*Topic updated on September 21, 2023*

# Trigger local variables, static variables, macros and event variables

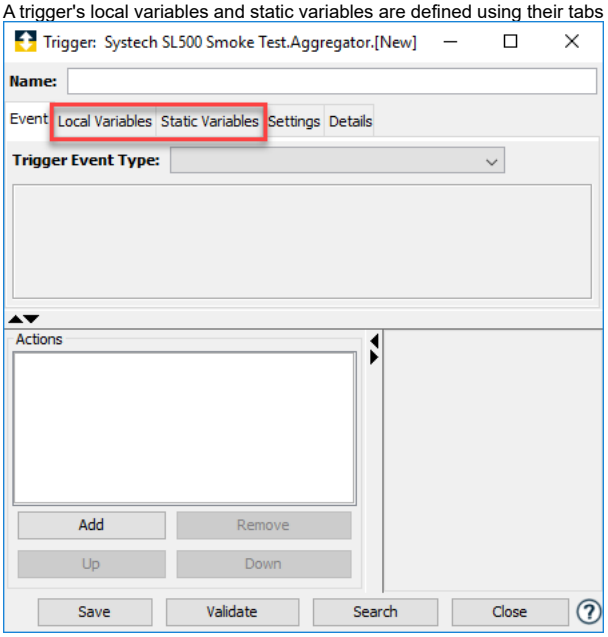
A trigger definition includes local variables and static variables for use as part of the trigger's application logic (in the trigger's actions). These local variables and static variables belong to the trigger and its actions and can not be referenced outside of the trigger.

A group of trigger macros are available for use as part of the trigger's application logic. These trigger macros can be used as the source variable for the trigger's actions.

The trigger's event type also identifies the event variables, both source variables and destination variables, that are available to the trigger's actions.

## Local variables and static variables

A trigger's local variables and static variables are defined using their tabs in the trigger window.




These trigger variables can be used as part of the trigger's application logic when a variable is needed to hold a run time value that does not need to be available outside of the trigger. Examples of local and static variable uses include:

- A temporary calculation
- A loop counter

An important difference between these trigger (local and static) variables and device variables is that the trigger variables are fully contained within the trigger's execution and are not accessible outside of the trigger. Device variables reside in a device, and can be accessed by anything that can access the device, including different triggers, the device's application code, and the device's programming tool.

A trigger's local variables and static variables have the same definition and run time behavior, with the following differences:

- Local variables have a life cycle, or scope, of a single execution instance of the trigger. This means that the variable is created when an instance of the trigger is executed and the variable is destroyed when that instance of the trigger ends its execution. A running counter would lose its value when the trigger instance ends and be reinitialized to its default value when the next instance of the trigger is executed.
- Static variables have a life cycle, or scope, of the duration that the trigger is in a Started state and a Loaded status. This means that the trigger is started and its project is also started. A static variable's value is retained during this life cycle, so a running counter would retain its value when the trigger instance ends and be available to the next execution instance of the trigger.

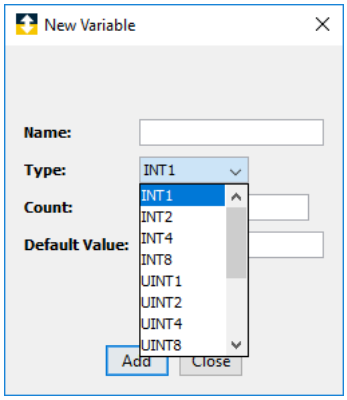
### Adding local or static variables to a trigger

1. Using either the List Editor or the Canvas Editor, from the trigger window select the **Local Variables** or **Static Variables** tab.

   This example will use the **Local Variables** tab.

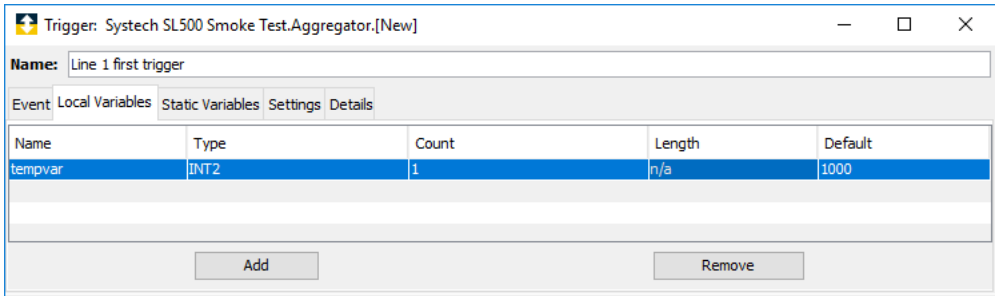
2. Select **Add**.

   The New Variable window appears.

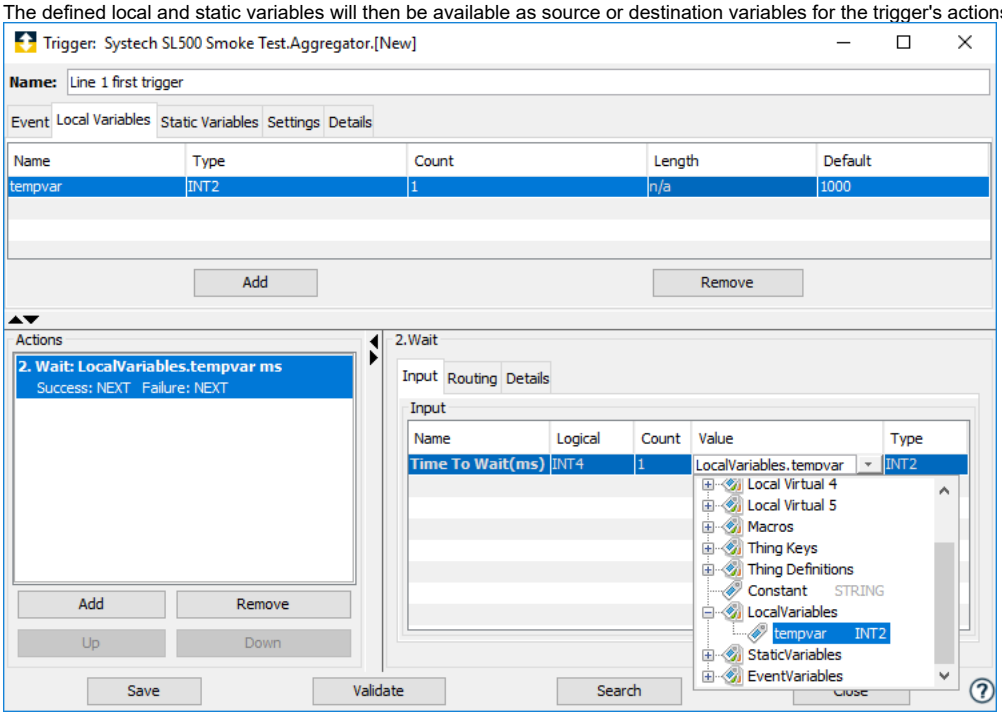



3. **Name** the variable, and then select the **Type** (for this example INT2).
4. In the **Count** parameter, enter a value of 1 (the default) to indicate a scalar variable, or enter a value greater than 1 to indicate an array variable.
5. In the **Default Value** parameter, type the value to use to initialize the variable.
6. Select **Add**.

   The variable is added to the **Local Variables** tab.




The defined local and static variables will then be available as source or destination variables for the trigger's actions, for example:




## Additional variable considerations

Additional considerations when using local variables, static variables or device variables include:

- A trigger's local or static variables are useful for temporary variables that do not need to be accessed outside of the trigger. They do not require the access overhead of a device driver code path, network overhead, and device response time to read or write.
- Local variables have a scope of a trigger's one execution instance, so the variable's value can not be retained and passed between trigger execution instances.

  This does allow the local variable, since it is private to one execution instance of the trigger, to not be affected by multiple concurrent execution instances of the trigger.

- Static variables have a scope of the trigger being in the Loaded status until it changes to the Unloaded status. This means that the variable's value is retained and is shared by all execution instances of the trigger. If multiple instances of the trigger can be executed concurrently, the variable may have indeterminate values based on the timing of the different execution instances referencing the variable.

  > **Warning**
  >
  > When editing a started trigger, the trigger is reloaded upon save. This causes all static variables to be reset to their initial value.

- Device variables referenced in any of a trigger's actions are read once when the trigger execution instance begins. The device variables are written to an internal buffer during the trigger's execution of its multiple actions and are only written (flushed) to the device when the trigger execution instance ends.
- This initial reading of all device variables and final flushing of the buffered writes may not result in the desired device variable access that the trigger's application logic requires.
  For more information on device variable access, see Demand Read, Demand Write, and Device Commit.

---