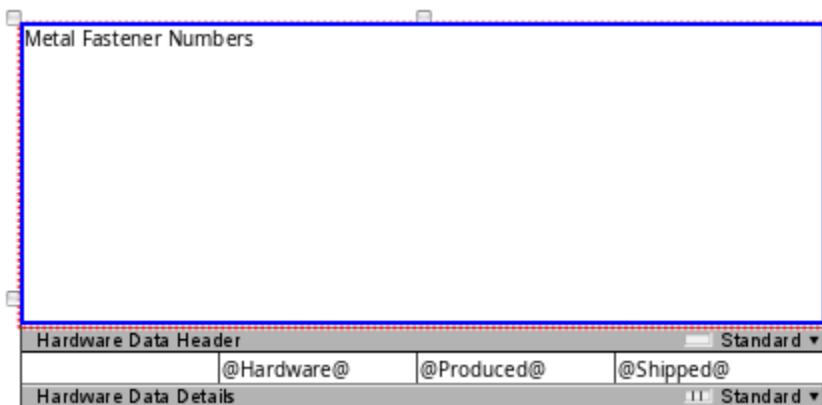
4. Enter the Header name (i.e., Metal Fastener Numbers Report) in the left column of the Data Header row.

The screenshot shows the Report Designer interface. The 'Design' tab is active. On the left, the 'Key Browser' and 'Property Inspector' panels are visible. The 'Configure Table' panel is open, showing the 'Data Key' set to 'Hardware Data'. In the 'Rows' section, the 'Header' checkbox is checked. The main workspace displays a table structure with three rows: 'Metal Fastener Numbers' (Header), 'Hardware Data Header' (Header), and 'Hardware Data Details' (Details). The 'Hardware Data Header' row contains three columns labeled '@Hardware@', '@Produced@', and '@Shipped@'.

5. Select the Header row and click on the **Row Structure** icon to make the row unstructured so you can add a chart.

Note: You can toggle between a structured and unstructured row by simply clicking the Structure Row icon.

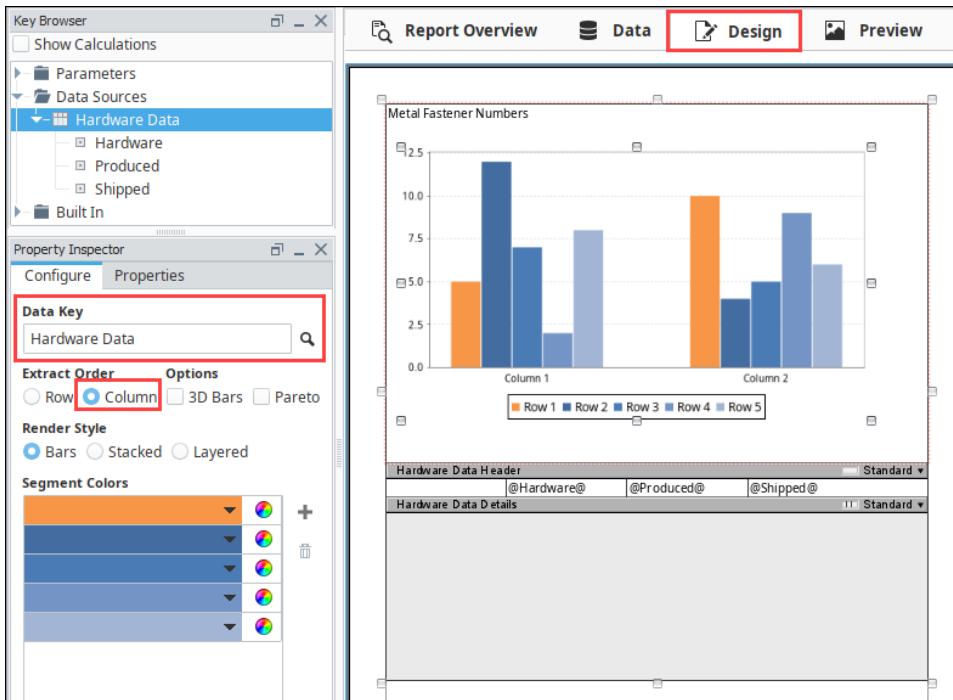
6. With the Header row selected, drag the row down the page to make room for the chart: click and drag on the dark gray bar titled "Hardware Data Header".
 7. Drag in a Bar Chart from the Report Design Component Palette into the unstructured Header row and expand it. While dragging the chart into the header, you'll notice a blue outline around the Header as you hold the mouse button down while the cursor is in front of the header. This signifies that the chart will be placed directly into the header row, as opposed to in front of the table.



8. With the chart is selected, drag a **Datasource** (i.e., Hardware Data) to the **Data Key** of the **Configure Tab**. Set the **Extract Order** as 'Column'.

Extract Order

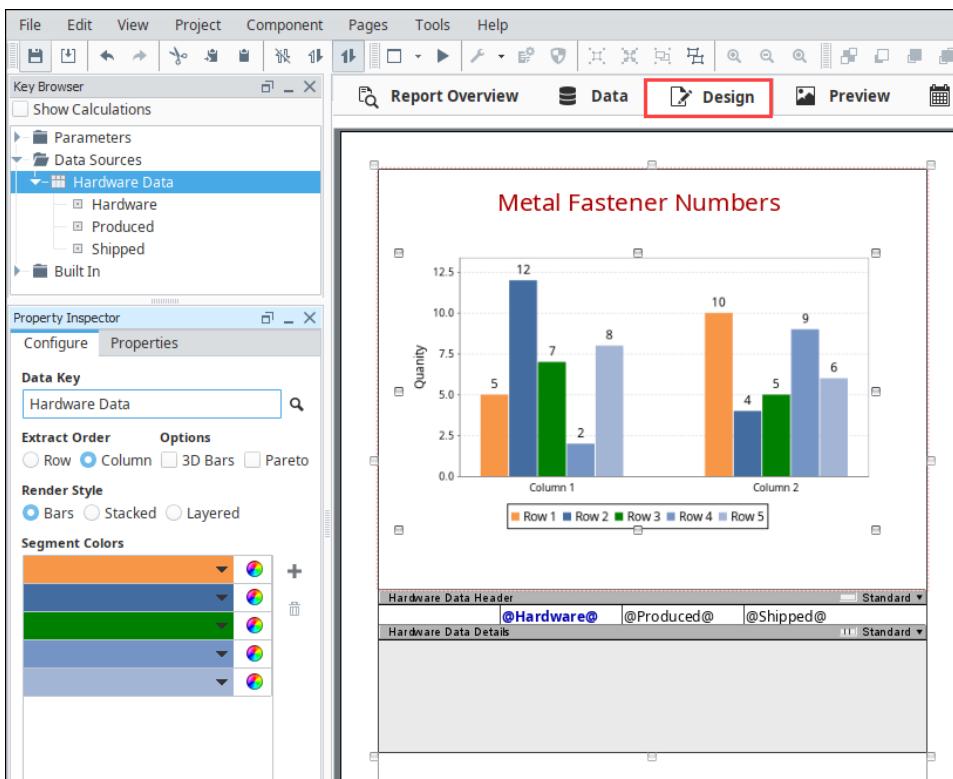
For a Bar Chart, Extract Order is the order in which the data is extracted from the data key, by Row or Column. By default, Row is selected. Use Row when columns in the data key define the series of data. Use Columns when rows in the data key define the series of data; each column is a new value for the same series.



9. Now, let's put your creativity to work and make your report stand out by changing properties. This example changed the following properties:

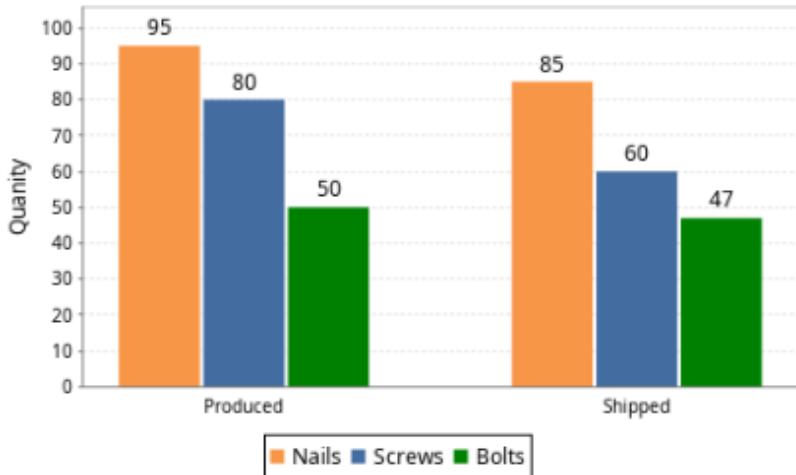
- In the **Configure Tab** - change the third bar color to green.
- Select **Metal Fastener Numbers Report** header - change the Font Size to 24 pixels, the Text Color to red, and the Horizontal Alignment to Center in the Properties tab.
- Select the **Bar Chart** - enter a name for the Axis Label (i.e., Quantity), and set the Bar Labels to 'true' in the Properties Tab.
- Select each **Data Details cells** (i.e., Hardware, Produced, and Shipped), change the Font Size to 14 pixels, and the Hardware Text to blue in the Properties tab.

Here is what your report will look like in the Design panel after changing the properties above.



10. Go to the **Preview** panel to view your report.

Metal Fastener Numbers



Nails	95	85
Screws	80	60
Bolts	50	47

Adding a Chart Inside a Details Row

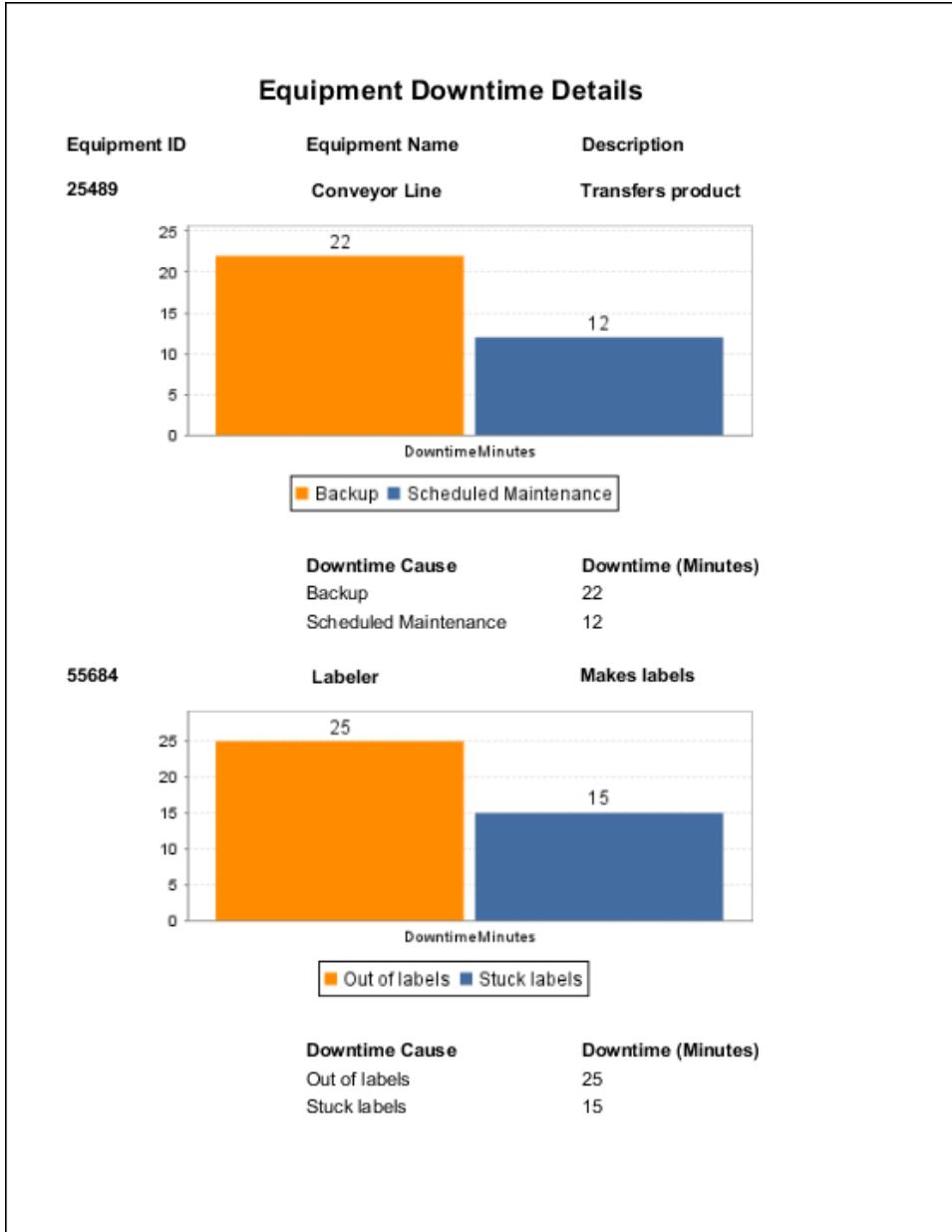
Adding a chart inside a Details row, duplicates the chart for the number of Detail rows you have in your table, and displays the respective data for each row in the chart. This example shows the Equipment Downtime Details for each piece of equipment: Cause and Downtime in Minutes. It also adds a Bar Chart after the Details row using an unstructured row, and a nested query so that each row has a chart with data from the query. Go to the [Nested Queries](#) page and complete the Equipment Downtime example at the bottom if you want to use the same datasets for this example.

1. Once you have your Data set up from the above example, click on the **Design** tab and add a table component to your report.
2. With the table created, add a Header row to the table. Drag all of your Equipment details into the table, and add header titles for each column.
3. Select the **Details** row, make the row **unstructured** , and drag the row down the page to make room for a chart.
4. Drag a chart from the Report Design Component Palette into the unstructured row and expand it.
5. With the chart selected, drag a **Datasource** (i.e., EquipDowntime) from the **Key Browser** to the **Data Key** field in the **Configure Tab**.

6. If you are using a Bar Chart, set the **Extract Order** to **Column**.

The screenshot shows the Report Designer application with the 'Design' tab selected. In the center, there is a report section titled 'Equipment Downtime Details'. It contains a chart with two groups of bars. The first group, labeled 'Column 1', has one orange bar (labeled 'Row 1') and four blue bars (labeled 'Row 2', 'Row 3', 'Row 4', 'Row 5'). The second group, labeled 'Column 2', has one orange bar (labeled 'Row 1') and four blue bars (labeled 'Row 2', 'Row 3', 'Row 4', 'Row 5'). Below the chart is a table with two columns: 'Downtime Cause' and 'DownTime (Minutes)'. The table header is 'EquipmentDowntime Header' and the body is 'EquipmentDowntime Details'. At the bottom of the report section, it says 'Table Group'. On the left side, there is a 'Project Browser' window showing a tree structure with 'Equipment' selected. Below it is a 'Property Inspector' window showing 'EquipmentDowntime' is selected.

7. Go to the **Preview** panel to view the report. If you want to make any changes to the chart like changing Segment Colors, adding Bar Labels, or adding a page break, etc., go back to the **Design** panel to modify any of the chart properties. When you're finished, return to the **Preview** panel. You'll notice that for each piece of equipment there is a chart and data. You can make each piece of equipment have its own page by setting the Page Break option on the Table under the Configure Table tab.



Using a Chart in a Table with Nested Queries

When working with data from nested queries, representing the sub-query data on a chart in a table row is fairly similar to using two unrelated data sources.

In this example, we used content from two different tables, as showing below

The Tables

Equipment

This table contains identifiers and descriptions for multiple pieces of equipment. A query named "equipment_list" is used by the report to retrieve this information.

equipment_id	equipment_description	equipment_name
1	North Tank	Tank 101
2	South Tank	Tank 202

Equipment_Downtime

This table identifies different downtime events, and the id for the equipment that went down. A query named "downtime_list" is used by the report to retrieve this information.

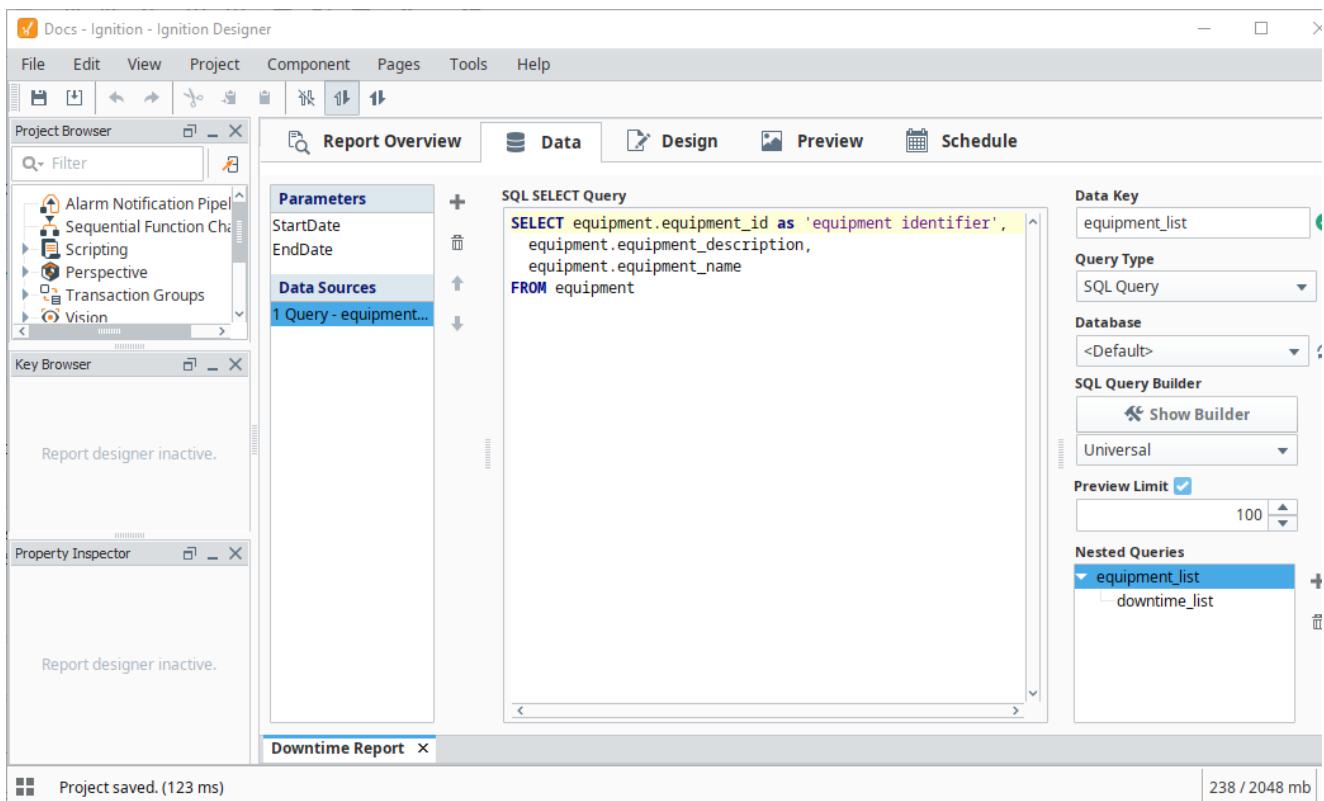
downtime_id	equipment_id	downtime_cause	downtime_minutes
1	1	Tank full	30
2	1	Pump failure	14
3	2	Changeover	60
4	2	Pump failure	34
5	2	Tank full	10

Queries

The queries used by this example are in the expand panel below.

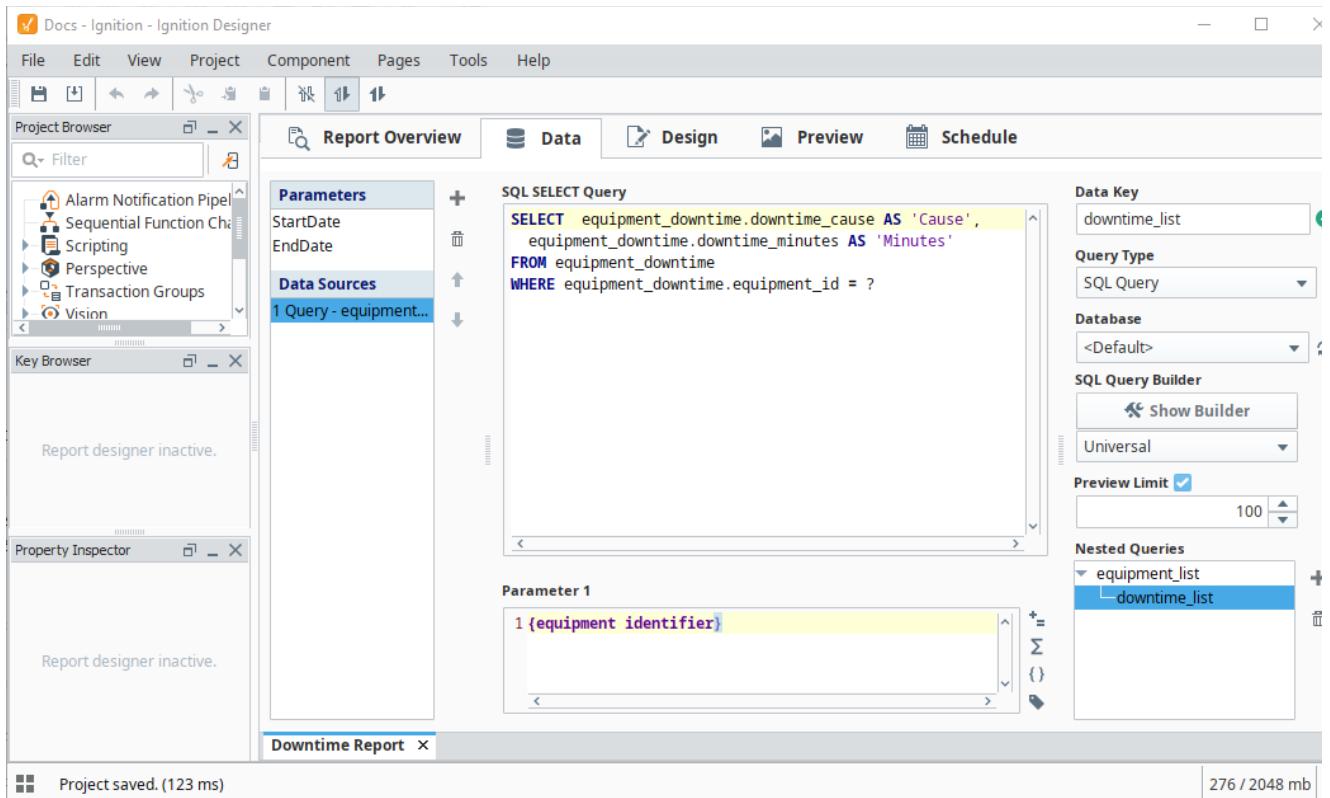
The query for the 'equipment' data source (a "SQL Query" type query) looks like the following:

```
SELECT equipment.equipment_id as 'equipment identifier',
       equipment.equipment_description,
       equipment.equipment_name
  FROM equipment
```



The query for the "equipment_downtime" table (also a "SQL Query" type query) looks like the following:

```
SELECT equipment_downtime.downtime_cause AS 'Cause',
       equipment_downtime.downtime_minutes AS 'Minutes'
  FROM equipment_downtime
 WHERE equipment_downtime.equipment_id = ?
```



Configuring the Chart

In this section, we are going to have a Category Chart component show the results of our downtime_list query. Because we're using a table, and a nested query, each row on the table will represent a different piece of equipment from our equipment table.

For this to work, the table component must be assigned the results of the parent query (which would be equipment_list in this case), and then assign downtime_list to our chart.

1. Create a new table component, and assign "equipment_list" as the **Data Key** for the table.
2. Make the Details row **unstructured**.
3. Add the equipment_name key to the details row, so we know which piece of equipment is represented by the row.
4. Add a **Category Chart** component to the details row of the table.
5. Set the **Data Key** on the chart to "downtime_list."
6. Set the **Extract Order** of the chart to "Column."
7. Switch to **Preview** panel, you should see each both pieces of equipment, represented as different charts, each showing their totaled downtime reasons.

Example Images

Docs - Ignition - Ignition Designer

File Edit View Project Component Pages Tools Help

Project Browser

Report Overview Data Design Preview Schedule

CategoryChart

Tank Summary

equipment_list Header

Column 1	Column 2
Row 1: 5.0	Row 1: 10.0
Row 2: 7.5	Row 2: 9.0
Row 3: 2.5	Row 3: 5.0
Row 4: 12.5	Row 4: 8.0
Row 5: 8.0	Row 5: 6.0

equipment_list Details

Downtime Report

Components

- Table
- CrossTab
- Simple Table
- Labels
- Barcode
- Image
- Graphs & Charts
- Timeseries Chart
- XY Chart
- Bar Chart
- Pie Chart

Shapes

- Text
- Line
- Rectangle
- Ellipse
- Star
- Polygon
- Pencil

Key Browser

Show Calculations

Data Sources

equipment_list

- equipment identifier
- equipment_description
- equipment_name
- downtime_list

 - Cause
 - Minutes

Property Inspector

Configure Properties

Data Key: downtime_list

Extract Order: Column

Options: 3D Bars, Pareto

Render Style: Bars

Segment Colors:

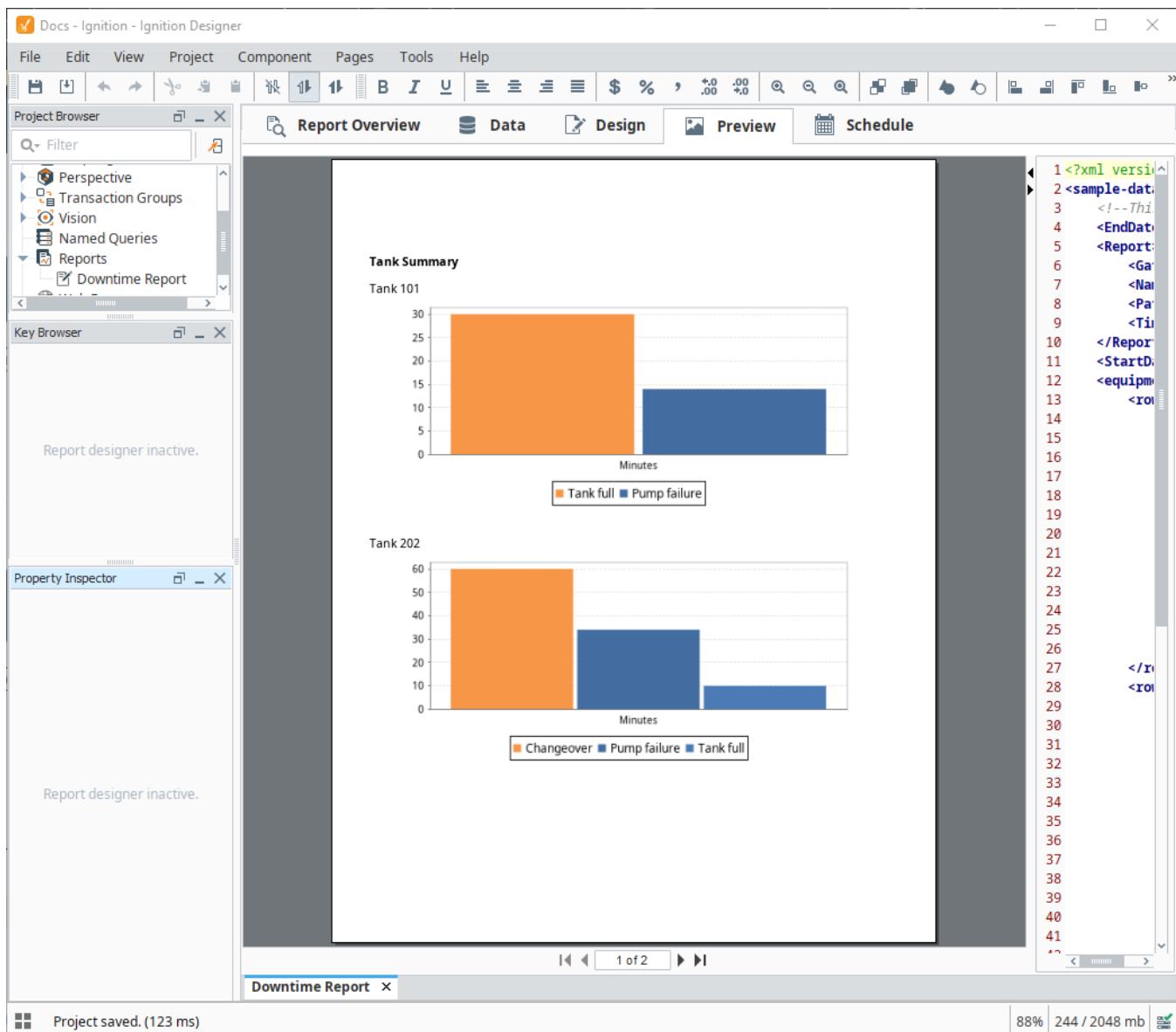
- Row 1: Orange
- Row 2: Dark Blue
- Row 3: Medium Blue
- Row 4: Light Blue
- Row 5: Very Light Blue

Scripting

Enabled: Edit Script

Project saved. (123 ms)

78% 226 / 2048 mb



Related Topics ...

- Grouping Data Inside of Tables
- Nested Queries
- Table Rows

Grouping Data Inside of Tables

Another important feature of tables is the ability to separate a single dataset into different categories or groups. When using a table in a report, you can group data in the table by a specific column. With Dataset Grouping, you can break tables down by data keys that share a common value (i.e., if you have a table that shows addresses, you can group the rows by the city, state, zip code, or any combination of the columns). This is done by dragging and dropping any of your data keys from the Key Browser to the Grouping list under the Configure Table tab.

When you add a data key to the Grouping list, a corresponding Details row will be added to your table component. Using dataset grouping allows you to use data keys to organize and arrange your data into different categories, organizing the results based on the values of a key. Each group can have its own [Header, Details, and Summary rows](#). Additionally, the keys from [Show Calculations](#) and other [keychain](#) functions are supported for any level of grouping.

Note: Table Groups and Grouping Data in Tables are two completely different things despite having similar names. Table Grouping involves using multiple datasets in the same Table component while Grouping Data Inside of Tables (this page) sorts the rows inside a single dataset.

Demonstration

Assuming an initial table that looks like the following:

Type	Count
Type 1	100
Type 2	45
Type 2	450
Type 4	123
Type 3	50
Type 1	250
Type 3	871
Type 2	984

We could utilize Dataset Grouping to group the results in the table by unique "Type" values. By adding a grouping on the Type column, and some additional formatting, we can produce a table that looks like the following:

On this page ...

- [Demonstration](#)

[Grouping Data Inside of a Table Example](#)
[Separating Groupings using Page Breaks](#)



Dataset Grouping

[Watch the Video](#)

Type 1	Count
	100
	250
Type 2	Count
	45
	450
	984
Type 3	Count
	50
	871
Type 4	Count
	123

Notice that we're no longer listing each type individually. Instead, the type acts as a sub-header for each group of data. See the example below for a how-to.

Grouping Data Inside of a Table Example

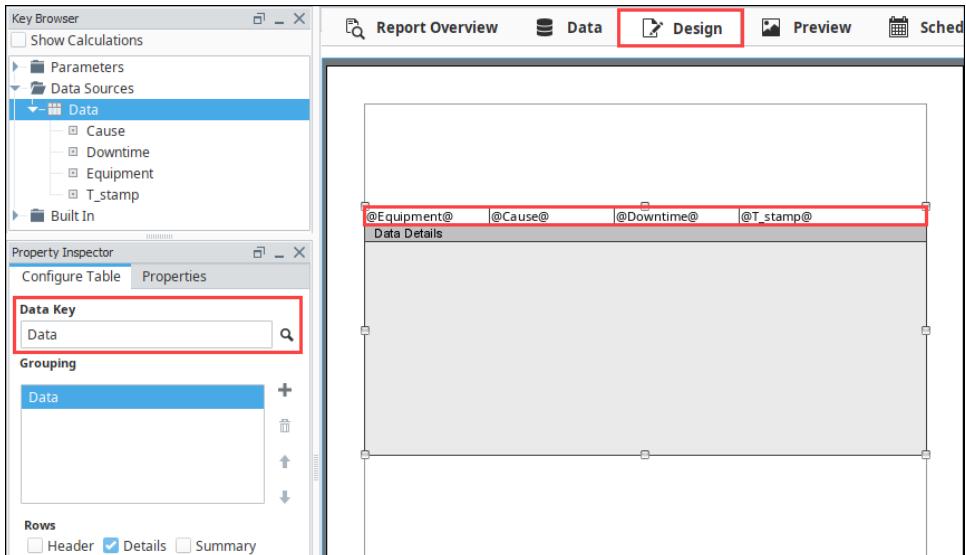
This example begins with a table similar to the one created in the [Report Workflow Tutorial](#). This example will demonstrate how to group the existing downtime report by equipment, collect downtime totals, and introduce some formatting techniques.

1. In the **Data** panel, create a Data Source that has a Timestamp, Equipment, Downtime, and Cause. Shown below is the text needed for a **Static CSV** datasource used for this example.

Data Source for Equipment Downtime Report

```
T_stamp, Equipment, Downtime, Cause
"Jan 20, 2017 17:55", "Labeler", 50, "Out of labels"
"Feb 20, 2017 18:40", "Filler", 120, "Overflow"
"Feb 28, 2017 12:45", "Palletizer", 21, "Misalignment"
"Feb 12, 2017 20:13", "Labeler", 98, "Stuck labels"
"Jan 21, 2017 18:15", "Conveyor Line", 27, "Backup"
"Feb 25, 2017 16:22", "Filler", 2, "Scheduled Maintenance"
"Feb 13, 2017 19:19", "Conveyor Line", 21, "Scheduled Maintenance"
"Jan 20, 2017 15:30", "Palletizer", 241, "Misalignment"
```

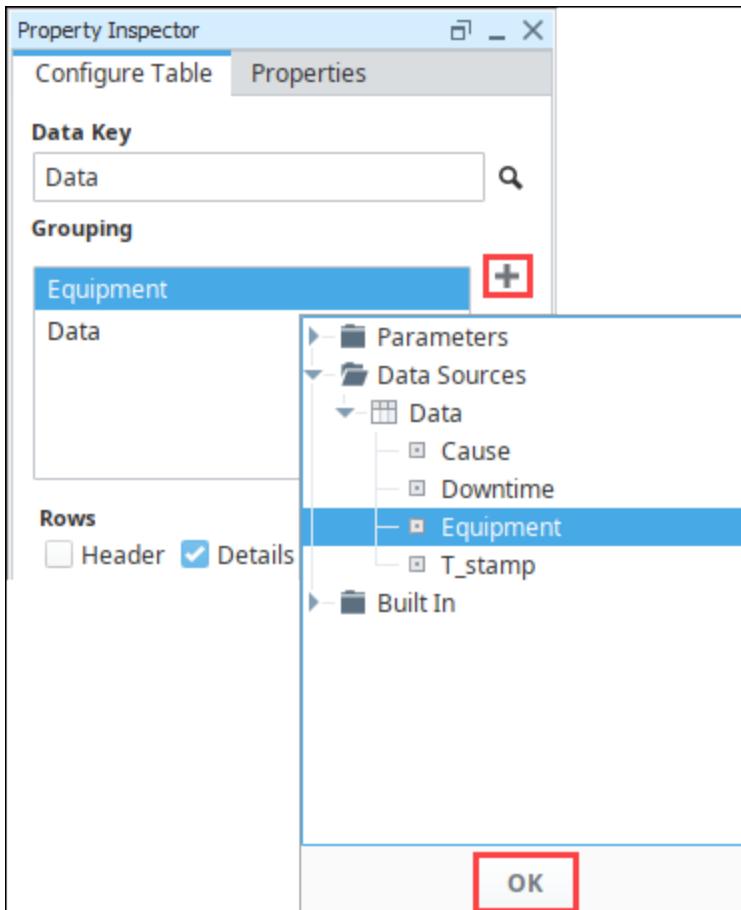
2. In the **Design** panel, drag a **Table** component to your report.
3. With the Table selected, drag the **Data** data source to the **Data Key** under the **Configure Table** tab of the Property Inspector.
4. Drag the each of the **data keys** (i.e., Equipment, Cause, Downtime, and T-Stamp) to any of the columns in the table row.



5. Click on the **Preview** panel to see what the report looks like. You'll notice that all your data is there, but it's a little hard to read because it's not organized.

Labeler	Out of labels	50	Jan 20, 2017 17:55
Filler	Overflow	120	Feb 20, 2017 18:40
Palletizer	Misalignment	21	Feb 28, 2017 12:45
Labeler	Stuck labels	98	Feb 12, 2017 20:13
Conveyor Line	Backup	27	Jan 21, 2017 18:15
Filler	Scheduled	2	Feb 25, 2017 16:22
	Maintenance		
Conveyor Line	Scheduled	21	Feb 13, 2017 19:19
	Maintenance		
Palletizer	Misalignment	241	Jan 20, 2017 15:30

6. Let's sort each row in the table by equipment and group all the equipment together. Go to the Property Editor, and in the **Configure Table**, click the plus icon next to **Grouping**, and a window will open with a list of data keys. Select the '**Equipment**' row, and click **OK**. You'll notice 'Equipment' was added to the Grouping list, and an Equipment Details row was immediately added to the table.



7. Go to the **Preview** panel to see that the report is now sorting by equipment name.

Equipment	Cause	Count	Date
Labeler	Out of labels	50	Jan 20, 2017 17:55
Labeler	Stuck labels	98	Feb 12, 2017 20:13
Filler	Overflow	120	Feb 20, 2017 18:40
Filler	Scheduled Maintenance	2	Feb 25, 2017 16:22
Palletizer	Misalignment	21	Feb 28, 2017 12:45
Palletizer	Misalignment	241	Jan 20, 2017 15:30
Conveyor Line	Backup	27	Jan 21, 2017 18:15

8. Let's remove the Equipment Name from each row in the table and add a Header. Go to the **Design** panel and cut '@Equipment@' from the Data Details row and paste it in the **Equipment Details** row. You can bold it to make it standout.
9. While the table component is selected, go to the **Configure Table** tab in the Property Inspector and select the **Equipment** item in the grouping list and check both the **Header** and **Summary** boxes. Then select the **Data** group and select the **Header** and **Summary** checkboxes for it.
10. Next, make the Equipment header an **unstructured row** and add text as a title for your report using the **Text Shape** in the component palette. Unstructuring the row allows you to easily center the title of your report.

11. Now add header text for each of the Data Details columns (Cause, Downtime, and Date) by typing into the **Data Header** row.

12. Set the **Show Calculations** checkbox at the top of the **Key Browser**. Drill down into the Downtime column and drag the '@total.downtime@' to both the **Data Summary** and **Equipment Summary** rows. You can also add any text outside of the @ symbols. In this example, we added 'minutes' after the total downtime in our cell to '@total.downtime@ minutes'.

13. Lastly, do the same in the Equipment Summary row. We also added the word "Total" to the beginning of the cell: 'Total: @total.downtime@ minutes'.

Note: In any **Summary** row, '@total.Downtime@' is a sum of all downtime at that level of grouping; (i.e, in the **Data Summary** row it is the total downtime grouped by equipment. In the **Equipment Summary** row, '@total.Downtime@' is the sum of all downtime for all equipment groupings).

14. Click over to the **Preview** panel to check out your report. If you want to make any changes, go back to the **Design** panel to update your report. Notice that the 'total' data key respects both groupings.

Equipment Downtime Report

Grouped by Equipment

Labeler

Cause	Downtime	Date
Out of labels	50	Jan 20, 2017 17:55
Stuck labels	98	Feb 12, 2017 20:13
148 Minutes		

Filler

Cause	Downtime	Date
Overflow	120	Feb 20, 2017 18:40
Scheduled Maintenance	2	Feb 25, 2017 16:22
122 Minutes		

Palletizer

Cause	Downtime	Date
Misalignment	21	Feb 28, 2017 12:45
Misalignment	241	Jan 20, 2017 15:30
262 Minutes		

Conveyor Line

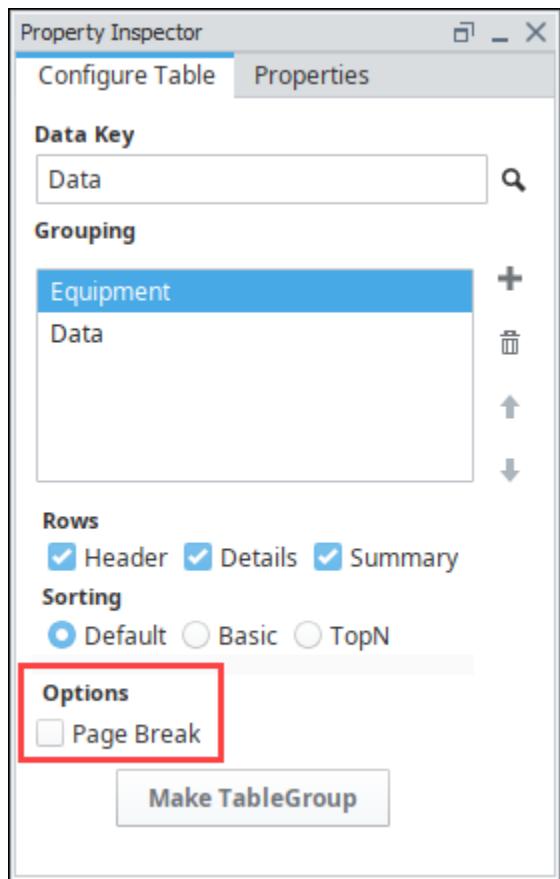
Cause	Downtime	Date
Backup	27	Jan 21, 2017 18:15
Scheduled Maintenance	21	Feb 13, 2017 19:19
48 Minutes		

Total: 580 Minutes

Separating Groupings using Page Breaks

In the **Configure Table** tab of the **Property Inspector**, there is **Page Break** option that can be set to create breaks between each Grouping. Each new instance of that level of grouping creates a new page in the report. In the example above, we could add a page break in-between each grouping of equipment type, which would further delineate each grouping of data. This is especially useful if you are [adding charts](#) or other images at the

beginning of each group.



Related Topics ...

- [Table Groups](#)

Table Groups

Table grouping is an easy way to add multiple datasources to a single Table component showing the results of all datasources in one table. Think of it like a single component that contains many tables, each with its own unique data. There are two types of Table Groups you can create within a Table component: **Peer Tables** and **Child Tables**. Peer Table Groups allow the second table to begin exactly where the first table ends. If multiple tables are used, they can be configured as multiple page templates, providing a page break between tables. Child Table Groups allow you to nest one table inside another, and work especially well with any **Nested Queries**. What's really nice about Table Groups and Nested Data Sources is that you can create Summary Tables for categories of items or drill-down charts all in one report.

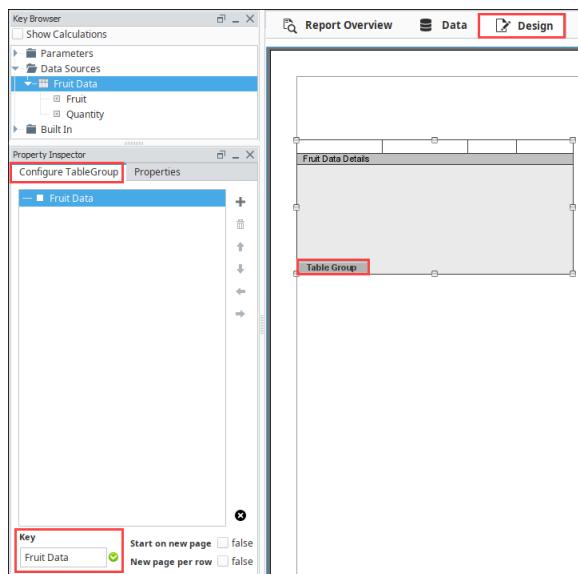
Note: Table Groups and Grouping Data in Tables are two completely different things despite having similar names. Table Grouping (this page) involves using multiple datasets in the same Table component while Grouping Data Inside of Tables sorts the rows inside a single dataset.

On this page ...

- Making a Table Group
 - Navigating Within Table Groups
 - Child versus Peer
- Peer Tables
- Child Tables
- Using Child Tables - An Example

Making a Table Group

To make a Table Group, you simply need to click the **Make TableGroup** button in the Configure Table tab of your table. You will be able to tell that it worked when you notice a **Table Group** icon displayed in the lower left corner of the table component, and your **Configure Table** tab has now changed into a **Configure TableGroup** tab.



Make TableGroup

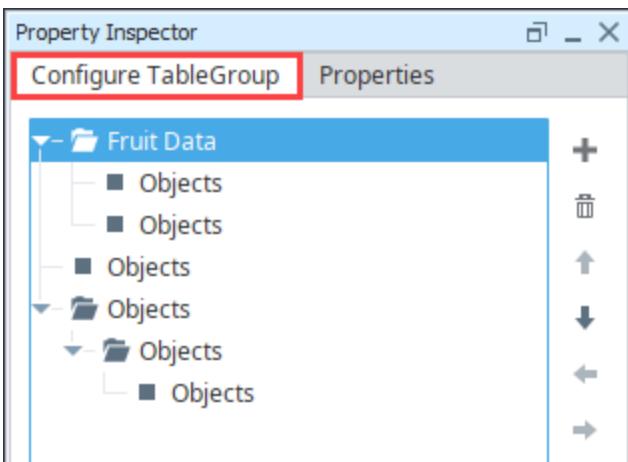


Table Groups

[Watch the Video](#)

We can see our original table listed in the **Configure TableGroup** tab, and we have the option to add additional tables by clicking the plus + icon. Adding a child table will add a table that is a child to the table that is currently selected in the Table Group list. Adding a peer table will add a table that is a peer to the currently selected table. We can add as many of each of these as we need.

For example, in the image below we added two child tables to the original Fruit Data table. We then have two peer tables to Fruit Data, one of them is a standalone table, while one has a child table with the child also having a child table.



As you can see from the image above, the Table Group hierarchy can get quite confusing very quickly. It is important to understand how each part of the Table Group works, and to always be aware of which table you are actually looking at inside the Table Group. To view a different table within the Table Group, simply select it from the hierarchy. The table displayed in the design area will switch depending on which table you selected.

Navigating Within Table Groups

This allows you to get a quick preview of how each of the tables within the Table Group are setup. To go in and configure an individual table within the Table Group, select it from the hierarchy and then click on the table object in the design area. The Configure TableGroup tab in the Property Inspector should change to the Configure Table tab, and will display properties relevant to the table you had selected. This allows you to add a Data Key, set grouping, configure relevant row styles, and format the table in any way that you want. After configuring the table, we can navigate back to the Table

Group configuration by clicking on the **Table Group** icon in the bottom left corner of the table.

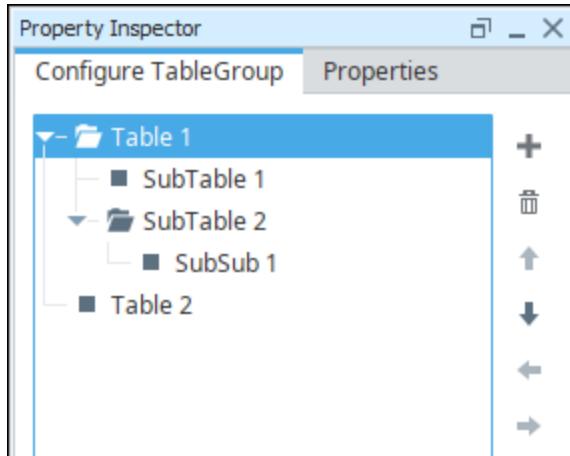
At the bottom of the Configure TableGroup tab, there is an option called **Start on new page**. With this option enabled, we can force the next table in the list to start on the next page instead of immediately after the previous table. This will only apply to the table that is currently selected.

In the Configure TableGroup tab, we can modify the hierarchy of the Table Group using the **Arrow** icons to the right of the table list. This allows us to change the order that the tables appear in, change a peer table into a child table, or change a child table into a peer table. If at some point, you realize you have added too many tables, you can also use the **Delete** icon to remove it. Be careful, as removing a table that has children will also remove the child tables.

Finally, if you ever want to cancel the Table Group click the **Delete** icon to do so.

Child versus Peer

When adding tables to a Table Group, they can be added as either a "Child" or "Peer" table. These terms are always relative to adjacent tables in the group's hierarchy. Thus a table that is a child to one table may have some peers, as well as its own child tables. You can always tell the relationship between two tables by the indentation of each table.



Both Table 1 and Table 2 are peers. This is denoted by the matching indentation. In regard to the image above, the following statements are true:

- SubTable 1 and SubTable 2 are both children to Table 1, because they are indented over from their parent.
- SubTable 1 and SubTable 2 are also peers to each other, because they have a matching indentation.
- SubSub 1 is a child to SubTable 2, and does not have any peer or children tables.

When the report is generated, the first row of **Table 1** will appear first. Each row of **Table 1** will generate an instance of **SubTable 1** and **SubTable 2**. Furthermore, each row of **SubTable 2** will generate an instance of **SubSub 1**. After all of **Table 1** has been represented on the report, **Table 2** will appear on the report.

The behavior of Peer and Child tables are further described below.

Peer Tables

Peer Tables are inserted one after another in the parent table. One table could potentially show details for each piece of equipment, with the peer table showing something completely unrelated starting at the end of the table before it. This helps keep the flow between two different datasources. Typically, each datasource would get its own table component. The issue with that is where to place them with how the tables automatically grow with more data. You could potentially have large amounts of blank space between the two tables because the first table only has a few records. This allows all the data to flow from one datasource to the next, even with unrelated datasources. You can see in the example below, the first image shows what a single individual table looks like, while the second image is a Table Group of two completely unrelated datasources. Each table inside the Table Group is also unique, with their own formatting, style, and even row versions.

Individual Datasource in Singular Table

Fruit	Quantity
Apples	60
Oranges	45
Melons	25

Table Group with Two Different Datasources

Fruit	Quantity
Apples	60
Oranges	45
Melons	25

Customer	Order Quantity
John	47
Mary	82
Tom	14
Peter	31

Child Tables

Child table groupings allow you to nest one table inside of another table. One table could potentially show details for each piece of equipment, but with a child table under each row, you can show all of the relevant downtime occurrences for that piece of equipment. These types of tables work well with [Nested Queries](#).

Using Child Tables - An Example

Creating child tables requires having a complex dataset. Go to the [Nested Queries](#) page and complete the Equipment Downtime example at the bottom if you want to use the same datasets for this example.

1. Once you have your Data set up, click on the Design tab and add a **Table** component to your report.
2. With the table created, add a **Header** row to the table. Drag all of your Equipment details into the table, and add header titles for each column.

Parent Table - 'Equipment'

3. Now that we have a base table, click the **Make TableGroup** button at the bottom of the Property Inspector.
4. Once you make your TableGroup, you're ready to create a Child Table. In the **ConfigureTable Group** tab, make sure the Equipment table is selected and click the plus icon and select '**Add child table**' The table you create becomes a child of the Equipment table. By default, the child table will be called **Objects**.
5. A Child Table has its own datasource and uses a nested query to pull data from the database. Double click on the child table to select it and see the details for that table. **Note:** You can switch back to the Configure TableGroup tab by clicking on the Table Group button in the lower left of the Table component.
6. Drag the child datasource (i.e., EquipDowntime) from the Key Browser to the **Key** field in the Configure TableGroup tab.
7. Next, add a header row with column name text.
8. Drag your columns from the Key Browser to the appropriate columns in your table.

Child Table - 'EquipDowntime'

Key Browser

- Show Calculations
- Parameters
- Data Sources
 - Equipment
 - EquipmentDescription
 - EquipmentID
 - EquipmentName
 - EquipDowntime
 - DowntimeCause
 - DowntimeMinutes
- Built In

Report Overview Data Design Preview Sch

Equipment Downtime Details

	Downtime Cause	Downtime (Minutes)
EquipDowntime Header	@EquipDowntime.DowntimeCause@	@EquipDowntime.DowntimeMinutes@
EquipDowntime Details		
EquipDowntime Summary		

Property Inspector

Configure TableGroup Properties

Equipment

EquipDowntime

Table Group

Key: EquipDowntime Start on new page: false New page per row: false

9. Go to the **Preview** panel to view the report. You'll notice the Child Table is now embedded in the parent table.

Equipment Downtime Details

Equipment ID	Equipment Name	Description
25489	Conveyor Line	Transfers product
	Downtime Cause	Downtime (Minutes)
	Backup	22
	Scheduled Maintenance	12
55684	Labeler	Makes labels
	Downtime Cause	Downtime (Minutes)
	Out of labels	25
	Stuck labels	15
88456	Palletizer	Makes pallets
	Downtime Cause	Downtime (Minutes)
	Misalignment	38
	Misalignment	55
626145	Filler	Fills tanks
	Downtime Cause	Downtime (Minutes)
	Overflow	50
	Scheduled Maintenance	40

Related Topics ...

- [CrossTab and Simple Tables](#)
- [Nested Queries](#)

Images in Reports

Images in Reports

You can spice up your reports by adding images, whether it's your corporate logo, or embedding icons in tables to help users find data quickly. Images can be added to reports in several different ways:

- Drag and drop an image from your desktop/computer to the report - No data key is required.
- Drag an image from the Image Management Tool to a report - No data key is required.
- Drag an Image component from the Report Design Palette to a report. This option requires that you enter a path from wherever the image is located (i.e., local drive, shared drive, or webpage) to the image in the Key property. You can either manually type in the path or reference a parameter, thus, making it static or dynamic.

[Data Keys](#) can help you make the images in reports more dynamic. By dragging a parameter or a datasource column from the Key Browser into the Key property of the Property Inspector, a data Key can resolve to a byte array, such as an image retrieved from a database, or a URL that points to an image on a webpage, or an image file stored on a shared drive. Reports are generated on the Gateway so the image source needs to be accessible from the Gateway computer.

[Keychain Expressions](#) can not be used in the data Key, but the Key can be a parameter or Data Source that dynamically constructs a path to an image. This allows you to easily change images in reports without changing the actual image component.

Caution: Whether using a parameter or typing in a path name in the Key property, always use double quotes around the path name.

Before we dive-in to show you how to configure the paths for your images, please read the Notes contained in this section carefully because path names require a specific format. There are also a few helpful hints about configuring images in reports.

Image Formats

The Report Module supports the following image file formats:

- .gif
- .tiff
- .jpeg
- .jpg
- .png
- .bmp
- .pdf

Drag and Drop Images

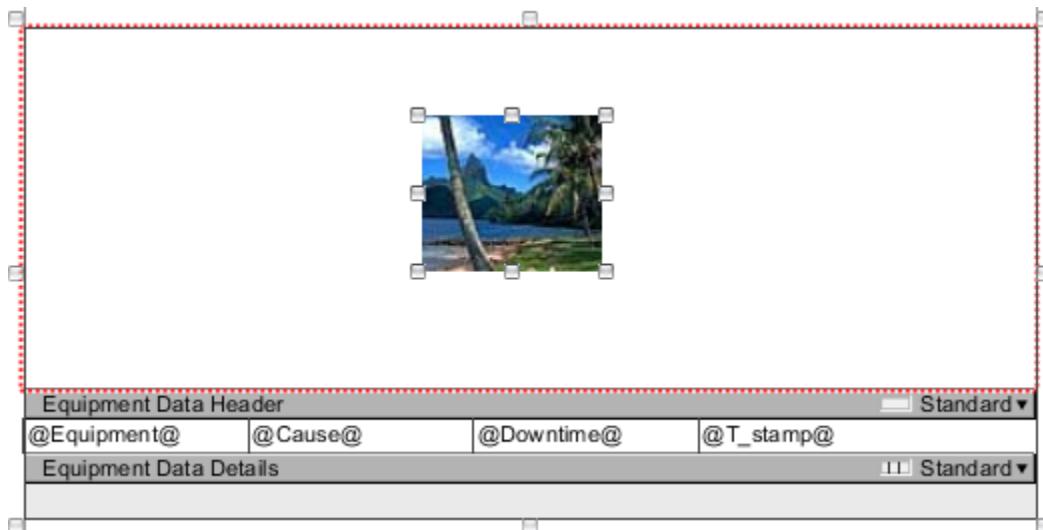
You can use the drag and drop feature to drag images from a local computer, shared drive, webpage, or the Image Management Tool to add images on a report. The report will display blue highlights around your report when you drop your image onto the report. No data key is required, and the dropped image will be used. Below, you can see a table header highlighted to drop an image in.

Equipment Data Header			
@Equipment@	@Cause@	@Downtime@	@T_stamp@
Equipment Data Details			

On this page ...

- [Images in Reports](#)
 - [Image Formats](#)
- [Drag and Drop Images](#)
 - [Drag and Drop an Image from a Desktop](#)
 - [the Image Management Tool](#)
- [The Key Property](#)
 - [Static File Path](#)
 - [Dynamic Key Property](#)

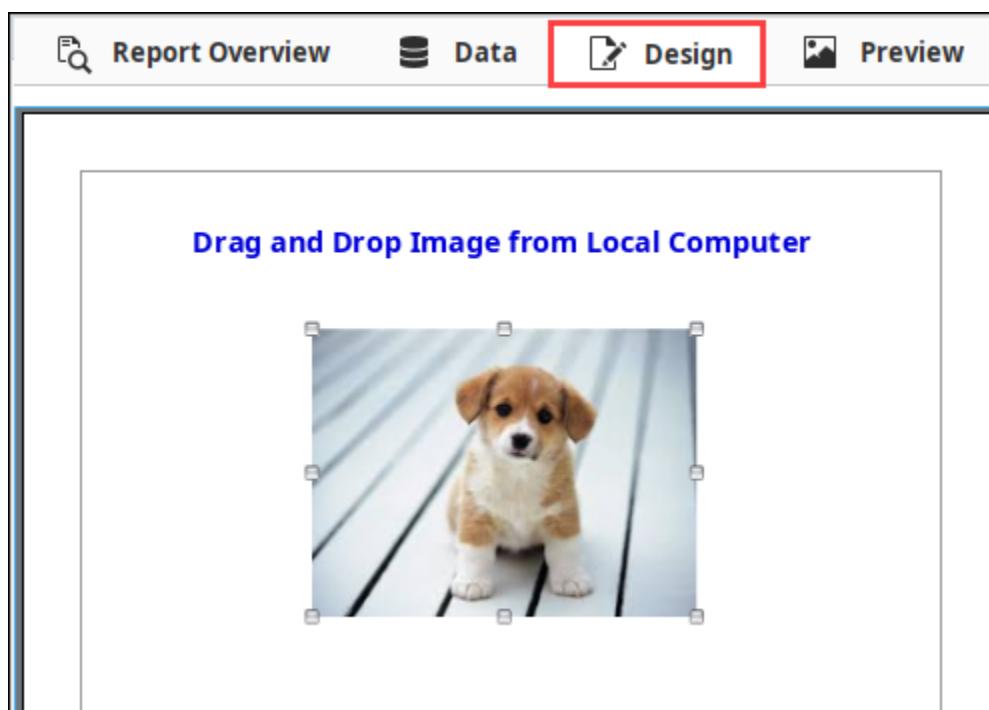
If you associate a key with an image component, the original placeholder image (as shown below) will still be seen in the Design panel, but will be replaced with the image associated with the data Key in the Preview panel. If the key does not link to a valid image, no image will be shown in the Preview panel.



Drag and Drop an Image from a Desktop

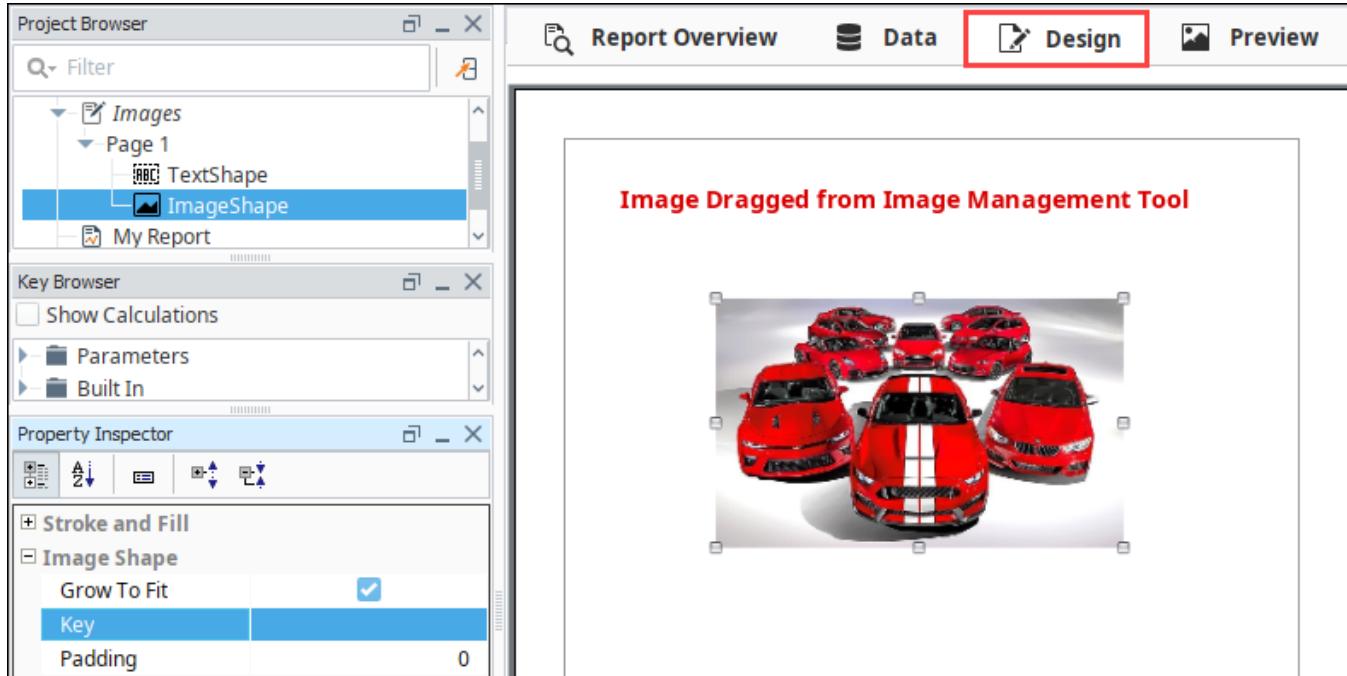
If you have a collection of images stored on your local computer, you can easily drag and drop any image into a report. No Key is required. You'll notice the **Key** property is blank. This doesn't mean you can't associate a Key with an image component. You most definitely can! The image will be replaced with the image associated with the Key when the report is generated. If you're going to use images from a local computer, it's a good idea to import those images to the Image Management Tool to make them visible and available to all project users.

To drag and drop images into a report from your desktop, head to the **Design Panel**. Find an image on your local computer, and drag the image directly onto the report. A new Image component will be created, and will be filled with your selected image. Note that the resulting Image component's Key property will be blank. This is expected behavior, as the image is embedded directly on the page, and not dependent on a key or file path.



the Image Management Tool

Images may also be dragged and dropped directly from the [Image Management Tool](#). Go to your report and open the **Design** panel. Drag an image from the Image Management tool to your report. You'll notice that the Key property is blank. Your image will be displayed directly in both the Design and the Preview panels.



The Key Property

Using the Image component is one way of adding an image in a report. You can make the image path static or dynamic by entering the path name manually, or referencing a parameter. The following sections describe how to setup the image path to use both static and dynamic data keys.

Note:

- Python is used to handle expressions when creating new parameters. Because the backslash has a special meaning in Python, formatting a path name for local drives and shared drives is a little different. Each backslash must be represented by two backslashes, for example, place four backslashes at the beginning of a network path and two backslashes between folders in the path. Here are a few examples;
 - Accessing a local drive - "\\\C:\\Images\\veggies.jpg"
 - Accessing a shared drive - "\\\\ComputerName\\Folder\\Images\\image.png"
- URLs from a webpage, use a forward slash, so use the link as is - "<http://i.stack.imgur.com/WCveg.jpg>"

Static File Path

A static file path is created by dragging an Image component from the Report Design Palette into the Design panel, and manually typing the image path name into the Key property. The image path can point to a drive on a user's local machine, shared drive, or webpage. In this example, the image path is pointing to a user's local machine.

Note: To paste a link into the Key property, use the Ctrl V command. Right clicking with your mouse to paste a link in the Key property is not an available function.

In the **Key** property, enter the path name where the image is located. You can type it in manually, or paste (**Ctrl-V**) the path name in the Key property making sure you have double quotes around the path name. **Note:** The path name uses double backslashes separating the drive and folders (i.e., "C:\\Images\\Images\\veggies.jpg").

The placeholder image will be seen in the Design panel, and the image associated with the Key property will be displayed in the Preview panel or when the report is executed.

Key Browser

- Show Calculations
- Parameters
- Data Sources
- Built In

Property Inspector

Stroke and Fill

- Fill: Black
- Fill Color: Black
- Opacity: 1
- Stroke Style: Hidden
- Stroke: None

Image Shape

- Grow To Fit: Checked
- Key: "C:\Images\veggies.jpg"
- Padding: 0
- Page Index: 0
- Preserve Aspect Ratio: Checked

Rectangle Shape

Basic Properties

Report Overview Data Design Preview Schedule

Placeholder Image

	Vegetable	Qty
Vegetables Header		
	@Vegetable	@QTY@
Vegetables Details		

Go to the **Preview** panel and you'll see the placeholder image was replaced by the image associated with the Key property.

Image Associated with Key Property

Vegetable	Qty
Squash	10
Zucchini	18
Carrots	25
Potatoes	8
Beans	33

Dynamic Key Property

The Key property can be a parameter that dynamically constructs a path to an image which allows you to easily change images in reports without changing the actual image component. You can make an image path dynamic by dragging an Image component from the Report Design Palette into the Design panel, and creating a parameter for the image path. The expression for the Image Path parameter can be a path to a local drive, shared drive, or webpage.

Note, that reports **always** execute on the Gateway, so the file path is always relative to the Gateway.

To create a parameter, go to the **Data** panel, and click the plus icon to add a parameter. In the expression block, enter in a link from a webpage using double quotes around the path name.

The screenshot shows the Data panel with a red box highlighting the 'Parameter Name' field containing 'Car Image Path'. Below it, the 'Parameter Type' is set to 'String'. The 'Default Value (expression)' field contains the URL '1 "http://i.ndtvimg.com/i/2016-10/car-of-the-year_827x510_51476626479.jpg"' with a red box around it. The left sidebar lists 'Parameters' (StartDate, EndDate, Image Path, Car Image Path) and 'Data Sources' (1 Static CSV - Used Car Inventory, 2 Query - tag_history).

Go to the **Design** panel, click on the placeholder image, and expand the **Parameters** folder in the **Key Browser**. Drag your new parameter (i.e., Car Image Path) into the **Key** property of the **Property Inspector**. The image associated with the parameter in the Key property will be displayed in the **Preview** panel or when the report is executed.

The screenshot shows the Design panel. On the left, the Key Browser displays 'Parameters' with 'Car Image Path' selected, and the Property Inspector shows 'Image Shape' with 'Key' set to 'Car Image Path'. The main preview area shows a placeholder image of a car, which is replaced by a real image of a tropical landscape when the parameter is applied. The report title is 'Used Car Inventory' and the date is 'March 2020'. The table structure includes columns for 'Make' and 'Quantity'.

Go to the **Preview Panel**, and you'll see the placeholder image was replaced by the image associated with the parameter in the Key property.

Used Car Inventory

March 2020



Make	Quantity
Ford	13
Toyota	22
Mercedes	2
Buick	6
Lexus	7
Jeep	11

Related Topics ...

- [Data Keys](#)
- [Keychain Expressions](#)

CrossTab and Simple Tables

In the other sections of the Report Table, we talked about creating tables using the Table component, commonly referred to as Standard Tables. Standard Tables focus on a dynamic number of rows with static columns, however, there are other table types that behave a little differently. These other table types are the Simple Table and CrossTab Table.

The CrossTab Table, although similar to standard tables and charts, are commonly used to summarize the relationship between two categories of data by showing summaries of cross sections of the datasource. The Simple Table allows you to easily create a grid-like table structure with the rows and columns being dynamic. Both table types are described below.



Working with Simple and CrossTab Tables

The examples on this page assume you already have datasources created. If not, you can copy the data from the Code Blocks in each example to create your own datasources to be used with the following Simple Table and CrossTab Table examples.

On this page ...

- [CrossTab Table](#)
- [Simple Table](#)

CrossTab Table

The Cross-Tabulation or CrossTab component is a tabular data element like the [Table](#) and [Charts](#). Also known as *Contingency Tables*, they are commonly used to summarize the relationship between two categories of data by showing summaries of cross sections of the data source. To be useful, crosstab data should have the following:

- Lots of repetitive data. Sums, Averages, and other Aggregating functions are well represented by CrossTab Tables.
- A data source that provides at least two columns of data which are repetitive compared to the number of rows.
- One or more columns that represent a value that requires calculation. Examples are: summing money, displaying average response times, counting occurrences, etc. These calculations may be provided as columns or calculations of the data source, or as [Keychain Expressions](#).

The CrossTab component is much simpler than the Table component. By default, it just shows a single cell. This is usually configured with an aggregate key, like "@total.getAmount@". After that, grouping keys are dragged to the horizontal and vertical axis.

The objective of this CrossTab example is to show the number downtime time events by site and equipment. Let's give it a try!

1. In the **Data** panel, create a CSV data source named **Downtime_by_Site**. You can copy the data from the Code Block below to create your own CSV Data Source.



CrossTab Table

[Watch the Video](#)

Downtime_By_Site

```
Equipment,Time,Site
Motor,15,Site A
Motor,23,Site A
Conveyor Line,148,Site B
Pallet Wrapper,58,Site A
Motor,96,Site C
Conveyor Line, 23, Site B
Palletizer,40,Site B
Conveyor Line,56,Site A
Pallet Wrapper,45,Site C
Motor,43,Site C
```

CSV Data Source

The screenshot shows the Project Browser on the left with items like Downtime, Downtime Report, Downtime Site Report, and Equipment DD. The Data panel is active, showing parameters StartDate and EndDate, and a data source named '1 Static CSV - Downtime by Site'. The data grid displays the following data:

Equipment	Time	Site
Motor	15	Site A
Motor	23	Site A
Conveyor Line	148	Site B
Pallet Wrapper	58	Site A
Motor	96	Site C
Conveyor Line	23	Site B
Palletizer	40	Site B
Conveyor Line	56	Site A
Pallet Wrapper	45	Site C
Motor	43	Site C

- In the **Design** panel, drag a **CrossTab** component to your report.
- With the CrossTab component still selected, drag-and-drop your Data Key (i.e., Downtime by Site) from the **Key Browser** to the **Data Key** property in the **Property Inspector**.

The screenshot shows the Key Browser on the left with 'Downtime by Site' selected. The Design panel on the right shows a CrossTab component in the report area.

- In the **Key Browser**, set the **Show Calculations** property to **true**.
- Next, add some data keys to the cells in the CrossTab Table component. From the **Key Browser**, drag the '@Equipment@' to the top cell, '@Site@' to the leftmost cell, and '@count@' to the remaining cell. This will show the downtime count of each piece of equipment by site. In the **Design** panel, the CrossTab should look like the following:

The screenshot shows the Key Browser on the left with 'Show Calculations' checked. The Design panel on the right shows a CrossTab component in the report area. The table structure is as follows:

@Equipment@	
@Site@	@count@

- Switch to the **Preview** panel and your CrossTab Table will display the results.

CrossTab Table

	Motor	Conveyor Line	Pallet Wrapper	Palletizer
Site A	2	1	1	0
Site B	0	2	0	1
Site C	2	0	1	0

Simple Table

The Simple Table is a grid-like table structure that dynamically creates new rows and columns for rows returned by the Data Keys on the component. With the Simple Table you can very quickly add a table inside a report.

The objective of this Simple Table example is to show each piece of equipment including the total amount of downtime, occurrences, and the average duration of each occurrence, broken down by each piece of equipment. Let's get started!

**INDUCTIVE
UNIVERSIT**

Simple Table

[Watch the Video](#)

1. In the **Data** panel, create a CSV data source named **equipment_downtime**. You can copy the data from the Code Block below to create your own CSV Data Source.

```
equipment_downtime

equipment,downtime
conveyor line,78
filler, 68
labeler,84
palletizer,27
```

CSV Data Source

Report Overview **Data** **Design** **Preview** **Schedule**

Parameters
StartDate
EndDate

Data Sources
1 Static CSV - equipment_downtime

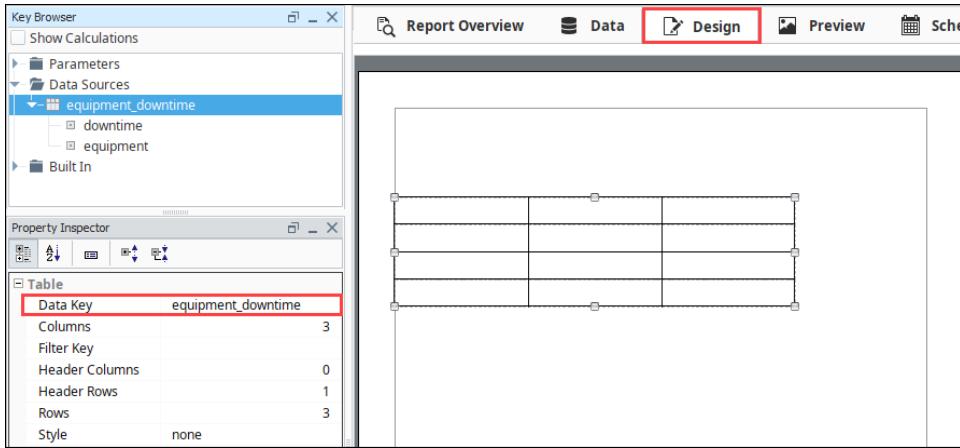
Data Key
equipment_downtime

Data

equipment,downtime
conveyor line,78
filler, 68
labeler,84
palletizer,27

2. In the **Design** panel, drag a **Simple Table** component to your report.

3. With the Simple Table component selected, drag and drop your Data key (i.e, equipment_downtime) from the **Key Browser** to the **Data Key** property of the **Property Inspector**.



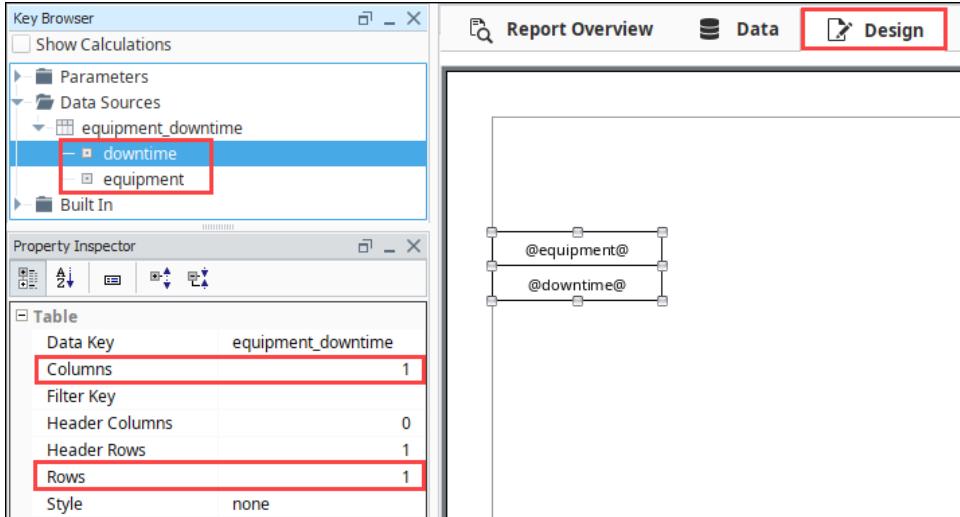
4. From the **Key Browser**, drag the '@equipment@'key to the top row of the Simple Table, and the '@downtime@'key to the second row.
Note: The data keys determine how many columns and rows get created in your table. The Simple Table will add a new column for each value in the 'equipment' key.

Header Row

It's important to note that the top row of the Simple Table is a **Header Row**. In the Property Inspector, there is one Header Row, but you can always add more rows by changing the value, or even setting it to '0' to remove the Header Row.

5. In the **Property Inspector**, change the number of **Rows** and **Columns** to 1. If you need to add or remove columns or rows, you can always change the Column and Row property values.

The Simple Table will look like the following image in the **Design** panel.



6. Next, switch to the **Preview** panel to check out your report. You can always go back to the **Design** panel to edit your table and update your report format.

Report Overview Data Design **Preview** Schedule

Simple Table

conveyor line	filler	labeler	palletizer
78	68	84	27

Report Schedules

Perhaps one of the most compelling features of the Reporting module is the Scheduling system. Once a report has been designed, it can be run and delivered automatically without using a client. Reports run on the Gateway, giving you total flexibility when reports should run, and how they should be delivered.

Scheduling a Report

For any Scheduled Report, there are three settings that need to be configured:

1. When the Scheduled report executes (i.e., date, time, and frequency). There can be multiple schedules on a report, and each schedule can be refined down to a minute resolution.
2. What Parameters will be used to determine the report data. Can use the default parameter values or have new values passed in.
3. The Actions that occur following report generation. There can be multiple actions per schedule.

On this page ...

- [Scheduling a Report](#)
- [Schedule Table](#)
 - [Schedule Tab](#)
 - [Parameters Tab](#)
 - [Actions Tab](#)



Schedules executed on the Gateway

It is important to note that schedules execute on the Ignition Gateway, not in the client. This means that schedules should be created to Gateway local time, not client or Designer time. Reports can also be run in the client using the [Report Viewer component](#).



INDUCTIVE
UNIVERSIT

Report Schedule Tab

[Watch the Video](#)

Schedule Table

The Schedule Table is at the top of the Schedule tab and will display a list of all currently configured schedules and actions. A single report can have multiple scheduled times and actions configured, allowing it to be saved with different parameters, at different times, and with different actions.

Creating a scheduled report is easy. To add a new schedule click on the **Add** icon on the top-right corner of the Schedule panel. In doing so, you've created a new Schedule which can now be configured. To remove any rows, simply select the row in the table and click the **Delete** icon on the right side of the panel. To configure the schedule, select it, and fill in the tabs below the table: **Schedule**, **Parameters**, and **Actions**.

Schedule Tab

The **Schedule** tab is where the scheduled time is set for the **report** to run. Schedules are driven via Cron formatted strings, a popular scheduling format used in computing. We've created an intuitive user interface that allows you to set schedules easily, even if you are not familiar with Cron. The Schedule GUI provides some convenient pre-made **Common Settings**. If there isn't a setting for you in the Common Settings combo box, choose one that is close and then simply customize it using any of the selection boxes below. For more information on the Unix Crontab scheduling, see the [Scheduled Backups](#) section.

You can also enable or disable the schedule by checking the **Enabled** option.

The screenshot shows the 'Schedule' tab selected in a top navigation bar. Below it is a table with one row, showing a scheduled task: 'At 12:00 am, on day 1 and 15 of the month (disabled)'. A 'Save File' button is next to it. To the right are a '+' icon and a trash bin icon. Below the table is a panel with tabs: 'Schedule' (selected), 'Parameters', and 'Actions'. The 'Schedule' tab contains a 'Common Settings' dropdown set to 'On the 1st and 15th of Every Month (0 0 1,15 * *)'. Underneath are five fields: Minutes (0 :00 (0)), Hours (0 Midnight (0)), Days (1,15 On the 1st and 15th of the Month (1,15)), Months (* Every Month (*)), and Weekdays (* Every Day (*)). At the bottom left of this panel is a red box highlighting the 'Options' section, which contains a checkbox labeled 'Enabled' with a checked state. A 'Downtime' tab is visible at the bottom.

Parameters Tab

The **Parameters** tab is where you can override the default values of your parameters when the schedule runs. You can alter these default values to tailor scheduled reports without having to change the Report's design or configuration. For instance, imagine a report which summarizes how many widgets a factory produced during a given shift. Rather than create a separate report for each shift, you can create multiple schedules (one for each shift) and simply alter a shift parameter. Using Parameters and Schedules, users can avoid creating multiple reports while keeping projects more maintainable.

Each parameter of the report will be listed. They can either be set to their default parameter value by selecting the checkbox, or they can be customized by deselecting the checkbox and specifying a parameter value to pass in at the time the scheduled action executes.

In this example, the parameter **Line** is being overridden to the value of the **Reporting/ActiveRun Tag**.

The screenshot shows the Ignition Designer interface with the Actions tab selected. A schedule named "Every hour" has one action assigned: "Save File". The Actions tab is active, and the configuration panel shows the action details. Below the actions, there is a script editor with the code "1 {Reporting/ActiveRun}".

Actions Tab

The **Actions** tab is where you can setup actions that will run following the generation of a scheduled report. There can be any number of Actions associated with a Schedule, covering virtually any requirement for automatically storing, distributing, or notifying upon completion of the report. Each action has its own custom configuration interface to make adding and editing Actions simple. To learn more about how to configure Actions, refer to [Scheduling Actions](#).

The Actions you can perform following report generation are the following:

- **Print File** - Configure print settings which execute when the report is generated. This is often used to create hard copies of reports automatically.
- **FTP** - Send your report to a file server for backups or storage. Automatically backs up your reports to a service.
- **Save File** - An easy way to save a report to a location on your local Gateway computer or shared network drive.
- **Email** - Email your report to a list of email addresses or users with specific roles. You can configure the Subject, Filename, and Body of the email.
- **Run Script** - A Run Script Action provides the ability to fully customize how a finished report is handled. The Run Script Action provides the report's data as well as the bytes generated according to the **Format** option in the configuration panel.

With any of these scheduling actions, you also have the option of running the report immediately, by clicking the **Double Arrow** icon next to the action.

Report Overview Data Design Preview Schedule

Schedule	Actions
At 12:00 am	Save File, Email

Schedule Parameters Actions

Save File Email

From Address: user.name@mail.com

Mail Server: <Select One>

Format: PDF

Retries: 1

Address Source: Email Addresses

Subject: 1 {Report.Name}

Attachment Filename: 1 {Report.Name}
2 + " - "
3 + dateFormat(now())
4 + ".pdf"

Recipient and ReplyTo Emails

Address	Method
user.name2@mail.com	To

Body: 1 "Report attached"

This screenshot shows the 'Schedule' tab of a report configuration interface. It includes a header with tabs for Report Overview, Data, Design, Preview, and Schedule. Below the tabs is a table with two columns: 'Schedule' and 'Actions'. The 'Schedule' column contains a single entry: 'At 12:00 am'. The 'Actions' column contains a single entry: 'Save File, Email'. Below this table is a detailed configuration panel with tabs for Schedule, Parameters, and Actions. The Actions tab is selected. On the left is a sidebar with a tree view showing 'Save File' and 'Email' (which is currently selected). The main configuration area has several sections: 'From Address' set to 'user.name@mail.com', 'Mail Server' dropdown set to '<Select One>', 'Format' set to 'PDF', 'Retries' set to '1', and 'Address Source' set to 'Email Addresses'. Below these are sections for 'Subject' (containing the expression '1 {Report.Name}'), 'Attachment Filename' (containing the expression '1 {Report.Name}\n2 + " - "\n3 + dateFormat(now())\n4 + ".pdf"'), and 'Body' (containing the expression '1 "Report attached"'). To the right of these sections is a 'Recipient and ReplyTo Emails' table with one row: 'user.name2@mail.com' under 'Address' and 'To' under 'Method'. There are also '+' and '-' buttons for adding or removing items from the configuration.

In This Section ...

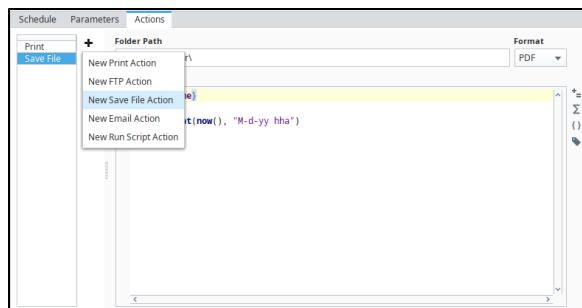
Scheduling Actions

Scheduling Actions

Actions can be configured to run once the report has generated at the scheduled time. Each action has its own custom configuration interface to make adding and editing Actions simple. The Actions you can perform following report generation are Print, FTP, Save, Email, and Run Script. You can even have multiple actions on the same schedule. So you can save the report to the hard drive, as well as email it out to multiple users.

Before creating any Scheduling Actions, you must first [create a schedule](#).

To create an Action, click the **Actions** tab, click the plus icon  and select an **Action**. Actions can be deleted using the trash  icon, and executed immediately using the **Double Arrow**  button.



On this page ...

- [Scheduling Actions](#)
 - [Format Options](#)
 - [Print Action](#)
 - [FTP Action](#)
 - [Save File Action](#)
 - [Email Action](#)
 - [Configuring an Email Action](#)
 - [Run Script Action](#)
 - [Arguments](#)
 - [The dataMap Argument](#)

Format Options

Many of the actions will create a file out of the report. The following file formats are available.

- PDF (recommended)
- HTML
- CSV
- RTF
- JPEG
- PNG
- XML
- XLS
- XLSX

A Note on xls and xlsx Formats

The XLS and XLSX format options may return less than pixel perfect results. This is due to how many spreadsheet programs interpret the resulting file. As a result, the PDF format is recommended in most cases.

Print Action

The Print Action is used to send a report to a printer that is accessible from a computer Ignition is installed on. Here are a list of property descriptions for the **Print Action**.

Property Name	Description
Primary Printer	The primary method of printing the report.
Backup Printer	A backup method of printing the report. Will print using this option if the Primary Printer fails. [Optional]
Print Mode	The mode to print the report in. Can be either Vector or Raster.

	Print Mode	Description
	Vector	Uses math to draw shapes using points, lines, and curves. The most common types of vector graphics are fonts and logos. PDF is a popular vector type. Vector based graphics like SVG image files show images with no pixelation when the size is changed.
	Raster	Are composed of thousands of pixels or dots. Set the dpi (dots per inch). Common raster file format extensions are jpg, jpeg, png, tiff, bmp, and gif.
Copies		The number of copies of the report that will print.
Print on both sides		Will attempt to print on both sides of a sheet of paper, if supported by the printer.
Collate		Orders the pages so that a complete report prints before the next copy prints, if applicable.
Use AutoLandscape Mode		Evaluates the page dimensions and determines portrait or landscape orientation.
Page Orientation		The orientation of the page. Can either be Portrait or Landscape.

The screenshot shows a software interface for printing a report. At the top, there are three tabs: Schedule, Parameters, and Actions. The Actions tab is active. On the left, there's a sidebar with a tree view showing 'Print' (selected) and 'Save File'. The main area contains the following settings:

- Primary Printer:** Default Printer
- Backup Printer:** Adobe PDF
- Print Mode:** Vector (radio button selected)
- Copies:** 2
- Options:**
 - Print on both sides
 - Collate
 - Use AutoLandscape Mode
- Page Orientation:** Portrait

FTP Action

The **FTP Action** can be used to automatically upload your reports to a file server for backups or storage. Here are a list of property descriptions for the **FTP Action**.

Property Name	Description
Server Address	The server address where the report file will be transferred.
Port	The port of the file transfer.
Folder Path	The folder path that the report file will be transferred to.
Format	The file format of the report.

Username	The username that will be used to access the FTP server.
Password	The password that will be used to access the FTP server.
SSL	Will use SSL encryption if True.
Filename	The name of the report file. The Filename property is constructed using the expression language.

Save File Action

The **Save File Action** will save a copy of the report to any folder the Ignition server has access to, such as a local folder or network shared drive. Here are a list of property descriptions for the **Save File Action**.

Property Name	Description
Folder Path	The folder path to save the report files to. This folder path is for the Ignition Gateway server.
Format	The file format of the report.
Filename	The name of the report files. The Filename property is constructed using the expression language.

Scheduling Actions - Save

[Watch the Video](#)



Email Action

The Email Action distributes a report via email when the report is finished executing. There is a Recipients Source property that allows you to send emails using either Email Addresses or User Roles. The 'From Address,' 'Subject,' 'Body,' and 'Attachment Filename' are all configurable. The

Subject, Filename, and Body editors can utilize Expressions to dynamically add content or change names.

Email Server settings must first be configured on the Gateway webpage under **Configure > Networking > Email Settings** page, or in **Email Actions** and clicking the **Create new server** link. Once you create and save an SMTP profile, you can test your email settings for your mail server on the Gateway webpage under **Email Settings**.



INDUCTIVE
UNIVERSITY

Scheduling Actions - Email

[Watch the Video](#)

Creating an email server

Before you setup any reports to be emailed, an email server must be configured. To create an email server if one doesn't exist, use the '**Create new server**' link. This link will take you to **Configure > Email Settings** on the Gateway webpage. There, you will be able to create an SMTP server. For more information, refer to [Gateway Settings](#).

Here are a list of property descriptions for the **Email Action**.

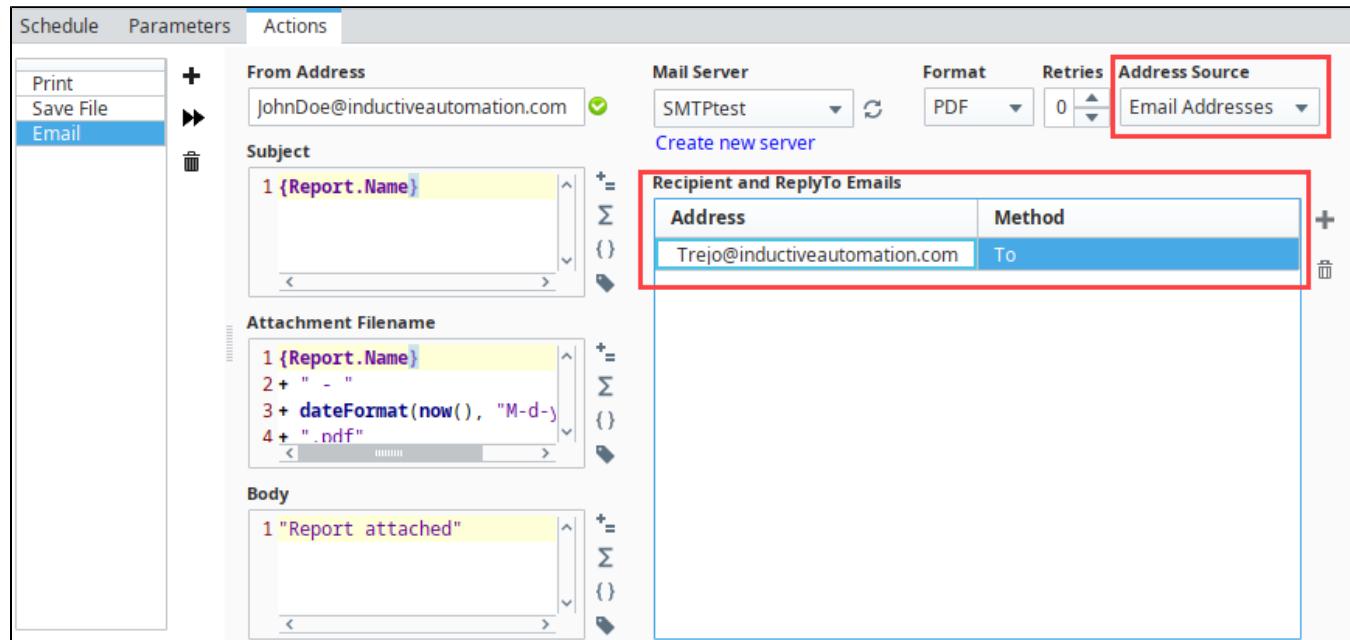
Property Name	Description																
From Address	The Email address from which the report is sent from.																
Mail Server	The mail server to use to email the report. If one doesn't exist, click on the Create new server link. Refer to Email Settings for more information on that page.																
Format	The file format of the report.																
Retries	The number of retry attempts if the email that was sent failed to be delivered the first time.																
Address Source	Will decide how email addresses are collected. Can be either Email Addresses or User Roles. There is a 'ReplyTo' Email function that allows you to reply to email actions using the Email Address and User Roles. This simply adds those emails to the "ReplyTo" header of the email sent to the recipient list, so that if recipients choose to reply to that email, their reply is sent to those email addresses as well.																
	A table of email addresses with a method that determines how those addresses will be used. Click the plus icon on the right side of the window to add additional rows. <table border="1"><thead><tr><th>Property Name</th><th>Description</th></tr></thead><tbody><tr><td>Addresses</td><td>A list of email addresses.</td></tr><tr><td>Method</td><td>The corresponding method of what to do with the email addresses. Options are To, CC, BCC, and ReplyTo.</td></tr></tbody></table> A list of roles where anyone with the given role in the specified user source with an email address will receive an email. <table border="1"><thead><tr><th>Property</th><th>Description</th></tr></thead><tbody><tr><td>Recipient User Source</td><td>The User Source to pull users from that match the Recipient Roles to get an email.</td></tr><tr><td>Recipient Roles</td><td>A list of roles to match with users. Any user that has any of the listed roles will get an email.</td></tr><tr><td>ReplyTo User Source</td><td>The User Source to pull users from that match the ReplyTo Roles that will be listed in the reply to of the email.</td></tr><tr><td>ReplyTo Roles</td><td>A list of roles to match with users. Any user that has any of the listed roles will have their email listed in the reply to of the email that gets sent out.</td></tr></tbody></table>	Property Name	Description	Addresses	A list of email addresses.	Method	The corresponding method of what to do with the email addresses. Options are To , CC , BCC , and ReplyTo .	Property	Description	Recipient User Source	The User Source to pull users from that match the Recipient Roles to get an email.	Recipient Roles	A list of roles to match with users. Any user that has any of the listed roles will get an email.	ReplyTo User Source	The User Source to pull users from that match the ReplyTo Roles that will be listed in the reply to of the email.	ReplyTo Roles	A list of roles to match with users. Any user that has any of the listed roles will have their email listed in the reply to of the email that gets sent out.
Property Name	Description																
Addresses	A list of email addresses.																
Method	The corresponding method of what to do with the email addresses. Options are To , CC , BCC , and ReplyTo .																
Property	Description																
Recipient User Source	The User Source to pull users from that match the Recipient Roles to get an email.																
Recipient Roles	A list of roles to match with users. Any user that has any of the listed roles will get an email.																
ReplyTo User Source	The User Source to pull users from that match the ReplyTo Roles that will be listed in the reply to of the email.																
ReplyTo Roles	A list of roles to match with users. Any user that has any of the listed roles will have their email listed in the reply to of the email that gets sent out.																
Subject	The subject of the Email. The Subject property is constructed using the expression language.																
Attachment	The name of the attached report. The Attachment Filename property is constructed using the expression language.																

Filename	
Body	The body of the email. The Body property is constructed using the expression language.

Configuring an Email Action

1. In the **Schedule panel**, create a **Schedule** to automatically email a report by clicking on the plus icon , if you don't already have one.
2. Next, click on the **Actions** tab.
3. Click on the plus icon , and select the **New Email Action** from the dropdown list.
4. Enter the sender's email address in the **From Address** field.
5. Select the **Mail Server** from the dropdown list. If one does not exist, click the '**Create new server**' link to create one.
6. Select the **Format** from the dropdown list.
7. Enter the number of **Retry** attempts in the event the email failed to be delivered the first time.
8. You can send emails to users using either Email Addresses or User Roles. Under **Address Source** enter either **Email Addresses** or **User Roles**.
Note, email recipients can choose to reply to the email if they prefer, since the email address is added to the 'Reply To' header of the email.
 - a. **Email Addresses** - enter individual email addresses under in the **Recipient and ReplyTo Emails** area. To add multiple addresses, click the plus icon  on the right side of the window. Next, specify the **Method** of how to send the email: **To**, **CC**, **BCC**, or **ReplyTo**.
 - b. **User Roles** - select the User Source from the dropdown in the **Recipient User Source** field.
 - i. In the **Recipient Roles** field, begin typing a configured role and Ignition will validate it.
 - ii. In the **Reply to User Source**, select the User Source from the dropdown. (Optional)
 - iii. In the **ReplyTo Roles** field, enter the role(s) you want listed in the 'ReplyTo' header of the email. (Optional)
9. Enter in values for the **Subject**, **Attachment Filename**, and **Body** fields, or use the defaults.

Recipients and ReplyTo Emails



The screenshot shows the Ignition Schedule panel with the 'Actions' tab selected. On the left, there is a sidebar with options: Print, Save File, and Email (which is currently selected). The main area has several sections:

- From Address:** Set to "johnDoe@inductiveautomation.com".
- Mail Server:** Set to "SMTPtest".
- Format:** Set to "PDF".
- Retries:** Set to "0".
- Address Source:** Set to "Email Addresses".
- Recipient and ReplyTo Emails:** This section is highlighted with a red box. It contains a table with one row:

Address	Method
Trejo@inductiveautomation.com	To
- Attachment Filename:** Contains the expression language code: "1 {Report.Name} 2 + " - " 3 + dateFormat(now(), "M-d-") 4 + ".pdf"
- Body:** Contains the expression language code: "1 "Report attached"

Recipients Source - User Roles

Run Script Action

This **Run Script Action** allows you to store your report in a database, provide special email code, or anything else you can think of. Run Script exposes the function **handleFinishedReport()** which gives you the report name and path, a mapping of the report parameters and datasets, and the bytes in whatever format you want.

Here are a list of property descriptions for the **Run Script Action**.

Property Name	Property Description
Run Script	An area where a script can be created to do something at the scheduled time.
Format	The file format that the reportBytes parameter should be.



```

def handleFinishedReport(reportName, reportPath, dataMap, reportBytes):
    """
    Provides an opportunity to perform script-based manipulation, review or
    action on a generated Report following its execution. Scheduled Report
    Actions execute on the Ignition Gateway and will have Gateway scope, file
    paths, etc.

    Arguments:
        reportName: The name of the report for which this script should run
        reportPath: The path of the report in your project
        dataMap: A PyDictionary containing the data Objects that were supplied
            to the report
        reportBytes: The generated Report's byte array in the file format
            specified in the Format selection box
    """

```

Arguments

The handleFinishedReport function has the following arguments:

- **String reportName** - The name of the report for which this script should run.
- **String reportPath** - The path to the report in your project.
- **PyDictionary dataMap** - The Python Dictionary containing the Parameters and Data Sources that were supplied to the report. This argument allows you to directly access Parameters and Data Sources in the report. **Note** that once handleFinishedReport() has been called, the report has already been generated, so changing the parameters from this function will not alter the resulting report. Instead, parameters should be altered from the **Parameters** tab.
- **byte[] reportBytes** - The report, presented in a byte array. The format of the report depends on the format specified in the **Format** dropdown list.

The dataMap Argument

There is a special argument in the RunScript Action called **dataMap** that may be used to review the raw data that was used to generate the report. Below is a demonstration of using dataMap.

Using dataMap

```
# The dataMap argument is simply a Python Dictionary with the name of each Parameter and Data Source acting as a key.

# Assuming a Report Parameter named 'shift', the value of 'shift' may be accessed with the following
dataMap['shift']

# Similar syntax may be used to extract the value from a Data Source.
data = dataMap['myDataSource']

# Rows objects, while similar in nature to a dictionary, are different objects.

# Individual rows in the Data Source may be accessed by index.
firstRow = data[0]

# getKeys() may be called on a row to list all of the column headers in the row.
firstHeader = firstRow.getKeys()[0]

# getKeyValue() may be used to access the value of a column in the row.
firstColumnInRow = firstRow.getKeyValue(firstHeader)
```

Common Reporting Tasks

This section contains examples for items we've identified as common tasks: undertakings that many users are looking to utilize when first starting out with a specific module or feature in Ignition. Additionally, this section aims to demystify some of the more complex or abstract tasks that our users may encounter.

The examples in this section are self-contained explanations that may touch upon many other areas of Ignition. While they are typically focused on a single goal or end result, they can easily be expanded or modified after the fact. In essence, they serve as a great starting point for users new to Ignition, as well as experienced users that need to get acquainted with a new or unfamiliar feature.

On this page ...

- [Report Workflow](#)
- [Generate Barcodes](#)
- [Converting Legacy Reports](#)

Report Workflow

Creating a sample report from start to finish. The [Reports Workflow Tutorial](#) will take you through all the different steps on how to bring in data, design a page, and schedule your report. This is a great place to start if you have used the Report module in the past and need a refresher, or if you are new to it and prefer concrete examples over the normal reference style pages of the manual.

Generate Barcodes

Creating sheets of product labels. The report module can quickly and easily generate [Labels with Embedded Barcodes](#). Using one of the special components in the Report Designer components list, you can easily set up standard sizes of paper to print multiple labels from a dataset and even include scanable barcodes and QR codes. Once printed, these labels can be used for shipping, product identification, or tracking.

Converting Legacy Reports

Convert old reports to the current format. The report module was updated in 7.8 and any reports older than that were designed in a different way. The current report module has a built in converter to [Convert Legacy Reports](#) to the new reporting format so you can start scheduling those reports without recreating them from scratch. This example demonstrates the simple conversion process to get you started in the new format.

[In This Section ...](#)

Tutorial: The Report Workflow

Steps to Amazing Reports

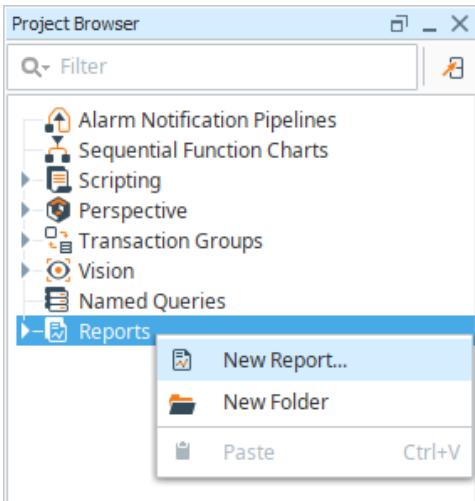
There are three fundamental steps to creating Reports:

1. Gather Data.
2. Design the Report.
3. View or Schedule Delivery.

Simple, right? This workflow is really the essence of what it means to create a report in the Reporting Module. Conveniently, the Reporting interface makes it simple to complete each step and view your result. This tutorial provides a walkthrough of creating a simple report, and reinforce some of the concepts and terminology involved. If you are already familiar with the basics of how to create a Report, you may be interested in the [Reporting Reference in the Appendix](#).

Creating Your First Report

To create your first report, you will need the [Reporting Module installed](#). Once installed, launch the [Ignitio n Designer](#), and let's get started! To create a new report, right click on the Reports node in the **Project Browser**, select **New Report**, and enter a name for your report.



On this page ...

- [Steps to Amazing Reports](#)
- [Creating Your First Report](#)
- [Setting Up Data Sources](#)
- [Design the Report](#)
 - [Adding a Table](#)
 - [Configuring Data for your Report](#)
 - [Adding a Header, Date, and Page Numbers to a Report](#)
 - [Adding a Chart to the Report](#)
 - [Edit Data Sources to Accommodate Report Requirements](#)
- [Viewing Reports in Runtime](#)
- [Creating a Schedule](#)
 - [Scheduling a Save Action](#)
 - [Disabling the Schedule](#)



Simple Report

[Watch the Video](#)

The **Report Workspace** will open, and you'll be in the **Report Overview Panel**. The top of the workspace has a series of stylized panels which guide you through the creation of a new report. It's no coincidence that our Report Workspace is laid out in such a way that lets you intuitively follow the three fundamental steps to creating reports with a little help along the way: gathering data, designing reports, and setting up schedules to run and distribute reports.

The screenshot shows the Report Overview interface with a red box highlighting the top navigation bar. The bar includes tabs for Report Overview, Data, Design, Preview, and Schedule. Below the bar, the title "Productivity Summary" is displayed. Underneath the title, there are sections for "LAST SCHEDULED RUN" (Information not available) and "NEXT SCHEDULED RUN" (No runs scheduled). Two large empty boxes labeled "REPORT DESCRIPTION" and "REPORT SNAPSHOT" are present. The left sidebar contains the Project Browser, Key Browser, and Property Inspector, all showing "Report designer inactive". The bottom status bar shows "Productivity Summary X".

Setting Up Data Sources

For any Report, you will first need to specify one or more Data Sources in the **Data** panel.

1. Click the plus icon to add a simple **Static CSV** Data Source, but if you would like to use data from a database connection, feel free to add a **SQL Query data source** instead.

The screenshot shows the Data panel with a red box highlighting the "Data" tab. The panel includes sections for "Parameters" (StartDate, EndDate) and "Data Sources". A context menu is open at the bottom right of the Data Sources section, with the "Static CSV" option highlighted. The menu options are: New Parameter, Named Query, SQL Query, Basic SQL Query, Tag Historian Query, Tag Calculation Query, Alarm Journal Query, Script, and Static CSV.

2. With the **Static CSV** Data Source editor open, copy and paste the CSV data below into your Data Source editor, and give your data a meaningful name in the **Data Key** field: we named ours **WidgetProduction Data**. This example models data collected from all the

International Widget Factories, complete with production capacity, number of widgets produced, and the number of minutes it took to produce them.

```
Static CSV Data Source

Facility, Capacity, Produced, Minutes
"California", 2000, 665, 345
"Texas", 3424, 1674, 924
"South Africa", 734, 232, 154
"Brazil", 1131, 882, 325
"China", 5324, 2764, 297
"Nozway", 436, 143, 383
"Kenya", 1431, 423, 164
"Italy", 543, 524, 234
"Romania", 154, 78, 45
"Peru", 624, 523, 732
```

Here is what the data looks like in the Static CSV Data Source editor.

The screenshot shows the 'Data' tab selected in the top navigation bar. The 'Data Key' field contains 'WidgetProduction Data' with a green checkmark. The 'Data' pane below lists the CSV data:

```
Facility, Capacity, Produced, Minutes
"California", 2000, 665, 345
"Texas", 3424, 1674, 924
"South Africa", 734, 232, 154
"Brazil", 1131, 882, 325
"China", 5324, 2764, 297
"Nozway", 436, 143, 383
"Kenya", 1431, 423, 164
"Italy", 543, 524, 234
"Romania", 154, 78, 45
"Peru", 624, 523, 732
```

3. Next go to the **Design panel**, and expand the **Data Sources** folder in the **Key Browser**. You'll see that each column of data in the Static CSV data source is represented by its own Data Key. These **Data Keys** are automatically generated based on the table of data fed to the report system. A key may have subkeys, or 'child-keys'. Accessing the data from a child-key is accomplished using the path of a key in relation to its parent, sometimes referred to as a Keychain is simply the path to your data key using 'dot notation'. For instance, the Keychain dragged into a report for our **Facility** key would be @WidgetProduction Data.Facility@.

The Key Browser window shows the following tree structure:

- Parameters
- Data Sources
 - WidgetProduction Data
 - Capacity
 - Facility
 - Minutes
 - Produced
- Built In

The Power of Data Keys

You can have multiple keys in any text field. In addition, you can use multiple keys and common numeric operators to do calculations within the @ symbols! These are called [Keychain Expressions](#). Expressions are not something this introductory tutorial will cover, but you can head over to the [data key documentation](#) page for more information.

Now that you have some data, let's create a simple design and see how the Data Keys can be used to create dynamic reports.

Design the Report

The Design Panel is where we start building the report. This section of the report has many components, such as tables and charts, that can display the results of your data sources and parameters. Components in a report are typically assigned one or more keys, which then feeds the results of our data sources and parameters directly to the component. To create comprehensive reports, you will likely find yourself combining different components. Fortunately, this is pretty easy to do using the visual Designer, so let's continue with your first report and see how easy it is.

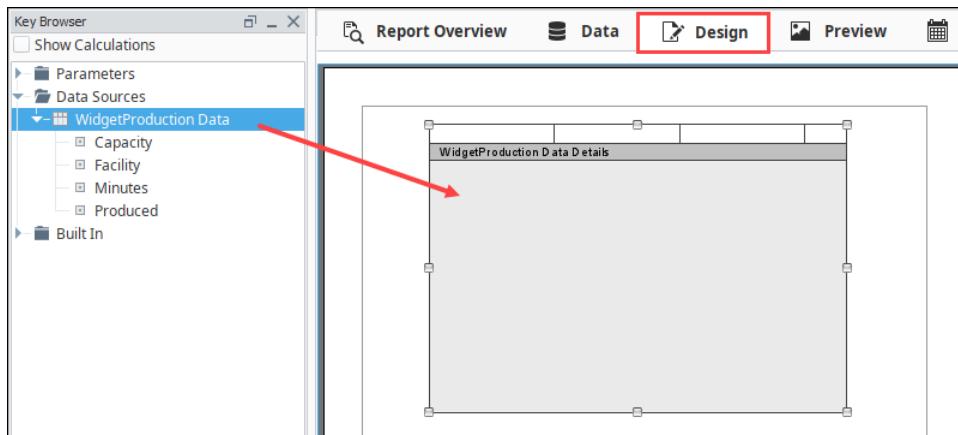
For this report, we will aim for the following requirements:

1. A [Table](#) that gives a summary of how many widgets each factory produced, and the totals for each data column.
2. Factory efficiency on a units/minute basis which needs to be calculated.
3. A Header / Title for the report.
4. Page numbers in case the report gets to be too long.
5. A [Bar Chart](#) that visualizes widget production. This will use a separate data source that summarizes the Table's data.

Adding a Table

Let's add a table and Datasource.

1. In the **Design panel**, drag and drop the **WidgetProduction Data** key from the Key Browser onto the page. When you let go of the mouse button, this will create a Table component.



2. With the Table selected, in the bottom left corner of the **Property Inspector**, you will see the **Configure Table** tab. Note that the Data Key property on the table was automatically assigned to our **WidgetProduction Data** data source. Additionally, you'll see the Details Row (the dark gray bar on the table component) displays the name of our data source. This means that the table was properly assigned a data source.



Alternatively, you may drag a Table component from the Component Palette on the right side of the interface. If you take this approach, you must manually assign a data source by interacting with the Data Key property on the Configure Table tab: click the icon next to the Data Key property, and select the **WidgetProduction Data** data source. It should look like the image below.

Configuring Data for your Report

Now, let's add some keys to the table.

1. In the **Design panel**, drag and drop data keys from the **Key Browser** (i.e., Facility, Capacity, Minutes, Produced) into the **Details** columns. You'll notice that the keys you dropped are surrounded by '@' symbols. The @ symbols tell the report engine that text inside is a key, and it should try and find the key's value when the report is generated.
2. Next, add a header for each of the columns by enabling the **Header** checkbox under the **Configure Table** tab of the Property Inspector. Next, select each header column, type in your header name, and if you like, you can make it standout by bolding the text in the **Text Editor** pane.
3. Go to the **Preview panel**, to check the data in your report.

Facility	Capacity (Units)	Run Time (Min)	Units Produced
WidgetProduction Data Header	@Capacity@	@Minutes@	@Produced@
WidgetProduction Data Details			

Facility	Capacity (Units)	Run Time (Min)	Units Produced
California	2000	345	1674
Texas	3424	924	
South Africa	734	154	232
Brazil	131	325	880
China	5324	297	2764
Norway	43	383	143
Kenya	1431	164	423
Italy	43	234	524
Romania	154	45	78
Peru	624	732	523

Let's take it a step further. Say you would like to calculate the efficiency of units produced per minute of production runtime. You can use the [keychain expression](#) `@Produced/Minutes@`, which will return the value of the Produced key divided by the value of the Minutes key. Note that this calculation is executed for each row, so it is always taking the current row's values to determine the quotient. To do this, you need to add another column in the Details Row and Header Row, using the following steps:

1. With the Table selected, double click the **Details Row** to select it. This opens the **Property Inspector** containing all the Details properties. Under the **Properties tab**, change the **Column Count** property from '4' to '5'. The new column will be added on the right side of your table. You may have to adjust your column widths to see it.
2. Click on the new cell and enter '`@Produced/Minutes@`'. (Refer to the screenshot in the Design panel below).
3. Double click the **Header Row**, and add a column by changing the **Column Count** property from '4' to '5'.
4. In the **Table**, adjust your column widths, and enter 'Efficiency - Units/Min' in the new Header column.

Lastly, let's add a total for the number of widgets produced from all facilities to your report using the following steps:

1. With the **Table** selected, mark the **Summary** checkbox in the **Configure Table** tab of the **Property Inspector**. A new Summary row will appear.
2. Double click on the new **Summary Row**, and under the **Properties tab**, change the **Column Count** property from '4' to '2'.
3. In the **Key Browser**, set the **Show Calculations** property to 'true' making the Built-in keys available.
4. Expand the Datasources WidgetProduction Data Produced object. Drag the 'total' key from **Produced** to the second column of your new **Data to Summary** row.
5. Select the cell you just added to and type the following text in front of your key: '**Total Widgets Produced**'. In the Text Editor you can make it bold so it stands out.
6. Go to the **Preview panel**, to check your report. You should see all your data in the report.

The screenshot shows the Key Browser, Property Inspector, and Preview panels of a report configuration tool.

- Key Browser:** Shows the 'WidgetProduction Data' source expanded, with the 'Produced' object selected. The 'total' key is highlighted with a red box. Other keys shown include 'Capacity', 'Facility', 'Minutes', and various aggregation functions like 'average', 'max', 'min', 'running.total', 'running.average', and 'running.count'.
- Property Inspector:** Shows the 'WidgetProduction Data' data key selected. The 'Configure Table' tab is active, with the 'Rows' section showing 'Header' and 'Summary' checked, and 'Summary' checked in the 'Rows' dropdown. The 'Sorting' section shows 'Default' selected. The 'Options' section shows 'Page Break' checked.
- Preview Panel:** Shows the report design with a table. The table has columns: Facility, Capacity (Units), Run Time (Min), Units Produced, and Efficiency - Units/Min. A summary row is added at the bottom with the text 'Total Widgets Produced: @total.Produced@'. The preview shows the data and the summary row with the calculated total.

Facility	Capacity (Units)	Run Time (Min)	Units Produced	Efficiency - Units/Min
California	2000	345	665	1.93
Texas	3424	924	1674	1.81
South	734	154	232	1.51
Asia	1131	325	882	2.71
China	5324	297	2764	9.31
Norway	436	383	143	0.37
Kenya	1431	164	423	2.58
Italy	543	234	524	2.24
Romania	154	45	78	1.73
Peru	624	732	523	0.71
Total Widgets Produced: 7908				

Adding a Header, Date, and Page Numbers to a Report

Header

The [Text Shape](#) is great to use for a report title, although it can be used anywhere in the report to add text to the page.

1. To activate, click on the **Text Shape** icon on the Component Palette.
2. Once selected, click and drag on the top of the page to create a Text Shape. Give your report a title by typing into the field of the shape.
3. You'll notice the **Edit Text** tab in the lower left corner of the **Property Inspector**, along with some buttons that let you customize the look and layout of the text. This configuration area will change depending on the selected component on the report. In the same area, you'll notice there is a tab titled **Properties**. The Properties tab provides access to a component's various properties. Feel free to experiment with the

settings like font in the editor or property table to customize your title and/or text shapes.



Date and Page Numbers

In addition to a title, let's add some metadata, such as the date the report was generated. However, you don't want to type the date into the text field, because all reports you run (could be today, tomorrow or next year!) will show what date you typed. Instead, you want the date to reflect the day the report was generated.

1. If you look at your **Key Browser**, you'll see a folder called '**Built In**'. Expand this folder and you'll see a number keys that are common in reports.
2. Drag the '**Date**' key and drop it on the report just below your title. This key represents the date and time the report is executed. By default, Text Shapes initially show just the date, but this can be modified by via the **Date Format** property, which is located in under the **Properties** tab when the Text Shape is selected. This example will use the default Date Format, but feel free to modifying this.
3. Next, drag the '**Page of PageMax**' key to the bottom of your report.

Adding text on a page shared with a repeating component (such as a table) will add the text to all pages created by the component. In some cases, like page numbers, date, and title at the top, this is desirable.

Here is what the report look likes in the Design and Preview panels after the header and page numbers were added to the report.

The screenshot shows the Report Designer interface with two main panels: Design and Preview. In the Design panel, a table is being created with columns for Facility, Capacity (Units), Run Time (Min), Units Produced, and Efficiency - Units/Min. A summary row at the bottom contains the formula '@total.Produced@'. In the preview panel, the final report is shown with the same table and a total value of 7908. Red boxes highlight the date placeholder '@Date@' in the header, the summary formula '@total.Produced@' in the table, and the total value 'Total Widgets Produced: 7908' in the preview.

Adding a Chart to the Report

To finish the report, we just need to add a chart. You may need to resize the table a little bit and add a chart right under the page header.

1. In the **Design** panel, drag a **Bar Chart** component from Report Palette into the space above the table. You will need to resize the Bar Chart and the Table to your report page.
2. Next, drag the **Data Source** (i.e., WidgeProduction Data) from the **Key Browser** to the **Data Key** on the **Configure** tab.
3. Go to the **Preview** panel to see how your report looks. Instantly, you'll notice the Bar Chart adds a bar for each column in the data, where our goal for this report is to only show the number of widgets produced. If you want to change the colors of the bars, go back to the Design panel and edit the **Segment Colors** under the Configure Tab.

The screenshot shows the Report Designer interface with two main panels: Design and Preview. In the Design panel, a bar chart is positioned above a table. The table has columns for Facility, Capacity (Units), Run Time (Min), Units Produced, and Efficiency - Units/Min. A summary row at the bottom contains the formula '@total.Produced@'. In the preview panel, the final report is shown with the same table and bar chart. Red boxes highlight the 'WidgetProduction Data' entry in the 'Data Key' dropdown of the Configuration tab, and the chart area in both panels.

Edit Data Sources to Accommodate Report Requirements

If we were using Query data sources, we would create an additional data source or use a nested source to get the two columns we want. For this example, let's create a subset of our original dataset (i.e., WidgeProduction Data) so we only get the data we are interested in.

1. In the **Data** panel, add another **Static CSV** data source (i.e., **FacilityProduced Data Source**) that includes only the Facility and Produced columns. For this example, copy the following code for your data source.

FacilityProduced Data Source

```
Facility, Produced
"California", 665
"Texas", 1674
"South Africa", 232
"Brazil", 882
"China", 2764
"Norway", 143
"Kenya", 423
"Italy", 524
"Romania", 78
"Peru", 523
```

An easy way to do this is to select the chart and drag the 'FacilityProduced Data Source' key onto the **Data Key** field in the Property Inspector. Now when we generate our report with the new data, we only get a Bar Chart with the items produced for each facility.

Facility	Capacity (Units)	Run Time (Min)	Units Produced	Efficiency - Units/Min
WidgetProduction Data Header	@Facility@	@Capacity@	@Minutes@	@Produced@
	@Capacity@	@Minutes@	@Produced@	@Produced/Minutes@
WidgetProduction Data Details				Total Widgets Produced: @total.Produced@
WidgetProduction Data Summary				

Facility	Capacity (Units)	Run Time (Min)	Units Produced	Efficiency - Units/Min
California	2000	345	665	1.93
Texas	3424	924	1674	1.81
South Africa	734	154	232	1.51
Brazil	1131	325	882	2.71
China	5324	297	2764	9.31
Norway	436	383	143	0.37
Kenya	1431	164	423	2.58
Italy	543	234	524	2.24
Romania	154	45	78	1.73
Peru	624	732	523	0.71

Total Widgets Produced: 7908

Viewing Reports in Runtime

Once your report is created, you can view the report from the Ignition runtime or Preview Mode of the Designer with the **Report Viewer** component. You can add the Report Viewer component to any Perspective View and Vision window. In Perspective, enter source path for the report by doing a copy path on the report. In Vision, select the name of your report from the **Report Path** parameter dropdown. To learn more, refer to [Reporting in Perspective](#) and [Reporting in Vision](#) pages.

Saving a report from the Report Viewer is simply a matter of right clicking on the report in the Ignition runtime or Preview Mode of the Designer, and selecting the format you wish to save it as. Selecting from the menu will open a Save or Print dialog in the Client window as well as in Preview Mode.

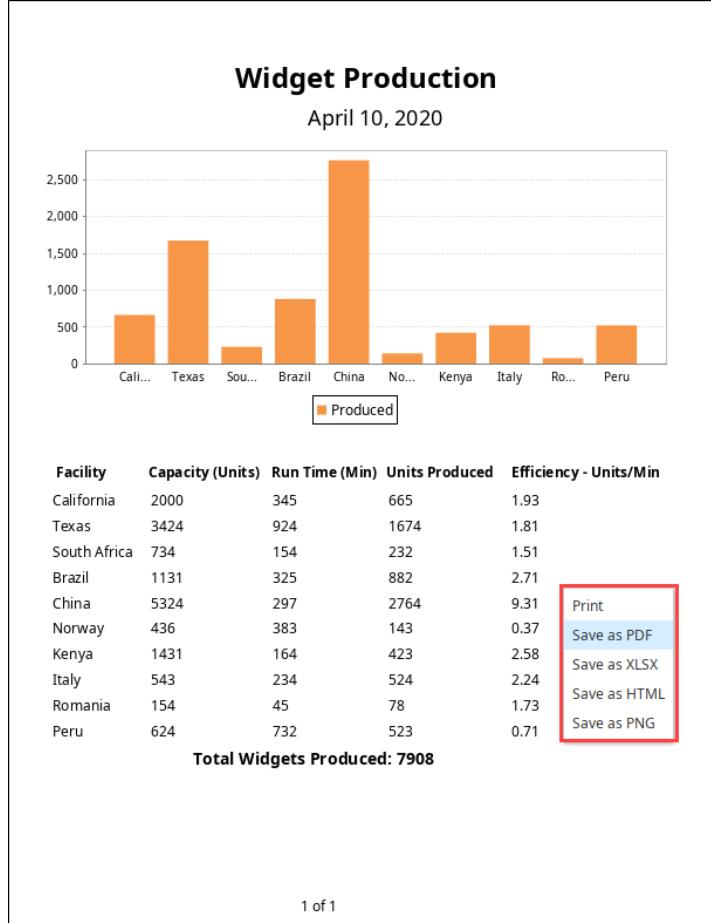
Saving a report from the Report Viewer in Vision is simply a matter of right clicking and selecting the format you wish to save it as from the dialog box in the Client and Preview Mode. In Perspective, you have the option to download a [report](#) to your local device or print a [report](#) to your local printer using the built-in controls at the bottom of the report.

The Report Viewer component supports the following options:

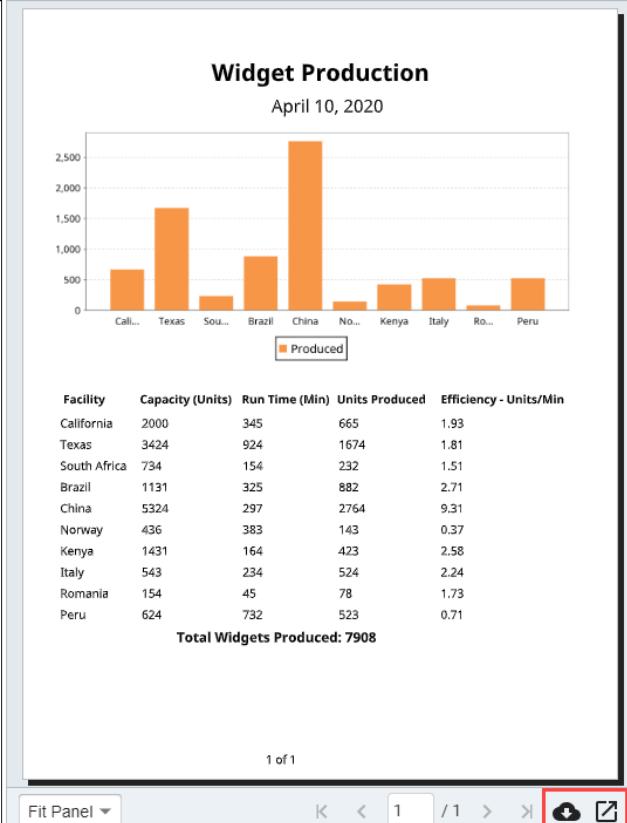
- **Print** - send your report to a connected printer

- **PDF** - formatted for viewing in PDF Viewers
- **HTML** - an HTML document viewable in your web browser
- **PNG** - save as an image

Vision Report Viewer



Perspective Report Viewer



Creating a Schedule

Once a report is designed, you can have a list of scheduled times and actions that will execute automatically at specified times using the [Report Schedule](#) functionality.

Use the following steps to create a schedule for your report.

1. In the **Schedule panel**, click on the plus icon on the right side of the panel. A new row is added to the table. You'll notice the user interface is split into two sections. On the top is a table which contains a list of all the schedules for your report as well as the Actions that will occur following report generation.
2. The **Schedule tab** is where you setup the schedule for your report to run. The UNIX Crontab format is used to setup schedules, but there are some common schedules available in the dropdown list to select from. To set a schedule, select a row in the Schedule Table. Once the row is selected, the Schedule section becomes editable.

The screenshot shows the Ignition Report Overview interface with the 'Schedule' tab selected. A new schedule is being created, indicated by the plus icon (+) in the top right corner. The 'Common Settings' dropdown is set to 'Twice Per Day (0 0,12 * * *)'. The 'Enabled' checkbox is checked.

- To create a new schedule, select a schedule from the Common Settings dropdown list, or choose from one of the Common Settings and customize it using any of the selection boxes. In this example, the **Hours** dropdown was set to **Every 12 hours (*12)**, which will run the report daily every time the current hour changes to 12 (i.e. midnight and noon). Now that a schedule has been created, we need to specify an Action to occur at the scheduled times.

Scheduling a Save Action

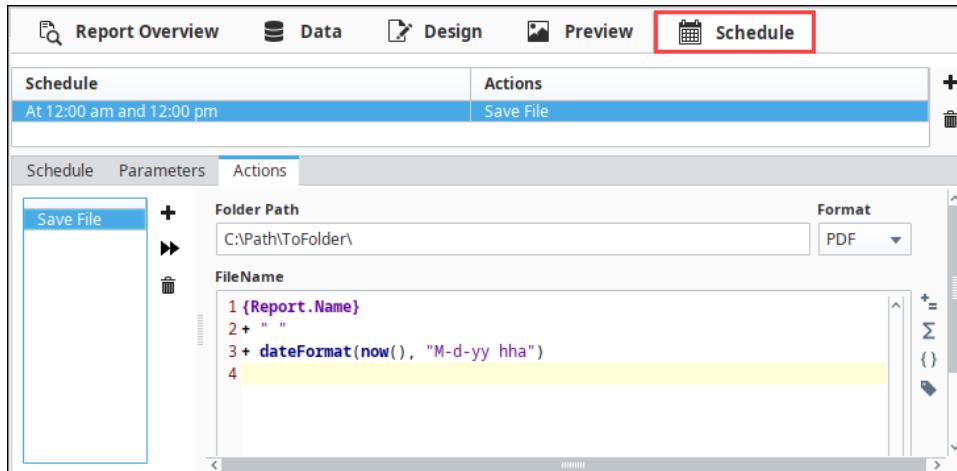
The **Save File Action** saves a copy of the report to any folder Ignition has access to whether it's on your local computer or a shared drive.

Here are a few simple steps to saving a report:

- In the **Schedule** panel, create a **Schedule** to automatically save a report by clicking on the plus icon if you don't already have one.
- Next, click on the **Actions** tab.
- Click on the plus icon and select **New Save File Action** from the dropdown list.

The screenshot shows the Ignition Report Overview interface with the 'Actions' tab selected. A new save file action is being created, indicated by the plus icon (+) in the top left of the actions panel. The 'Folder Path' is set to 'C:\Reports\{date}' and the 'Format' is set to 'PDF'.

- Enter the **Folder Path** where you want to save your report to. You can save them to your local computer or a shared drive. Here are a couple of examples of a folder path for a local computer and a shared drive: "C:\Reports\" and "Share_Drive\Folder\". This path is always relative to the Gateway, so you should always take the Gateway server's operating system into consideration when specifying the path.
- Select the file **Format** from the dropdown list that you want to save the report as. PDF is a very common format.
- By default, your file will be saved with your report name followed by month, day, year, and hour. If you prefer not to use the default filename, you can change it.



7. Save your project. You've just completed configuring a scheduled! You can test out the action by clicking on the **Run Immediately** button (▶)

Disabling the Schedule

Now that your schedule is complete, you may want to disable the schedule so that a new copy of this example report is not created every 12 hours. The **Schedule** tab has an **Enabled** checkbox that can be disabled to prevent the schedule from occurring.

Related Topics ...

- [Reporting in Perspective](#)
- [Reporting in Vision](#)
- [Report Design Components](#)

Labels with Embedded Barcodes

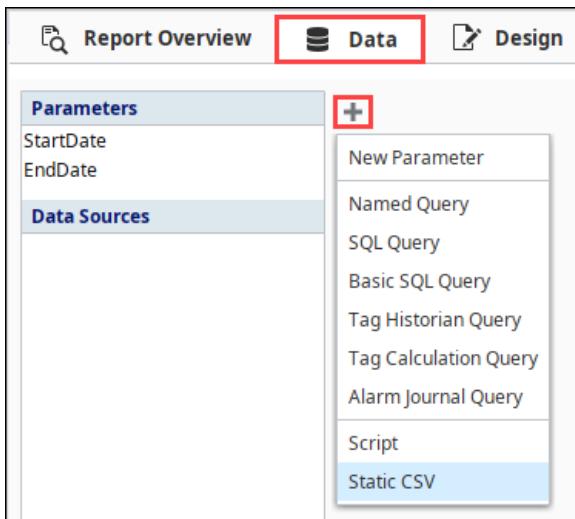
Using labels with embedded barcodes is very common practice. Barcodes are typically used to print out mailing and shipping labels, and can be used to identify and track almost anything. When creating any type of label, you can specify your own dimensions or use standard Avery label sheets. The size of the labels component is based on the values in the Label Attribute section of the Configure Labels tab. When you update any of the Label Attributes (i.e., Rows, Columns, Width, Height, and Spacing), the labels automatically resize based on the values that were specified.

The best way to explain how to setup labels with embedded barcodes is to use an example. Let's create a set of custom labels for the University Research Lab to better track the DNA Primers used in experiments. We want the labels to contain information about each Primer and where it belongs. In addition, we want the barcode to contain the Primer name, number of the lab, and the location within the lab that the samples belong to. We also want the labels to include a sequence number to identify the process order for each step in the experiment. A barcode will be embedded into the template label to create QR Codes, and a QR Code Scanner will be used to track the Primers.

Setting Up Data Sources

Before setting up any labels, you first need to specify a **Data Source** in the **Data panel**.

1. Click the plus icon  to add a **Static CSV** Data Source. This example will use the static data listed below, but could easily substitute real data from a database by using a [SQL Query Data Source](#) data source instead.



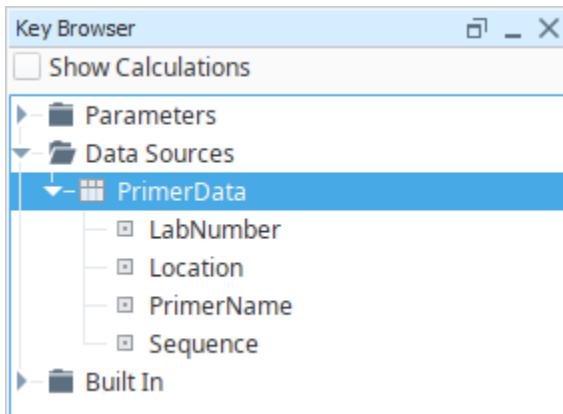
2. With the **Static CSV** Data Source editor open, copy and paste the CSV data below into your Data Source editor, and give your data a meaningful name (i.e., PrimerData) in the **Data Key** field.

PrimerData Datasource
PrimerName, Sequence, LabNumber, Location 16F, CGG TTA CCT TGT TAC GACT T, 2, Cooler 2 3AOX, GCA AAT GGC ATT CTG ACA TCC, 3, Freezer EGFPC1R, CAT TTT ATG TTT CAG GTT CAG GG, 2, Cooler 2 pGLrev, CTT TAT GTT TTT GGC GTC TTCC, 1, Cooler 1 M13R, CAG GAA ACA GCT ATG ACC, 3, Freezer SV40-promoter, TAT TTA TGC AGA GGC GAGG, 1, Cooler 1 pBabe5, CTT TAT CCA GCC CTC AC, 2, Cooler 2 EGFPC1R, CAT TTT ATG TTT CAG GTT CAG GG, 2, Cooler 2 pGLfor, GTA TCT TAT GGT ACT GTA ACT G, 3, Walk-in Shelf A EF-1xForward, TCA AGC CTC AGA CAG TGG TTC, 1, Cooler 1

3. Next, go to the **Design panel**, and expand the **Datasources** folder in the **Key Browser**. You'll see that each column of data in the Static CSV datasource is represented by its own Data Key.

On this page ...

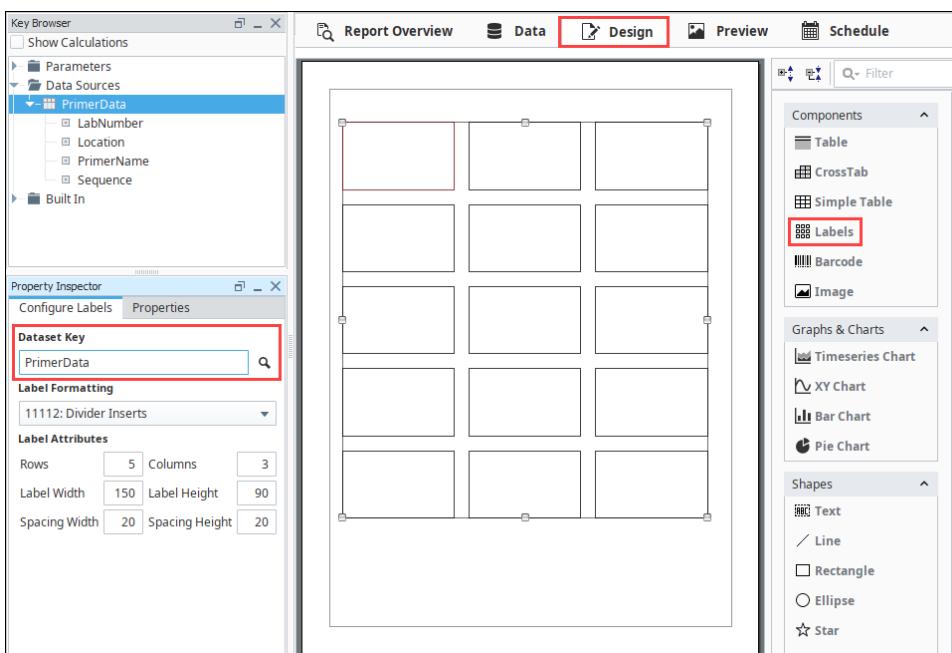
- [Setting Up Data Sources](#)
- [Configuring the Label and Embedded Barcode](#)
- [Printing Labels](#)



Configuring the Label and Embedded Barcode

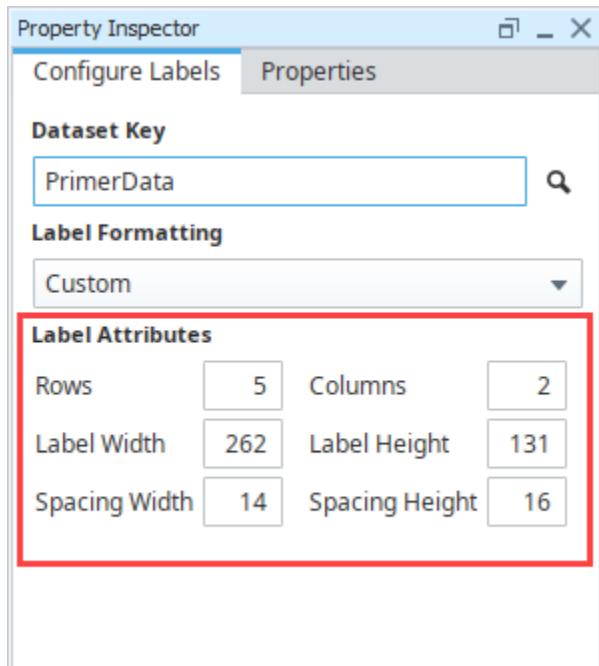
Now that you know what information you want on your labels, you have to decide what you want your labels to look like.

1. In the **Design** panel, drag a **Labels** component from the Report Component Palette to your workspace. By default, 15 labels will display in your workspace.
2. Specify the Data Key that maps to the Data Source that you want to drive the label. Drag your **Datasource** (i.e., PrimerData) to the **Dataset Key** in the Configure Labels tab of the Property Inspector.



3. While the Label component is selected, edit the number and size of the labels in the **Configure Labels** tab. The size of the label is based on the values for the Label Attributes: Rows, Columns, Label Width, Label Height, Spacing Width, and Spacing Height. When any of the Label Attributes are changed, the labels on the page are automatically resized based on these values.

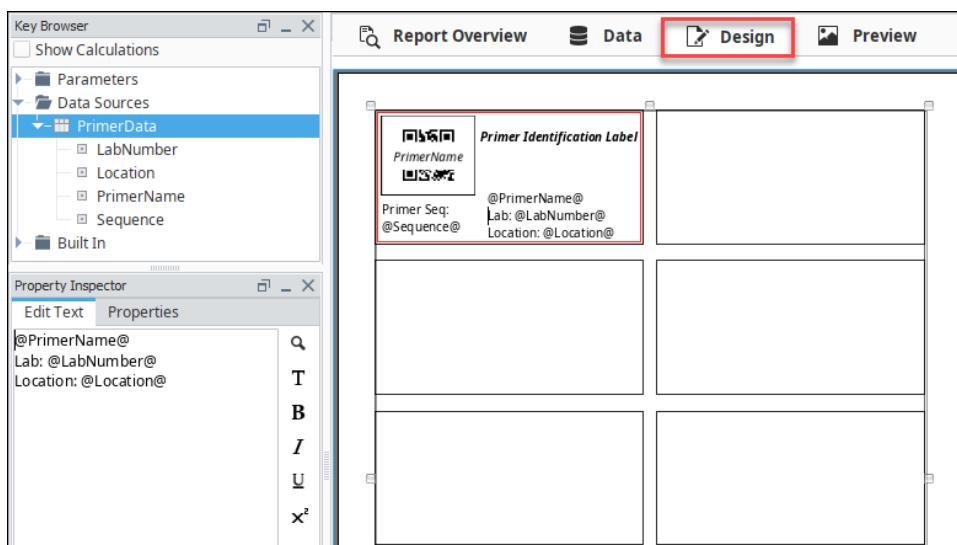
Note: To commit values to any of the Label Attributes, enter a value and hit **Enter**.



i Template Label

The top-left label in the Label component is the template label. Here is where you'll configure your label. To edit a component or shape within a template label, super-select it by double clicking on it. When you **Preview** and **Print** your labels, the template label pulls all the data from the dataset and populates all your labels.

4. Next, drag a **Barcode** component and place it in the upper left corner of the template label.
5. Now you're ready to drag your data keys (i.e., PrimerName, LabNumber, and Location) from the Key Browser to the label template.
Note: Once you drag the first data key to your label template, the **Edit Text** tab will open. You can drag the other data keys directly into the Edit Text tab so they make one shape. This makes editing a little easier.
Next, drag the **Sequence** data key into the lower left corner of the label.

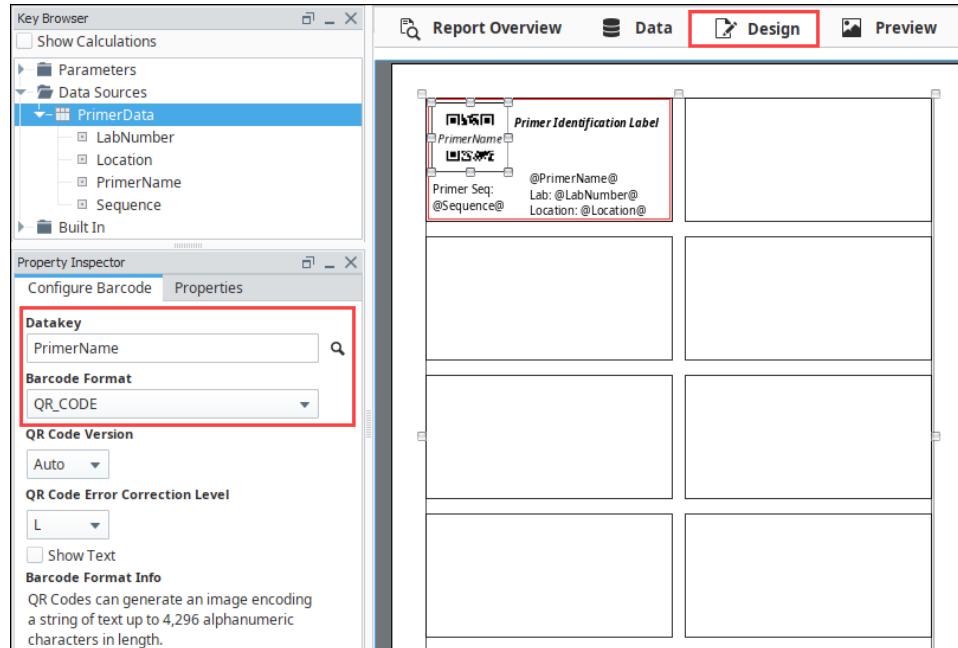


6. Go to the **Preview panel** to verify if all your data populated all the labels, and check how all the data fits on the label because you may need to resize the barcode or text components.
7. Go back to the **Design panel**, click on the **barcode** component, and specify the data key you want to map to the Data Source to drive the barcode. In this example, we want the barcode to encode the PrimerName to the Data Key field of the barcode.
Next, choose a **Barcode Format** from the dropdown list based on your label requirements (i.e., product type, number of characters required, label spacing, etc.). For this example, select the **QR_CODE** barcode format. Note: If you want to push multiple fields into the barcode text,

you can concatenate them together in the Data section of the report.

- Add a Title for your label by dragging a **Text Shape** component to the top of the label. Enter a title inside the text box (i.e., Primer Identification Label).

In addition, in the Edit Text tab, you can mix plain text and data keys, change the font, size, and style, and even bold text that you want to stand out.



- Go to the **Preview Panel**, to view the finished design layout and the HTML code. It's not uncommon to go back to the Design panel to resize components and shapes on the label several times so they fit correctly and the data is readable. This may take several iterations.

```

1 <?xml version="1.0" ?>
2 <sample-data>
3   <!--This is the raw data used to
4   <EndDate>2020-04-15 17:22:32.050
5   <PrimerData>
6     <row-0>
7       <PrimerName>16F</PrimerN
8       <Sequence>CGG TTA CCT TG
9       <LabNumber>2.0</LabNumbe
10      <Location>Cooler 2</Loca
11    </row-0>
12    <row-1>
13      <PrimerName>3A0X</Primer
14      <Sequence>GCA AAT GGC AT
15      <LabNumber>3.0</LabNumbe
16      <Location>Freezer</Loca
17    </row-1>
18    <row-2>
19      <PrimerName>EGFPC1R</Pri
20      <Sequence>CAT TTT ATG TT
21      <LabNumber>2.0</LabNumbe
22      <Location>Cooler 2</Loca
23    </row-2>

```

Printing Labels

You can print your labels by creating a **Print Action** on the Schedule tab, and run it by clicking the "Run Immediately" button. If you receive new shipments of Primers on a set schedule, you can setup the labels to print automatically. For this example, we want to simply print our labels on

demand.

The screenshot shows a software interface with a top navigation bar containing 'Report Overview', 'Data', 'Design', 'Preview', and 'Schedule'. The 'Schedule' tab is highlighted with a red border. Below the navigation is a table with two columns: 'Schedule' and 'Actions'. A single row is visible, showing 'At 12:00 am' in the Schedule column and 'Print' in the Actions column. To the right of the table are '+' and '-' buttons. The main content area is titled 'Actions' and contains a 'Print' configuration panel. This panel includes a 'Print' section with a '+' button and a red-bordered 'Run' button. It also has sections for 'Primary Printer' (set to 'Default Printer'), 'Backup Printer' (set to 'None'), 'Print Mode' (set to 'Vector'), 'Copies' (set to '1'), and 'Options' (checkboxes for 'Print on both sides', 'Collate', and 'Use AutoLandscape Mode' (checked)). A 'Page Orientation' dropdown is set to 'Portrait'.

Converting Legacy Reports

Overview of Converting Legacy Reports

The complete rebuild of the Reporting Module in Ignition 7.8 brought many improvements that would have been impossible to add to the legacy module. To preserve report functionality and prevent problems with backward compatibility, any existing reports will function as they always have. To get the most out of your reports and enable new functionality in such as Scheduled Reports, you'll have to convert your old Vision Report Panel components to new Report Resources. In an effort to minimize barriers, we created a Report Conversion process that will attempt to convert Ignition 7.7- reports into 7.8+ reports.

We encourage users to convert their reports to the new format if they feel they would benefit from the added functionality, but in doing so, it's important to keep some things in mind:

Some Components have Imperfect Conversion

There are a number of components that have been upgraded or completely rebuilt. Due to the changes, some components and/or configurations may not convert perfectly to the newer module. Specifically, the upgrade to the Barcode component and addition of 2D barcodes utilizes a new encoding system that does not have perfect parity with the legacy encoder. The new component does not explicitly support encoders for some of the Narrow and Extended codes, as well as MSI. Reports which require the Narrow or Extended Code 39/Codabar or MSI barcodes will not convert perfectly. Lastly, the changes to the Charts brought many improvements, but will look a little different and may require some configuration.

Data Sources will need Configuration

Data Sources are a huge improvement in the new module. It's now far easier to collect and use data from nearly anywhere in Ignition. Unfortunately, the custom properties in the Legacy Report module do not directly map over to the new Data Sources. When a report is converted, its custom properties will be converted to [Parameters](#). Parameters are great in that they allow for a quick conversion and enable things like [Scheduled Reports](#), but Data Sources will need to be manually configured if desired.

These are the two major caveats to be aware of when converting. The conversion tool has been tested with a variety of legacy reports, but there may be additional factors preventing conversion. We encourage users to either visit the [Inductive Automation forum](#) or contact our [support department](#), and let us know – improvements to the conversion tool can only occur if we are made aware of errors!

On this page ...

- [Overview of Converting Legacy Reports](#)
 - Some Components have Imperfect Conversion
 - Data Sources will need Configuration
- [Conversions are Non-Destructive](#)
- [How to Convert](#)



INDUCTIVE
UNIVERSITY

Converting Legacy Reports

[Watch the Video](#)

Conversions are Non-Destructive

With the potential issues covered, it's important to note that report conversions [do not](#) destroy or alter the original report. If the conversion isn't successful for some reason, the new report can simply be deleted and the old one exists as it always did. If the conversion is successful, the old report can be saved to an unused window or exported as a backup before being deleted. Conversions may not always be perfect, but there is no risk in trying.

How to Convert

Converting a legacy report is quite simple. In the Ignition Designer, open the window where the Report Viewer component exists. If you right click on the component, the new "**Convert...**" option is at the top of the popup menu. Selecting it will start the conversion process which first prompts for the "OK" to proceed, then asks what the new Report Resource should be called.



INDUCTIVE
UNIVERSITY

Sample Legacy Report Conversion

[Watch the Video](#)



Upon conversion, a new Report Viewer will be added to the existing window in addition to the old report. If you check the Project Browser, you'll see the new report under the Reports tree node. Open the newly created Report Resource to make any edits, to add data sources, or add Scheduled Actions!

Related Topics ...

- [Report Data](#)

Alarm Notification

The Alarm Notification module enables you to send a message to a group of users when an alarm becomes active or cleared. This functionality is enabled by having the Alarm Notification module installed with alarm pipelines and Email notification set up. Additional modules can be installed which add [SMS](#) notification and [Voice](#) notification capabilities

Five Steps to Alarm Notification

Regardless of what type of notification you decide to use, there are five simple steps to getting alarm notification setup on an Ignition Gateway.

Notification User

Setting up [users with contact information](#) is the first step to sending out alarm notifications. Most users will have already added contact information when they set up their username and password. This step is typically done in the Config section of the Gateway Webpage.

On-Call Roster

The second step to sending out alarm notifications is creating an [On-Call Roster](#). An On-Call Roster is a collection of users that are notified when an alarm occurs. When an alarm is triggered, it is sent to a designated On-Call Roster where it evaluates the users schedules, and only notifies those users that have an active schedule. Users that are off-schedule will not be notified. The On-Call Rosters are setup in the Config Section of the Gateway Webpage.

Alarm Notification Profile

The [Notification Profile](#) is the third step, and determines what type of notification (Email, SMS, or Voice) is sent out. The notification profile typically connects to an outside system that can do the actual notification. Each alarm notification type can be configured for both one-way and two-way notifications. A notification profile will be set up in the Config section of the Gateway Webpage.

Alarm Notification Pipeline

The next step is setting up an [Alarm Notification Pipeline](#). The Alarm Notification Pipeline creates a list of actions that the alarm will do. A pipeline can send out notification using pre configured Notification Profiles and On-Call Rosters, and can have logic to determine who to notify and when. Alarm Notification Pipelines have a section in the Ignition Designer where they are created.

Adding Pipelines to Tags

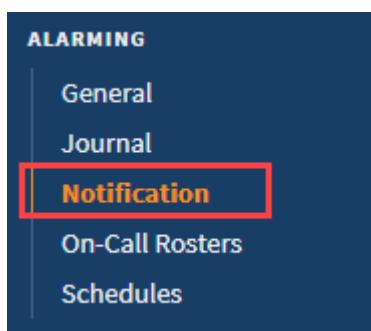
The last step of the alarm notification process is to [add an Alarm Notification Pipeline to a Tag](#). Every alarm on a Tag can have a pipeline set up that the alarm will use when the alarm becomes active, which is most common, but can also use a pipeline to notify users when an alarm becomes acknowledged or gets cleared. This last step of the process is configured on individual alarms on Tags.

Notification Testing

Before assigning Pipelines to alarms, the notification system can inject test alarms to help troubleshoot any problems. A notification will be sent out to any On-Call Rosters in the selected pipeline, so it is recommended to either enable test mode on any Notification blocks, or create a testing-only roster.

To test alarms, follow these simple steps:

1. Go to the Gateway webpage and select the [Config](#) section.
2. From the menu on the left, select [Alarming > Notification](#). The Alarm Notification Profiles page will open.



On this page ...

- [Five Steps to Alarm Notification](#)
 - [Notification User](#)
 - [On-Call Roster](#)
 - [Alarm Notification Profile](#)
 - [Alarm Notification Pipeline](#)
 - [Adding Pipelines to Tags](#)
- [Notification Testing](#)

- Click on **Test Pipelines and Notification Profiles**.

The screenshot shows the Ignition Config interface with the 'Config' tab selected. In the center, the 'Alarm Notification Profiles' page is displayed. It lists two profiles: 'Email Notifications' (Running) and 'Voice' (Registered with VOIP Host). At the bottom of the list, there are two links: 'Create new Alarm Notification Profile...' and 'Test Pipelines and Notification Profiles...'. The second link is highlighted with a red box.

- Select your configured Pipeline, and click the **Submit** button.

The screenshot shows the 'Test Pipelines and Notification Profiles' page. It includes a note: 'Use this page to inject a fake alarm event into a pipeline to test out your pipeline and notification profile settings.' Below this is a form with fields for 'Pipeline' (set to 'project:Core:/pipeline:Basic Pipeline'), 'Display Path' (set to 'TestPage/TEST_ALARM'), and 'Priority' (set to 'Diagnostic'). A large blue 'Submit' button is at the bottom, which is highlighted with a red box.

- Once the system has been tested successfully, the last step is to assign a pipeline to your alarms. This can be done by [editing your Tags in the Designer](#). Go to your **Tag Browser** and double click on your Tag.
- The **Tag Editor** will open. Click on **Alarming**.
- Select an alarm and go to the Notification section, and choose a pipeline (i.e., Escalate) to assign to your alarm. Click **OK**.

The screenshot shows the Ignition Tag Editor for a tag named 'WriteableFloat1 > Alarms'. On the left, under 'Alarms', 'Fault1' is selected. In the main panel, the 'Properties' section is shown for 'Fault1'. Under the 'Notification' section, the 'Ack Pipeline' and 'Active Pipeline' fields are both set to 'Core/Basic Pipeline'. The 'Active Pipeline' field is highlighted with a red box. At the bottom right are 'Commit' and 'Revert' buttons.

Now, when an alarm occurs, an alarm notification will be sent to the users in the On-Call Roster whose schedule is active.

In This Section ...

Notification Contact Info

Ignition provides the capability to send out alarm notifications to a list of people in Email, SMS, and Phone (voice dial-out). There are several steps to get alarm notifications sent out to a list people. The first thing to do is create an [Alarm Notification Profile](#) for [Email](#), [SMS](#) or [Voice](#), whichever method you intend to use. Then, create the users who are going to receive the alarms. Each user must have a name and contact information in order for Ignition to notify them of an impending alarm.

Add a User

1. Go to the **Config** section of the Gateway Webpage.
2. Scroll down to **Security > Users, Role**.

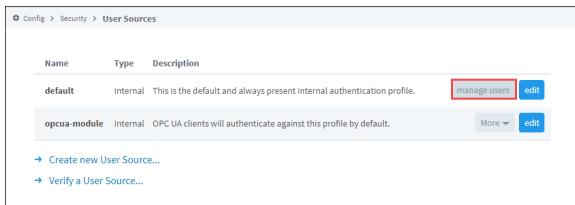


On this page ...

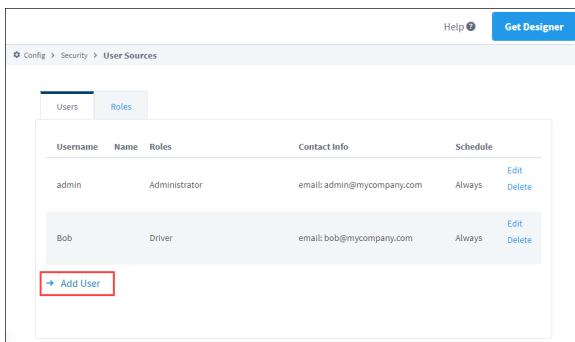
- [Add a User](#)
- [User Sources Settings](#)

The Inductive University logo is displayed, featuring a green laurel wreath around a blue hexagon containing the letters 'IU'. To the right of the logo, the text 'INDUCTIVE UNIVERSITY' is written in a bold, sans-serif font. Below the logo, there is a section titled 'Notification Users' with a blue link labeled 'Watch the Video'.

3. Select **manage users** from the appropriate user source.



4. Look for the blue arrow and select the **Add User** link to add a user.



5. Complete the fields that correspond to the user's information. for example:

The screenshot shows the 'User Properties' section of the 'User Sources' configuration screen. It includes fields for Username (HJames), Password, Re-type password, First Name (Harry), Last Name (James), Roles (Administrator, Operator, Driver), Schedule (Always), and Language (English).

6. In the **Contact Info** section, click **Add Contact info**.
7. Choose the delivery method from the dropdown list, and enter the appropriate Value such as an email address.

The screenshot shows the 'Contact Info' dialog. The 'Type' dropdown is set to 'E-Mail' (highlighted with a red box). The 'Value' field contains 'HJames@company.com'. The 'Save' button is highlighted with a red box.

8. Click **Add User**. The new user will now appear under the Users tab.

User Sources Settings

The following table describes all the settings on the User Source screen.

User Properties	
Setting	Description
Username	User name
Password	Password for this user.
Password	Re-type the password for verification
First Name	First name of the user (Optional).
Last Name	Last name of the user (Optional).
Roles	Select the roles this user will be assigned.
Schedule	Select the schedule for the user. Default is Always.
Language	Select the language for the user. Default is English.
Notes	Any notes about the user. (Optional).
Badge	Badge ID information (Optional).
Contact Info	
Type	Dropdown list of contact information types, such as Email.
Value	Contact information (i.e., email address).
Extended Properties	
Security PIN	Security PIN for the user (Optional).

Related Topics ...

- [User Schedules](#)

On-Call Rosters

The On-Call Roster lets you create user groups to be notified when an alarm occurs. Each group includes a list of users in a specific order. Alarm pipeline's notification blocks must choose an on-call roster which identifies the users to notify for that notification block. Depending on the alarm notification profile used, the users can be notified one at a time (sequential), or all at once (parallel). It is important to remember that when an on-call roster is used for alarm notification, only those users on the roster whose [schedules](#) are active will be notified.



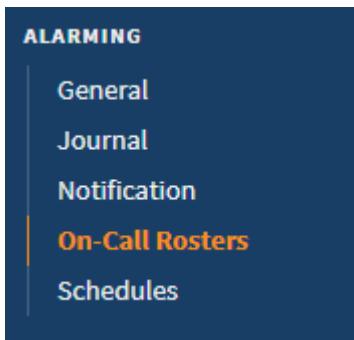
INDUCTIVE
UNIVERSITY

On-Call Rosters

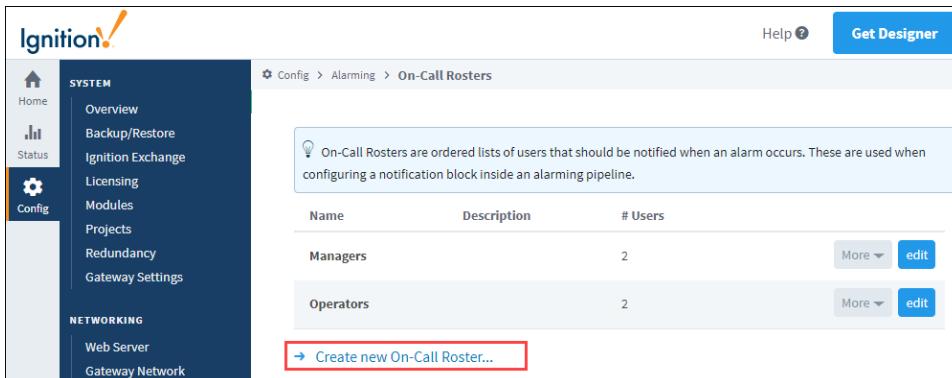
[Watch the Video](#)

Create and Manage an On-Call Roster

1. Go to the **Config** section of the Gateway Webpage.
2. Scroll down to **Alarming > On-Call Rosters**.



3. Click on **Create new On-Call Roster**.



Help ⓘ Get Designer

Config > Alarming > On-Call Rosters

On-Call Rosters are ordered lists of users that should be notified when an alarm occurs. These are used when configuring a notification block inside an alarming pipeline.

Name	Description	# Users	More	Edit
Managers		2	More	edit
Operators		2	More	edit

[Create new On-Call Roster...](#)

4. On the **New On-Call Roster**, enter a **Name**, and click **Create New On-Call Roster**.

The screenshot shows the Ignition Config interface. The left sidebar has 'Config' selected. The main area is titled 'On-Call Rosters'. A 'Properties' section contains 'Name: MyRoster' and 'Description: Test on-call roster.' A blue button at the bottom right says 'Create New On-Call Roster'.

- Now back on the On-Call Rosters page, click on the **More** button, and select **manage** from the dropdown list to the right of the roster name.

The screenshot shows the 'Manage Roster' page for 'MyRoster'. It lists users: Managers (2), MyRoster (0), and Operators (2). The 'Operators' row has a 'manage' button highlighted with a red box. A note at the bottom says 'Create new On-Call Roster...'.

- The **Manage Roster** page is displayed. Here you can add one or all users from a source list, and put them in any order you wish. You can also remove users from the On-Call Roster.

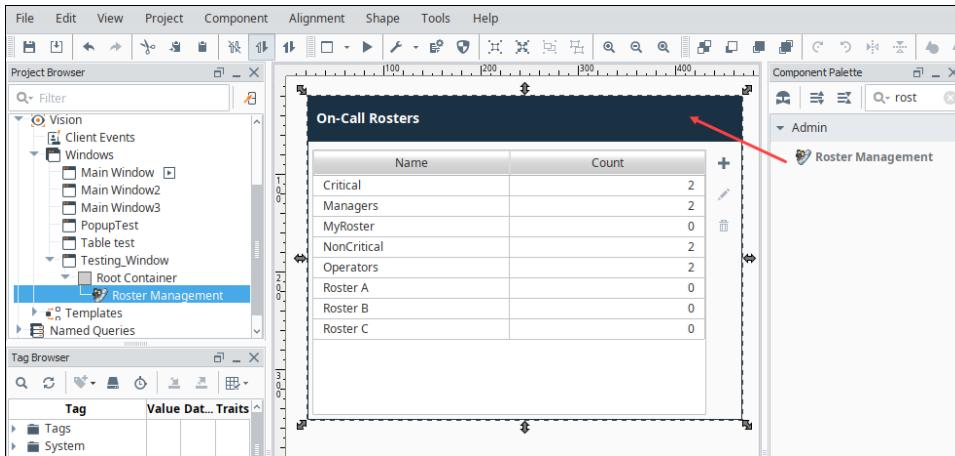
The screenshot shows the 'Manage Roster' page. It has two columns: 'All Users from' (source) and 'On-Call Roster' (target). In the source list, users are listed with arrows to move them: admin, Bob, guest, and Sal. In the target list, Bob is already moved with an arrow. A note at the top says 'Click the right arrow on each user in the user source column (left) to move them to the on-call list (right). Re-order the users on the on-call list using the up and down arrows to control who gets called first.' A 'Save' button is at the bottom.

- Click on **Save**.

Manage the Roster from the Client

Roster management also can take place from within a Vision Client.

- Open your project in the **Designer** and navigate to a window the client has access to. If this is a new window, you will have to add it to your navigation.
- From the Component Palette, drag and drop a **Roster Management** component in your design window.



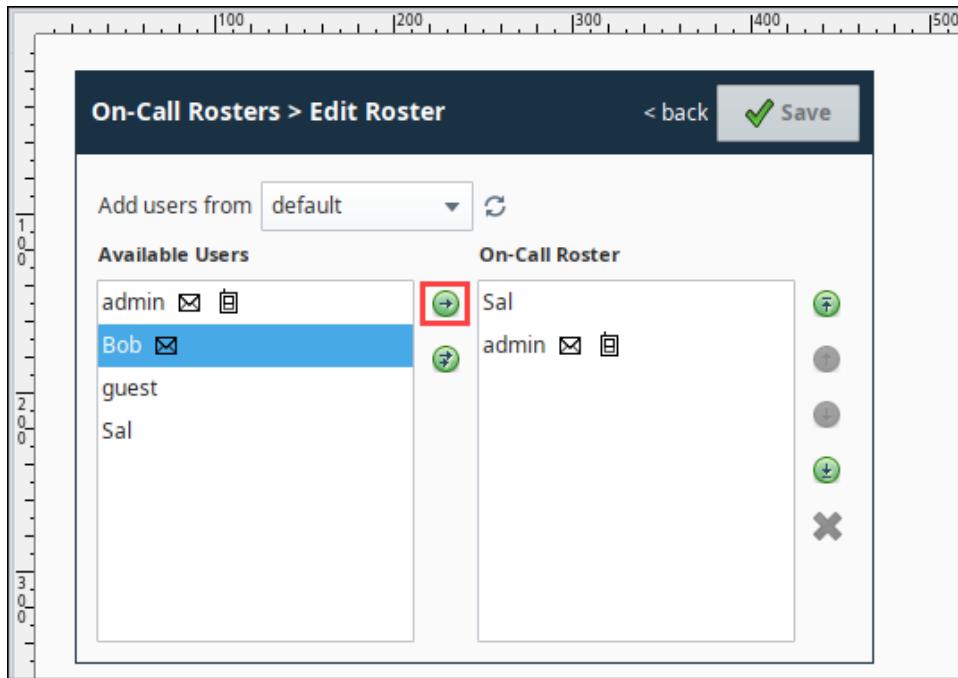
3. If you have more than one user source, go to the Property Editor and enter the one you want to be used by the roster in the **User Source** property.
4. **Save** your project.
5. To interact with the roster, either launch your Vision Client or put your Designer into Preview mode.
6. To edit the roster, double-click any where on the roster or click the **Edit** icon in the upper right.

This screenshot shows the 'On-Call Rosters' table in preview mode. The 'Managers' row is selected, and a red box highlights the edit icon (pencil symbol) in the header of the table.

Name	Count
Critical	2
Managers	2
MyRoster	0
NonCritical	2
Operators	2
Roster A	0
Roster B	0
Roster C	0

7. On the Edit Roster screen, you can add an available user to the On-Call Roster, by selecting the user and clicking the **Add User** icon.

Alternatively you can add all users by clicking the **Add All Users** icon.



8. To delete a user from the roster, select the user and click the **Delete**  icon.

For additional information, see the [Vision - Roster Management](#) component.

Notification Profile Types

Once you have alarms configured in Ignition, you can notify people when alarms occur. (See [Alarming in Vision](#) and [Alarming in Perspective](#).) Ignition provides three Notification Profile Types: Email, SMS or text message, and Voice through a phone call in Ignition. All three of these notification profiles can be configured to be two-way, which can be acknowledged via email, by text message, or by pressing 1 on the phone. Here is a brief description of each Notification Profile type:

- **Email Notification Profile** - Sends an email containing the alarm notification message. The two-way sends an email containing links that allow the user to acknowledge the alarm.
- **SMS Notification Profile** - Sends a text message to users notifying them of an alarm, and a link to acknowledge the alarm.
- **Voice Notification Profile** - Delivers alarm notifications to users via telephone. Acknowledgement is provided as a voice instruction as part of the call.

Details on how to create and configure each of these profiles are found on the following sub-pages.

Module Requirements

Each Notification Profile requires the [Alarm Notification Module](#). This is because Notification Pipelines are managed and defined by the Alarm Notification Module. The [SMS Notification Module](#) and [Voice Notification Module](#) are similar in concept to browser plug-ins in that they extend the functionality of another piece of software. The SMS Notification and Voice Notification are both separate modules, and can be downloaded from the [Ignition Downloads webpage](#). To learn more about installing these modules, refer to [Installing or Upgrading a Module](#).

Additional Requirements

Each Alarm Notification Profile needs a method to deliver the message. These methods are not included with the Alarm Notification Module, and some may require additional hardware to be purchased.

- **Email Notification** - The Gateway will need network access to a SMTP server.
- **SMS Notification** - A cellular modem with an active SIM card. Alternatively, a Twilio account may be used instead of the physical cellular modem.
- **Voice Notification** - Any SIP compatible phone system.

In This Section ...

Email Notification Profile

Alarm notification is the act of sending an email message to a group of people when an alarm becomes active or cleared. In Ignition, this functionality is enabled by having the Alarm Notification module installed which provides alarm pipelines and email notification.

Once you have your Email Notification Profile created in Ignition, you can notify people when alarms occur via Email. If the two-way setting is enabled, operators can also acknowledge the alarm by clicking the link in the body of the email.

For notifications to be sent, Ignition must be able to access an SMTP server that accepts user credential (username and password).

Let's get started setting up an Email Notification Profile.

On this page ...

- [Create an Email Notification Profile](#)
- [Email Notification Profile Settings](#)
 - [Email Settings](#)
 - [Two-Way Settings](#)
 - [POP3 Two-Way Settings](#)
 - [Auditing](#)
 - [Advanced](#)

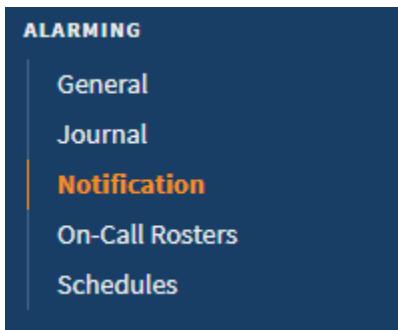


Email Notification Profile

[Watch the Video](#)

Create an Email Notification Profile

1. Go to the **Config** section of the Gateway Webpage.
2. Scroll down and select **Alarming > Notification** from the menu on the left side.

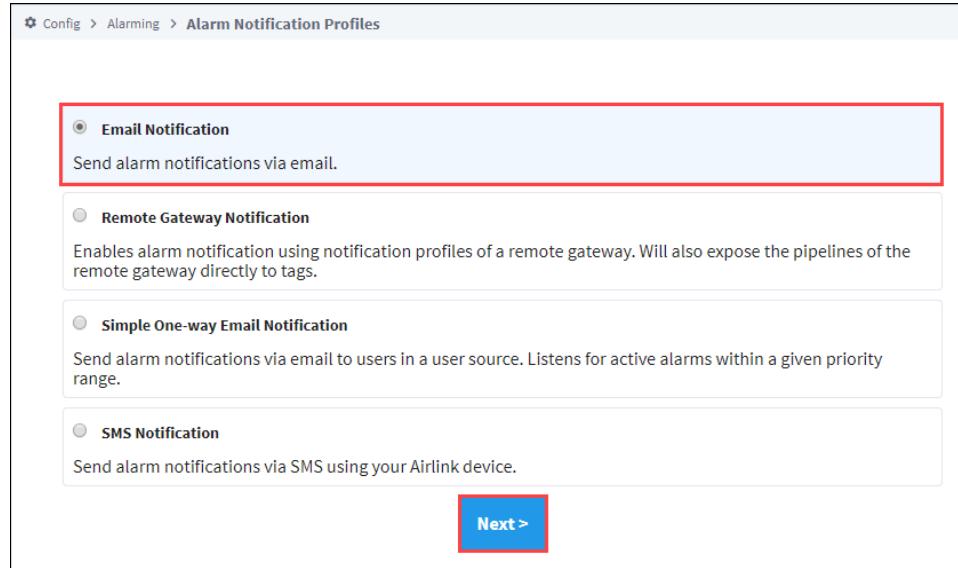


3. Click on the **Create new Alarm Notification Profile** link.

A screenshot of the Ignition Config interface. The left sidebar shows 'SYSTEM' with 'Overview', 'Backup/Restore', 'Ignition Exchange', 'Licensing', 'Modules', 'Projects', 'Redundancy', and 'Gateway Settings'. The 'Config' icon is selected. The main panel shows a table of 'Alarm Notification Profiles' with columns: Name, Description, Enabled, Type, and Status. There are two profiles listed: 'Email Notifications' (Status: Running) and 'Voice' (Status: Registered with VOIP Host). At the bottom of the main panel, there are two links: 'Create new Alarm Notification Profile...' (highlighted with a red box) and 'Test Pipelines and Notification Profiles...'. A 'Get Designer' button is in the top right corner.

4. A list of available alarm notification profiles appear. If any of the profiles are not displayed, it means the required module is missing or faulted. (To verify that the Alarm Notification Module is installed and running, click on **Modules** from the menu bar on the left side). Choose **Email**

Notification, and click **Next**.



5. Configure the settings for the Email Notification Profile:

- Enter a **Name**: Email 1
- Click the **Enabled** box to enable Email notifications.
- Enter the **Hostname**. This is the hostname of the SMTP server that will be responsible for sending the emails.
- Enter the **Port**. The default is **Port 25**, but your mail server may be different.
- Enable **SSL/TLS** if you are using encryption.
- Enter your **Username** and **Password** if they are required.
- Click the **Two-way Enable** box if you want to make your Email Notification Profile two-way so that the operator can acknowledge the email. There is a link to acknowledge the alarm in the email.

Note: You must have a connection back to the Ignition Gateway. So, if you are working from home and not on a VPN, you will either have to get on a VPN or expose your Ignition server on the Internet. You can do this with port forwarding, talk to your IT department about it.

- **Pop3 Two-way Settings:** Leave the defaults for now.
- **Auditing:** You have the option of attaching emails to an Audit profile so that Ignition can store all the events associated with the Email Notification Profile. (i.e., who is receiving alarm and who is acknowledging an alarm).

Config > Alarming > Alarm Notification Profiles

Main	
Name	Email 1
Description	
Enabled	<input checked="" type="checkbox"/> (default: true)

Email Settings	
Use SMTP Profile?	<input type="checkbox"/> If selected, this notification profile will use one of the gateway defined SMTP profiles. Otherwise, it will use the settings defined here. (default: false)
SMTP Profile	- none - If Use SMTP Profile is selected, alarm notifications will be emailed using this profile.
Hostname	mail@inductiveautomation.com Hostname of the SMTP server to send email through. Only used when Use SMTP Profile is false.
Port	25 Port SMTP service is running on. Only used when Use SMTP Profile is false. (default: 25)
Enable SSL/TLS	<input type="checkbox"/> Connect using SSL/TLS. Only used when Use SMTP Profile is false. (default: false)
Username	user@inductiveautomation.com Only used when Use SMTP Profile is false.
Password	*****
Re-type Password	***** Re-type password for verification.

Two-way Settings	
Two-way Enabled	<input checked="" type="checkbox"/> (default: false)
Gateway	localhost 8088 Address and port this gateway is reachable at. Will be used in notification emails. Example: 1.2.3.4:8088
Send HTTPS Link	<input type="checkbox"/> (default: false)

Auditing	
Audit Profile	- none - If an audit profile is selected, events such as emails and acknowledgements will be stored to the audit system. Note that alarm acknowledgements are also stored to the alarm journal.

Show advanced properties

Create New Alarm Notification Profile

6. Click **Create New Alarm Notification Profile**.
7. Once you create your Email Notification Profile, it will appear in the Alarm Notification Profiles list, and it will be running. You can edit the profile at any time to change any of the settings, and create as many Email Notification Profiles as you need.

Config > Alarming > Alarm Notification Profiles					
Successfully created new Alarm Notification Profile "Email 1"					
Name	Description	Enabled	Type	Status	
Email		true	Email Notification	Running	<button>delete</button> <button>edit</button>
Email 1		true	Email Notification	Running	<button>delete</button> <button>edit</button>
SMS		true	SMS Notification	Running	<button>delete</button> <button>edit</button>

Note:

Once your Notification profile is created, you can use it in an [Alarm Pipeline](#).

Email Notification Profile Settings

Email Settings

These settings specify the SMTP server that should deliver the emails. Multiple Ignition Gateways may be configured to use the same SMTP server.

Property Name	Description
Use SMTP Profile?	If selected, this notification profile will use one of the Gateway defined SMTP profiles. Otherwise, it will use the settings defined here. The setting should NOT be enabled unless a SMTP Profile has been configured elsewhere on the Gateway. Enabling this property causes the profile to ignore the Hostname and Port properties listed below
SMTP Profile	If the Use SMTP Profile property is enabled, alarm notifications will be emailed using this profile.
Hostname	Hostname of the SMTP server to send email through. Only used when Use SMTP Profile is false.
Port	Port SMTP service is running on. Only used when Use SMTP Profile is false.
Enable SSL/TLS	Connect using SSL/TLS. Only used when Use SMTP Profile is false.
Username	The username Ignition should use when communicating with the SMTP server. This is only required if the SMTP server expects authentication. Only used when Use SMTP Profile is false.
Password	The password Ignition will use to authenticate against the SMTP server. This is only required if the SMTP server expects authentication.

Two-Way Settings

These settings allow users to acknowledge alarms directly from the Email Notification. When enabled, a link is included in the email that will redirect to a page on the Gateway.

Property Name	Description
Two-way Enabled	Enables remote alarm acknowledgement. If disabled, users will not be able to acknowledge alarms from the email.
Gateway	Network address and port the Gateway is reachable at. Will be used in notification emails.
Send HTTPS Link	Specifies if the link should use a HTTPS link for SSL.

POP3 Two-Way Settings

These settings allows Notifications to be acknowledged from when retrieved from a POP3 mailserver. Multiple Ignition Gateways may be configured to use the same POP3 server.

Property Name	Description
Pop3 Two-way Enabled	Enable two-way notification via POP3.
Enable SSL/TLS	Enables SSL/TLS between the Gateway and the mailserver.
Delete On Acknowledge	If enabled, will delete messages from the inbox after the acknowledging an alarm.
Hostname	The hostname of the mailserver.
Port	The port the mailserver is running on.
Username	The username the Gateway will use when authenticating against the mailserver.
Password	The password the Gateway will use when authenticating against the mailserver.
Custom Message	Allows an opportunity to provide an additional message to the notification. This custom message is in addition to the Custom Messages configured on the Alarm and the Notification Blocks in the Alarm Pipeline.
Polling Interval	The interval (in milliseconds) that the Gateway will check for new notifications.

Auditing

Property Name	Description
Audit Profile	If an audit profile is selected, events such as emails and acknowledgements will be stored to the audit system. Note that alarm acknowledgements are also stored to the alarm journal.

Advanced

Property Name	Description
SMTP Timeout	Timeout (in milliseconds) to use when connecting to, reading from, and writing to the SMTP server.
Debug Mode Enabled	Enable email session debugging. Information is printed to standard output (wrapper.log).
STARTTLS Enabled	Enable use of the STARTTLS command, allowing the connection to be upgraded to an SSL or TLS connection if supported by the server. This is not necessary for connections that are already SSL/TLS. Ignored when Use SMTP Profile is checked.

Related Topics ...

- [SMS Notification Profile](#)
- [Alarm Notification Pipelines](#)

In This Section ...

Email Alarm Acknowledgement

Email Alarm Acknowledgement allows operators to acknowledge alarms via a link that is included directly in the email alarm notification. The email alarm acknowledgement process a simple two-step process:

- The operator receives an email in their inbox alerting them to an alarm(s).
- The link in the email directs the operator to a landing page describing the alarm(s) where the operator can add notes and acknowledge the alarm(s).

Once you have an [Email Notification Profile](#) created in Ignition, operators can be notified when alarms occur and acknowledge alarms via email. In order to acknowledge alarms in an email, the **Two-way Enable** option must be set to 'true' in the [Email Notification Profile](#).

On this page ...

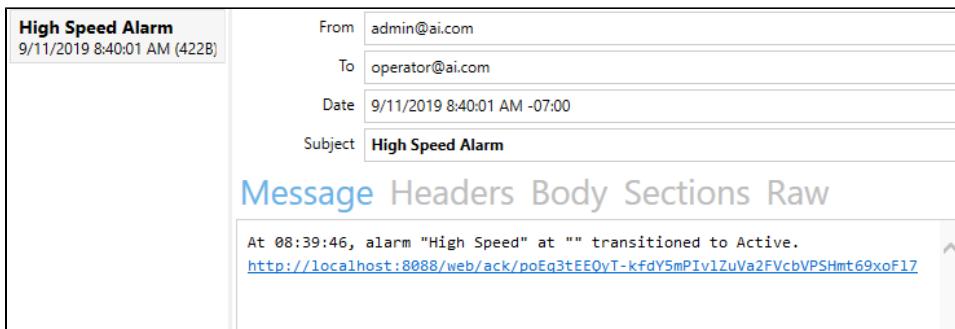
- [Using Email Alarm Acknowledgement](#)

Note: This page assumes an [Email Notification Profile](#), an [On-Call Roster](#) containing users with valid email addresses who will be receiving and acknowledging alarms, and an [Alarm Notification Pipeline](#) are configured, including assigning the [alarm to a pipeline](#). You can choose to send one email notification per alarm, or multiple alarms in one email notification using [Notification Consolidation](#).

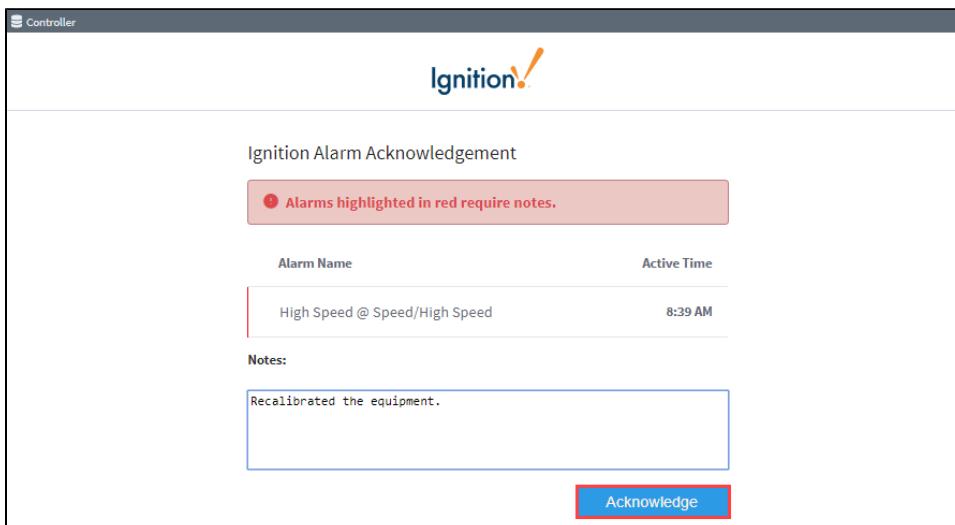
Using Email Alarm Acknowledgement

Acknowledging an alarm in an email is super simple!

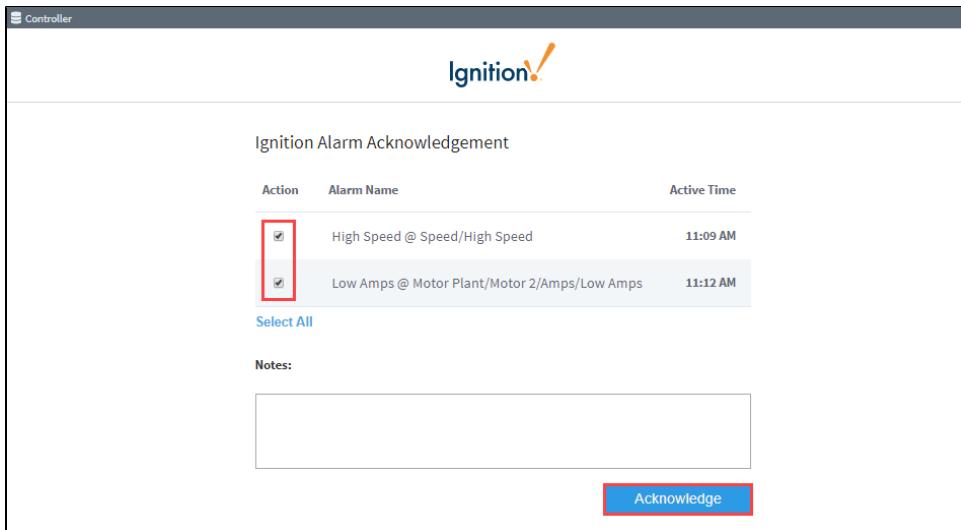
1. Open the the alarm notification email message, and click the link.



2. A new window opens describing the alarm. This example shows one email for one alarm.
3. If notes are required as in the following example, enter notes about the alarm.
4. Click the **Acknowledge** button.



5. If [Notification Consolidation](#) is configured, you can limit the number of email notifications that will be sent, thus, consolidating multiple alarms into one email for operators to acknowledge.



Note: If you are not receiving any email alarm notifications, check **Status > Logs** for any error messages. If you are getting errors saying emails are "unreachable," then check your [Email Notification Profile](#).

Related Topics ...

- [Alarm Notification](#)
- [Pipeline - Notification Block Consolidation](#)
- [Alarm Notification Pipelines](#)

Simple One-Way Email Notification Profile

The Simple One-Way Email Notification profile is another type of Email Notification profile that can be setup to notify people when alarms occur. This one-way email notification profile is just as the name suggests, one way. It listens for active alarms within a given priority range, and when the alarm criteria is met, it sends alarm notifications via email to a specified [On-Call Roster](#).

Although it's similar to the Email Notification Profile, it does not utilize [Alarm Pipelines](#) and does not include a link in the email message to allow recipients to acknowledge the alarm.

It is important to be aware that there is no way to filter which alarms get sent using this notification profile other than the priority, so if alarms are sent to the same roster through an alarm pipeline, users may see duplicate alarms.

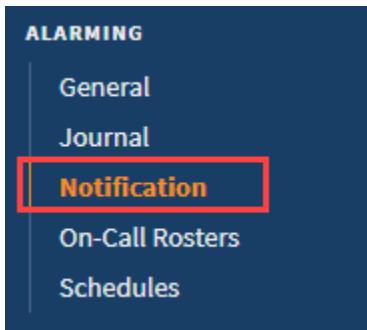
Note: The Simple One-way Email Notification requires the [Alarm Notification Module](#) to be installed.

On this page ...

- [Create a Simple One-Way Email Notification Profile](#)
- [Simple One-Way Email Notification Profile Settings](#)

Create a Simple One-Way Email Notification Profile

1. Go to the **Config** section of the Gateway Webpage.
2. Under **Alarming**, select **Notification** from the menu on the left side.



3. A list of Alarm Notification Profiles will appear. Click on the **Create new Alarm Notification Profile** link.

A screenshot of the Ignition Gateway Webpage under the 'Config' section. On the left, there is a sidebar with 'SYSTEM' and 'NETWORKING' sections. In the main area, the path 'Config > Alarming > Alarm Notification Profiles' is shown. A table lists three profiles: 'Email Notifications' (Email Notification, Running), 'SMS' (SMS Notification, Unknown), and 'Voice' (VOIP Voice Notification, Registered with VOIP Host). At the bottom of the list, there are two links: '**Create new Alarm Notification Profile...**' (highlighted with a red box) and '**Test Pipelines and Notification Profiles...**'.

4. A list of all the alarm notification profile types will appear.

Note: If any of the profiles are not shown, it means the required module is missing or faulted. (To verify the Alarm Notification Module is installed and running, go to the Config tab of the Gateway Webpage and click on **System>Modules**.)

5. Choose **Simple One-Way Email Notification**, and click **Next**.
6. Configure the profile settings for the Simple One-Way Email Notification Profile, and then click **Create New Alarm Notification Profile**.
 - **Name:** One-Way Email
 - **SMTP:** SMTP (Please see the [Email Settings](#) page for more information on creating an SMTP profile.)
 - **On-call Roster:** Managers (For more information, please see the [On-Call Rosters](#) page.)
 - **Minimum Priority:** High
 - **Maximum Priority:** Critical

This notification profile example below is configured to send email notifications to the users in the **Managers** On-Call Roster when the alarm criteria is **High** or **Critical**.

Main

Name	OneWay Email
Description	
Enabled	<input checked="" type="checkbox"/> (default: true)

Email Settings

SMTP Profile	SMTP
On-Call Roster	Managers

Alarm Criteria

Minimum Priority	High
Maximum Priority	Critical

Create New Alarm Notification Profile

- Once created, the Simple One-way Email Notification Profile appears in your Alarm Notification Profiles list, and it will be running. You can edit the profile to change the On-Call Roster or alarm criteria at any time. You can create as many Simple One-way Email Notification Profiles as you need.

Name	Description	Enabled	Type	Status	Actions
Email		true	Email Notification	Running	<button>delete</button> <button>edit</button>
Email 1		true	Email Notification	Running	<button>delete</button> <button>edit</button>
OneWay Email		true	Simple One-way Email Notification	Running	<button>delete</button> <button>edit</button>
SMS		true	SMS Notification	Running	<button>delete</button> <button>edit</button>

Simple One-Way Email Notification Profile Settings

Property Name	Description
Main	
Name	Name of the profile.
Description	Brief description of the profile. Optional.

Enabled	Whether the Simple One-Way Email Notification is enabled.
Email Settings	
SMTP Profile	The SMTP Profile that will be used to distribute the notifications.
On-Call Roster	The Roster of users that will be notified when an alarm meets the criteria set in this profile.
Alarm Criteria	
Minimum Priority	For a notification to be sent from this Notification Profile, its priority must be equal to or higher than the priority specified on this property. (Options are Diagnostic, Low, Medium, High, or Critical.)
Maximum Priority	For a notification to be sent from this Notification Profile, its priority must be equal to or lower than the priority specified on this property. (Options are Diagnostic, Low, Medium, High, or Critical.)

Related Topics ...

- [Notification Profile Types](#)
- [On-Call Rosters](#)
- [Alarming](#)

SMS Notification Profile

SMS Notification Module

Short Message Service (SMS) is the protocol that is used by text messaging systems. The [SMS Notification Module](#) allows you to deliver SMS alarm notifications via a cellular modem configured with a SIM card belonging to an active cellular account. If enabled, recipients of these messages can reply with a special code in order to acknowledge the alarm.

Note: The [SMS Notification Module](#) is dependent on the [Alarm Notification module](#). You must have both modules installed to send alarms.

Device Configuration

Inductive Automation officially supports five Airlink Devices: the RV50/RV50X, RV55/RV55X, the Raven XE, the LS300, and the LX40. Most devices from most manufacturers can be used, but some configuration settings might be different. You can do the basic configuration for the Airlink modem by importing a template settings file provided by Inductive Automation. To learn more, refer to the Knowledge Base article by clicking the link on the right side of this page.

For more information about The AirLink® LS300, see: <http://www.sierrawireless.com/products-and-solutions/gateway-solutions/ls300/>

For more information about The AirLink® RV50, see: <https://www.sierrawireless.com/products-and-solutions/routers-gateways/rv50/>

For more information about The AirLink® RV55, see: <https://www.sierrawireless.com/products-and-solutions/routers-gateways/rv55/>

Multiple Systems with One Modem

You can use one SMS modem with multiple Ignition systems for one-way (outgoing) messaging. If you want to use two-way messaging (incoming and outgoing), then only one Ignition system can receive those incoming responses.

On this page ...

- [SMS Notification Module](#)
 - [Device Configuration](#)
 - [Multiple Systems with One Modem](#)
- [Create an SMS Notification Profile](#)
- [SMS Notification Profile Settings](#)



AirLink LS300 Setup Guide

[Link to Knowledge Base Article](#)



AirLink RV50 Setup Guide

[Link to Knowledge Base Article](#)

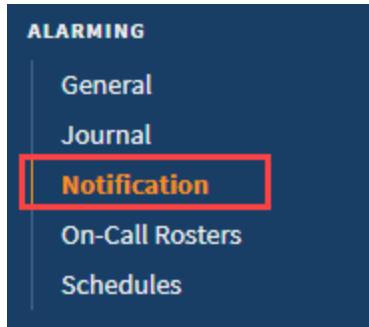
Create an SMS Notification Profile

1. Go to the [Config](#) section of the [Gateway Webpage](#)
2. Scroll down to [Alarming > Notification](#) from the menu on the left.



SMS Notification Profile

[Watch the Video](#)



3. On the Alarm Notification Profiles page, click on **Create new Alarm Notification Profile**.
4. Select the **SMS Notification** profile type and click **Next**.

The screenshot shows a configuration interface for creating a new alarm notification profile. The top navigation bar includes 'Config', 'Alarming', and 'Alarm Notification Profiles'. Below this, there are five options for notification methods:

- Email Notification**: Send alarm notifications via email.
- Remote Gateway Notification**: Enables alarm notification using notification profiles of a remote gateway. Will also expose the pipelines of the remote gateway directly to tags.
- Simple One-way Email Notification**: Send alarm notifications via email to users in a user source. Listens for active alarms within a given priority range.
- SMS Notification**: Send alarm notifications via SMS using your Airlink device. This option is highlighted with a red border.
- Twilio SMS Notification**: Send SMS alarm notifications via a Twilio account.
- VOIP Voice Notification**: Telephone notification using VOIP, compatible with most SIP based telephony systems.

A blue 'Next >' button is located at the bottom right of the form.

5. Set the following settings concerning your device, and then click on **Create New Alarm Notification Profile**.

- **Name:** SMS
- **Description:** optional
- **Enabled box:** Click to Enable SMS notifications.
- **Airlink Host Address:** The IP address of the airlink modem.
- **Send Port:** The default port is 17341, which is configured in the device.
- **Receive Port:** The default port is 17342, which is the port used by Ignition when two-way messaging is enabled. The port must not already be used by the host system, and must not be blocked by a firewall.
- **Two-way Enabled:** If enabled, the message recipients will receive a text message notifying them of an alarm, and a link to acknowledge the alarm. This is communicated to Ignition via UDP data sent from the modem. Therefore, the Airlink modem must be configured with the IP address of the system.
- **Numeric Only Ack Code** - If this is true, the ack code sent with the SMS message will be a numeric code instead of an alphanumeric code.
- **Audit Profile** - If an audit profile is selected, events such as SMS messages and acknowledgements will be stored to the audit system.
- Once all the settings are entered, click the **Create New Alarm Notification Profile** button.

Once you have the SMS Notification profile set up, you can add phone numbers to your contacts using the **SMS** Contact type. SMS notifications can also be sent with the [Twilio Module](#).

SMS Notification Profile Settings

The following table describes the settings on the Alarm Notification Profiles page for a Twilio SMS Notification profile.

Setting	Description
Main	
Name	Name for this <u>alarm</u> notification profile.
Description	Description of the profile.
Enabled	Whether the profile is enabled or disabled.
SMS Settings	
Airlink Host address	Enter the Airlink host address for this profile.
Send Port	The send port. (Default is 17,341.)
Receive Port	The receive port. (Default is 17,342.)
Two-Way Enabled	Whether or not two-way SMS is enabled.(Default is false.)
Numeric Only Ack Code	Whether to accept only numeric ack codes. (Default is false.)
Auditing	
Auditing Profile	Select an audit profile from the dropdown list.

Related Topics ...

- [Twilio Module](#)
- [Voice Notification Profile](#)
- [Alarm Notification Pipelines](#)

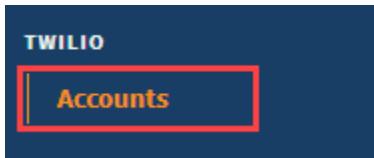
Twilio SMS Notification

SMS notifications can also be sent with the [Twilio Alarm Notification](#) module. This does not require a cellular modem, but the Gateway must have Internet access, and a Twilio account with SMS capabilities must be created (www.twilio.com). Once created, the account must be defined in Ignition.

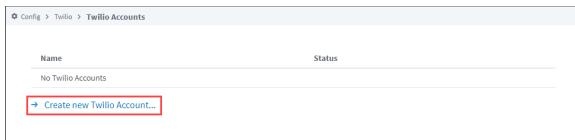
Note: The Twilio module is dependent on the [Alarm Notification](#) module. You must have both modules installed to send alarms.

Define a Twilio Account

1. Navigate to the [Config](#) section of the Gateway Webpage.
2. Scroll down and choose **Twilio > Accounts**.



3. On the Twilio Accounts page, click the [Create new Twilio Account](#) link.



4. Provide a **Name** for the account as well as the **Twilio Account Sid**, and **Twilio Auth Token**. Both the Account SID and Auth Token can be retrieved from the Account Settings menu in Twilio.
5. If desired, provide the **Public Hostname**. This is the hostname or IP address that inbound requests will be forwarded. If the users should be able to acknowledge alarms by responding to the SMS messages, then this property should list a publicly reachable hostname or IP address.
6. Click the [Create New Twilio Account](#) button.

Once the account has been defined, a Twilio SMS notification profile must also be created.

On this page ...

- [Define a Twilio Account](#)
- [Twillio Accounts Settings](#)
- [Create a Twilio SMS Notification Profile](#)
 - [Twilio SMS Notification Profile Settings](#)
- [Receiving Alarm Acknowledgement via Twilio](#)



AirLink LS300 Setup Guide

[Link to Knowledge Base Article](#)



AirLink RV50 Setup Guide

[Link to Knowledge Base Article](#)

Twillio Accounts Settings

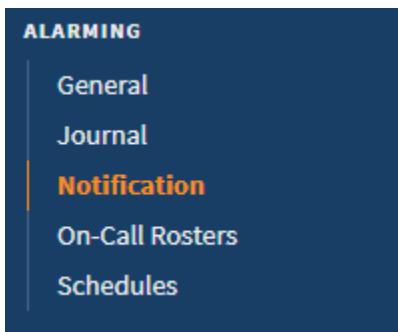
The following table describes the settings on the Twilio Accounts page.

Setting	Description
Twillio	
Name	Name for the Twilio Account.
Twilio Account Sid	Twilio account Sid: retrieve from the Account Settings menu in Twilio. (Required.)
Twilio Auth Token	Twilio auth token: retrieve from the Account Settings menu in Twilio. (Required.)

Inbound	
Public Hostname	The publicly reachable hostname or IP address that is configured to forward inbound requests to this Ignition gateway. This must be configured in order to receive inbound SMS. Leave blank if inbound SMS are not needed.
Public Port	The port on the Public Hostname to make the connection to. Default is 80.
Backup Public Hostname	The publicly reachable hostname or IP address that is used when the backup of a redundant pair becomes active.
Backup Public Port	The port on the Public Hostname to make the connection to that is used when the backup of a redundant pair becomes active. Default is 80.
HTTPS Enabled	True if the callback URL should use HTTPS instead of HTTP. Default is false.
Inbound Validation Enabled	Cryptographically validate that inbound requests over HTTPS are from Twilio. Default is false.

Create a Twilio SMS Notification Profile

1. Go to the **Config** section of the Gateway Webpage.
2. Scroll down and select **Alarming > Notification** from the menu on the left side.



3. Click on the **Create new Alarm Notification Profile** link.
4. Select the **Twilio SMS Notification** profile type, and click **Next**.

The screenshot shows a list of notification types:

- Email Notification
- Remote Gateway Notification
- Simple One-way Email Notification
- SMS Notification
- Twilio SMS Notification** (selected)
- VOIP Voice Notification

A blue 'Next >' button is at the bottom right.

5. Provide a name for the profile, and select a **Twilio Account** from the dropdown.
6. Set the other profile settings as desired. (See table below for a description of all settings.)
7. Click **Create New Alarm Notification Profile**.

Twilio SMS Notification Profile Settings

The following table describes the settings on the Alarm Notification Profiles page for a Twilio SMS Notification profile.

Setting	Description
Name	Name for this alarm notification profile.
Description	Description of the profile.
Enabled	Whether the profile is enabled or disabled.
Twilio Account	Select a Twilio account from the dropdown list.
Acknowledgement Allowed	Enables users to respond to notifications directly from the SMS message; also known as Two-Way notification . If enabled, the Public Hostname property in the Twilio Account must be configured.
Auditing Profile	Select an audit profile from the dropdown list.

Note: Once your Notification profile is created, you can use it in an [Alarm Pipeline](#).

Receiving Alarm Acknowledgement via Twilio

If you're using 2-way alarming, when a user acknowledges an alarm, Twilio's servers send the acknowledgement to your Ignition Gateway. This means your Ignition Gateway needs to be accessible from Twilio's servers. If you're running Ignition on the cloud, this is likely already the case. If Ignition is on-premise, you may need to explore one of the following options:

1. Make use of port forwarding
2. Put an Ignition alarming server in the cloud with a Gateway Network connection to your on-premise server.

Because there are often security implications of port forwarding, and a cost associated with adding an Ignition server in the cloud, often the SMS Notification module is chosen over the Twilio module for users running Ignition on-premise.

Related Topics ...

- [Voice Notification Profile](#)
- [Alarm Notification Pipelines](#)

Voice Notification Profile

Note: For the **Voice Notification Module** to work, a compatible [Voice Module](#) must also be installed.

The Voice Notification Module adds the ability to deliver alarm notifications to users via telephone, using any SIP compatible phone system (common for VOIP systems). Messages are constructed in text and are delivered through a high quality text-to-speech engine. The engine supports multiple voices and languages.

Core Features

Deliver voice calls through any SIP compatible phone system. No dedicated hardware required. Messages generated by high-quality text to speech, and not a canned set of prerecorded files.

- Supports multiple languages concurrently, based on user preference.
- Allows users to acknowledge events.
- Supports requiring a personal identification number for additional security.
- Ties into the audit log to audit call events, successful message delivery, and user acknowledgements.
- Supports message consolidation.

Voice Notification Configuration

About SIP and VOIP

The Session Initiation Protocol, is a highly popular specification for implementing Voice Over IP (VOIP) based phone systems. The protocol itself, as the name suggests, is responsible for initiating communication sessions, and then other protocols such as SDP and RTP are used to actually transfer voice data. Ignition has these protocols built in, SIP is a peer-to-peer protocol, where one side talks directly to the other. However, it is possible to have Gateways that repeat and route data between the two parties. Sometimes phone calls on VOIP networks stay purely in software, but often a Gateway will transition the call to a traditional phone line.

By leveraging SIP, Ignition can call physical phones, soft phones, be worked into more complex PBX schemes, while avoiding the high cost of traditional, dedicated voice cards. To get started, though, you'll need some sort of SIP Gateway. Asterisk is a popular, open source, system that is used by thousands of companies worldwide. If you simply want to connect to a phone line, there are several low cost devices that runs Asterisk.

Gateway Configuration

To get started with voice notification, add a new profile by going to **Config-> Alarming -> Notification** in the Ignition Gateway. You are only required to specify the host address of the SIP gateway, though depending on the Gateway, you may be required to enter a username and password.

There are additional settings that dictate how calls are managed, such as timeouts for answering, and the maximum amount of time that a call can take. Additionally, you can choose to link the notification profile to an audit profile in order to record important call lifecycle events.

After saving the profile, you should see the status update from "Unknown" to "Registered", indicating that the Gateway has successfully registered with the SIP Gateway. If you see an error, verify that the settings are correct, and that the username and password are correct. The system log console can also be useful in determining what is wrong.

Note: If you receive errors indicating that an "invalid parameter" has been used, try setting the local and public bind interface settings under the advanced options. These should be set to the IP address of the network card that is being used to communicate with the SIP Gateway. On some systems, especially Linux hosts, the default empty values result in this error.

Using Skype

If you don't have an existing VOIP system in place, using a hosted service is the quickest way to get going with the voice notification module. Skype offers SIP based service through their Skype Connect product. To get started, you must have a business account, which provides you access to the Skype Manager. From there, you can create a new Skype Connect channel. For more information, visit <http://manager.skype.com>.

Once you have created a Skype Connect channel, configuration in the Gateway is similar to that of any other PBX system. The host will be "<sip.skype.com>", and the user name and password will be those provided by skype during the registration process. Note: while skype allows to you specify how many "channels" may be used, Ignition currently only supports one channel at a time.

On this page ...

- **Core Features**
- **Voice Notification Configuration**
 - About SIP and VOIP
 - Gateway Configuration
 - Configuring Messages
 - The Call Lifecycle
 - Pipeline Configuration
- **Create a Voice Notification Profile**
- **VOIP Voice Notification Profile Settings**



Voice Notification Profile

[Watch the Video](#)

Configuring Messages

The message played to the user during the phone calls is defined in the call script. The script dictates the overall structure of the call, defining the phrases and options, and the possible responses. The messages for each alarm are built off of a message template that can reference properties in the alarm. The script can be edited by selecting "manage scripts" next to the voice notification profile. Note: Although the link appears next to a particular profile, the scripts are shared across all voice notification profiles.

The role of each phrase in the script is explained on the settings page. Some parts of the script, such as the phrase requesting the user's PIN number, will only be used if certain settings are configured on the notification block in the pipeline (in this case, the setting to require a PIN). As previously mentioned, the alarm message (for both active and clear) can reference any property of the alarm. The default message looks like this:

At {eventTime|hh:mm:ss}, alarm named {name} became {eventState} with a value of {eventValue}

In this case, the message refers to the alarm name, the eventState and eventValue (note: eventState is different than state. Event state is just the transition that triggered the alarm, such as "active", whereas "state" is the full current state, such as "active, unacknowledged"), and the eventTime. Notice that the event time is formatted to only use the time, and not include the current date.

Scripts can be created for different languages. When the system attempts to deliver a notification, it will look to see if the target user has a preferred language. If so, and the language has both a script defined, and a compatible voice installed, the user will be notified in that language.



Alarms also allow you to define a custom message, relevant to that particular state. If a custom message is defined, it will be used instead of the default message in the script.

The Call Lifecycle

A call is initiated when an alarm event enters the notification profile block in a pipeline. When this occurs, the target users are collected, based on the defined call roster, and the current schedule. Only users who have phone contact details defined will be selected for phone notification.

The voice system can only call one number at a time, and so it takes the first contact off of the queue and initiates the call. The user is given up to the "answer timeout" to answer. After picking up, the user will be asked to enter their pin number, or press any number to continue. The call is not considered "answered" until this action occurs, so the message will be repeated until the answer timeout expires. By acting in this way, the system is able to confirm that a human has actually answered, and the call will then be audited as successful.

Once past the initial challenge, the user will then hear the alarm messages. After each message, they will be asked to either acknowledge, ignore, or repeat the message. Selecting "acknowledge" will cause the alarm to be acknowledged in the alarming system, likely causing it to drop out of the alarm pipeline (dependant on pipeline settings). Ignoring the alarm indicates that the user has heard the message, but cannot or does not want to acknowledge the alarm.

Once the call has completed, the notification system will check the alarm events against the pipeline fallout conditions, and move on to the next call. The system will cycle through all alarms and all contacts (and all phone numbers for each contact) until everyone has been notified, or the pipeline settings have caused all events to drop out.

Pipeline Configuration

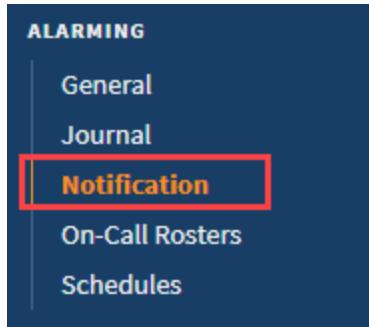
The voice notification system is accessed through the Notification Block in the Designer's pipeline configuration, like all other notification methods. As with other methods, you select a call roster, and can optionally turn on consolidation. There are only two options which can be set on the notification block:

Require PIN: If selected, the user will be required to enter their personal identification number in order to hear the message. The user's PIN number is specified on their profile in the user management system. If false, or if the user does not have a PIN specified, the user will only be required to press any key to continue.

Allow Acknowledgement: If true, the user will be given the opportunity to acknowledge the alarm. If false, they will only be able to hear the alarm and continue.

Create a Voice Notification Profile

1. Go to the **Config** section of the Gateway.
2. Select **Alarming > Notification** from the menu on the left side.



3. Click on the **Create new Alarm Notification Profile** link.

A screenshot of a 'Config > Alarming > Alarm Notification Profiles' page. It displays a table with two rows of existing notification profiles: 'Email Notifications' (Status: Running) and 'SMS' (Status: Unknown). Below the table are two buttons: a red-bordered 'Create new Alarm Notification Profile...' button and a blue 'Test Pipelines and Notification Profiles...' button.

4. Add Alarm Notification Profile Step 1 window will appear. Select the **VOIP Voice Notification** radio button and click **Next**.

A screenshot of the 'Add Alarm Notification Profile Step 1' window. It lists several notification types as radio buttons: 'Email Notification' (Send alarm notifications via email), 'Remote Gateway Notification' (Enables alarm notification using notification profiles of a remote gateway. Will also expose the pipelines of the remote gateway directly to tags), 'Simple One-way Email Notification' (Send alarm notifications via email to users in a user source. Listens for active alarms within a given priority range), 'SMS Notification' (Send alarm notifications via SMS using your Airlink device), 'Twilio SMS Notification' (Send SMS alarm notifications via a Twilio account), and 'VOIP Voice Notification' (Telephone notification using VOIP, compatible with most SIP based telephony systems). The 'VOIP Voice Notification' option is highlighted with a red rectangular border. At the bottom right is a blue 'Next >' button.

5. On the New Alarm Notification Profile page, configure your settings. Enter the Gateway Address or domain name of your SIP Gateway. You may be required to enter a username and password. There are also some additional call settings that may need to be setup depending on how you want to manage your calls, and based on your preference. Refer to the table below for a description of all settings.

6. When completed, click **Create the New Alarm Notification Profile**. Your new Voice Notification Profile will be displayed with a status of **Registered with VOIP Host**.

The screenshot shows a table of alarm notification profiles. The columns are Name, Description, Enabled, Type, and Status. The rows are:

- Email Notifications: true, Email Notification, Running, delete, edit
- SMS: true, SMS Notification, Unknown, delete, edit
- Voice: true, VOIP Voice Notification, Registered with VOIP Host, More ▾, edit

Below the table are two links: "Create new Alarm Notification Profile..." and "Test Pipelines and Notification Profiles...".

VOIP Voice Notification Profile Settings

The following table describes the settings on the Alarm Notification Profiles page for a VOIP Voice_Notification profile.

Setting	Description
Main	
Name	Name for this <u>alarm</u> notification profile.
Description	Description of the profile.
Enabled	Whether the profile is enabled or disabled.
VOIP Gateway Settings	
Gateway address	The ip address or domain name of your SIP gateway.
Username/Account	User or account name for this profile.
Password	Enter a password for the account.
Password	Re-enter password for verification.
Outbound Proxy	The proxy to use, if any.
Call Settings	
Max Call Duration (minutes)	The length of time, in minutes, that any particular call is allowed to take. The call will be terminated if it lasts longer than this limit. (Default is 5.)
Answer Timeout (seconds)	How long, in seconds, to wait for an answer before moving on to the next contact. (Default is 60.)
Voice Rate	Adjusts the desired voice rate. (Default is Normal.)
Auditing	
Auditing	If an audit profile is selected, events such as phone calls and acknowledgements will be stored to the audit system. Note that alarm acknowledgements are also stored to the alarm journal
Advanced Settings	
Authorization Id	Some services require a separate authorization name, in addition to the user/account name. If left blank, the username specified above will be used.
SIP Port	SIP Port. (Default is 5,060.)
RTP Port	RTP Port. (Default is 8,000.)
Local Bind Address	Local bind address for the profile.
Public Bind Address	Public bind address for the profile.

Media Debug Enabled

If true, call audio will be recorded to the Ignition temp directory. (Default is false.)

Note: Once your Notification profile is created, you can use it in an [Alarm Pipeline](#).

Related Topics ...

- [Notification Contact Info](#)
- [Alarm Notification Pipelines](#)

In This Section ...

Notification Security PIN

You can force an operator to enter a personal identification number (PIN) in order to hear a Voice alarm notification message. It's a good idea to validate that the person who is receiving the voice alarm is the correct person by requiring a PIN.

When alarms are sent out via voice, you typically want an operator to acknowledge the alarm by pressing '1' on their phone. Requiring a PIN also prevents other people who might be sharing a phone from acknowledging important alarms.

It is easy to setup Security PINs in Ignition in a couple simple steps. First, a PIN number will need to be created in each user's profile to listen to Voice messages. The second step is to configure the Voice.

On this page ...

- [User Profile](#)
- [Voice Notification Profile](#)



User Pin for Voice

[Watch the Video](#)

User Profile

Let's begin by configuring a Security PIN for each user that will be receiving a voice message and acknowledging SMS alarm messages.

1. Go to the **Config** section of the [Gateway Webpage](#).
2. Scroll down to **Security > Users, Roles** from the menu on the left.
3. Under **default** user source, select the **manage users** link.

The screenshot shows the 'User Sources' configuration page. It lists two user sources: 'default' and 'opcua-module'. The 'default' source is selected, and its details are shown in the main pane. The 'manage users' button is highlighted with a red box. Below the table, there are links to 'Create new User Source...' and 'Verify a User Source...'. The top navigation bar shows the path: Config > Security > User Sources.

Name	Type	Description	Actions
default	Internal	This is the default and always present internal authentication profile.	manage users edit
opcua-module	Internal	OPC UA clients will authenticate against this profile by default.	More ▾ edit

4. Here you will see list of all your users. For every user that is using Voice and/or SMS alarm notifications, you will need to edit their user profile to create a PIN. Select the **Edit** link for one of your users.

User Sources					
		Users	Roles	Contact Info	Schedule
admin	Administrator, Operator	email: admin@mycompany.com, phone: 555-555-5555	Always	Edit Delete	
Bob	Driver	email: bob@mycompany.com	Always	Edit Delete	
guest	Operator		Always	Edit Delete	
Sal	Administrator, Driver, Operator		Always	Edit Delete	
→ Add User					

- The user profile screen will open. Scroll down to the bottom of the screen, and verify that the contact information for Email and Phone is up-to-date.
- Create a **Security Pin**, and press **Save Changes**.

Now, this PIN is associated with this user.

User Sources					
Contact Info					
Type	Value	Up / Down Delete	Up / Down Delete	Up / Down Delete	Up / Down Delete
E-Mail	admin@mycompany.com				
Phone	555-555-5555				
Add Contact Info					
Extended Properties					
Security PIN	<input type="text" value="8765432"/>				
Cancel			Save Changes		

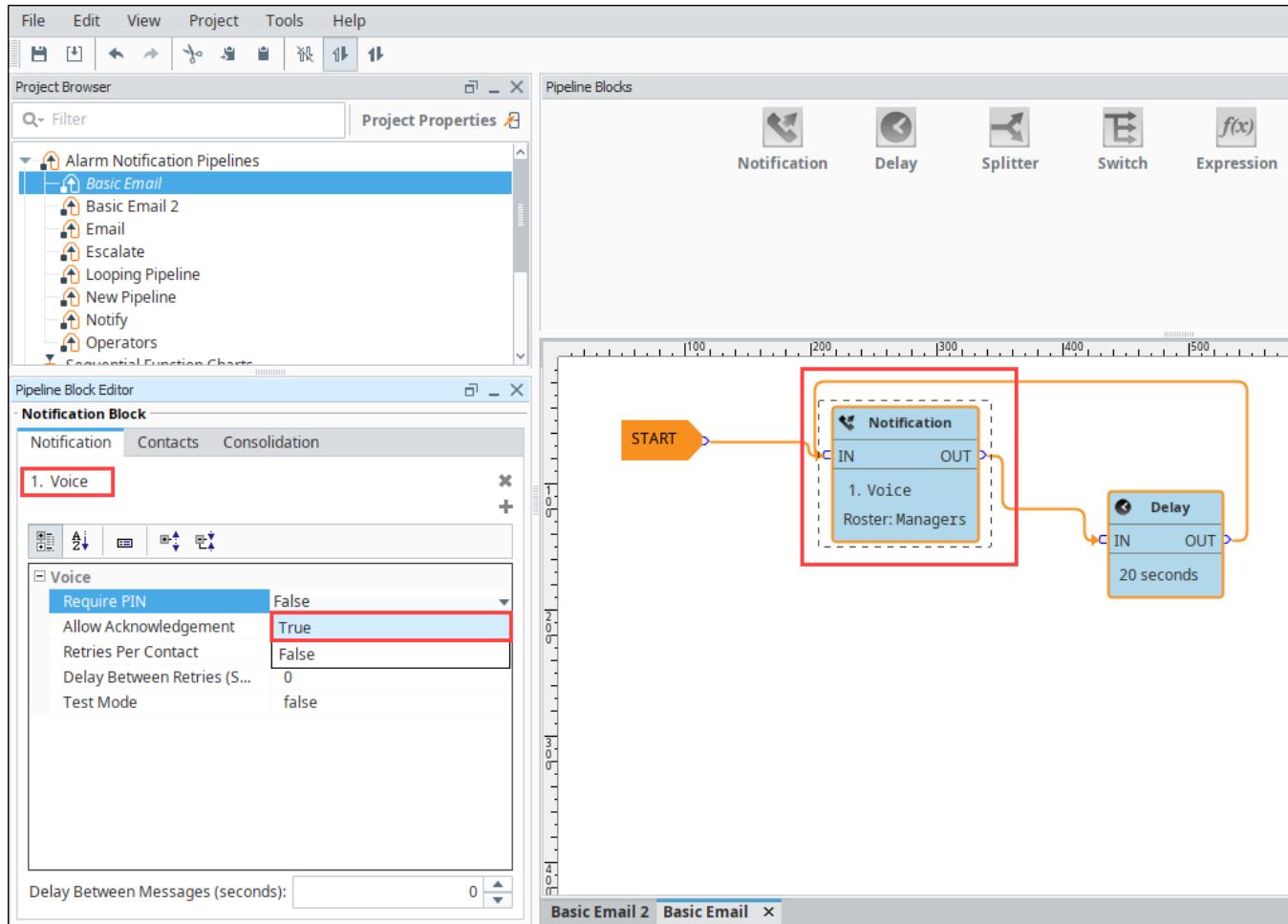
Voice Notification Profile

The next step is to configure the Voice Notification Profile to require a PIN.

- In the **Designer**, go to your **Alarm Notification Pipelines** and open your pipeline. This example uses the **Basic Email** pipeline.
- Select the **Notification Block** in the workspace.
- Click on the **Notification** tab, and choose **Voice**.

4. You'll see several settings for Voice. Set the Required PIN to **True**.

Now, when the operator receives a call, they will have to type in their PIN number followed by the pound sign to hear the message.



Related Topics ...

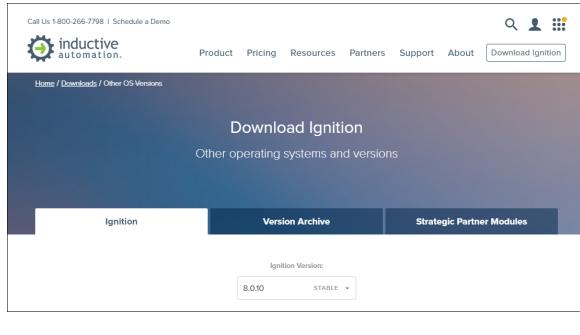
- [Voice Notification Scripts](#)

Voice Notification Scripts

Voice Languages Overview

When alarm events happen, Ignition uses the Voice Notification Module and the voice language modules to send alarm messages. The Voice Notification Module supports making calls using male and female voices in multiple languages. The language modules (i.e., English, Spanish, Italian, etc.) contain the text to speech engine that is specific for that language and gender.

You can download these modules from [Inductive Automation's Downloads page](#) on the website, and install them in your Gateway.



The screenshot shows the Ignition software download page. At the top, there are links for 'Call Us 1-800-266-7798 | Schedule a Demo', 'inductive automation.', 'Product', 'Pricing', 'Resources', 'Partners', 'Support', 'About', and 'Download Ignition'. Below this, the URL 'Home / Downloads / Other OS Versions' is shown. The main content area has a dark blue header with 'Download Ignition' and 'Other operating systems and versions'. Below the header, there are three tabs: 'Ignition', 'Version Archive', and 'Strategic Partner Modules'. Under 'Version Archive', there is a dropdown menu for 'Ignition Version' set to '8.0.10 STABLE'. A table lists several notification modules:

Notification	Version	Checksum
Alarm Notification Module (810KB)	5.0.10	sha-256
SMS Notification Module (197KB)	5.0.10	sha-256
Twilio Notification Module (8MB)	2.0.10	sha-256
Voice Notification Module (14MB)	5.0.10	sha-256

[Link](#)

On this page ...

- [Voice Languages Overview](#)
- [Modifying a Voice Notification Script](#)



Voice Languages

[Watch the Video](#)



Custom Email, SMS, and Voice Scripts

[Watch the Video](#)

The following table shows a list of the current Text-to-Speech voices that you can download after you have the Voice Notification Module installed. To download Text-to-Speech voices, go to the Knowledge Base article on [Voice Notification Downloads](#) where you can also find some good information.

Voices	Languages
Katherine	English
Laura	Italian
Sara	Spanish
Suzanne	French
Alex	German

Once you [installed the voice modules](#), you can configure the voice scripts for any language in the voice notification module. For example, you may want to have a different script that gets sent out when you're calling someone in English versus someone that you're calling in Spanish.

Note: When we refer to a "script" on this page, we are talking about a speech or a dialog for Ignition to read to your users over the phone. These scripts have nothing to do with the scripting language (Python) that you write code with.

Modifying a Voice Notification Script

From Voice Notification Profiles page, you can manage your voice scripts. Two examples are provided here; one in English, and the other in Spanish.

1. Go to your **Gateway Config** page, and select **Alarming > Notification**. To the right of the Voice Notification Profile, click on the **manage scripts** link.

Name	Description	Enabled	Type	Status	Actions
Email Notifications		true	Email Notification	Running	delete edit
Test		true	Email Notification	Running	delete edit
Voice		true	VOIP Voice Notification	Registered with VOIP Host	More ▾ edit

→ Create new Alarm Notification Profile...
→ Test Pipelines and Notification Profiles...

2. Here you can manage all the scripts for every language, and can even create new scripts. The languages that come with Ignition are English, Italian, French, and Spanish. There are a host of other languages that Ignition supports. For this example, click on the **edit** link for the **English** language.

Language	Country	Actions
English	General	edit
Italian	General	delete edit
French	General	delete edit
Spanish	General	delete edit

→ Create new script...

3. Default voice scripts are the same in all supported languages. This is the default script for English. It consists of a number of little scripts. Each script is what the recipient of the call will hear over the phone. These are instructions informing the recipient of one or more the alarm(s) depending on how the alarm notification is configured.

When you receive a voice notification, you will hear each script or instruction one at a time. For example, the first script you will hear is "Hello this is an Ignition alarm notification" which, of course, is the greeting. It will then go through each script and inform the recipient of the alarm name, date and time of the alarm, alarm status, and how to acknowledge the alarm. You can edit this default script to your requirements. When you're finished editing, click **Save Changes**.

You can stop any of these Call Scripts from being read by removing all text in that field.