Variables

Overview

Variables are used as placeholders of information which can be assigned "outside" of the template. A user selecting tag names for an on-demand report or the schedule setting the batch ID on the start of a batch are examples of setting "outside" of the template.

Five types of variables are provided:

- **Function**

  These read-write variables hold numbers, text and timestamps and are saved permanently so that they can be used over time and are retained even if the machine is restarted. Function variables give the highest fidelity but are the slowest to use because they are cached to a file. To configure see **Variable Editor** below.

  Usage:

  - Variable connections in a report template

  - Naming convention of a report template

- **User Defined**

  These read-write *Custom* variables hold numbers, text and timestamps that are not saved but rather persist only for a report cycle (~30 seconds). To configure see **Variable Editor** below.

  Usage:

  - On-Demand reports

- **Memory Variables**

  There are 2 types of Memory variables: **System** and **User**.

  - **System** variables are read only and provide information about the running system like the last error or how many actions are pending in the schedule.
  - **User** variables are read-write and can hold numbers or text. These values are saved in system memory and are persistent as long as the system is running. If the system is restarted or the *XLReporter Runtime* Service is restarted, the values are reset. To configure, see **Variable Editor** below.

  Usage:

  - Event condition in the Scheduler

- **Analytic**

  These read-write variables hold summary calculations of process readings. The variables can be optionally configured to maintain a history of the calculations. To configure see the **Analytics** document.

  Usage:

  - Snapshot of process values

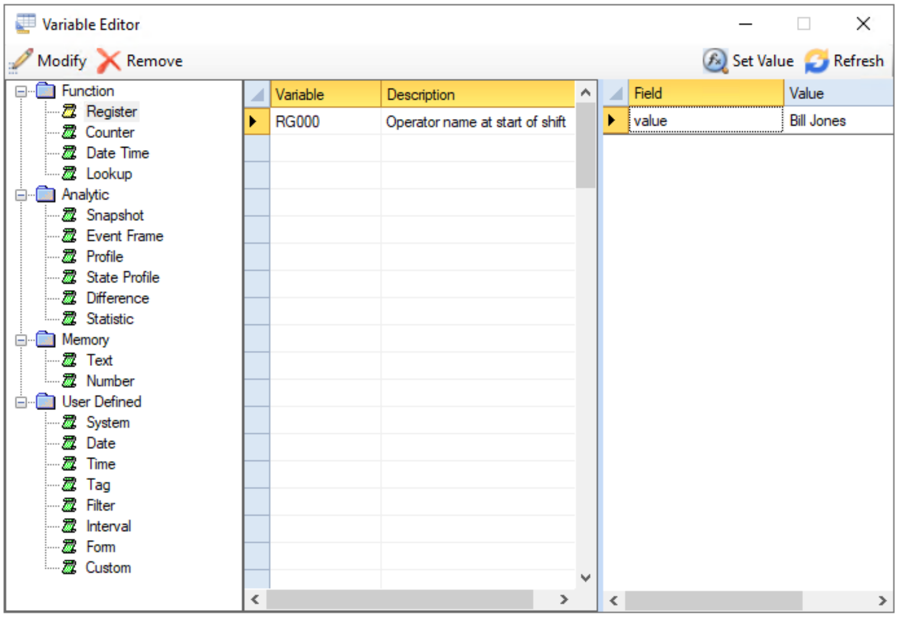
  - Statistics from process values over a period of time

- **Name Type**

  These read-only variables hold date and time values (see **Name Types** document).

Variable Editor

The **Variable Editor** is used to configure **Functions** and view the current values of **Functions** and **Analytics** (to configure see **Analytics**). The editor is opened from the right-side **Tools** tab of the **Project Explorer** in the **Connect** section by selecting **Variables**.



Variable	Description
RG000	Operator name at start of shift

Field	Value
value	Bill Jones

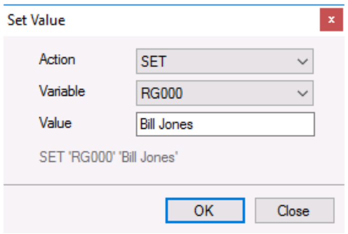
In the Left view are the variable types and categories. Select a category to display all the configured variables in the grid where **Name** is the name of the variable and **Description** is a description of the variable.

To add or modify variable, select a row and double-click, click the **Modify** menu option or select **Modify** from a right click. Note that **Analytic** variable cannot be edited in this dialog; they are edited in the **Analytic Designer**.

To remove a variable, select a row and press the **Delete** key, click the **Remove** menu option or select **Remove** from a right click. Note that **Analytic** variable cannot be deleted in this dialog; they are edited in the **Analytic Designer**.

The current value of a variable can be viewed by selecting the variable and have its value display in the Right view. Since they are multi-valued, the values are distinguished by a **Field**. Note that **User Defined** variables do not persist so no value is shown for those.

The **Function** and some **Memory** variables also permit their values to be edited directly from the display by clicking **Set Value**. This is helpful during development. In practice, their values would be modified by schedule actions.



Action: SET
Variable: RG000
Value: Bill Jones

SET 'RG000' 'Bill Jones'

Function Variable

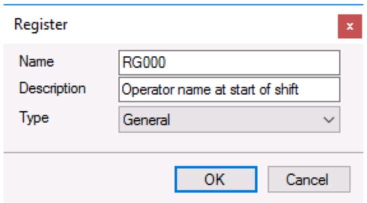
Function Variables are read-write variables that are saved permanently so that their values can be re-used at different times across different reports. The variables are used to hold numbers, text and timestamps.

The **Function Variables** provided are:

- **Register**
- **Counter**
- **Date Time**
- **Lookup**

Register

A **Register** variable is a placeholder for a value.



Name: RG000
Description: Operator name at start of shift
Type: General

To assign a value to a **Register**:

Action:	Set '*Variable Name*' '*Value*'
Scheduler:	Set a Value to Variable

In addition, a **Register** variable can be set using the **Data Management**, **Data Export To Variables** and **To Variable List** functions. These functions can be used to set **Register** variables to values in cells on the worksheet.

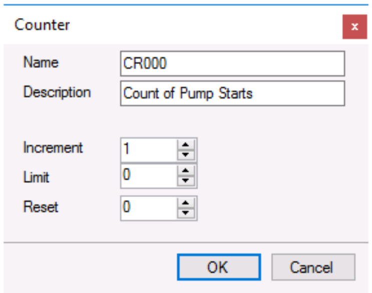
Example: Use a **Register** to hold the naming convention of a report and use it while the report is being updated over a period of time.

The variable provides the following **Fields**:

- **:Cval** (current value, optional)

Counter

A **Counter** variable maintains a count.



Name: CR000
Description: Count of Pump Starts
Increment: 1
Limit: 0
Reset: 0

When used, it increments its value by the **Increment** to a maximum value of **Limit** at which point its value is **Reset**. If the **Limit** is zero (default) then there is no limit.

To reset a **Counter**:

Action:	Reset *'Variable Name'*
Scheduler:	Reset a Variable

To change a value by the **Increment**:

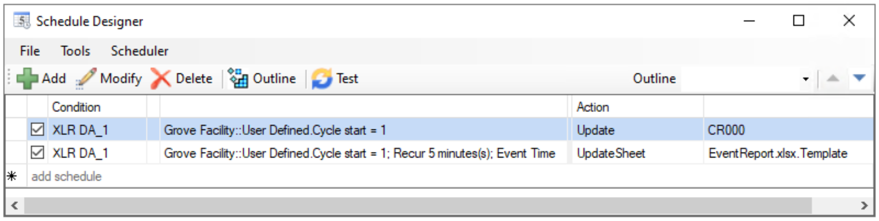
Action:	Update *'Variable Name'*
Scheduler:	Update a Variable

Note that if a counter is used in a template for the connection **Placement** then an **Update** is performed automatically when the report is updated.

The variable provides the following **Fields**:

- **:Cval** (current value, optional)
- **:Incr** (increment)
- **:Lmit** (limit)
- **:Rset** (reset)

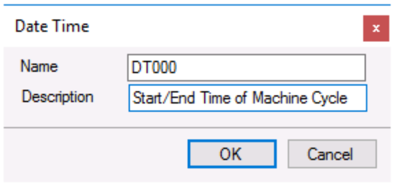
Example: Use a **Counter** as part of an expression. Using the **Counter** called *Index* to increment each time an event occurs and use it in an expression *EventReport_{CR000}* as the naming convention of an event report with the schedule configured similar to the image below:



	Condition		Action	
☑	XLR DA_1	Grove Facility::User Defined.Cycle start = 1	Update	CR000
☑	XLR DA_1	Grove Facility::User Defined.Cycle start = 1; Recur 5 minutes(s); Event Time	UpdateSheet	EventReport.xlsx.Template

Date Time

A **Date Time** variable is a placeholder for the start and end timestamps and durations of events.



Name: DT000

Description: Start/End Time of Machine Cycle

To store the start date and time to a **Date Time**:

Action:	Reset *'Variable Name'*
Scheduler:	Reset a Variable

To store the end date and time to a **Date Time**:

Action:	Update *'Variable Name'*
Scheduler:	Update a Variable

The DT variables *can* be **Set** from the *Set Value to a Variable* schedule action, however only the Date(:Date), Time(:Time), End Date(:Edat), and End Time(:Etim) fields can be set this way. The universal date time fields will play off these values.

The variable provides the following **Fields**:

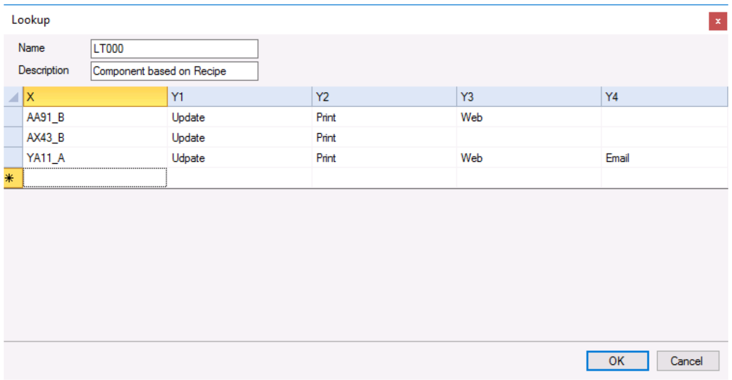
- **:Stdu** (start date and time in universal format YYYY-MM-DD HH:mm:ss)
- **:Stdt** (start date and time in local format)
- **:Date** (start date in local format)
- **:Time** (start time)
- **:Endu** (end date and time in universal format YYYY-MM-DD HH:mm:ss)
- **:Endt** (end date and time in local format)
- **:Edat** (end date in local format)
- **:Etim** (end time)

- **:Span** (span in the format DD:hh:mm:ss)
- **:Spdy** (span in days)

Example: A machine cycle report uses **Date Time** variable *MachineCycle* to hold the start and end time of the cycle. This variable can be used for the time period of a data group.

Lookup

A **Lookup** variable uses a lookup table to provide up to four text values from an input value (X). Note, that if X is set to a non-specified value it will be ignored and contain the last good specified value it was set to. X values are **case sensitive**.



Lookup

Name: LT000

Description: Component based on Recipe

X	Y1	Y2	Y3	Y4
AA91_B	Update	Print	Web	
AX43_B	Update	Print		
YA11_A	Udpate	Print	Web	Email

To assign values to a **Lookup**:

Action:	Set '*Variable Name*' '*X Value*'
Scheduler:	Set a Value to Variable

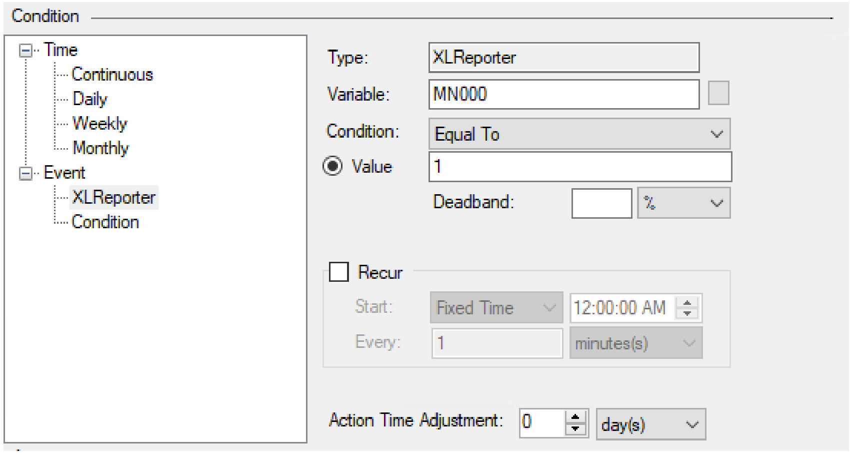
The variable provides the following **Fields**:

- **:X** (X value)
- **:Y(1 to 4)** (Y values)

Example: A template is configured with 10 group connections, each adding their own data to the report. In a batch cycle, a batch phase activates a combination of groups to produce the report for that phase. Use a Lookup function where X is the phase ID and Y are the group combinations.

Memory Variable

Memory Variables hold system information, numbers and text in memory but only persist until the system is restarted or the *XLReporter Runtime* service is restarted. These variables can be used as Event Based schedule conditions, where a particular memory variable is monitored for a specific value or change in value that in turn triggers the run of the specified schedule action.



Type: XLReporter

Variable: MN000

Condition: Equal To

● Value 1

Deadband: %

□ Recur

Start: Fixed Time 12:00:00 AM

Every: 1 minutes(s)

Action Time Adjustment: 0 day(s)

The **Memory Variable** types provided are:

- **System**
- **Number**
- **Text**

Memory variables can be set by using a **Set** action or by using the **Data Management**, **Data Export To Variables** and **To Variable List** functions.

System

System memory variables are preexisting values in all XLReporter projects. They are values generated by the process (e.g., Schedule Action Durations) that can be used to trigger events when a certain process or metric is out of compliance. For example, if an action is taking too long to process, an email can then be sent to notify an operator that the system needs attention to correct this. The following are available:

- **Error Date**
  The timestamp of the last error
- **Error Date Adjusted**
  The timestamp of the last error with the Scheduler **Action Time Adjustment** applied.
- **Error Code**
  The numeric error code of the last error.
- **Error Action**
  The action that triggered the last error.
- **Error Message**
  The error message of the last error.
- **Longest Action Duration**
  The duration (in seconds) of the longest action of the last 10 actions executed.
- **Longest Action**
  The action that took the longest of the last 10 actions executed.
- **Longest Action Date**
  The timestamp of the longest action of the last 10 actions executed.
- **Schedule Action Pending Count**
  The number of actions currently pending in the Schedule action queue.
- **Schedule Action Pending %**
  The percentage of actions currently pending in the Schedule action queue (note that the Schedule action queue can hold up to 256 actions at one time).
- **Analytic Action Pending Count**
  The number of actions currently pending in the Analytic action queue.
- **Analytic Action Pending %**
  The percentage of actions currently pending in the Analytic action queue (note that the Analytics action queue can hold up to 256 actions at one time).
- **Current Action Duration**
  The duration in seconds since the current action was initiated.

Note that the **Current Action Duration** can be used in a schedule as a Condition to trigger an action *For example,* when Current Action Duration > 30 secs, send an email.

When the condition is triggered, the action is performed and so in the above example the email is sent. In addition, the current Action (which caused the trigger) is ended. For more information, in the **Document Library** see *REPORTS, Deploy Schedule Reports.*

Number

A **Number** type memory variable can hold whole numbers ranging from -2,147,483,648 to 2,147,483,648. Decimal values will be truncated to only include the whole number portion of the decimal. There are 32 number variables available.

Text

A **Text** type memory variable can hold up to 64 characters. The following are illegal characters and cannot be pushed to Text Memory variables: **Semicolon** (;) and **Tilde** (~). There are 32 text variables available.

User Defined Variable

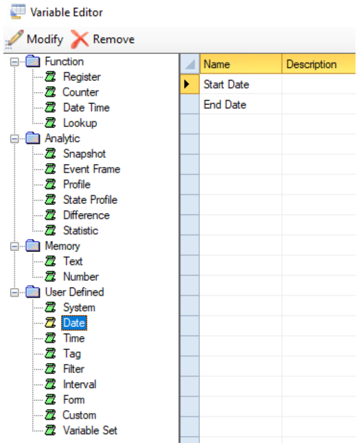
User Defined Variables are read-write *Custom* variables that hold numbers, text and timestamps that are not saved but rather persist only for a report cycle (*~30 seconds*). These variables are typically used on-demand but can be used for schedule reports when the variable values are set at runtime. This can be done by using the **Set** action in the **Scheduler**. In addition, these variables can be set by using the **Data Management**, **Data Export To Variables** and **To Variable List** functions.

The **User Defined Variable** types provided are:

- **Date**
- **Time**
- **Tag**
- **Filter**
- **Interval**
- **Form**
- **Custom**

Date

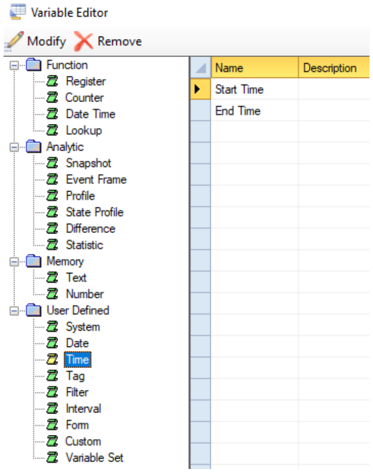
A **Date** type user defined variable can be referenced in a historical data group time period, database group filter, or other location to supply the value of a Date. For example, both Start Date and End Date are configured by default and used in the majority of On-Demand Reports.



Name	Description
Start Date	
End Date	

Time

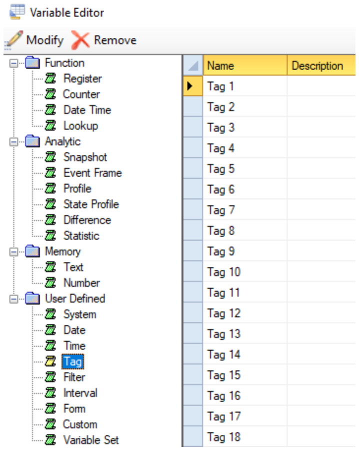
Similarly, a **Time** type user defined variable can be referenced in a historical data group time period, database group filter, or other location to supply the value of a Time. For example, both Start Time and End Time are configured by default and used in the majority of On-Demand Reports.



Name	Description
Start Time	
End Time	

Tag

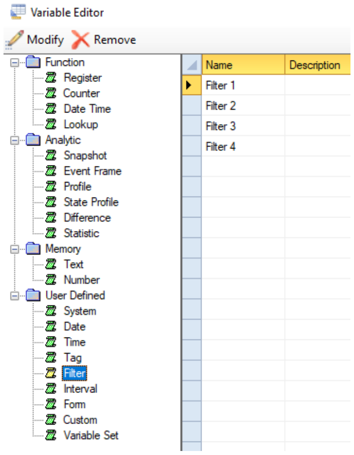
The **Tag** type variables are used in place of fully declared tag names when creating a new data group. These tags come predefined in every project.



Name	Description
Tag 1	
Tag 2	
Tag 3	
Tag 4	
Tag 5	
Tag 6	
Tag 7	
Tag 8	
Tag 9	
Tag 10	
Tag 11	
Tag 12	
Tag 13	
Tag 14	
Tag 15	
Tag 16	
Tag 17	
Tag 18	

Filter

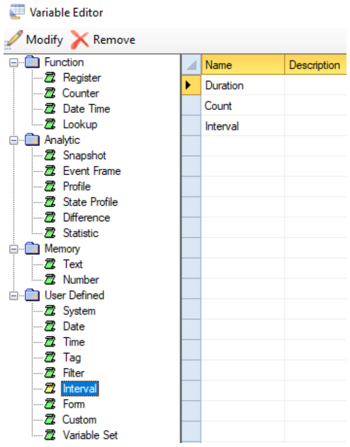
The **Filter** type variables are useable anywhere in the product, but the intended use is on the **Filters** tab of historical data groups and database groups.



Name	Description
Filter 1	
Filter 2	
Filter 3	
Filter 4	

Interval

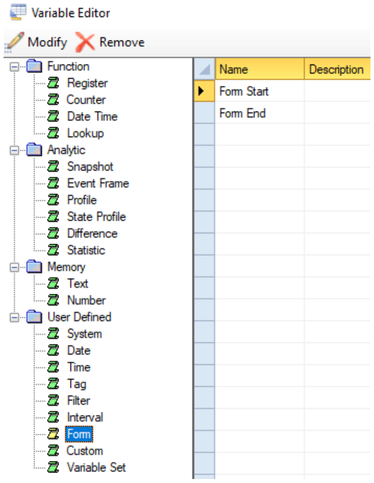
The **Interval** type variables are useable anywhere in the product, but the intended use is on the **Time Period** tab of a summary values historical data group in the **Interval** section.



Name	Description
Duration	
Count	
Interval	

Form

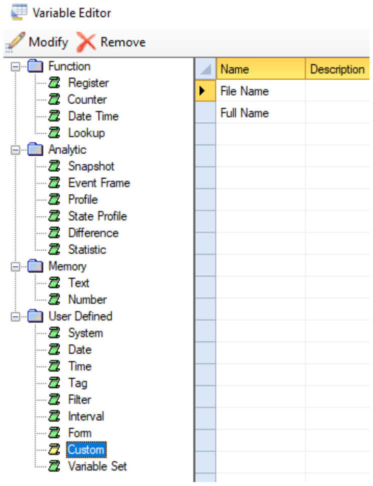
The **Form** type variables are used in the **Manual Data Entry Forms** templates. For example, the *Form Start* and *Form End* dates. These variables are configured by default in all projects.



Name	Description
Form Start	
Form End	

Custom

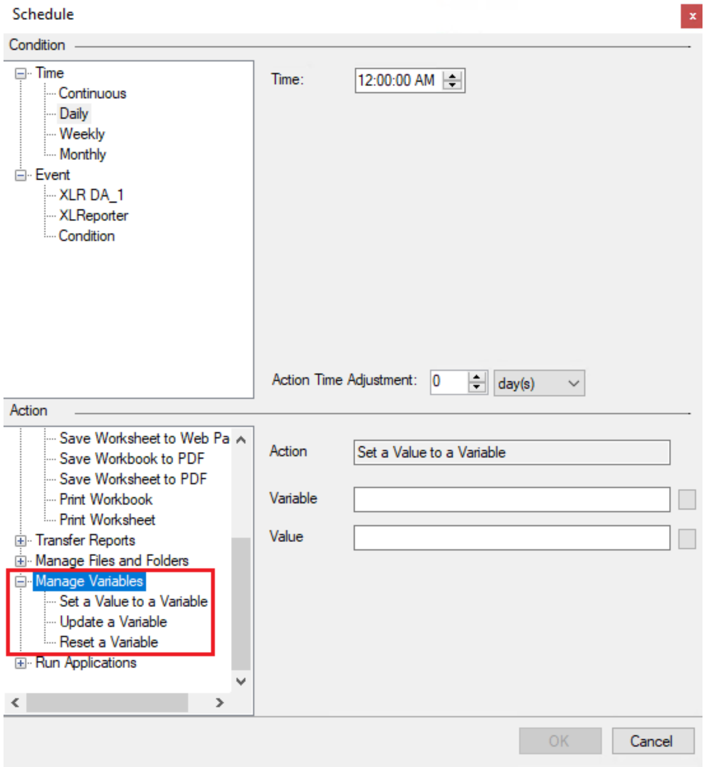
The **Custom** type variables are just that, custom fields that can be referenced anywhere in the product. The *File Name* and *Full Name* variables are included by default as they are used in the **Text Discrete** data connector.



Name	Description
File Name	
Full Name	

Scheduler Actions

The **XLReporter Scheduler** has built-in actions to manipulate variables at runtime. These actions include the **Set**, **Update**, and **Reset** actions found under the *Manage Variables* action category.



Time: 12:00:00 AM

Action Time Adjustment: 0 day(s)

Action: Set a Value to a Variable

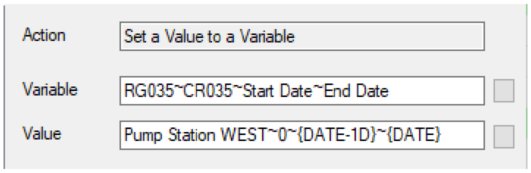
Variable:

Value:

Set a Value to a Variable

The **Set** action writes a *Value* to the specified *Variable* when the condition is met. The *Value* can be static text, an expression (i.e. {DATIM}), or a real-time process value from a connector defined in the same project.

This action can set up to four values to four different variables via a ~ separated list.



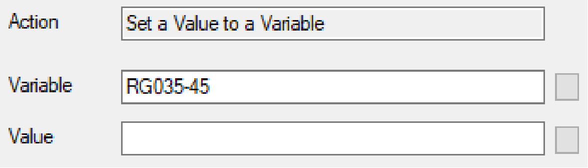
Action: Set a Value to a Variable

Variable: RG035~CR035~Start Date~End Date

Value: Pump Station WEST~0~{DATE-1D}~{DATE}

The configuration above would set *RG035* to “Pump Station WEST”, *CR035* to 0, *Start Date* to the previous day’s date, and *End Date* to the current date.

Alternatively, this action can also set a single value to an array of variables of the same type.



Action: Set a Value to a Variable

Variable: RG035-45

Value:

The configuration above would set the value of “” to Registers 35 through 45. Clearing them for the next cycle report.

Update a Variable

The **Update** action has two functions. It increments *Counter* variables according to the variables configured **Increment** value (default 1). For *Date Time* variables, it sets the *universal end date*, *local end date, end date, end time, Span, and Span (in days)*.

Reset a Variable

The **Reset** action, much like the update action, has two functions. As the name implies, it resets *Counter* variables back to their initial value. For *Date Time* variables it sets the *universal start date*, *local start date, start date,* and *start time.*

Both the *Update* and *Reset* actions can use the ~ separated lists or the CRXXX-CRYYY and DTXXX-DTYYY syntax respectively to apply the action to multiple variables.

Variable Parsing

Function and **User Defined** variables can be parsed to limit the length of the text when the variable is used. To specify parsing, the following syntax is used:

{*name*, *start*, *end*, *mode*}

where

name is the name of a variable

start a start position determined by a character (from the start)

end is an end position determined by a character (from the end)

mode is the parsing mode; 1=left, 2=mid, 3=right

Example: The variable **Tag1** contains the text value MIXER_FLOW_SETPOINT.

{**Tag1**,_,_,1} gives MIXER

{**Tag1**,_,_,2} gives FLOW

{**Tag1**,_,_,3} gives SETPOINT

{**Tag1**,X,T,2} gives ER_FLOW_SETPOIN