Types of Tags

There are several different types or scopes of Tags you can create in Ignition: Gateway Executed Tags, System Tags, and Client Tags. Each Tag belongs to a scope, and each scope plays a role in how Tags behave. All these Tag types are available in the Tag Browser.

When discussing "Tags", we commonly mean Gateway Executed Tags, but System and Client Tags also play an important role in the overall design of a project.

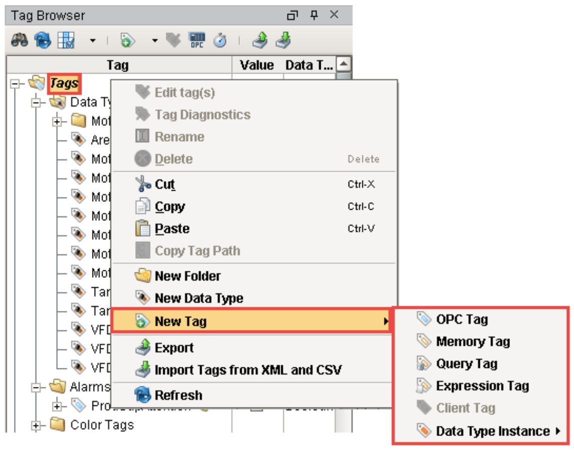
Let's examine the different types of Tags.

On this page ...

Gateway Executed Tags

Tags executed in the Gateway support all of the primary features of Tags: scaling, alarming, history, and role-based permissions. These Tags are all Gateway scoped, and the values of the Tags are shared among all running clients. They are identical in their configurations, apart from defining how the value is generated.

You can create each of these Tags in the **Tag Browser** by right clicking on the Tags folder, scroll down to **New Tag** and select the desired Gateway Executed Tag.



INDUCTIVE UNIVERSITY

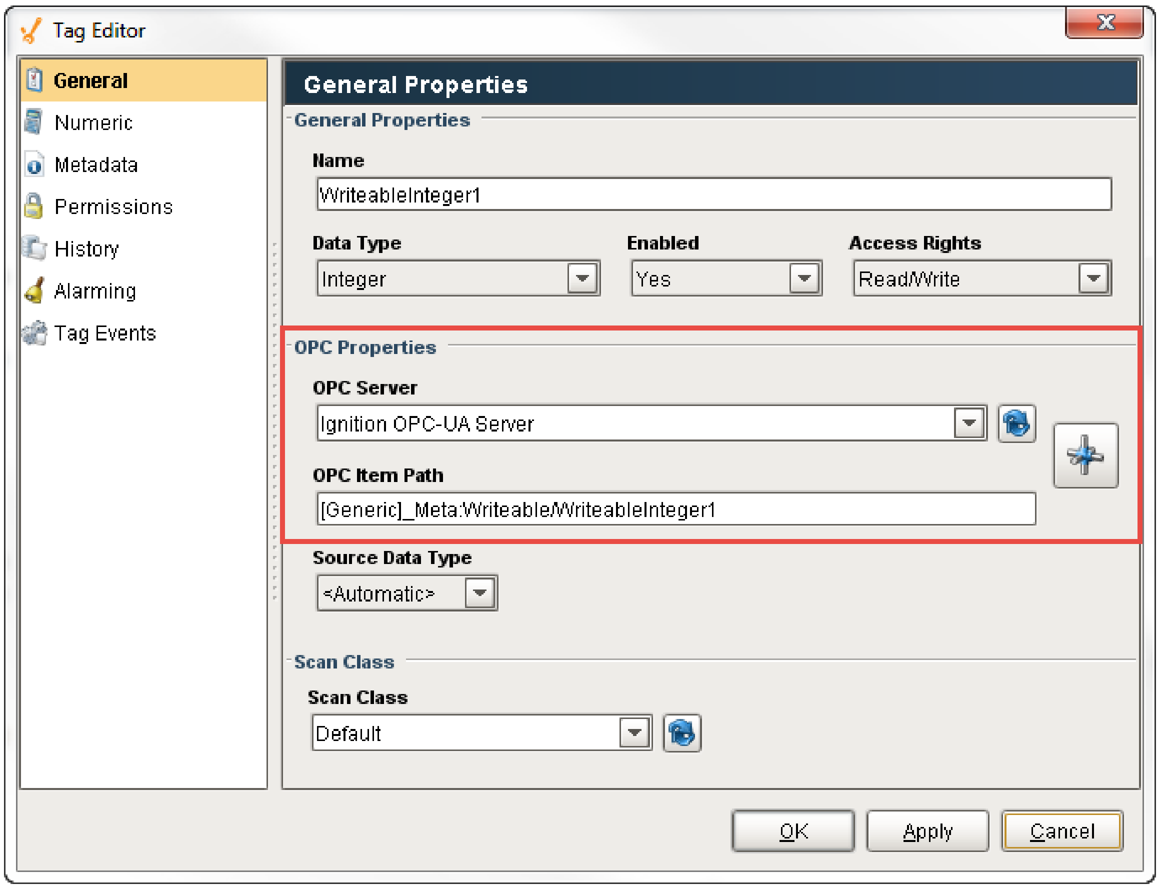
Types of Tags

Watch the Video

OPC Tags

An OPC Tag is an OPC server and address which drives values. The OPC server is typically a PLC or device, such as Ignition's OPC-UA server, or any third party OPC-UA server.

In the **Tag Browser**, double click on any existing OPC Tag, to see the the **OPC Server** name and **Item Path**.



General Properties

General Properties

Name: WriteableInteger1

Data Type: Integer

Enabled: Yes

Access Rights: Read/Write

OPC Properties

OPC Server: Ignition OPC-UA Server

OPC Item Path: [Generic]_Meta:Writeable/WriteableInteger1

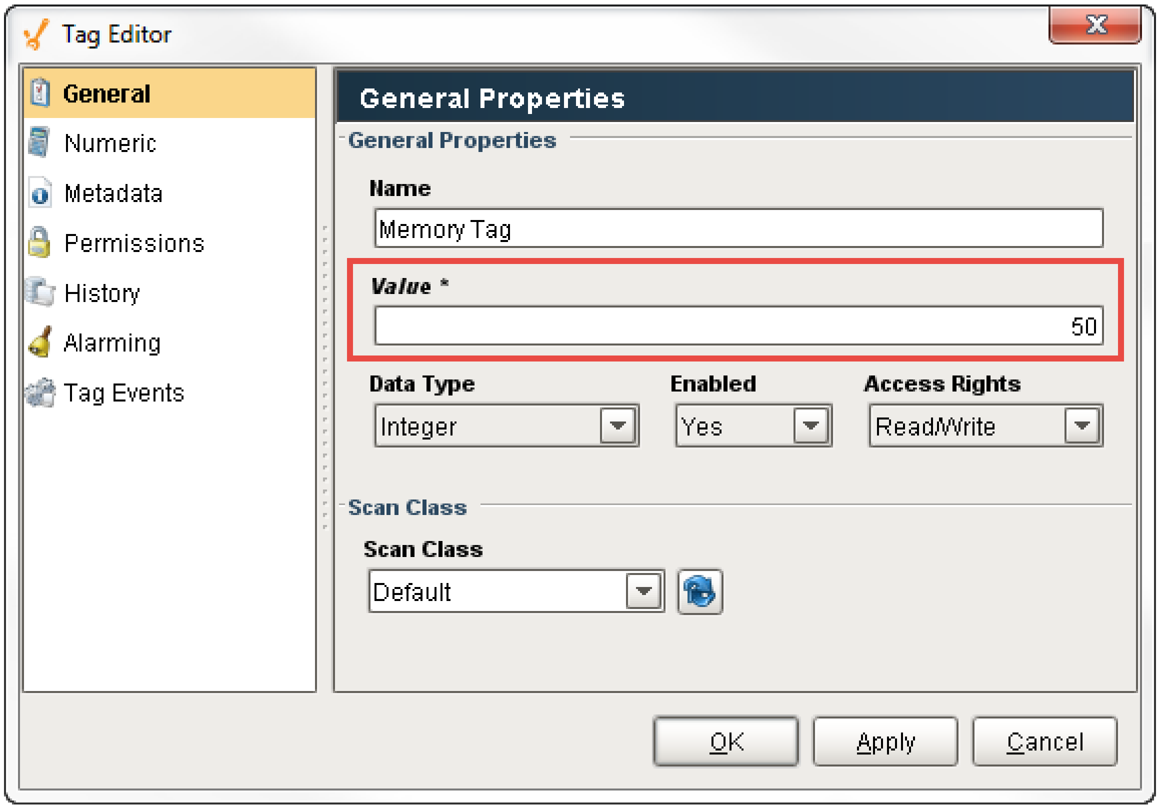
Source Data Type: <Automatic>

Scan Class

Scan Class: Default

Memory Tags

The Memory Tags are simply values. The value is specified during configuration, and is stored when written (if the Tag allows writing).



Name: Memory Tag

Value *: 50

Data Type: Integer

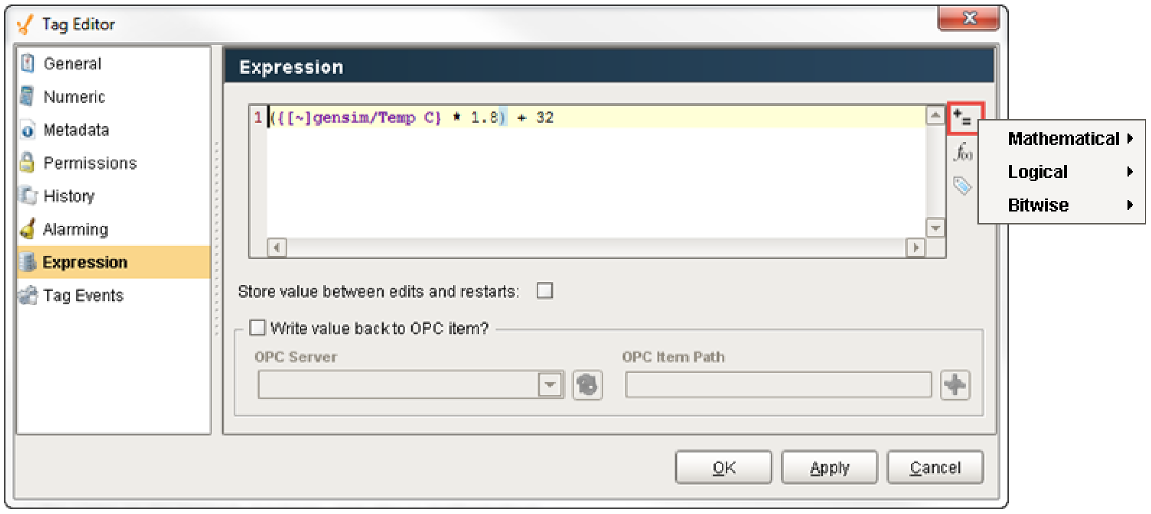
Enabled: Yes

Access Rights: Read/Write

Scan Class: Default

Expression Tags

The Expression Tags are driven by an expression, and its values are derived from a calculation. The expression syntax is the same as for property bindings, and allows mathematical operations, references to other Tags, logic operations, functions, and more. This expression example converts Celsius to Fahrenheit.



```
({[~]gensim/Temp C} * 1.8) + 32
```

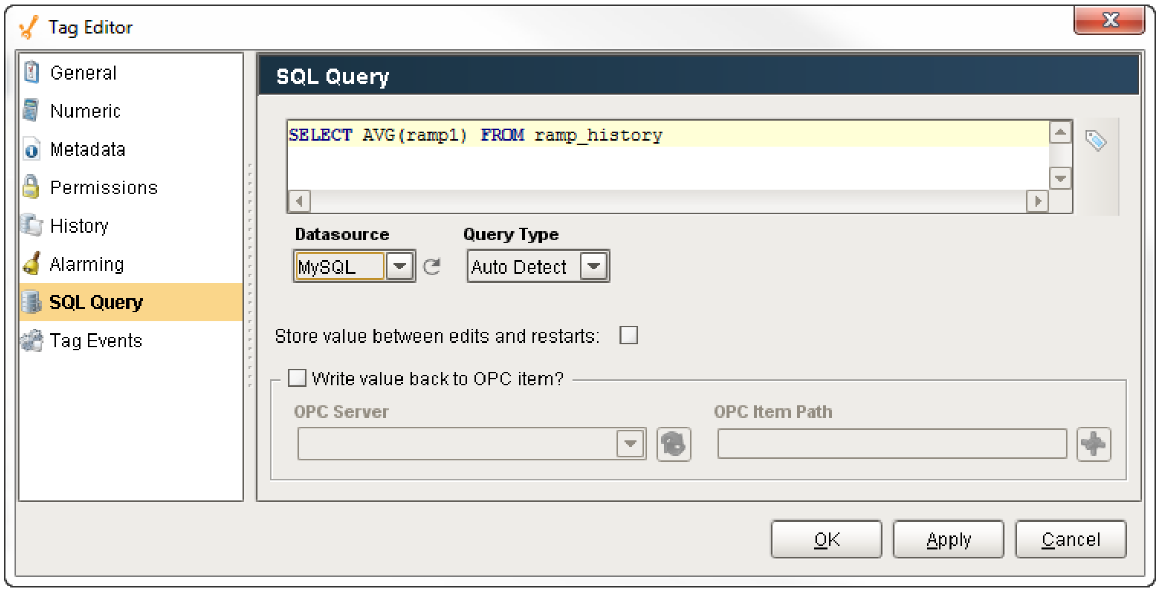
Store value between edits and restarts: ☐

☐ Write value back to OPC item?

SQL Query Tags

The SQL Query Tags execute a SQL Query, whose result provides the value for the Tag from the SQL database. Like an SQL binding in Vision, SQL Query Tags can reference other Tags to build dynamic queries. This SQL query example is getting the average of Ramp1 from the History table.

Each query Tag will run a separate query, so large numbers of this type of Tag will **dramatically** increase the number of queries running against the database. It is highly recommended to attempt to optimize and restrict the number of query Tags: instead of having several query Tags pulling values from the same table, have one query Tag configured to a Dataset data type that returns large portions of the table, and use several expression Tags to extract values from the individual elements in the dataset.

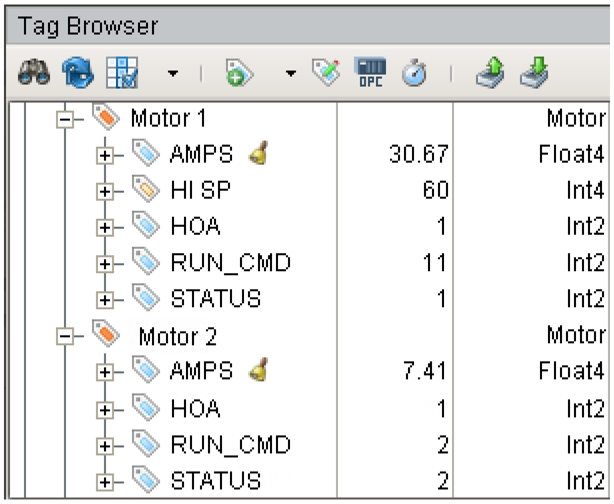


```
SELECT AVG(ramp1) FROM ramp_history
```

Datasource: MySQL
Query Type: Auto Detect
Store value between edits and restarts: ☐
☐ Write value back to OPC item?
OPC Server
OPC Item Path

Complex Tags (UDTs)

Complex Tags are created out of standard Tag types, but offer a variety of additional features. In simple terms, you can think of them as a way to create "data templates", where a particular structure of Tags is defined, and can then be created as if it were a single Tag. This UDT example shows two Motor instances, the data type Motor, and all the Tags that make up the structure (i.e., AMPS, HI SP, HOA, RUN_CMD, and STATUS).



Motor 1		Motor
AMPS	30.67	Float4
HI SP	60	Int4
HOA	1	Int2
RUN_CMD	11	Int2
STATUS	1	Int2
Motor 2		Motor
AMPS	7.41	Float4
HOA	1	Int2
RUN_CMD	2	Int2
STATUS	2	Int2

Derived Tags

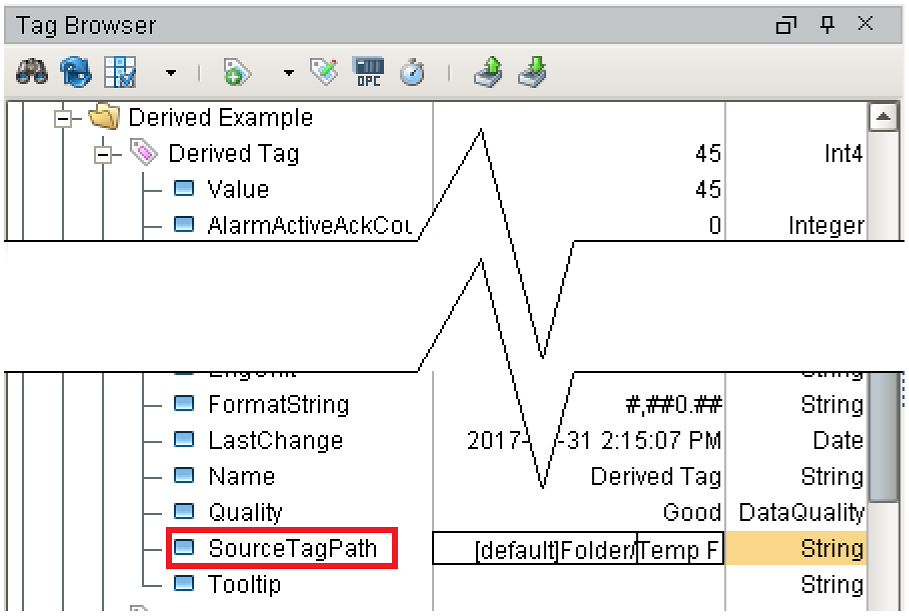
This feature is new in Ignition version **7.9.2**
Click here to check out the other new features

A Derived Tag is an abstracted Tag that refers to another Tag. They are similar conceptually to Expressions Tags that reference another Tag, but Derived Tags have some additional functionality. Namely, they can write back to the referenced Tag.

Source Tag Path

The Source Tag Path property of a Derived Tag depicts which Tag the Derived Tag should reference. This is similar to the OPC Item Path on an OPC Tag, except it points to an Ignition Tag.

Aside from editing the Tag from the Designer, the Source Tag Path on a Derived Tag may be modified by writing directly to the **SourceTagPa th** property on the Tag, or using a Tag Binding on a Vision component



Derived Example		
Derived Tag	45	Int4
Value	45	
AlarmActiveAckCou	0	Integer
FormatString	#,##0.##	String
LastChange	2017- -31 2:15:07 PM	Date
Name	Derived Tag	String
Quality	Good	DataQuality
SourceTagPath	[default]Folder/Temp F	String
Tooltip		String

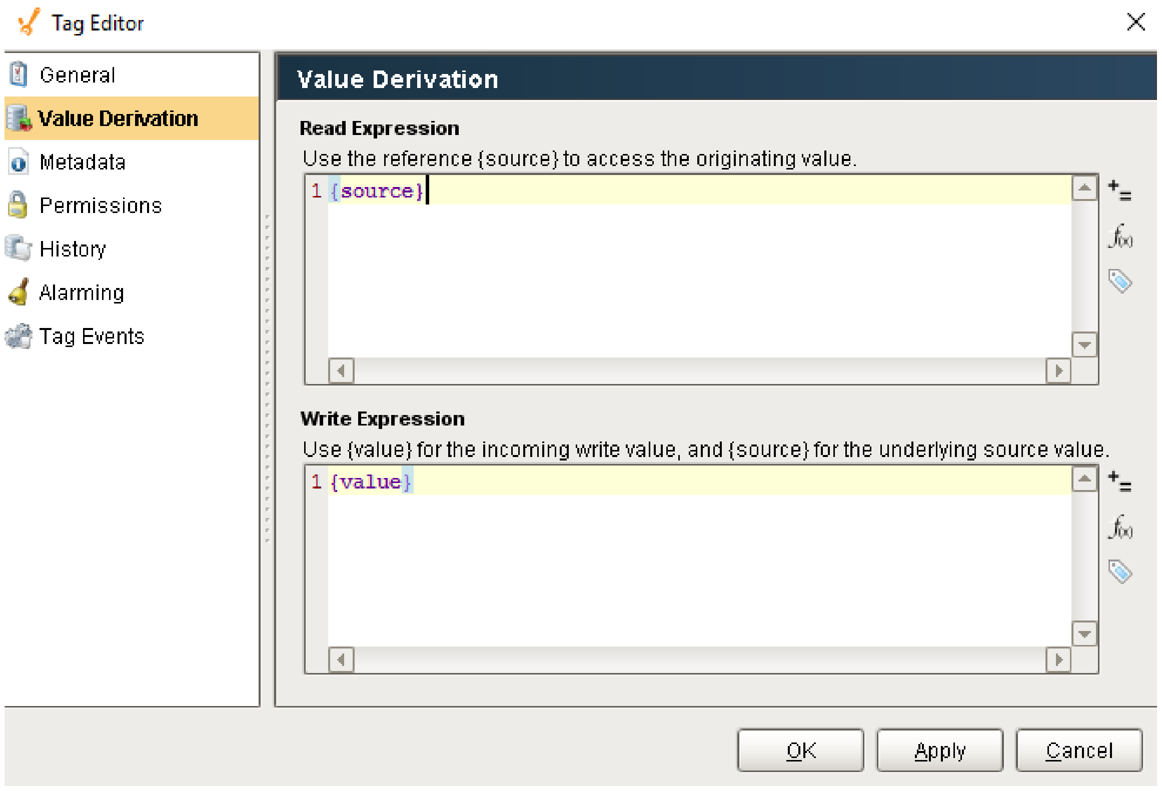
Additionally, the **SourceTagPath** property may be changed through scripting:

```
#Example of writing to the SourceTagPath attribute via Python Script
system.tag.write("Derived Example/Derived Tag.SourceTagPath", "[.]Folder/New Source Tag")
```

The Derived Tag will update at the rate of the source Tag.

Value Derivation

Instead of a Numeric section, Derived Tags have a Value Derivation section accessible in the Tag Editor. This section allows you to leverage Expressions when reading from or writing to the source Tag, which provides a greater degree of customization than the traditional Numeric scaling settings available to OPC Tags.



Read Expression

Use the reference {source} to access the originating value.

```
{source}
```

Write Expression

Use {value} for the incoming write value, and {source} for the underlying source value.

```
{value}
```

Read Expression	Determines what value should appear on the Derived Tag. The current value of the source Tag may be referenced with the **{source}** reference.
Write Expression	When writing to the Derived Tag, this expression determines what value should be written to the source Tag. The current value of the source Tag may be referenced with the **{source}** reference. The raw value passed to the Derived Tag may be referenced with the **{value}** reference.

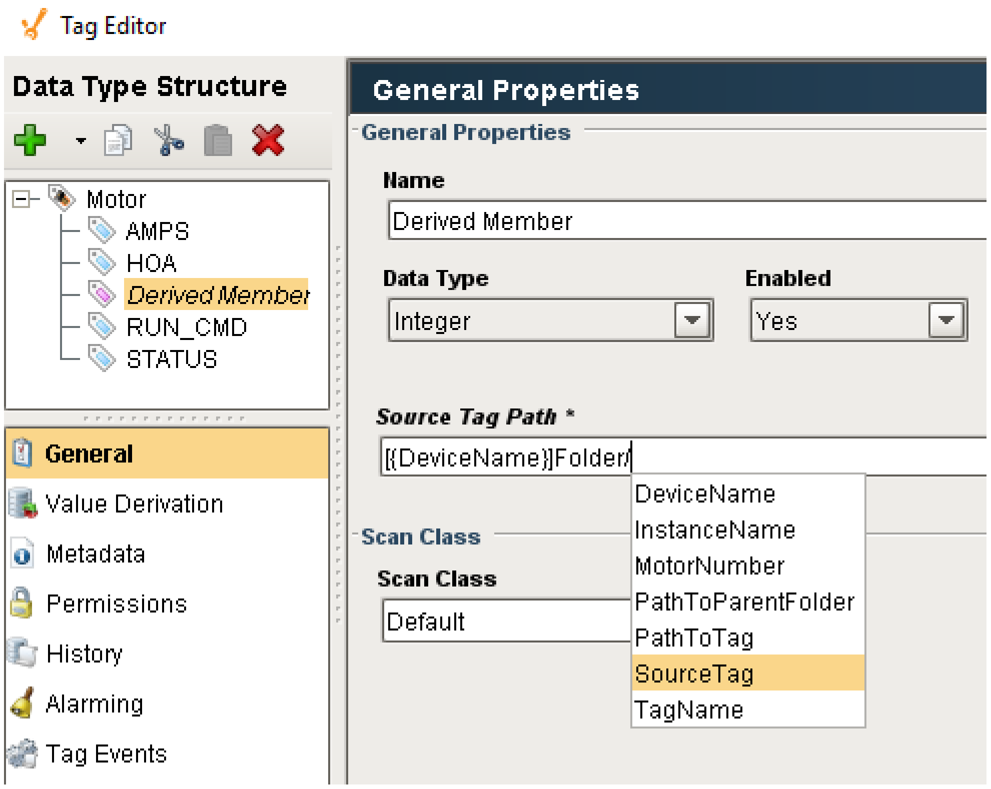
This interface provides an opportunity to scale the written and read value. For example, if the source Tag was in Fahrenheit, and the derived Tag should be Celsius, we could use the following expressions:

```
//Read Expression
({source}-32)*(5/9)

//Write Expression
{value}*(9/5) + 32
```

Derived Tags in UDTs

When added as a member of a UDT, Derived Tags may reference peer members. Additionally, UDT parameters may be used in the Source Tag Path.


Tag Editor

Data Type Structure

Motor
- AMPS
- HOA
- Derived Member
- RUN_CMD
- STATUS

General
Value Derivation
Metadata
Permissions
History
Alarming
Tag Events

General Properties

General Properties

Name: Derived Member

Data Type: Integer

Enabled: Yes

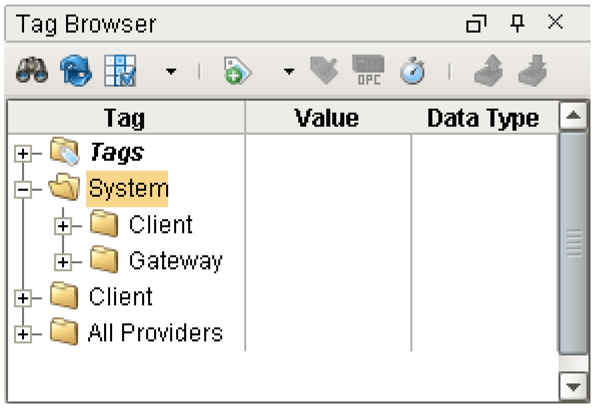
Source Tag Path *: [{DeviceName}]Folder/

Scan Class

Scan Class: Default

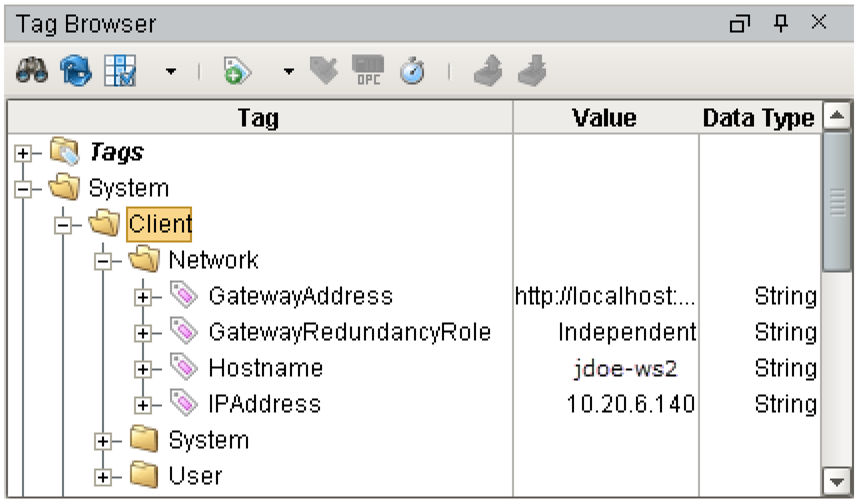
System Tags

System Tags are status Tags that Ignition provides about the Client and the Gateway. You cannot create any new Tags in the Client or Gateway folders. The Tag names and folder structure are managed by the Ignition Gateway. However, you can modify the Tags in the Gateway folder to perform alarming, history, and scaling. You cannot modify the Client-scoped System Tags.



Client-scoped System Tags

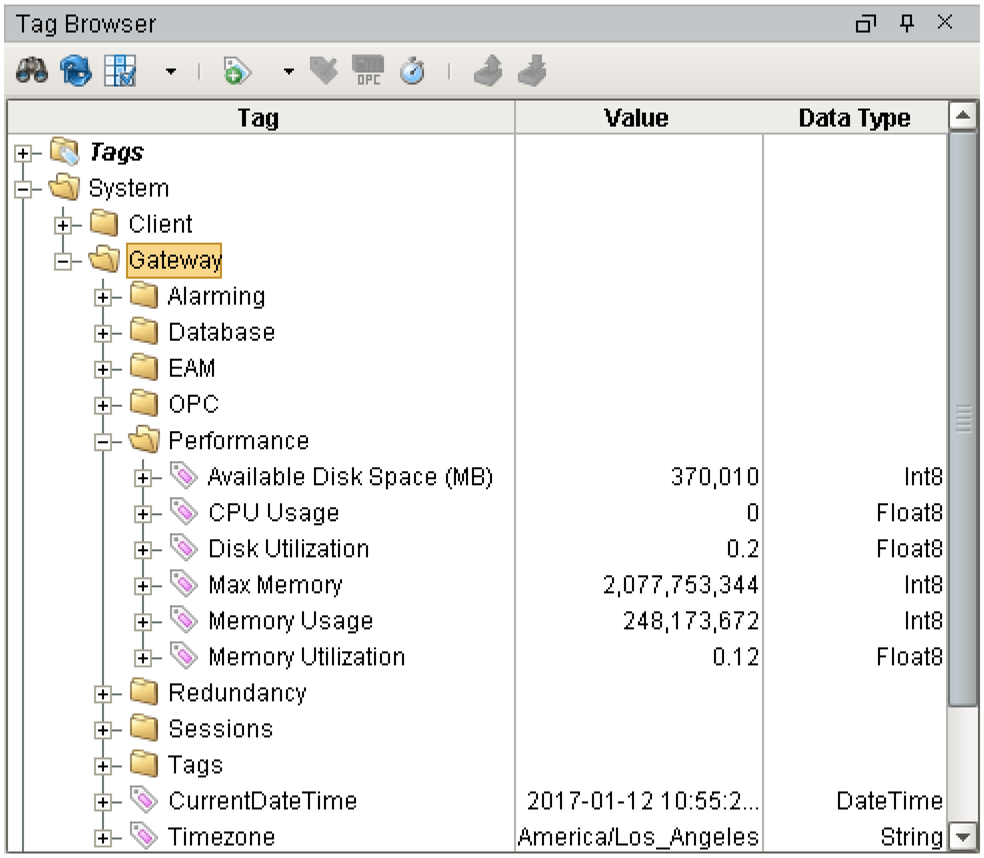
Client-scoped Tags provide status information about the client's system. Every individual client is going to have their own values like IP address, hosting name, username, and more.



Tag	Value	Data Type
Tags		
System		
Client		
Network		
GatewayAddress	http://localhost:...	String
GatewayRedundancyRole	Independent	String
Hostname	jdoe-ws2	String
IPAddress	10.20.6.140	String
System		
User		

Gateway-scoped System Tags

Gateway-scoped Tags provide status information about the Gateway like alarming, database, throughput, memory usage, performance of the server, and much more.



Tag	Value	Data Type
Tags		
System		
Client		
Gateway		
Alarming		
Database		
EAM		
OPC		
Performance		
Available Disk Space (MB)	370,010	Int8
CPU Usage	0	Float8
Disk Utilization	0.2	Float8
Max Memory	2,077,753,344	Int8
Memory Usage	248,173,672	Int8
Memory Utilization	0.12	Float8
Redundancy		
Sessions		
Tags		
CurrentDateTime	2017-01-12 10:55:2...	DateTime
Timezone	America/Los_Angeles	String

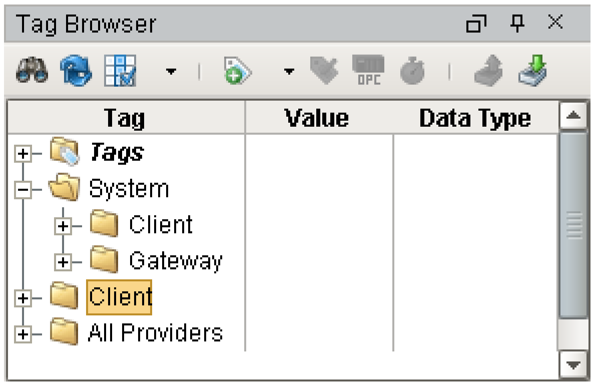
Other System Tags

There are potentially more of these System Tags depending on the modules that you have installed. For example, if the Mobile Module is installed you will see a **Mobile** folder under the Client and Gateway folders. This will include information about the Client in addition to the Client folder, but only if the client is opened with the Mobile Module.

Client Tags

Client Tags, as the name implies, are only available for use in Clients. They are scoped at the client level unlike the Gateway Executed Tags. All clients will have the same list of client Tags, however, the actual values are unique and independent for each running Client. In other words, even though client Tags are created in the Designer, each client will create their own instances. This makes them very useful as in-project variables for passing information between screens, and between other parts of the clients, such as scripting.

Client Tags support most of the data types that standard Tags do (including datasets), but excluding the array types. Additionally, Client Tags do not have a Scan Class property, so the value will only update when the polling property executes, or a reference in the Client Tag's expression updates.



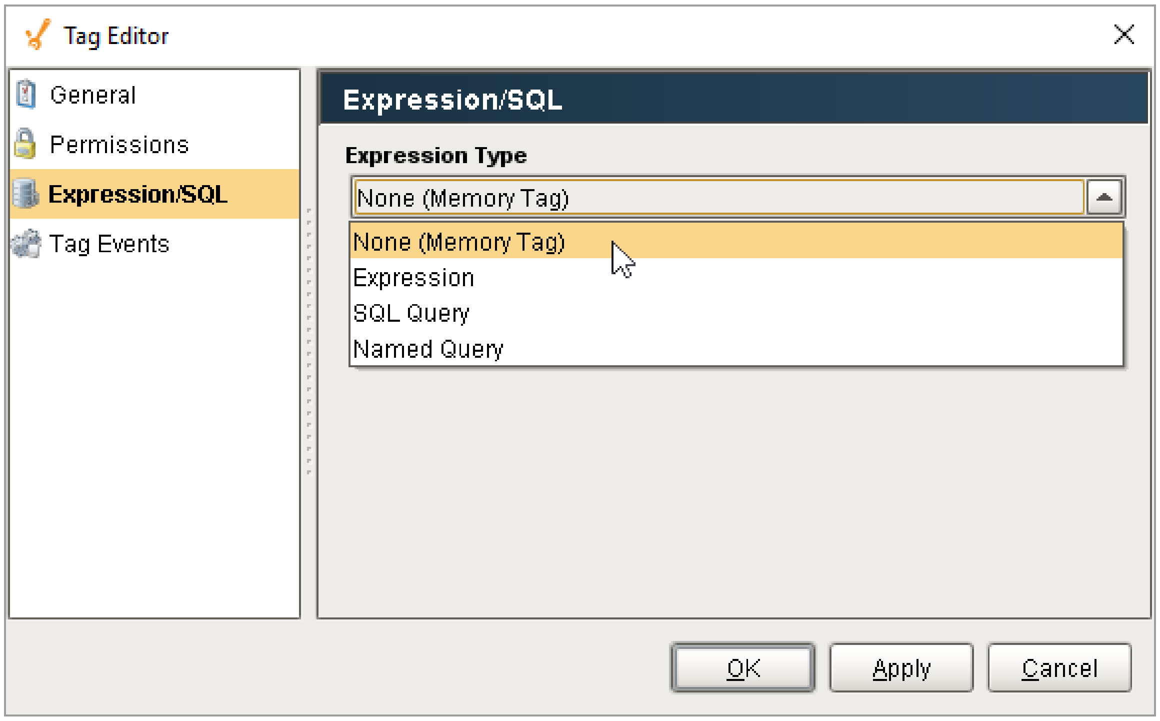
Expression Type

Client Tags can be configured in one of several ways, based on the **Expression Type** property. The following options are available.

- **None**: Causes the Tag to behave like a Memory Tag.
- **Expression**: Allows the Tag to utilize Ignition's Expression Language, much like an Expression Tag.
- **Query**: Executes a SQL Query, similar to a Query Tag.
- **Named Queries**:

  This feature is new in Ignition version **7.9.6**
  Click here to check out the other new features

  Client Tags may now call Named Queries



Expression Type: None (Memory Tag)

- None (Memory Tag)
- Expression
- SQL Query
- Named Query

Related Topics ...

- Keeping Tags Organized
- Creating Memory, Expression, and Query Tags
- System, Client, and Diagnostic Tags