KODARO

CSV Driver Guide

April 11, 2021

This documents usage of the CSV driver on the Niagara AX and N4 platforms.

OVERVIEW

This driver imports and exports Comma Separated Value (CSV) documents. Originally the driver only dealt with CSV but has been expanded to fixed field length and supports any separator character such as tabs and pipes.

CSV is a very common format. Most databases and database driven applications can import and export CSV data.

SYSTEM COMPATIBILITY

Niagara AX

This software will function on all Niagara AX – 3.n.nnn platforms.

Niagara 4

This software will function on Niagara 4 – 4.0.22.16 and higher platforms.

INSTALLATION

Niagara AX

Install csi3csv.jar on the computer where Niagara Workbench will be run. To install, place a copy of the file(s) in the modules directory of your Niagara AX installation. This is typically C:\Niagara\Niagara-3.n.nnn\modules.

Install the module(s) on the target station. Using a Niagara AX Workbench where the module has already been installed, connect to the station's platform service. Go to the Software Manager and install.

Niagara 4

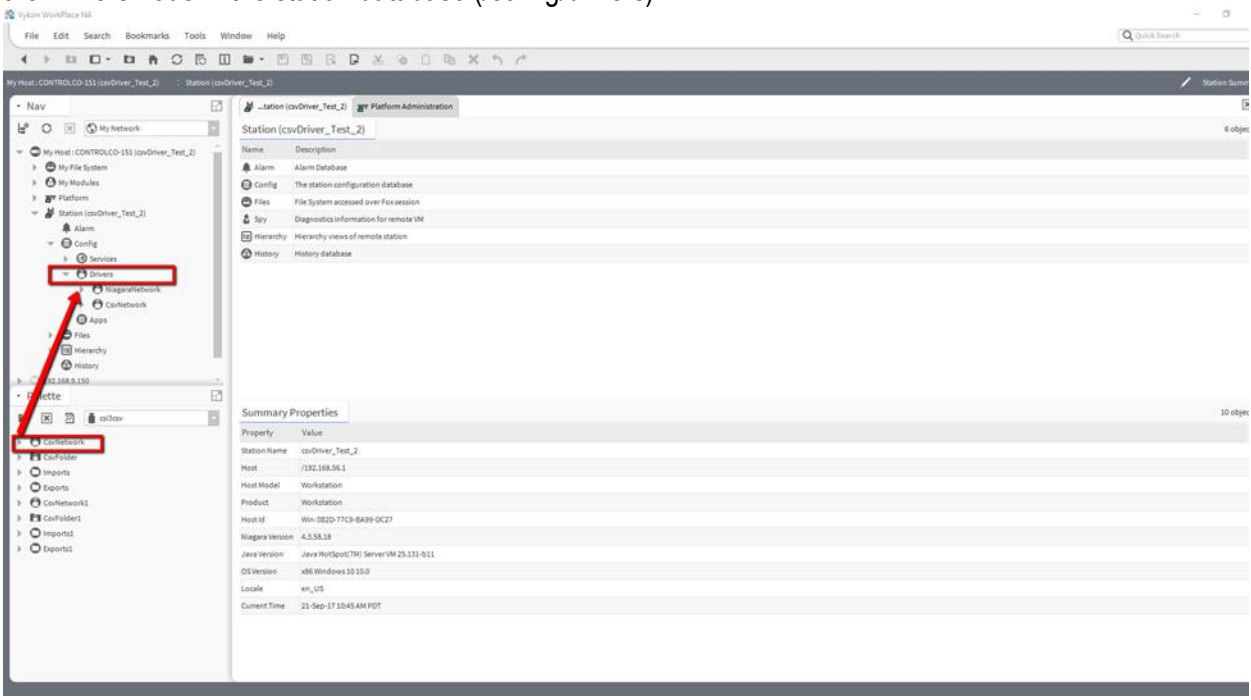
Install csi3csv-rt.jar and csi3csv-wb.jar on the computer where Niagara Workbench will be run. To install, place a copy of the files in the modules directory of your Niagara 4 installation. This is typically C:\Niagara\Niagara-4.n.nnn\modules.

Install the modules on the target station. Using a Niagara AX Workbench where the module has already been installed, connect to the station's platform service. Go to the Software Manager and install.

NETWORK INSTALLATION

Niagara AX and Niagara 4

Using the csi3csv palette in workbench, copy and paste (Or Drag) the CsvNetwork under the Drivers node in the station database (/config/drivers).



LICENSING

Licensing is managed on an object in your database. The licensing object is located on the property sheet of the CsvNetwork. It has the following properties.

- Product Code – Text automatically generated by the driver that is needed to generate a license key.
- Connections – The number of import or export objects.
- License Key – Where the key to validate the license must be entered.

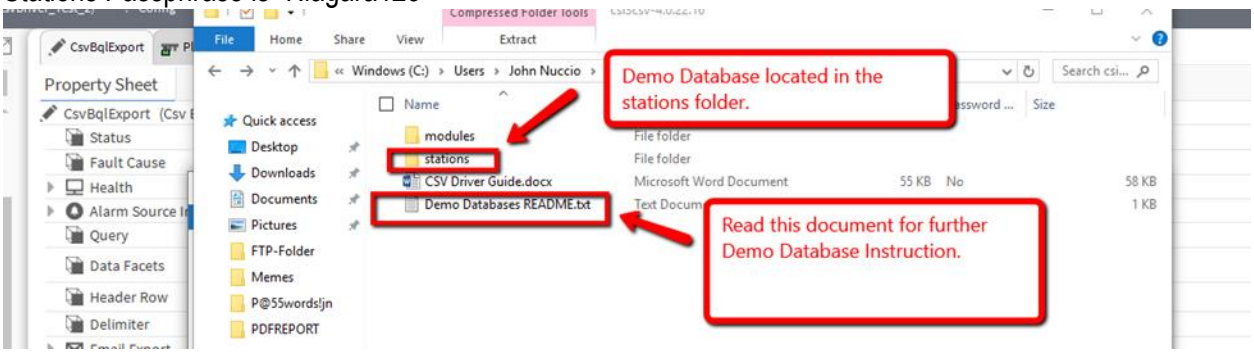
Set the number of connections you wish to purchase. Copy the value of the "Product Code" property that is automatically generated. You can highlight the value and copy it by pressing CTRL-C. Send the product code to your Kodaro representative. They will respond with a text string for you to enter in the "License Key" property.

You must restart the station after changing the "License Key".

The exact text and case of the product code and license key are critical. Do not send screen shots. Highlight the text, copy it using CTRL-C and paste into an email.

CSV DEMO, XML DEMO, AND DEMO STATION

The csi3csv Driver comes with a demo Station and database that is full of examples to get you familiar with the driver. The demo Station can be found in the Stations folder. Please note that the Stations Passphrase is “Niagara123”



The CsvDemo has the following properties.

- Requires the csi3csv module only.
- The user "admin" has no password.
- The station name must be CsvDemo.
- License Key – Where the key to validate the license must be entered.
- All objects are located under /Drivers/CsvNetwork.

To view the export webpages, visit.

- http://<host>/CsvPointExport
- http://<host>/CsvHistoryExport

The Xml Demo has the following properties.

- This requires both the csi3csv and csi3xml modules.
- The user "admin" has no password.
- This database only demonstrates how to convert XML to CSV. To understand other aspects of importing, use the CsvFileImport in the CsvDemo station.
- All objects are located under /Drivers/CsvNetwork.

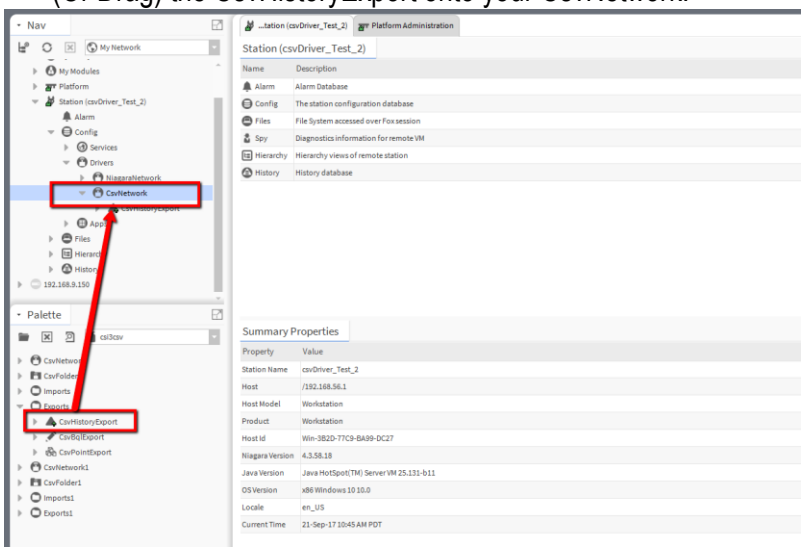
EXPORTING OVERVIEW

History Export

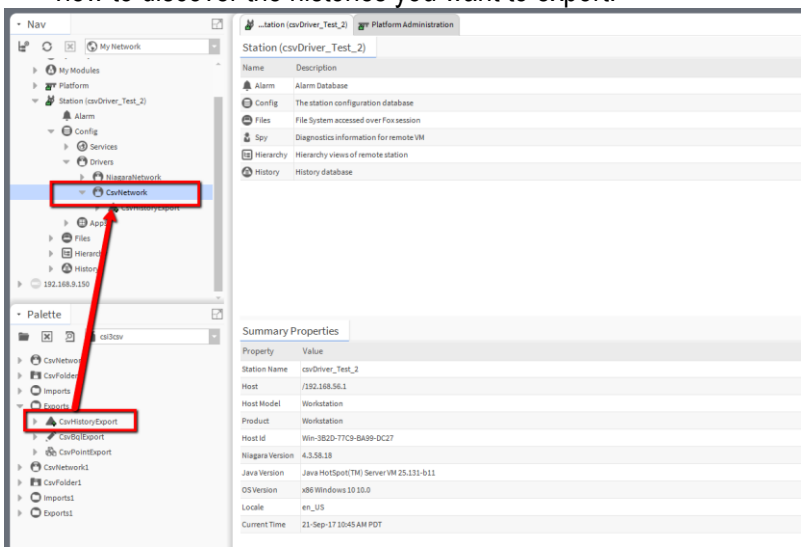
The CsvHistoryExport object converts histories into CSV documents and can send them or make them available through a variety of transports.

Setup Overview

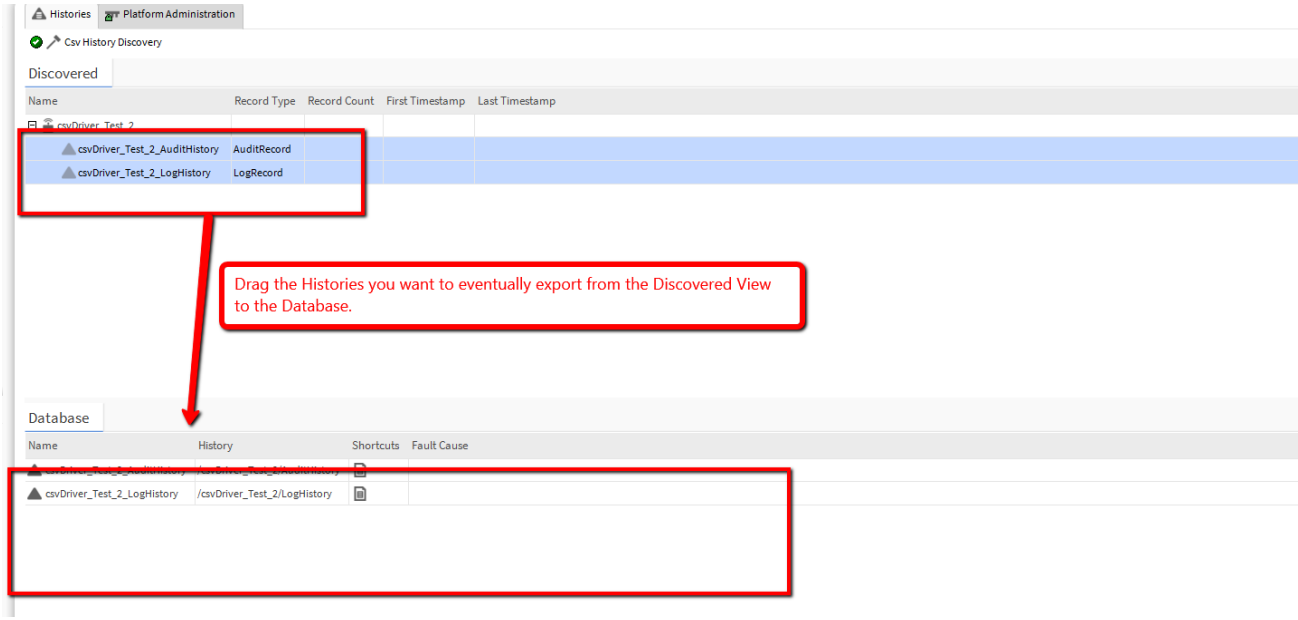
- Using the csi3csv palette in workbench, open the exports folder and then copy and paste (Or Drag) the CsvHistoryExport onto your CsvNetwork.



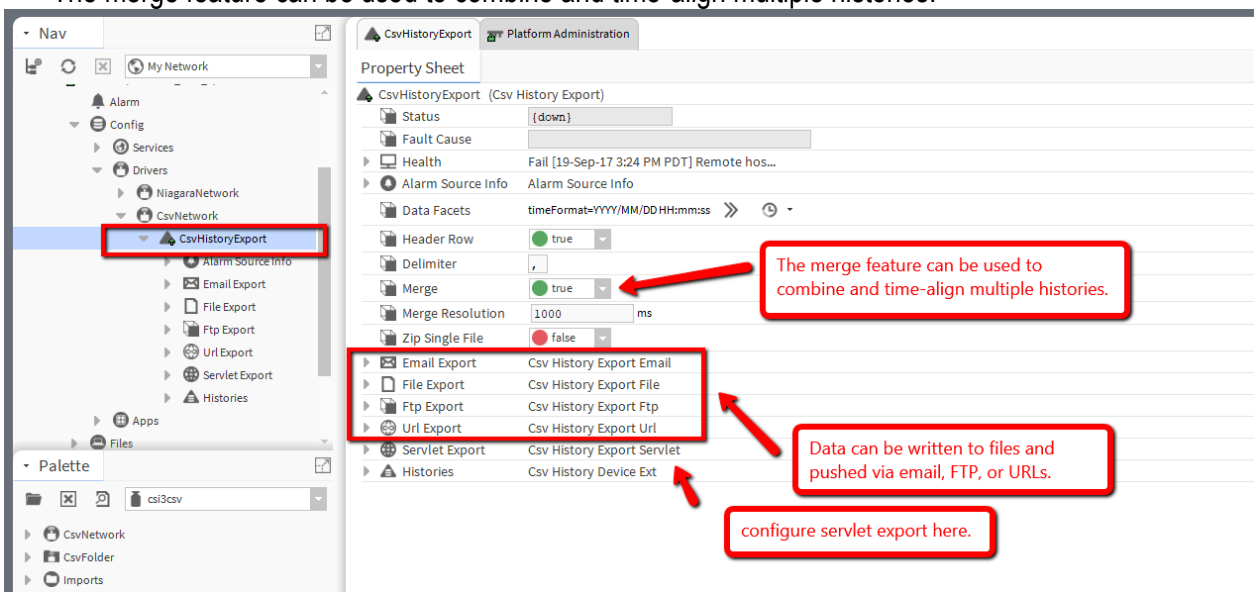
- Navigate in your CsvHistoryExport to Histories and then double click Histories to open the Csv Histories Manager view. Click Discover on the bottom of the Csv History Manager view to discover the histories you want to export.



- Highlight the Histories you want to eventually export and drag them into the Database view from the Discovered view.



- Enable and configure the servlet export and visit the servlet name URL to see what is available.
- Data can also be written to files and pushed via email, FTP, or URLs.
- The merge feature can be used to combine and time-align multiple histories.

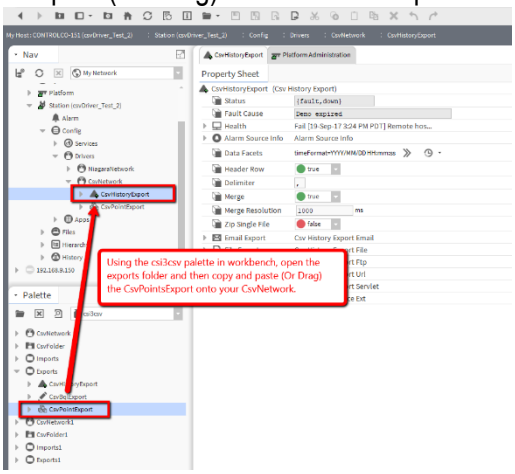


Point Export

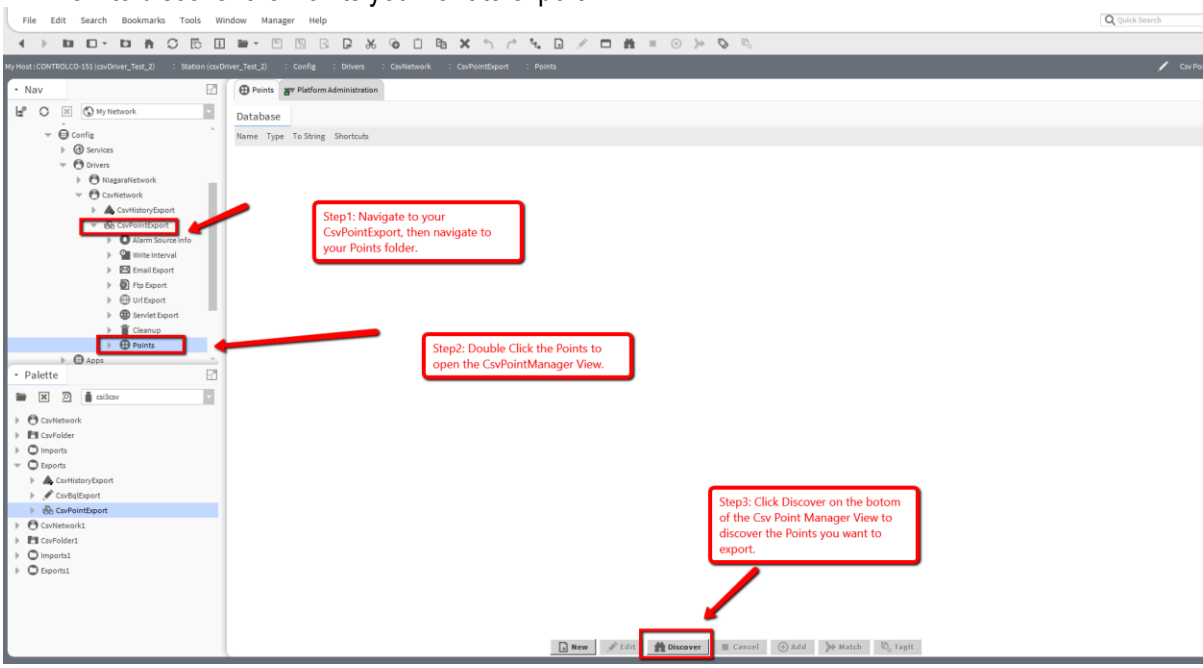
The CSVPointExport object takes a set of points and treats each as a column in a CSV document.

Setup Overview

- Using the csi3csv palette in workbench, open the exports folder and then copy and past (Or Drag) the CSVPointExport onto your CsvNetwork.



- Navigate in your CsvPointExport to Points and then double click Points to open the Csv PointManager view. Click Discover on the bottom of the Csv Point Manager view to discover the Points you want to export.



- Highlight the Points you want to eventually export and drag them into the Database view from the Discovered view.

- Set the directory and file properties. Consider encoding the time (see the time format section) into the file name so that you can create a file per day, week, month, etc.

- Click points to discover the points you want.

- Set the write interval property to append a row at a regular interval.
- Set the write cov property to write a row whenever a point changes.
- Enable and configure the servlet export and visit the servlet name URL to see what is available.
- Configure the cleanup object so that you don't run out of disk space.

IMPORTING OVERVIEW

Creating the Import Object

This driver is implemented as if it were a device driver. There are several types "devices" that pull documents and convert them into Niagara objects in your station:

- CsvFileImport
- CsvUrlImport
- CsvFtpImport
- FixedLengthFileImport
- FixedFieldUrlImport
- FixedFieldFtpImport

Setup Overview

- Set the *URL* or *File* properties on your device.
  - Ftp imports should only specify the host name. Do not include the schemes such as ftp: ftps: sftp:
- Define the file format:
  - On CSV import objects set the delimiter character if it isn't the default comma.
  - On Fixed Length import objects, set the column widths.
- Click on *columns* to discover and define the values you want to extract from the CSV.
- Set *Skip N Rows* on your import device object to skip leading non-data rows when polling the CSV.
- Click on *records* to configure how to control, trend and alarm data from CSV documents. You can create filters to conditionally control, trend and alarm.

- Set the *Poll Interval* on the import device object to being polling (importing the CSV).

Dynamic File Names and URLs

Both file and url properties both utilize Niagara BFormat syntax. You may be familiar with format syntax from animating widget properties on PX pages.

Often systems create CSV files in a path that indicate date information. The Niagara format makes this easy to achieve.

The following format string encodes the time:

The time is %time()%.

Which results in something like:

The time is 22-Jan-2009 05:06 PM PST.

You can access parts of the time such as:

The year is %time().year%.

Which results in something like:

The year is 2009.

The following data can be accessed from Niagara time. This is a subset of public API of javax.baja.sys.BAbsTime, more accessors can be found in the API docs for advanced usage.

Accessor	Description
day	The day of the month.
dayOfYear	The day of the year.
hour	0-23
millisecond	0-999
minute	0-59
month	January - December
month.monthOfYear	1-12
month.shortDisplayTag	Jan - Dec
nextDay	Same time, next day
nextMonth	Same day and time, next month
nextYear	Same month, day, and time, next year
prevDay	Same time, previous day
prevMonth	Same day and time, previous month
prevYear	Same month, day, and time, previous year
second	0-59
weekday	Sunday – Saturday
weekday.shortDisplayTag	Sun – Sat
year	Eg. 2009

The time can be more powerfully formatted using the facets property of the device object.

timeFormat – A special Niagara facet for converting Niagara time into a highly customizable string.

The default timeFormat in the device facets for example purposes only, it is MM-DD. Using it, the URL string:

http://www.example.com/%time()%/data.csv

Results in something like:

http://www.example.com/01-22/data.csv

The complete time format rules are discussed in another section.

Discovery and Column Creation

After setting up the import device object, the next step is to define how it will convert rows from a document into a Niagara object.

Double-click the import device's CvsColumnDeviceExt named columns, and press the discover button. The first eight rows (after the SkipNRows property value) are displayed. This should give a hint as to what each column represents.

The discovery view might be confusing because the columns are displayed as rows. This is because you will be converting CSV columns into properties on the CSV record objects.

Column Type

Columns from a document can be converted into Niagara numeric, boolean and string types. Timestamps are also supported.

Timestamp

Timestamps usually play an important part of tabular data. Much can be done with timestamps in this driver but there is one special timestamp column that needs to be pointed out.

If a column is named timestamp, it must be a CsvTimestampColumn. It will set the frozen timestamp property of all CSV record objects. If there is no column named timestamp, the timestamp for all rows of an imported CSV document will be the time the row was parsed from the imported CSV document. Because Java only has 100ms time resolution probably all rows will have the same timestamp.

Skipping Rows

Once your columns are defined, you must tell the import device object how many rows to skip before processing actual data. For example, given this CSV document,

```
Timestamp, Name, Value
2009-01-22 03:23:01, Temperature, 45.01
2009-01-22 03:24:01, Temperature, 45.02
```

The first row is called a header row and it must be skipped, SkipNRows should be 1.

Do not set this property until after discovery because column discovery also skips these rows. However, this may also help discover very unusual documents.

Working with Records

The columns determine the structure of CSV record objects. There are multiple CSV record objects but they all share the following:

- They have a frozen timestamp property which is either the time the row was read in the Niagara station, or set by a column named timestamp.
- For every defined column, they have corresponding properties.
- The column properties are linkable and can be used for control.
- CsvHistoryExt can be added to create histories.
- CsvAlarmExt can be added to create alarms.
- Modifying columns will modify (update) all CsvRecords under the same import device.

CsvRecordDeviceExt

All CSV record objects can be found under the CsvRecordDeviceExt named records.

The CsvRecordDeviceExt has a frozen record object named source. All rows from an imported CSV document are copied into the properties of this object. When a document is processed, the rows are processed so quickly that you will not see this. However, if a CsvHistoryExt were trending a column, every CSV row would have a Niagara history row.

Linking

All properties of a CsvRecord can be linked. When columns are modified, all CsvRecords under an import device are kept in sync with the defined columns. This could break prior links, be careful.

Histories

To trend values that "pass-thru" a CsvRecord, add a CsvHistoryExt for the desired columns. This history extension will allow you to trend COV, time interval or a combination of both.

Alarms

To create alarms when CsvRecords are changed, add CsvAlarmExts to CsvRecords. Unlike CsvHistoryExt, only one CsvAlarmExt is necessary.

Triggered Boolean

If all you care about is when imported CsvRecord passes some filter, link the record accepted topic of a filter to a CsvTriggeredBoolean. It is a configurable momentary.

Working with Filters

Filters are CSV record objects, but they do not copy incoming row values unless their filter rule is passed. This enables conditional control, trending and alarming based upon values in a row. There are numeric, boolean, string and timestamp filters.

When an incoming record meets the criteria of a filter, the values are copied into filter's corresponding column properties. The filter also fires a topic named record accepted which can be linked to actions, such as record in on additional filters and activate on the CsvTriggeredBoolean.

FORMATTING TIME

There are several places in this driver where time needs to be formatted into a String and vice versa. This can be achieved with time formats.

Time formats are special pattern strings. Within pattern strings, letters from "A" to "Z" and from "a" to "z" are interpreted as pattern letters representing the components of a date and time. All other characters are not interpreted; they're ignored.

The following are the available pattern letters:

Pattern	Description
YY	Two digit year
YYYY	Four digit year
M	One digit month
MM	Two digit month
MMM	Short tag month
D	One digit day of month
DD	Two digit day of month
h	One digit 12 hour

hh	Two digit 12 hour
H	One digit 24 hour
HH	Two digit 24 hour
m	One digit minutes.
mm	Two digit minutes
a	AM / PM
ss	Two digits seconds (and optionally milliseconds)
Z	Timezone

Examples:

Time String	**Pattern**
2004-01-02 03:04 AM	YYYY-MM-DD hh:mm:ss a
04-1-2 3:04 AM	YY-M-D h:m:s a
2-Jan-04 3:04	D-MMM-Y H:m:s
02-Jan-2004t3:04	DD-MMM-YYYYtH:m:s
1983-01-02	YYYY-MM-DD
2000\\01\\02	YYYY\\MM\\DD
16/1/2	YY/M/D
12:00AM 29-Apr-04	hh:mm:ssa DD-MMM-YY
0:00 29-Apr-04	H:mm:ss DD-MMM-YY
8:44:01PM 7-Jun-04	h:mm:ssa D-MMM-YY
7-Jun-04 20_44_01PM	7-Jun-04 20_44_01PM
8:44:01.234PM 7-Jun-04	h:mm:ssa D-MMM-YY

TIME RANGES

Time ranges can be specified in the URL of a CsvHistoryExport servlet. The following time ranges are supported:

Time Range String	Notes
today	?timeRange=today
yesterday	?timeRange=yesterday
thisWeek	?timeRange=thisWeek
lastWeek	?timeRange=lastWeek
thisMonth	?timeRange=thisMonth
lastMonth	?timeRange=lastMonth
thisYear	?timeRange=thisYear
lastYear	?timeRange=lastYear
current<N><Units>	The period including the current time. N is a positive number. Units are seconds, minutes, hours, days, weeks, months or years. ?timeRange=current3months
previous<N><Units	The period excluding the current time. N is a positive number. Units are seconds, minutes, hours, days, weeks, months or years. ?timeRange=previous3months
<startTime>;<endTime>	The time strings must conform to BAbsTime.decodeFromString rules. This text must strictly conform to the ISO 8601 standard format of "yyyy-mm-ddThh:mm:ss.mmm[+/-]hh:mm".
all	The entire history.

CSI3CSV ORD SCHEME

Import devices can be bound to Niagara tables in graphics. The ord scheme to do so is “csi3csv” and an example ord you could bind with is:

station:|slot:/Drivers/CsvNetwork/CsvImportUrl|csi3csv:

COMPONENT GUIDE

CsvNetwork

This is the root component of the driver. There can only be one CsvNetwork object in a station database.

Properties

- License – See the licensing section of this document.
- Thread Pool – You should allocate at least one threat per import/export object. However, if you want to throttle imports/exports, you can use fewer threads.

Actions

- Enable – Enables the network and any device objects that have enabled property set to true.
- Disable – Disables the entire network.

CsvFolder

Used to organized device objects.

CsvPointExport

A “device” level object in the Niagara driver architecture. This object creates Csv documents from a set of points.

Each point represents a column. The CsvPointExport object appends rows to an output file either at a specified interval and/or on changes of value of any of the points. This file can then be pushed or pulled by a variety of export methods.

Properties

- Directory – A Niagara Format string that must resolve to a directory to create the CSV documents in. This directory should only contain documents created by this device.
- File Name – A Niagara Format string that once resolved creates the file name. The name is resolved every time a write occurs.
- Current File – The resolved file path most recently written to.
- Last File – The resolved file path prior to the current file.
- File Facets – Facets used to resolve the directory and file name properties.
- Data Facets – The default facets used to encode the point values. The point facets of each point override these.

- Append File – Whether or not to append to the current file or clear the file each time a write occurs.
- Header Row – Whether or not to write a header row at the beginning of the file.
- Delimiter – The character used to separate column values in each row. Must be a single visible character.
- Write Interval – Can be used to write values at a regular interval.
- Timestamp Header – The header string for the timestamp column.
- Write Cov – Whether or not to write a row whenever any of the points change value.
- Show Status – When true, the status of each point will be included in each column. The status will be in the standard Niagara style {down,fault,alarmIf commas are present, the column will be wrapped in quotes.
- Points – The set of points to export.
- Email Export – This can email files at a configurable interval.
- File Export – This can write files to the file system at a configurable interval.
- FTP Export – This can push files at a configurable interval.
- Url Export – This can push files at a configurable interval.
- Servlet – The makes files accessible to web clients.
- Cleanup – Can be used to delete old files.

Actions

- Enable – Enables the export.
- Disable – Disables the export.
- Write – Write a row of values. This can be linked for more advanced writing.

Email Export

This emails a single file or a zip of all files at a configurable interval.

Properties

- Enabled – Enables/disables the auto export mechanism.
- File – Which file to send. Current file, previous file, or zip all files in the directory.

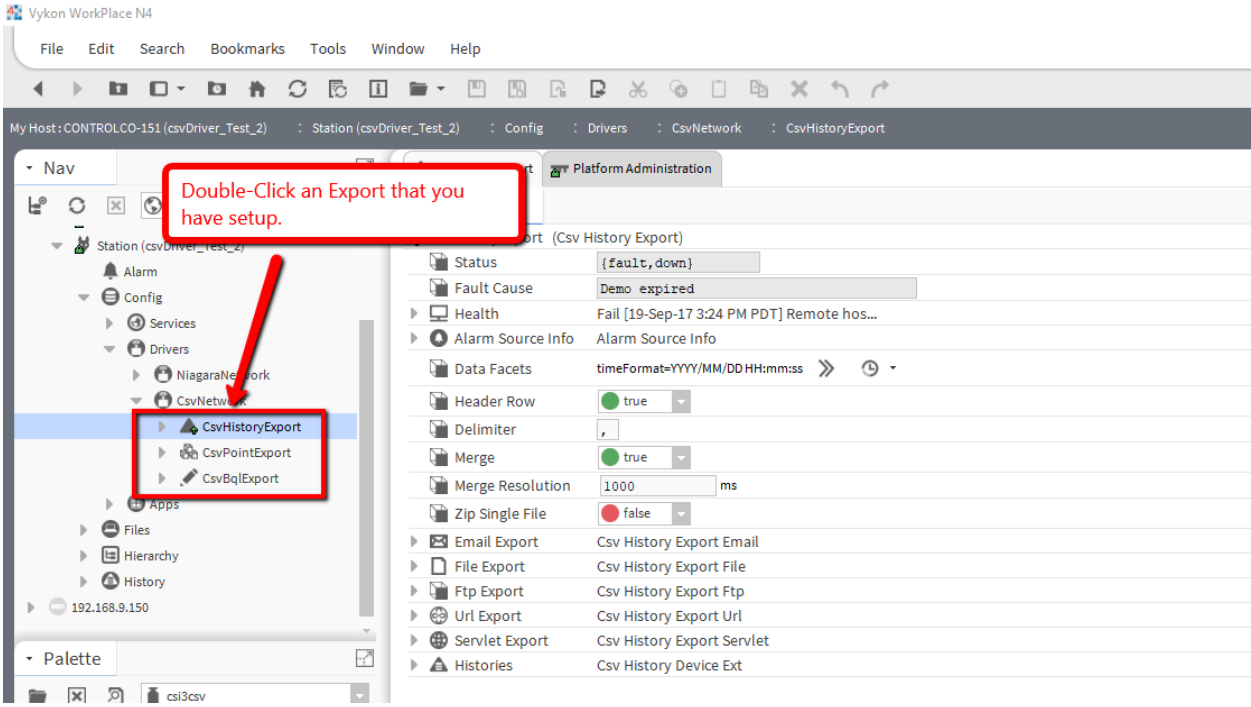
- Send Previous File Cov – If true, send the Previous File whenever it changes.
- Interval – Can be used to send emails at a regular interval.
- Recipients – Semi-colon separated list of email addresses.
- From – Reply to address
- Subject – Format for creating the email subject.
- Subject Facets – Used to resolve the subject.
- Clean After Export – If export is successful (queued in the email service) then run cleanup.

Actions

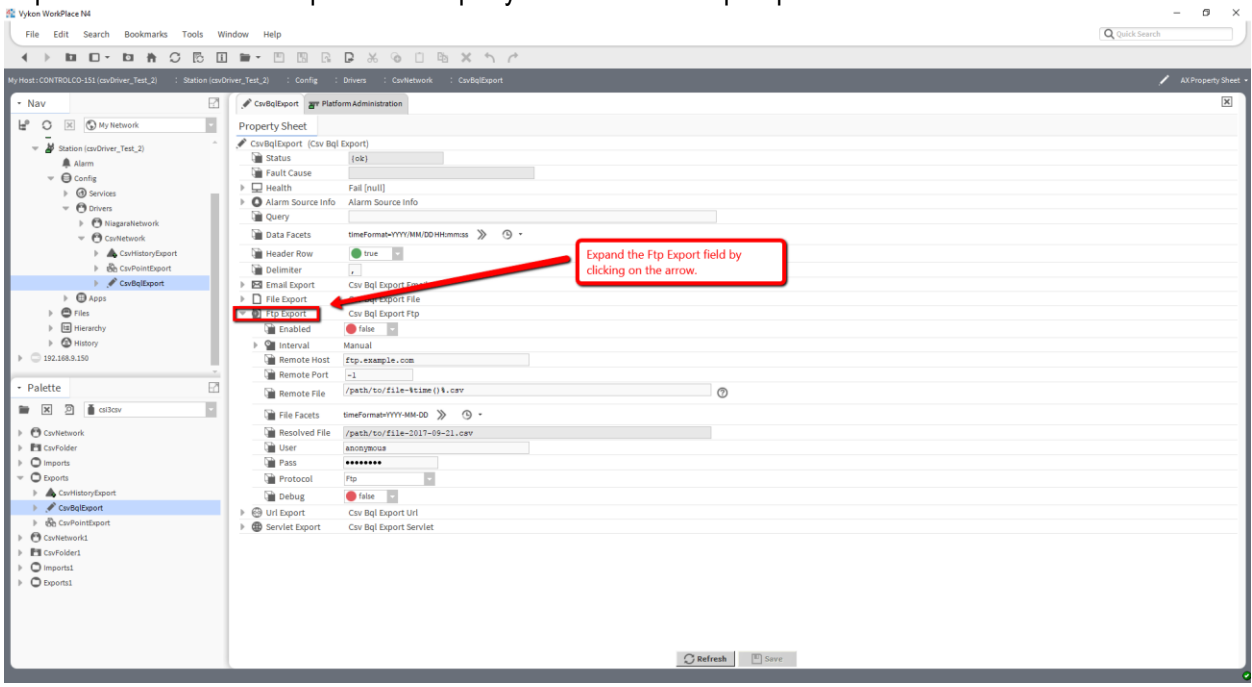
- Send – Can be manually invoked or linked.

Ftp Export

This pushes a single file or a zip of all files at a configurable interval. To create an Ftp Export, double click on a CsvHistoryExport, CsvPointExport, or CsvBqlExport that has been added to your CsvNetwork. This will open the AX Property Sheet for the double-clicked Export.



Expand the field on the Exports AX Property Sheet labeled “Ftp Export.”



Properties

- Enabled – Enables/disables the auto export mechanism.
- File – Which file to send. Current file, previous file, or zip all files in the directory.
- Send Previous File Cov – If true, send the Previous File whenever it changes.
- Interval – Can be used to send emails at a regular interval.
- Remote Host – FTP server. Can be an IP address or host name. Should not be a URL, there should be no scheme such as ftp:
- Remote Port – Leave at -1 for the default FTP port (21).
- Remote File – A Niagara Format representing the file path on the remote server.
- File Facets – Used to resolve the Remote File.
- User – User name for authentication.
- Pass – Password for authentication
- Protocol – Which FTP protocol to use.

- Connect – Mode Active or Passive. If one mode isn't working, try the other mode.
- Clean After Export – If export is successful (successful transmission and if HTTP, a response code in the 200s) then run cleanup.

Actions

- Send – Can be manually invoked or linked.

URL Export

This pushes a single file or a zip of all files at a configurable interval.

Properties

- Enabled – Enables/disables the auto export mechanism.
- File – Which file to send. Current file, previous file, or zip all files in the directory.
- Send Previous File Cov – If true, send the Previous File whenever it changes.
- Interval – Can be used to send emails at a regular interval.
- Url – A Niagara Format string that must resolve to valid URL.
- Url Facets – Used when resolving the URL.
- Http User – User name for HTTP authentication only. For FTP authentication, encode the username and password into the URL.
- Http Pass – Password for HTTP authentication only.
- Http Proxy Host – For passing through proxies.
- Http Proxy Proxy – For passing through proxies.
- Http Proxy User – For passing through proxies.
- Http Proxy Pass – For passing through proxies.
- Clean After Export – If export is successful (successful transmission and if HTTP, a response code in the 200s) then run cleanup.

Actions

- Send – Can be manually invoked or linked.

Servlet Export

This enables the pulling of CSV files from web clients.

Properties

- Enabled – Enables/disables web access.
- Servlet Name – the resolved Servlet Name Format. This translates to the URL for this export servlet. If the servlet name is "csv", then visiting http://<host>/csv will list the download options.
- Servlet Name Format – Niagara Format object for generating the servlet name.
- Content Type – Sometimes "text/plain" maybe desired for browser access.
- Zip File Name – The file name to download all files in the directory as a single zip. If the servlet name is "csv" and the zip file name is "directory.zip", the URL to download the file would be http://<host>/csv/directory.zip

Actions

- Configure Guest Access – Configure guest access for the servlet.
- Update Servlet Name – Resolves the servlet name format.

Cleanup

This enables the pulling of CSV files from web clients.

Properties

- Enabled – Enables/disables auto cleanup.
- Max File – How many files to retain in the directory.
- Interval – When to automatically clean the directory.

Actions

- Clean – Can be manually invoked or linked.

CsvBqlExport

A "device" level object in the Niagara driver architecture. This object creates CSV documents from Niagara ords.

Properties

- Query – Any Niagara Ord can be used, but was designed primarily for BQL.
- Data Facets – The default facets used to encode the point values. The point facets of each point override these.
- Header Row – Whether or not to write a header row at the beginning of the file.
- Delimiter – The character used to separate column values in each row. Must be a single visible character.

Email Export

This emails a file of all histories at a configurable interval.

Properties

- Enabled – Enables/disables the auto export mechanism.
- Interval – Can be used to send emails at a regular interval.
- From – Single from address.
- Recipients – Supports multiple email addresses.
- Subject – Format for creating the email subject.
- Subject Facets – Used to format the subject.
- Attachment – Niagara Format representing the file name of the attachment.
- Attachment Facets – Used to resolve the Attachment.

Actions

- Send – Can be manually invoked or linked.

File Export

This creates a file of all histories on the local file system (will probably not work with a distributed file system), at a configurable level.

Properties

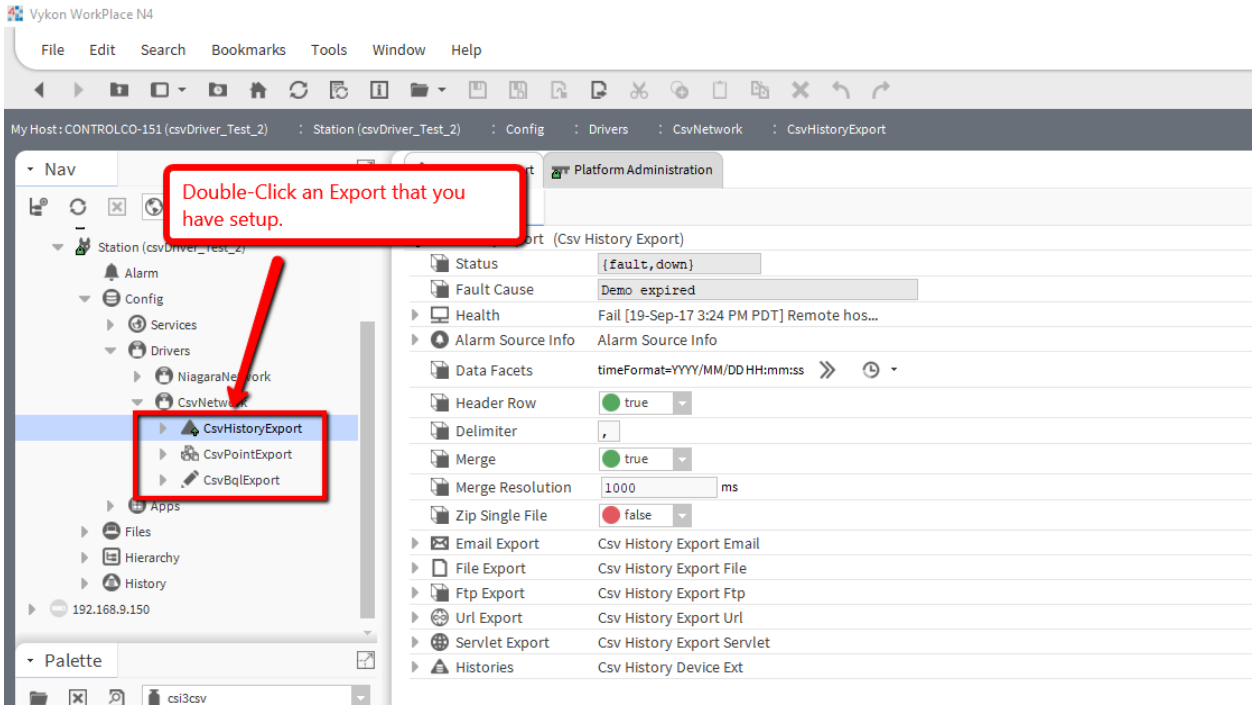
- Enabled – Enables/disables the auto export mechanism.
- Interval – Can be used to send emails at a regular interval.
- File – A Niagara Format string that must resolve to valid file ORD.
- File Facets – Facets used when resolving the file.

Actions

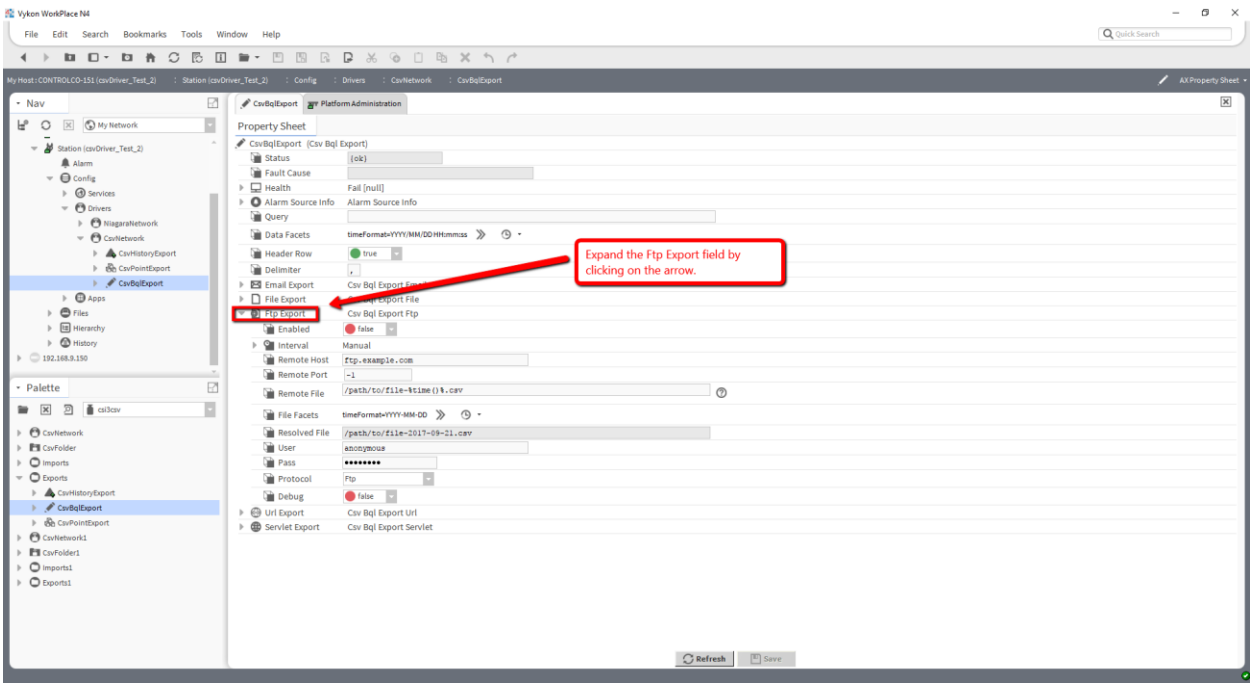
- Export – Can be manually invoked or linked.

Ftp Export

This pushes a single file or a zip of all files at a configurable interval. To create an Ftp Export, double click on a CsvHistoryExport, CsvPointExport, or CsvBqlExport that has been added to your CsvNetwork, this will open the AX Property Sheet for the double-clicked Export.



Expand the field on the Exports AX Property Sheet labeled “Ftp Export.”



Properties

- Enabled – Enables/disables the auto export mechanism.
- File – Which file to send. Current file, previous file, or zip all files in the directory.

- Send Previous File Cov – If true, send the Previous File whenever it changes.
- Interval – Can be used to send emails at a regular interval.
- Remote Host – FTP server. Can be an IP address or host name. Should not be a URL, there should be no scheme such as ftp:
- Remote Port – Leave at -1 for the default FTP port (21).
- Remote File – A Niagara Format representing the file path on the remote server.
- File Facets – Used to resolve the Remote File.
- User – User name for authentication.
- Pass – Password for authentication.
- Protocol – Which FTP protocol to use:
- Connect – Mode Active or Passive. If it's not working, try the other mode.
- Clean After Export – If export is successful (successful transmission and if HTTP, a response code in the 200s) then run cleanup.

Actions

- Send – Can be manually invoked or linked.

URL Export

This pushes a file of all histories at a configurable interval. This uses Java's URL class which supports multiple schemes besides HTTP. HTTP is the only supported scheme by this driver, but others are available if you are familiar with Java's URL class and protocol handlers.

Properties

- Enabled – Enables/disables the auto export mechanism.
- Time Range – Defines which data to export.
- Url – A Niagara Format string that must resolve to valid URL.
- Url Facets – Used when resolving the URL.
- Http User – User name for HTTP authentication only. For FTP authentication, encode the username and password into the URL.
- Http Pass – Password for HTTP authentication only.

- Http Proxy Host – For passing through proxies.
- Http Proxy Proxy – For passing through proxies.
- Http Proxy User – For passing through proxies.
- Http Proxy Pass – For passing through proxies.

Actions

- Export – Can be manually invoked or linked.

Servlet Export

This enables the pulling of CSV files from web clients. Visit the URL defined by the servlet name property to see the download options.

Time ranges can be specified as a query parameter in the URL. The query parameter name is “timeRange”. The value is discussed in another section of this document. The web page designated by the servlet name has several examples of this.

Properties

- Enabled – Enables/disables web access.
- Servlet Name – the resolved Servlet Name Format. This translates to the URL for this export servlet. If the servlet name is “csv”, then visiting http://<host>/csv will list the download options.
- Servlet Name Format – Niagara Format object for generating the servlet name.
- Content Type – Sometimes “text/plain” maybe desired for browser access. This is applicable only to individual histories. Zip files will be “application/zip”.

Actions

- Update Servlet Name – Resolves the servlet name format.

CsvHistoryExport

A “device” level object in the Niagara driver architecture. This object creates CSV documents from Niagara histories.

Properties

- Data Facets – The default facets used to encode the point values. The point facets of each point override these.
- Header Row – Whether or not to write a header row at the beginning of the file.

- Delimiter – The character used to separate column values in each row. Must be a single visible character.
- Merge – If true, all histories are merged into a single CSV document for the email, URL, FTP and file exports.
- Merge Resolution – Defines how to round the time stamp of each row, but it does not rollup multiple rows. The intent is to align data that is very close in time, but not at the exact moment. Should be smaller than the smallest interval.
- Zip Single File – When only one history is being exported, it can be either as a text file, or as a zip.
- Email Export – This can email files at a configurable interval.
- File Export – This can write files to the file system at a configurable interval.
- FTP Export – This can push files at a configurable interval.
- Url Export – This can push files at a configurable interval.
- Servlet – The makes files accessible to web clients.
- Cleanup – Can be used to delete old files.

Email Export

This emails a file of all histories at a configurable interval.

Properties

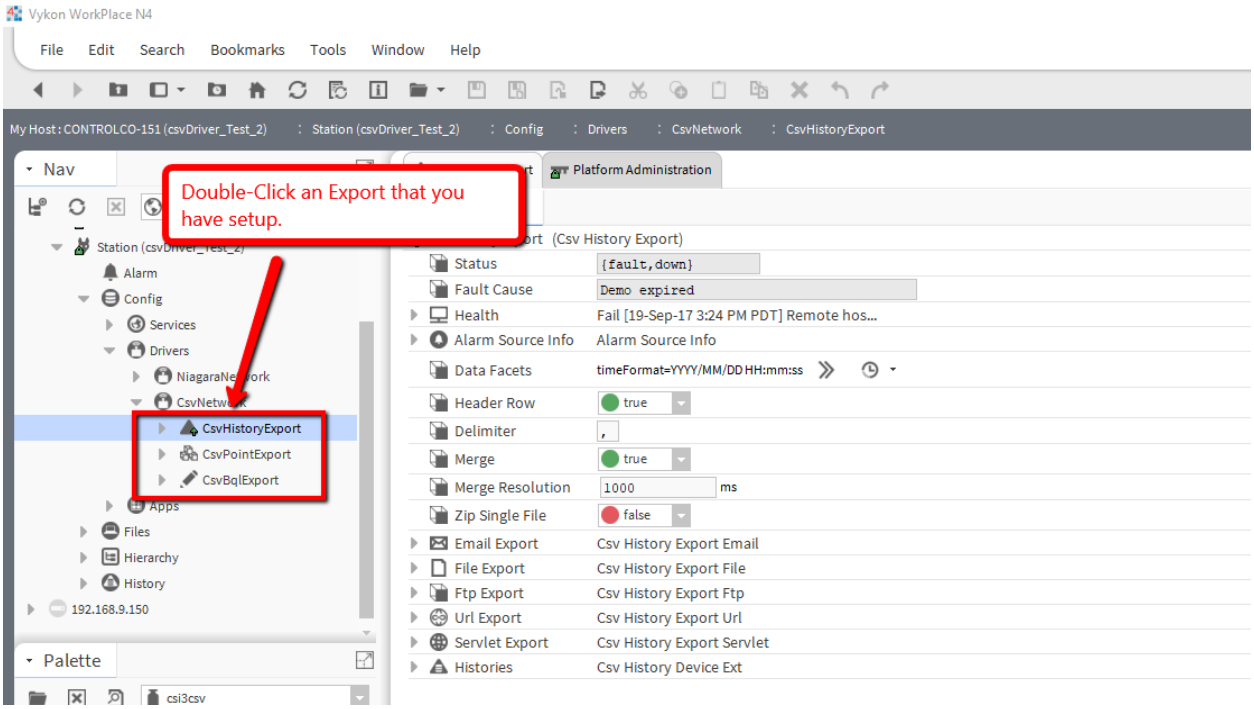
- Enabled – Enables/disables the auto export mechanism.
- Time Range – Defines which data to export.
- Interval – Can be used to send emails at a regular interval.
- Recipients – Semi-colon separated list of email addresses.
- Subject – Format for creating the email subject.
- Subject Facets – Used to format the subject.
- Attachment – Niagara Format representing the file name of the attachment.
- Attachment Facets – Used to resolve the Attachment.

Actions

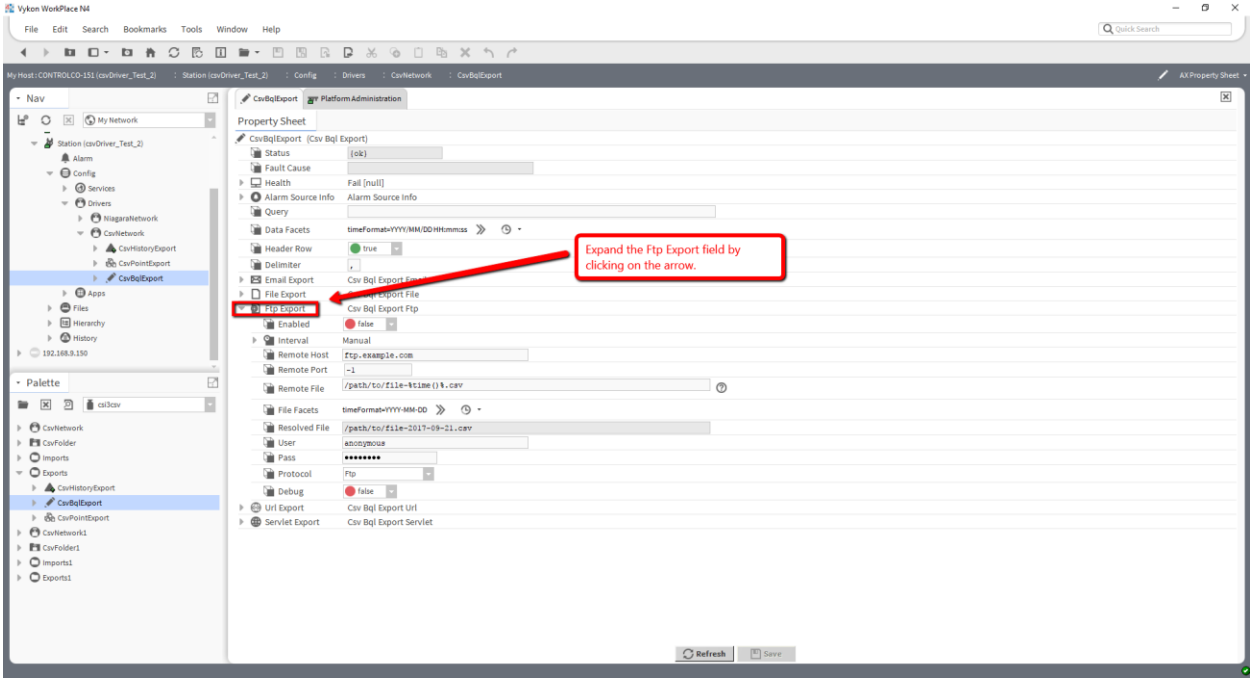
- Send – Can be manually invoked or linked.

Ftp Export

This pushes a single file or a zip of all files at a configurable interval. To create an Ftp Export, double click on a CsvHistoryExport, CsvPointExport, or CsvBqlExport that has been added to your CsvNetwork, this will open the AX Property Sheet for the double-clicked Export.



Expand the field on the Exports AX Property Sheet labeled “Ftp Export.”



Properties

- Enabled – Enables/disables the auto export mechanism.
- File – Which file to send. Current file, previous file, or zip all files in the directory.
- Send Previous File Cov – If true, send the Previous File whenever it changes.
- Interval – Can be used to send emails at a regular interval.
- Remote Host – FTP server. Can be an IP address or host name. Should not be a URL, there should be no scheme such as ftp:
- Remote Port – Leave at -1 for the default FTP port (21).
- Remote File – A Niagara Format representing the file path on the remote server.
- File Facets – Used to resolve the Remote File.
- User – User name for authentication.
- Pass – Password for authentication.
- Protocol – Which FTP protocol to use:
- Connect – Mode Active or Passive. If it’s not working, try the other mode.

- Clean After Export – If export is successful (successful transmission and if HTTP, a response code in the 200s) then run cleanup.

Actions

- Send – Can be manually invoked or linked.

Url Export

This pushes a file of all histories at a configurable interval. This uses Java's URL class which supports multiple schemes beyone HTTP. HTTP is the only supported scheme by this driver, but others are available if you are familiar with Java's URL class and protocol handlers.

Properties

- Enabled – Enables/disables the auto export mechanism.
- Time Range – Defines which data to export.
- Interval – Can be used to send emails at a regular interval.
- Url – A Niagara Format string that must resolve to valid URL.
- Url Facets – Used when resolving the URL.
- Http User – User name for HTTP authentication only. For FTP authentication, encode the username and password into the URL.
- Http Pass – Password for HTTP authentication only.
- Http Proxy Host – For passing through proxies.
- Http Proxy Proxy – For passing through proxies.
- Http Proxy User – For passing through proxies.
- Http Proxy Pass – For passing through proxies.

Actions

- Export – Can be manually invoked or linked.

Servlet Export

This enables the pulling of CSV files from web clients. Visit the URL defined by the servlet name property to see the download options.

Time ranges can be specified as a query parameter in the URL. The query parameter name is "timeRange". The value is discussed in another section of this document. The web page designated by the servlet name has several examples of this.

Properties

- Enabled – Enables/disables web access.
- Servlet Name – the resolved Servlet Name Format. This translates to the URL for this export servlet. If the servlet name is “csv”, then visiting http://<host>/csv will list the download options.
- Servlet Name Format – Niagara Format object for generating the servlet name.
- Time Range – The default time range when one is not specified in the query string.
- Content Type – Sometimes “text/plain” maybe desired for browser access. This is applicable only to individual histories. Zip files will be “application/zip”.
- File Name Facets – Used to resolve file names.
- Merged File Name – Niagara Format representing the name for the merge of all histories.
- Zip File Name – Niagara Format representing the name for the zip of all histories.
- Zip File Name Facets – Used to resolve the Zip File Name.

Actions

- Update Servlet Name – Resolves the servlet name format.

CsvFileImport

A “device” level object in the Niagara driver architecture. This object reads CSV documents from the file system.

Properties

- Enabled – Enables/disables the auto poll mechanism.
- Encoding – Character set of the input file.
- Facets – Used to format the file property.
- Skip First N Rows – If the CSV being imported has header rows, set this to skip those rows after the columns have been discovered and defined.
- Include Rows – After the first N rows are skipped, this determines which rows are converted into records.
- Delimiter – Allows you to change the column delimiter. Must be a single visible character.
- Poll Interval – When to import the CSV.

- Debug – Will print information during the processing of a CSV.
- Columns – Used to define the record structure of Last Record.
- Records – Where rows from the import CSV which conform to the Skip First N Rows and Include Rows rules are sent.
- File – A Niagara Format string that must resolve to a valid file Ord. The facets property will be used to resolve the format string.
- Resolved File – Shows how the file string was last resolved.
- Delete File – If true, deletes the file after processing it.
- Fail If Not Found – Whether or not to go into a fault condition if the file is not found.

Actions

- Enable – Enables the import.
- Disable – Disables the import.
- Poll – Processes the CSV.

CsvFtpImport

A “device” level object in the Niagara driver architecture. This object reads CSV documents from FTP servers.

Properties

- Enabled – Enables/disables the auto poll mechanism.
- Encoding – Character set of the input file.
- Facets – Used to format the File format string.
- Skip First N Rows – If the CSV being imported has header rows, set this to skip those rows after the columns have been discovered and defined.
- Include Rows – After the first N rows are skipped, this determines which rows are converted into records.
- Delimiter – Allows you to change the column delimiter. Must be a single visible character.
- Poll Interval – When to import the CSV.

- Debug – Will print information during the processing of a CSV.
- Columns – Used to define the record structure of Last Record.
- Records – Where rows from the import CSV which conform to the Skip First N Rows and Include Rows rules are sent.
- Remote Host – FTP server. Can be IP address or host name. Should not be a URL, there should be no scheme such as ftp:
- Remote Port – Leave at -1 for the default FTP port (21).
- Remote File – A Niagara Format representing the file path on the remote server.
- Resolved File – Shows how the File format was last resolved.
- User – User name for authentication.
- Pass – Password for authentication.
- Protocol – Which FTP protocol to use.
- Connect Mode – If it's not working, try changing this.

Actions

- Enable – Enables the import.
- Disable – Disables the import.
- Poll – Processes the CSV.

CsvUrlImport

A "device" level object in the Niagara driver architecture. This object reads CSV documents from URLs.

Properties

- Enabled – Enables/disables the auto poll mechanism.
- Encoding – Character set of the input file.
- Facets – Used to format the File format string.
- Skip First N Rows – If the CSV being imported has header rows, set this to skip those rows after the columns have been discovered and defined.

- Include Rows – After the first N rows are skipped, this determines which rows are converted into records.
- Delimiter – Allows you to change the column delimiter. Must be a single visible character.
- Poll Interval – When to import the CSV.
- Debug – Will print information during the processing of a CSV.
- Columns – Used to define the record structure of Last Record.
- Records – Where rows from the import CSV which conform to the Skip First N Rows and Include Rows rules are sent.
- URL – A Niagara Format string that must resolve to valid URL. The facets property will be used to resolve the format string.
- Resolved URL – Shows how the file string was last resolved.
- Http User – User name for HTTP authentication only. For FTP authentication, encode the username and password into the URL.
- Http Pass – Password for HTTP authentication only.

Actions

- Enable – Enables the import.
- Disable – Disables the import.
- Poll – Processes the CSV.

FixedLengthFilmeImport

A “device” level object in the Niagara driver architecture. This object reads field length documents from the file system.

Properties

- Enabled – Enables/disables the auto poll mechanism.
- Encoding – Character set of the input file.
- Facets – Used to format the file property.
- Skip First N Rows – If the CSV being imported has header rows, set this to skip those rows after the columns have been discovered and defined.

- Include Rows – After the first N rows are skipped, this determines which rows are converted into records.
- Poll Interval – When to import the CSV.
- Debug – Will print information during the processing of a CSV.
- Columns – Used to define the record structure of Last Record.
- Records – Where rows from the import CSV which conform to the Skip First N Rows and Include Rows rules are sent.
- Column Widths – A comma delimited set of integers that define the column lengths of the input file.
- File – A Niagara Format string that must resolve to a valid file Ord. The facets property will be used to resolve the format string.
- Resolved File – Shows how the file string was last resolved.
- Delete File – If true, deletes the file after processing it.
- Fail If Not Found – Whether or not to go into a fault condition if the file is not found.

Actions

- Enable – Enables the import.
- Disable – Disables the import.
- Poll – Processes the CSV.

FixedLengthFtpImport

A “device” level object in the Niagara driver architecture. This object reads field length documents from URLs.

Properties

- Encoding – Character set of the input file.
- Facets – Used to format the File format string.
- Skip First N Rows – If the CSV being imported has header rows, set this to skip those rows after the columns have been discovered and defined.
- Include Rows – After the first N rows are skipped, this determines which rows are converted into records.

- Poll Interval – When to import the CSV.
- Debug – Will print information during the processing of a CSV.
- Columns – Used to define the record structure of Last Record.
- Records – Where rows from the import CSV which conform to the Skip First N Rows and Include Rows rules are sent.
- Column Widths – A comma delimited set of integers that define the column lengths of the input file.
- Remote Host – FTP server. Can be IP address or host name.
- Remote Port – Leave at -1 for the default FTP port (21).
- Remote File – A Niagara Format representing the file path on the remote server.
- Resolved File – Shows how the File format was last resolved.
- User – User name for authentication.
- Pass – Password for authentication.

Actions

- Enable – Enables the import.
- Disable – Disables the import.
- Poll – Processes the CSV.

FixedLengthUrlImport

A “device” level object in the Niagara driver architecture. This object reads field length documents from URLs.

Properties

- Encoding – Character set of the input file.
- Facets – Used to format the File format string.
- Skip First N Rows – If the CSV being imported has header rows, set this to skip those rows after the columns have been discovered and defined.
- Include Rows – After the first N rows are skipped, this determines which rows are converted into records.

- Poll Interval – When to import the CSV.
- Debug – Will print information during the processing of a CSV.
- Columns – Used to define the record structure of Last Record.
- Records – Where rows from the import CSV which conform to the Skip First N Rows and Include Rows rules are sent.
- Column Widths – A comma delimited set of integers that define the column lengths of the input file.
- URL – A Niagara Format string that must resolve to valid URL. The facets property will be used to resolve the format string.
- Resolved URL – Shows how the file string was last resolved.
- Http User – User name for HTTP authentication only. For FTP authentication, encode the username and password into the URL.
- Http Pass – Password for HTTP authentication only.

Actions

- Enable – Enables the import.
- Disable – Disables the import.
- Poll – Processes the CSV.

CsvColumnDeviceExt (Columns)

A "device extension" level object in the Niagara driver architecture. This object is used to define how data is extracted from a CSV row to create CSV record objects. Double-click columns under an import device object to discover and define columns.

CsvBooleanColumn

Used to convert a column from a CSV row into a Boolean value.

Properties

- Index – Column index, starting at 0.
- Facets – True text and false text must match the values that will be found in specified column.
- Default Value – Value to use when the column has empty text.

CsvNumericColumn

Used to convert a column from a CSV row into a numeric value.

Properties

- Index – Column index, starting at 0.
- Default Value – Value to use when the column has empty text.

CsvStringColumn

Used to extract a column from a CSV row into a string value.

Properties

- Index – Column index, starting at 0.
- Default Value – Value to use when the column has empty text.

CsvTimestampColumn

Used to convert a column from a CSV row into a timestamp. It is very important to set the timeFormat facet. See the Time Format section for more details.

Name the column "timestamp" to set the frozen timestamp property of CSV record objects. This timestamp is the one used by the CsvHistoryExt for trending.

Properties

- Index – Column index, starting at 0.
- Default Value – Value to use when the column has empty text.
- Time Zone – If not null, will be applied to the parsed time. Leave null if the time zone is being parsed.

CsvRecordDeviceExt (Records)

A "device extension" level object in the Niagara driver architecture. This is where records are located. Double-click "Records" under an import device object to see the records and to create filters.

CsvRecordSource

As a frozen child of records, it represents every row parsed from the CSV document. It will have a property for every defined column. Each property will change value as each row is parsed from CSV document. This cannot be viewed by a human, but if CsvHistoryExts configured for change of value are add to this object, a record will be created for every row parsed from the file.

Properties

- Debug – Prints information to the console. Can be used to trouble shoot the logic of complex filters.
- XXX – There will be a property for every defined column.

Actions

- Add History Ext – Convenience for creating a CsvHistoryExt.
- Add Alarm Ext – Convenience for creating a CsvAlarmExt.
- Update Columns – Properties representing columns are automatically updated whenever columns are changed. However, since properties can be modified on a slot sheet the record could be out of sync with the defined columns. This action will fix the record.

Topics

- New Record – After all properties have been set for a single row, this topic is fired. It can be linked to record input actions on filter objects.

CsvBooleanFilter

Like the CsvRecordSource, it has properties for every defined column. However, it will not set those properties unless its boolean rule is passed.

Records that pass the rule are fired out of the record accepted topic. If they fail, they are fired out of the record rejected topic.

Properties

- Debug – Prints information to the console. Can be used to trouble shoot the logic of complex filters.
- Status – Will be {fault} if there is an error testing the rule with the specified column.
- Fault Cause – Will attempt to explain any fault.
- Facets – Used for the display of the Value property.
- Column – Column – Which column of the incoming record to test.
- Value – The value the specified column of the incoming record must equal.

Topics

- Record Accepted – Fired after the incoming record passes the rule and is copied into the appropriate properties. Can be linked to the record input action of other filter objects or the TriggeredBoolean object.
- Record Rejected – Fired only when the incoming record does not the rule. Can be linked to the record input action of other filter objects or the TriggeredBoolean object.

CsvNumericFilter

A record filter that compares numeric columns to a number.

Like the CsvRecordSource, it has properties for every defined column. However, it will not set those properties unless its numeric rule is passed.

Records that pass the rule are fired out of the record accepted topic. If they fail, they are fired out of the record rejected topic.

Properties

- Status – Will be {fault} if there is an error testing the rule with the specified column.
- Fault Cause – Will attempt to explain any fault.
- Facets – Used for the display of the Value property.
- Column – Which column of the incoming record to test.
- Rule – Determines how to compare the incoming record with the value property.
- Value – The set of values the specified column of the incoming record is compared to.

Topics

- Record Accepted – Fired after the incoming record passes the rule and is copied into the appropriate properties. Can be linked to the record input action of other filter objects or the TriggeredBoolean object.
- Record Rejected – Fired only when the incoming record does not the rule. Can be linked to the record input action of other filter objects or the TriggeredBoolean object.

CsvStringFilter

Like the CsvRecordSource, it has properties for every defined column. However, it will not set those properties unless its string rule is passed.

Records that pass the rule are fired out of the record accepted topic. If they fail, they are fired out of the record rejected topic.

Properties

- Status – Will be {fault} if there is an error testing the rule with the specified column.
- Fault Cause – Will attempt to explain any fault.
- Facets – Used for the display of the Value property.
- Column – Which column of the incoming record to test.

- Rule – Determines how to compare the incoming record with the value property.

- Value Format – A Niagara format for automatically generating the Value property. Leave empty for manually entering the Value property. This is not resolved for each record, it is resolved at station start or when this property changes.

- Value – The value the specified column of the incoming record is compared to. Do not modify if the Value Format property is not empty.

Topics

- Record Accepted – Fired after the incoming record passes the rule and is copied into the appropriate properties. Can be linked to the record input action of other filter objects or the TriggeredBoolean object.

- Record Rejected – Fired only when the incoming record does not the rule. Can be linked to the record input action of other filter objects or the TriggeredBoolean object.

CsvTimestampFilter

This is a CsvTimestampFilter, but it first creates a timestamp before applying the filter. Like the CsvRecordSource, it has properties for every defined column. However, it will not set those properties unless its time rule is passed.

Records that pass the rule are fired out of the record accepted topic. If they fail, they are fired out of the record rejected topic.

Properties

- Status – Will be {fault} if there is an error testing the rule with the specified column.

- Fault Cause – Will attempt to explain any fault.

- Facets – Used for the display of the Value property.

- Column – Which column of the incoming record to test.

- Rule – Determines how to compare the incoming record with the value property.

Topics

- Record Accepted – Fired after the incoming record passes the rule and is copied into the appropriate properties. Can be linked to the record input action of other filter objects or the TriggeredBoolean object.

- Record Rejected – Fired only when the incoming record does not the rule. Can be linked to the record input action of other filter objects or the TriggeredBoolean object.

CsvTimestampBuilder

Creates a time using Niagara format syntax. This is a CsvTimestampFilter, but it first creates a timestamp before applying the filter. This was created for instances where the date and time are in separate columns.

1. Create the time string in the Time Format property. You can and should probably use Niagara formats. This string with formats will be resolved against this instance, so you can access other properties (columns).
2. Create the Time Pattern. See the Formatting Time section.

Properties

- Time Format – The Niagara format to create a time string.
- Time Pattern – The rules for converting the resolved time format into time. See the Formatting Time section of this document.
- Time Zone – Leave null to use the local time zone or a time zone from the formatted string.
- Status – Will be {fault} if there is an error testing the rule with the specified column.
- Fault Cause – Will attempt to explain any fault.
- Facets – Used for the display of the Value property.
- Column – Which column of the incoming record to set the time and to test.
- Rule – Determines how to compare the incoming record with the value property.

Topics

- Record Accepted – Fired after the incoming record passes the rule and is copied into the appropriate properties. Can be linked to the record input action of other filter objects or the TriggeredBoolean object.
- Record Rejected – Fired only when the incoming record does not the rule. Can be linked to the record input action of other filter objects or the TriggeredBoolean object.

CsvFilterFolder

This folder has record input action and record output topics. It can be used to group a complex set of filters.

Actions

- Record In – Records enter the folder through this action. The folder has a child named in which functions exactly like the source of the CsvRecordExt.

Topics

- Record Accepted – Fired when a filter in the folder sends a record to the folder accepts action of the out child.
- Record Rejected – Fired when a filter in the folder sends a record to the folder rejects action of the out child.

CsvAlarmExt

Can create alarm records whenever one of the record objects is changed.

Properties

- Status – Whether or not there was a problem
- Fault Cause - Will attempt to explain any fault.
- Enabled – Setting to false disables the creation of alarms.
- Duplicate Alarms – How to handle multiple open alarms for a single CsvAlarmExt.
    1. Ignore. If there is an open alarm (acked or not), do not create another alarm.
    2. Update. If there is an open alarm, update its data.
    3. New Alarm. Create a new alarm no matter what.
- Alarm Class – Where to route alarms.
- Alarm Message – Niagara format syntax can be used.
- Alarm Priority – Niagara alarm priority.

CsvHistoryExt

A CsvRecord history extension. Creates trend records for a single column.

- CsvRecords can have multiple CsvHistoryExts.
- Can be used for boolean, numeric and string trends.
- Can trend changes of value, on time intervals, or both.
- The timestamp property of the CsvRecord is used as the timestamp of the trend record.

Properties

- Status – Whether or not there was a problem
- Fault Cause - Will attempt to explain any fault.

- Enabled – Setting to false disables writing history records.
- Timestamp Column Format – When not blank, will be resolved and the value will be placed in the Timestamp Column property. Only resolved at startup or when this value changes.
- Timestamp Column – Which column represents the timestamp.
- Value Column Format – When not blank, will be resolved and the value will be placed in the Value Column property. Only resolved at startup or when this value changes.
- Value Column – Which column represents the value to trend.
- History – The resolved history ID.
- Dynamic History – If true, the History Device and History Name properties are resolved for every record that is written.
- History Device – Niagara format allowing you to specify the history device. Leave blank to use the local station.
- History Name – Niagara format for defining the history name.
- System Tags – Niagara formatting can be used here. Multiple tags must be separated by semi-colons.
- Capacity – History record capacity.
- Full Policy – What to do when the capacity is a record count and that count is reached.
- Active Period – Determines when history records can be created.
- COV – When true, creates records for changes of value.
- Interval – When greater than 0, creates trend records on the specified interval.
- Change Tolerance – Values are trended only when the difference is greater than this value. If the value is 0, every record is trended. If the record type is not numeric, make sure this value is greater than 0 if only real changes of value are desired.
- Facets – Copied into the history.
- Ignore Null – Ignore records whose status is null.
- Ignore Duplicate Timestamps – Ignore records whose timestamp is identical to the last record in the history.

- Ignore Duplicate Records – Ignore records whose timestamp, value and status are identical to the last record in the history.

- Clear History When Older – Clear the entire history when the timestamp of a record is older than the last record in the database.

Actions

- Update History ID – Invoke this to update the value of the History property. This is for informational purposes only, the history ID is automatically updated when writing records to the database.

Triggered Boolean

This object can be triggered to set a boolean output for a configurable amount of time. Simply link the past topic from the CsvRecordDeviceExt or a CsvFilter to the activate action on the TriggeredBoolean object.

Properties

- Duration – How long the out value should remain on.

- Expires – Informational, when the out value will turn off.

- Out – The boolean output. Normally false, activation causes it to be on.

Actions

- Activate – Turn on the output.

- Cancel – Turn off the output.

HISTORY

November 5, 2020: 4.8.0.110
- Update for newer Niagara

September 11, 2018: 4.2.36.34.5
- Changed last record resolve technique to account for possible duplicate time stamps during history creation.

March 14, 2018: 4.2.36.34.4
- Fix Logging for N4.3 Security Manager

September 28, 2017: 4.2.36.34.3
- Fix Logging for N4.3 Security Manager

July 07, 2017: 4.2.36.34.2
- Update FTP to work with N4 Security Manager

September 17, 2015
- Added Niagara 4 support.

November 21, 2014
- Added Ftps Explicit.

April 4, 2014: 3.7.106.3
- Removed date from history export file in zip since date may not reflect the data.

March 17, 2014: 3.7.106.2
- Buffering COV point exports

March 6, 2014: 3.7.106.1
- BQL Export

September 11, 2013: Module version 3.6.47.9
- The column name of single history exports was changed from “value” to the name of the history export descriptor.
- Columns were ordered alphabetically, not they retain the order added by the user.
- Fixed interpolation bug in merged exports.
- Fixed bug in the history ext that would create duplicate rows if the same file was imported twice.
- Fixed bug with secure FTP.
- Data.csv in zip files now has the date as part of the name.

January 30, 2012: Module version 3.6.36.1
- Added Encoding property to all import objects

October 7, 2011: Module version 3.6.36
- Added systemTags to the history extension
- Fixed ignoreNull on the history extension
- Added action to create historyExt for each column on the record / filter objects.
- Duplicate timestamps could still bypass the rules on the CsvHistoryExt.
- Numbers using parents for negativity are supported.
- Performance enhancements to the CsvHistoryExt
- Merge logic would be skipped if only one history selected

May 6, 2011: Module version 3.5.34.1
- CsvTimestampColumn and CsvTimestampBuilder now have a Time Zone property

April 18, 2011: Module version 3.5.34
- CsvFtpImport now supports ftps and sftp.
- CsvHistoryExt is now highly dynamic and has many new properties.
- Leading rows could be accidently skipped.
- Delimiter was ignored when merging histories.
- Point discovery could fail in the Niagara Network.
- More robust number parsing.

May 12, 2010: Module version 3.5.24
- History merge of non-trend histories would break export.
- If showStatus is true for any history export, now {ok} will be included.
- Added Value Format to StringFilter

March 12, 2010: Module version 3.5.11.1
- Point exports could be incorrectly stuck in a stale state.
- There was a memory leak in the FTP import object.
- Added merging of histories to the history export.

October 23, 2009: Module version 3.4.51.2
- In point export, if “Show Status” is set to true, all cells are wrapped in quotes.
- Added FTP to point and history exports.
- Added CsvFtpImport and FixedLengthFtpImport
- Removed the zip file properties on the history export and moved all file naming down into each export type.
- History Export can now send a single file as a zip or plain text.

August 25, 2009: Module version 3.4.51.1
- Added proxy support to URL exports.
- Imports were not stripping surrounding quotes from CSV cells.

August 6, 2009: Module version 3.4.51
- Many dynamic text fields are now automatically resolved when the format patterns are changed.
- There were all kinds of problems if the history export object was duplicated.
- History export URLs would break if the export descriptor name had invalid URL characters.
- History export discovery would break if histories names had spaces or other invalid slot names.
- Point export has the new property “Show Status” which when true, will include the status in each column.

May 28, 2009: Module version 3.3.27.4
- Added clearHistoryWhenOlder to the history extension.
- May 23, 2009: Module version 3.3.27.3
- Added cleanAfterExport to the point email and url exports
- Url Facets were being ignored.
- HTTP URL exports could fail because the content type was incorrect.
- Added some time utilities to the palette.

March 10, 2009: Module version 3.3.27.2
- Added FixedLength imports objects.

March 5, 2009: Module version 3.3.27.1

- Licensing update.

February 27, 2009: Module version 3.3.37
- First release.