Understanding Tags

Where are Tags Configured?

Tags provide a consistent data model throughout Ignition, and offer the easiest way to get up and running creating realtime status and control systems. Despite their low initial learning curve, however, Tags offer a great amount of power in system design and configuration. The ability to aggregate Tags from a variety of installations means that you can build widely distributed SCADA systems more easily than ever before with a high level of performance and relatively easy configuration.

While the goal of Tags in Ignition is to create an easy yet powerful model, the variety of options and terminology can sometimes make configuration confusing. Tags are created and controlled using both the Gateway and the Designer for configuration.

- In the **Designer**, you will create the **Tags**. There are several types of Tags such as an OPC Tags, Memory Tags, and more. Each tag has many properties and other functionality such as alarming, history, etc. Once your tags are created, you can use them in your windows, reports, and more.
- In the **Gateway**, you create and modify **Tag Providers**. You can create these Realtime Providers to store groupings of Tags for use in your projects either internally in Ignition or in an external database. There are also Historian Providers used to store historical data for the tags, but these are automatically created for each datasource you have. These Tag Provider configurations in the Gateway apply globally to all your projects.

Creating Tags

By default, there is one Internal Tag Provider created for you when Ignition is installed. Because of this, you can get started right away with creating Tags.

Tags are created in the Designer, either manual by right-clicking in the Tag Browser and selecting New Tag, or if you already have a device set up, by opening the OPC Browser and dragging tags or folders in. See Browsing and Creating OPC Tags for more information.

Manually Create Tags
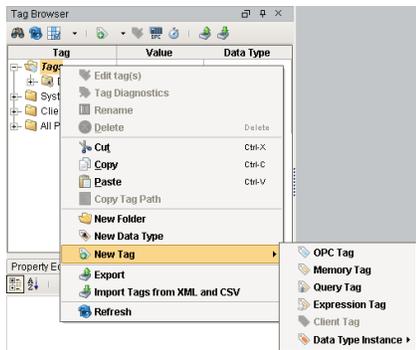
Using the OPC Browser
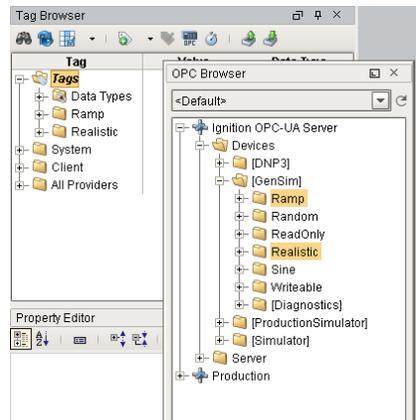
Working with Tags

Once you have Tags in your Tag Browser, they will automatically start executing. The Designer is live, and you can see your Tags update in real time and write values directly to them. The most commonly used type of tag is an OPC Tag, which gets its value from a device like an OPC Server. You can connect to just about any type of device out there and show the data on screen, as history, and write back whenever you want. Tags make Ignition extremely flexible and easy to show data. See the Quick Start Guide to get some tags created and in your client.

There are many things you can do with Tags to keep everything organized and give you exactly what you need.

- Learn about the different types of Tags

- Organize your Tags
- Set Metadata properties
- Set scaling
- Import/export

Once you have some Tags created, you can edit them to include even more features. You can:

- Set up Alarms
- Store Historical data
- Add Security
- Add Scripting to Tag Events

In addition to all that, Ignition allows you to create UDTs (User Defined Types) out of your Tags to rapidly develop projects with structured objects. This gives you the ability to create a master type then quickly add many instances. UDTs allow you to make a change in one place and see it automatically propagated to every instance.

What are Tag Providers?

Tag Providers and Drivers

At the highest level of configuration is the Tag Provider. A provider is a tag database (a collection of tags) and a name. An Ignition Gateway can have any number of tag providers, and therefore the name is used to distinguish which provider a tag comes from. Tag Providers can be set up with security or even disabled independent of each other.

Every provider holds tags, but not every provider is a Tag driver, or a tag executor. Some tag providers simply make tags available to use, and the tag execution is performed by a different driving provider elsewhere. For example, the Database Provider is a Tag provider that simply watches a tag database stored in an external database. It does not execute tags, it only monitors the values of the tags present. Somewhere else there is a tag driver such as a Driving Datasource Provider or a legacy FactorySQL that is executing the tags and storing the values to the database.

Realtime vs. Historian Providers

As discussed above, all Tags reside in a tag provider and provide realtime values. Additionally, there is the concept of tag historian providers, which can store and query historical data for tags. Each tag can optionally have a historian provider assigned to it to whom it will report value changes for historical storage.

Realtime Providers

The Tags "tag database", or how Tag's tag configuration is stored, can take a few different forms. For internal Tags, the configuration is stored in the Ignition Gateway. With the database form, Tags are stored in a SQL database, outside of the Ignition Gateway. Finally, Remote Tag Providers allow Tag Providers of one Gateway to be accessed on a different Gateway. The different storage mechanisms have different pro and cons, and so it is a good idea to acquaint yourself with each of them.

Internal Tag Provider

The Internal Tags Provider manages the Tags inside the Ignition Gateway. This is the most common method and is optimized for the best scaling and performance.

It is possible to create multiple internal providers per Gateway. The Tags can be exposed to other Gateways on the network through the Gateway's OPC-UA settings.

Remote Tag Provider

The Remote Tag Provider works by creating a link from the current gateway to a tag provider on another Gateway in the Gateway Network. This link can be to either an internal provider or a database provider. The catch is that the two Gateways must be linked through the Gateway Network. However, unlike an OPC connection which pulls tags from the OPC server, a remote provider will keep all of the Tag's settings intact. So, things like alarms and history will still be present, and can be used in the Gateway. By default, the remote tag provider will fall under the Default Security Zone and be read only.

Remote Tag Provider

Watch the Video

There are two types of Database Tags providers in Ignition:

- Tag Consuming Provider
- Tag Driving Provider (provided by the SQL Bridge module)

While these two tag providers used to be referred to as external tag providers, they function in the same way that they used to. The Driving Provider will execute Tags as well as make available Tags driven by other Tag Drivers looking at the same database, such as other Ignition Gateways using the Driving Provider, or legacy FactorySQL installations, whereas the Consuming Provider will only read the Tags and will not write values back to the database..

The primary benefit of database providers is that the data is stored in a central database, and is therefore accessible to other data consuming systems that do not have the capabilities to connect to Ignition's OPC-UA server as an OPC-UA client, yet still require the need to consume realtime Tag information.

The negative side to external providers is that all Tag values must be written to the database and then polled for change, which is a less efficient method than internal provider's method. In other words, the inherent architecture of the external model does not scale as well as the internal provider model. For more information about those tables and how they work, see the External Tag Provider Reference.

Configuring Realtime Providers

Realtime Tags providers are configured in the Gateway's **Configure** section under **Tags > Realtime**. After installation, the Ignition Gateway will start with an internal provider defined. You can edit its name and settings by selecting **edit** to the right of its entry in the table, or create new providers by selecting **Create new Realtime Tag Provider** below the table.

There are four types of **Realtime Tag Providers** that you can choose from:

- **Standard Tag Provider**
- **Remote Tag Provider**
- **Tag Consuming Provider**
- **Tag Driving Provider**

Standard Tag Provider

Tags are stored inside of Ignition and executed by the system.

Name	The name of the provider.
Description	The description of the provider.
Tag Editing Role (s)	Users must belong to one of these roles in order to edit this provider's tags in the Designer. Multiple roles can be specified by separating them with commas. If blank, tag editing for this provider will not be restricted in the Designer.
Default Database	The default database connection to use for expression tags that run SQL queries. All query tags with default database providers selected with run their queries against this database source.

Remote Tag Provider

Tag Provider from one Gateway is brought in to another Gateway.

Name	The name of the provider.
Description	The description of the provider
Tag Editing Role (s)	Users must belong to one of these roles in order to edit this provider's tags in the Designer. Multiple roles can be specified by separating them with commas. If blank, tag editing for this provider will not be restricted in the Designer.
Gateway	The name of the Gateway on the Gateway Network that this provider is coming from.
Provider	The name of the provider as it is on the remote Gateway. This does not have to be the same as its name on the new Gateway.

History Access Mode	This setting dictates how tag history is queried for remote tags. Gateway Network will go through the Gateway Network to query history. Database will query the database directly.
History Datasource	The datasource to query when History Access Mode is set to Database.
History Driver	If querying the database directly, this is the gateway name of the remote system. It is used to identify data from that system in the database.
History Provider	If querying the database directly, this is the name of the tag provider on the remote system. It is used along with driver name to identify the correct tags in the database.
Alarms Enabled	If true, alarms configured on the remote gateway will be enabled on the new Gateway.
Alarm Mode	How alarm state should be monitored. In 'queried', state will be queried through the gateway network when necessary. In 'subscribed', the state will be subscribed, and updates will be sent asynchronously. Subscribed provides better performance, but uses more memory.

Tag Consuming Provider

Reads tags stored in an external database and driven by a different instance. Only consumes tags provided by other systems, does not write tag values back to the database.

Name	The name of the provider.
Description	The description of the provider.
Tag Editing Role (s)	Users must belong to one of these roles in order to edit this provider's tags in the Designer. Multiple roles can be specified by separating them with commas. If blank, tag editing for this provider will not be restricted in the Designer.
Database	The database connection where the tag database resides.
Poll rate	The rate in milliseconds at which value and configuration changes will be polled.
Poll overlap	The amount of time, in milliseconds, to overlap the polling time span.

Tag Driving Provider

Stores and executes tags in an external database.

Name	The name of the provider.
Description	The description of the provider.
Tag Editing Role (s)	Users must belong to one of these roles in order to edit this provider's tags in the Designer. Multiple roles can be specified by separating them with commas. If blank, tag editing for this provider will not be restricted in the Designer.
Database	The default database connection where the tag database resides.
Driver name	The unique name of this driver. Since the tags are stored in a central database, there may be multiple providers and drivers operating on them. This name will be used to identify this driver instance, and the tags that it executes. While the driving provider will read all of the tags stored in the database, it will only execute those tags that are assigned to it.
Poll rate	The rate in milliseconds at which value and configuration changes will be polled.
Poll overlap	The amount of time to overlap polls by. If set to 0, the config scan will look for changes only since the last execution. However, on databases that do not support millisecond resolution or are performing less-than-optimally, this could result in missed changes. This setting will expand the window in order to avoid missing these changes.
Enable browsing	Allows remote browsing of the OPC servers available to this driver over TCP/IP. This allows other Gateways to remotely browse and add tags assigned to this driver into the central database.

Browse port	The port to listen on for remote connections. This port must not be in use by any other entity on the machine. Also, each driving provider that wishes to support browsing must have its own port.
Browse address	The IP address/network name that remote Gateways will use when browsing. Therefore, care must be taken that the address is available from the Gateways that will try to connect. Also, since it is used for access by remote systems, it should not be the loopback address (localhost or 127.0.0.1). The browse address and port will be stored in the Tags database so that other Gateways can easily look them up. After the settings are configured, you should immediately see the driver name in the OPC browse list for the external provider on other systems looking at the same database. **Note:** When using remote browsing, make sure that the local firewall has an exception for the port that is used to listen. Otherwise, remote machines will not be allowed to connect.

Enum Reference

Almost every property in Ignition that is a dropdown list has an enumeration that starts at 0 and counts down the list. Below are the definitions of Tag property selections:

Basic Tag fields

Tag Type	
0	OPC Tag
1	DB Tag
2	Client Tag
6	Folder Tag

Datatype	String Value	Integer Value
Byte	Int1	0
Short	Int2	1
Integer	Int4	2
Long	Int8	3
Float	Float4	4
Double	Float8	5
Boolean	Boolean	6
String	String	7
DateTime	DateTime	8
Text	Text	10
Byte Array	Int1Array	17
Short Array	Int2Array	18
Integer Array	Int4Array	11
Long Array	Int8Array	12
Float Array	Float4Array	19

Quality

Data Quality	
0	Bad Data from OPC
4	CONFIG_ERROR
8	NOT_CONNECTED
12	DEVICE_FAILURE
16	SENSOR_FAILURE
20	Bad, showing last value
24	COMM_FAIL
28	OUT_OF_SERVICE
32	WAITING
64	UNCERTAIN
68	UNCERTAIN showing last value
80	SENSOR_BAD
84	LIMIT_EXCEEDED
88	SUB_NORMAL
28	SERVER_DOWN
192	Good Data
216	Good, with local override
256	OPC_UNKNOWN
300	Config Error
301	Comm Error

Alarms

Comparison Mode	
0	Equal
1	Not Equal
2	Less Than
3	Less Than Equal
4	Greater Than
5	Greater Than Equal

Alert Flags	
0x01	Low Exclusive
0x02	Low Infinite
0x04	High Exclusive
0x08	High Infinite
0x10	Any Change
0x20	Low Driven
0x40	High Driven

Double Array	Float8Array	13
Boolean Array	BooleanArray	14
String Array	StringArray	15
DateTime Array	DateTimeArray	16
Binary Data	ByteArray	20
Dataset	DataSet	9
Document	Document	29

310	Expr Eval Error
330	Tag exec error (fsql)
340	Type conversion error
403	Access Denied
404	Not Found
410	Disabled
500	Stale
600	Unknown (loading)
700	Write Pending

Access Rights

0	Read Only
1	Read/Write
2	Custom

Scan Class Modes

0	Direct
1	Driven
2	Leased

Write Response

0	Failure
1	Success
2	Pending

Related Topics ...

- Browsing and Creating OPC Tags
- Creating Memory, Expression, and Query Tags
- Creating User Defined Types - UDTs
- Scan Classes

In This Section ...