

# Option CloudGate Asset Gateway Step-by-Step Guide

This is Step-by-Step guide that will walk you through installing and setting up an Option CloudGate Asset Gateway with deviceWISE Asset Gateway software. As well using the deviceWISE Workbench to control an Option CloudGate Asset Gateway. deviceWISE is an open platform that allows you to control and monitor any device remotely. In this guide you will be working with the Asset Gateway side of deviceWISE, however there is also the Small Device, Modem, M2M Service, and Enterprise Gateway sides of deviceWISE and you can find out more information about these other aspects of deviceWISE at [help.devicewise.com](http://help.devicewise.com).

\*Note that this guide is for Option CloudGate firmware version up to m2m-1.2x.

## **Section 1**

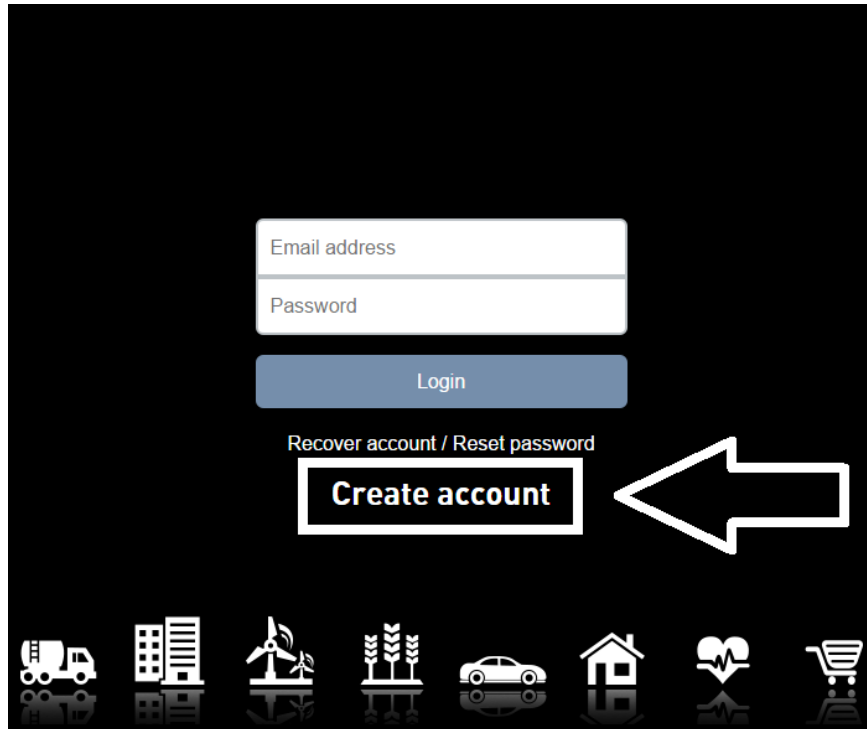
### **Creating your deviceWISE Management Portal Account**

The deviceWISE Management Portal is a web based environment that allows developers, project managers, IT managers, and end Users to easily view and manage their devices and gateways from anywhere in the world, either out in the field or right in office. The Management Portal is also the place where you can decide who can see which device or gateway. Giving you the ability to dynamically restrict or expand the scope of your team easily through the web interface. The Management Portal is also where you can find device resources such as: updated or new deviceWISE firmware packages and device driver packages. Giving you the versatility to keep your devices up-to-date and relevant by adding new device to your existing hardware.

In this section you be walked through the process of creating a new deviceWISE Management Portal account.

- 1) Open the web browser of your choice; navigate to [portal.telit.com](http://portal.telit.com)

- 2) At the Login page for the **deviceWISE Management Portal** select **Create Account**.



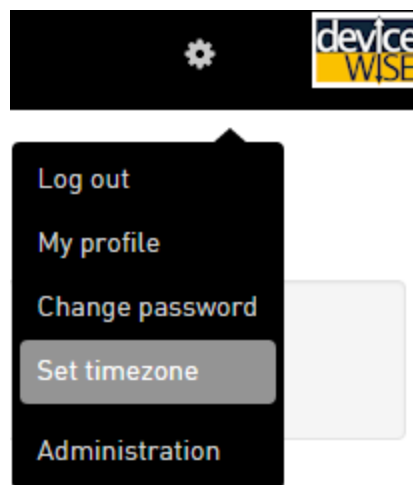
- 3) Enter your email address, then select **Send email** to start the account creation process. A confirmation email will be sent to your email address; depending on your email account settings your confirmation email could be in your **Spam** folder.

- 4) When you receive your signup email, select the link in the message to continue the account creation process.

- 5) Fill in the account registration form.

\* The Organization Key parameter is a unique string that identifies your organization within the M2M Service. You should pick something meaningful like your company name or the project name. If one of your colleagues has already created their account with the Organization Key that you wish to use, then you need to specify a different string for the Organization Key.

- 6) Once you have finished filling out your information, select the **Create account** button to complete your registration process.
- 7) Once you have completed your registration process, log into the **deviceWISE Management Portal** to confirm that you have been successfully registered.
- 8) Change your **Management Portal** time zone to your local time, by clicking on the “gear” icon in the top right corner of the portal window.
- 9) Select the **Set Time Zone** option, then pick your local time zone.



\* An alternative to creating an account using the above steps is to have a colleague that has an account in an organization use the **Administration -> Users -> Invite user** feature to create your account in that existing organization. If you use this alternative method, an email will be sent to your email ID with information to complete the creation of your account. For this method to work, your colleague will need **Administrative** privileges within the organization you wish to join.

\*Once your account has been created, you also have the option of being invited to another organization; there is no limit to the amount of organizations you can work within. However, the work or changes you make within an organization will stay within that organization.

## Section 2

### Installing deviceWISE Workbench

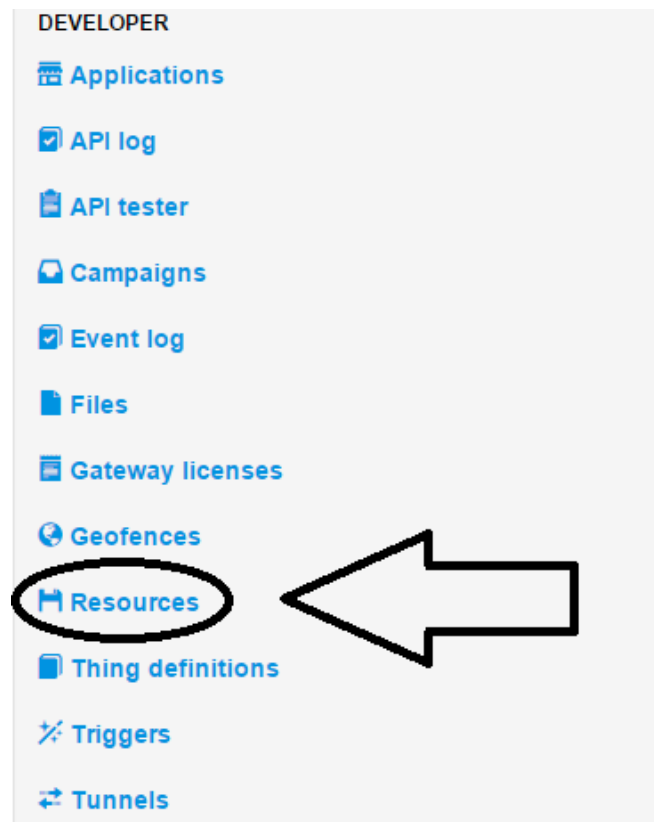
The deviceWISE Workbench is the main tool used to configure an asset or enterprise gateway running deviceWISE. The Workbench provides full development capabilities as well as the ability to administer and debug an individual device in the field. The Workbench itself does not have deviceWISE running inside it, but rather acts as a viewer to the User, giving the User the ability to see and manage multiple gateways from one simple to use tool.

In this section you will walk the process of installing the deviceWISE Workbench.

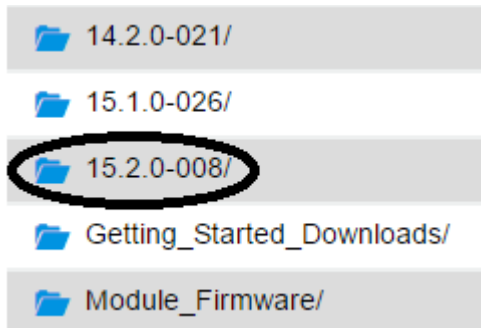
- 1) Log into the **deviceWISE Management Portal** and then select **Developer**.



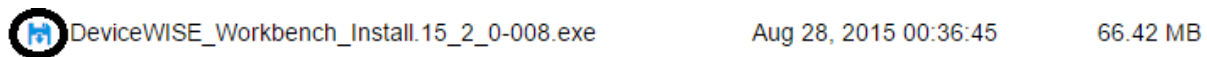
- 2) Under the **Developer** Menu select the **Resources** option.



- 3) Select the latest build folder.



- 4) Click the save icon next to the “**DeviceWISE\_Workbench\_Install.X\_X\_X-X.exe**” to download the deviceWISE Workbench installer (where XXX is the version of the Asset Gateway software you want to download).



- 5) Navigate to where on your computer you downloaded the Workbench Installer.
- 6) Open and run the Workbench Installer follow the on-screen instructions do not change any of the default settings. When the deviceWISE Workbench has completed installing, select **Finish** to close the installer.
- 7) Open the deviceWISE Workbench by double-clicking on the shortcut on your desktop or by finding the Workbench in your programs folder under “deviceWISE”.

You now have the **deviceWISE Workbench** installed on your computer.

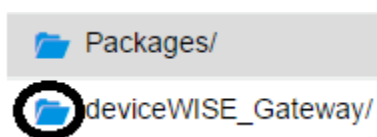
## Section 3

### Installing deviceWISE on the Option CloudGate

The Option CloudGate Asset Gateway comes in two different deviceWISE variants, deviceWISE Enabled and deviceWISE Ready. A deviceWISE Ready Asset Gateway is an Asset Gateway that has been certified to be compatible and fully functional with the deviceWISE Asset Gateway software; however, the deviceWISE software is not installed by Option. A deviceWISE Enabled Asset Gateway is an Asset Gateway that has been pre-loaded with deviceWISE Asset Gateway software by Option or by a third party distributor. This section will walk you through how to install the deviceWISE Asset Gateway software onto an Option CloudGate. If you have a deviceWISE Enabled Asset Gateway you should follow these steps as well to update your gateway to the latest version of deviceWISE or the version you have been told to use.

### Obtaining the Option CloudGate Asset Gateway Installation Files.

- 1) Log into the deviceWISE Management Portal and then select **Developer** option.
- 2) Under the **Developer Menu** select the **Resources** option.
- 3) From the list of files select the appropriate build number (If you are not sure which build is appropriate for you select the newest build).
- 4) Select **Option** folder.
- 5) Select the **deviceWISE\_Gateway** folder.

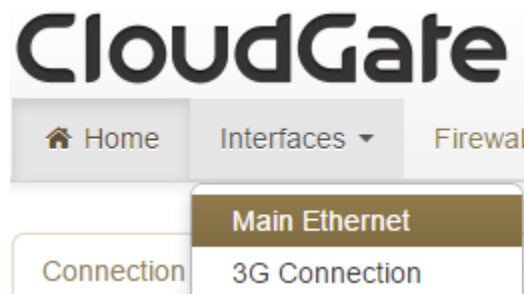


- 6) Download the **DWGateway\_Install.Linux-ARM-Option.X\_X\_X-X.zip** (where **XXX** is the version of the Asset Gateway software you want to download).

## Installing deviceWISE Asset Gateway Software

Before trying to install the deviceWISE Asset Gateway software, please follow the included instructions for your Option CloudGate Asset Gateway. Make sure your gateway is powered on and activated on a cellular network.

- 1) Plug your Option CloudGate Asset Gateway into your computer via Ethernet; either directly or over a local LAN Network.
- 2) Open a web browser and log back into the CloudGate web user interface with the default username and password.
- 3) Select **Main Ethernet** from within the **Interface** pull down.



- 4) Under **General** make sure:
  - a) **Enabled:** Yes
  - b) **Mode:** LAN
  - c) **WAN/LAN Switchover:** Yes

- 5) If you are using **DHCP** to connect to your gateway, make sure the **DHCP sever** is enabled; otherwise disable the **DHCP server**.
- 6) Scroll down to the bottom of the page, click on the **Save changes** button to save your settings.
- 7) Select **System** from the web toolbar.
- 8) Under **Time Settings** set the time zone to your local time zone, then save your changes.
- 9) Select **Provisioning** from the web toolbar.
- 10) Scroll down to settings and click “**NO**” next to “**Enable automatic provisioning**”.
- 11) If you or your organization has decided to use the CloudGate web interface to load the deviceWISE Asset Gateway software, proceed with this step. Alternatively, your organization may direct you to use the Option provisioning server to load the deviceWISE Asset Gateway software.
  - a) Select **Provisioning** from the web toolbar.
  - b) Scroll down to “**Upload device provision file**” and click “**Choose File**”.
  - c) Navigate to where you downloaded the **DWGateway\_Install.Linux-ARM-Option.X\_X\_X-X.zip**.



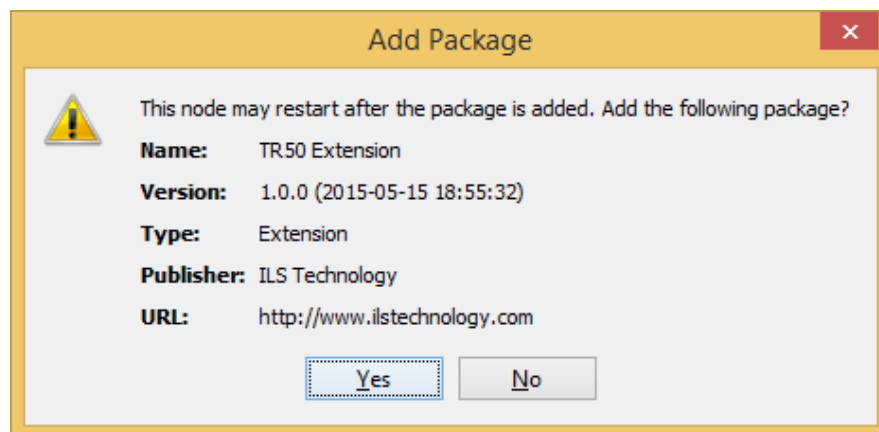
- d) Open deviceWISE install file.
  - e) Click the **Upload** button on the web interface.
  - f) A **Restart Device** option will be displayed. Select **Restart** to have the gateway restart. The deviceWise Asset Gateway software will be installed after the gateway.
- 12) If your organization has decided to use the CloudGate provisioning server, <http://cloudgate.option.com/> , to load the deviceWISE Asset Gateway software and perform other gateway configuration tasks, the details of these tasks will come from your organization and depends on how they have decided to administer and support your gateways.

## Updating a package (Optional)

Select the **Packages** tab under **Administration** on your gateway if you see the status of a package is “ERROR: Incompatible interface”, then the current package installed is incompatible with the new version of deviceWise. If everything has a status is “OK” skip to the Section 4.

- 1) Log into the deviceWISE Management Portal and then select **Developer** option.
- 2) Under the **Developer Menu** select the **Resources** option.
- 3) Navigate back to the **Option** folder under the same release version folder on the Management Portal.
- 4) Select the **Package** folder.

- 5) Depending on what package you are looking for it could be in either the **Release** or the **Prototype** folder.
- 6) Whatever package you are you looking will be named as **“dw.dwPackageName.pkg”** (The Package Name is name you see in the package tab). Download any and all package files as needed.
- 7) On the **Package** Tab click the **Add** button.
- 8) Navigate to where you downloaded the package file and select the package file.
- 9) A confirmation window will appear click the **“Yes”** button to begin installation. Your gateway will reboot.



- 10) Repeat steps 5-9 for any additional packages you need to install.

Congratulations deviceWISE is now installed on your Option CloudGate Asset Gateway.

## Section 4

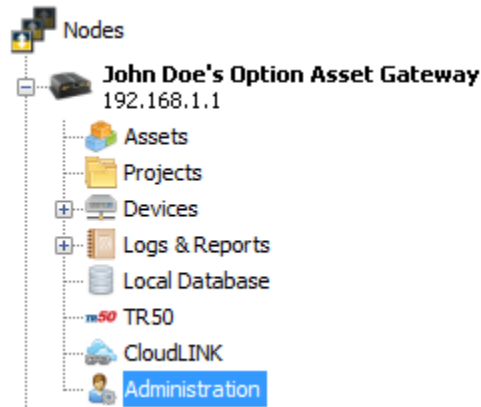
### Connecting to the Your Asset Gateway

Now that you have the deviceWISE Asset Gateway software installed on your Option CloudGate Asset Gateway, you can connect to your gateway using the deviceWISE Workbench. Make sure that your computer's Ethernet adapter (Local Network) is still configured in the same way that it was in Section 3. With the same **Class C Network ID** (192.168.1) as your gateway, but with a different **Host ID** than your gateway.

- 1) Open the deviceWISE Workbench by clicking the Windows **Start** menu, select **All Programs > deviceWISE > Workbench > Workbench**. You can also double-click the Workbench icon on your desktop.
- 2) If you are prompted to login the default username and password are both: **admin**.
- 3) The first time you log into the deviceWISE Workbench you will not see any devices; to be able to see a device on the workbench you have to first have to **Scan** for a device. At the top left of the workbench window right-click on the **Nodes** icon, then select **Scan**. Each device you work with on the workbench is referred to as a **Node**.
- 4) A window will appear asking you to select where you want to scan for your device; select **Network**, because you are currently connected to your Option CloudGate Asset Gateway locally through an Ethernet connection.
- 5) A dialog window will appear showing you different ways of scanning for a device. In the **Address/Host Name** box type: **192.168.1.1** ; then select **Scan**.

\*If you are working with a device that has been assigned a different IP Address other than the default IP Address, then you have to scan for that IP Address.

- 6) An icon representing your Gateway will appear below the **Nodes** icon, with the name “**New Node**”. To change the name of your device double-click on your gateway’s icon to expand your gateway’s list of features. Select the **Administration** icon within your gateway’s list of features.



- 7) Select the **Node Administration** Tab.

- 8) In the **Name** box type in whatever you want your Gateway’s name to be and then click the **Save Details** button.

Details

Name:

John Doe's Asset Gateway

Version:

1.0

Description:

Save Details

## Section 5

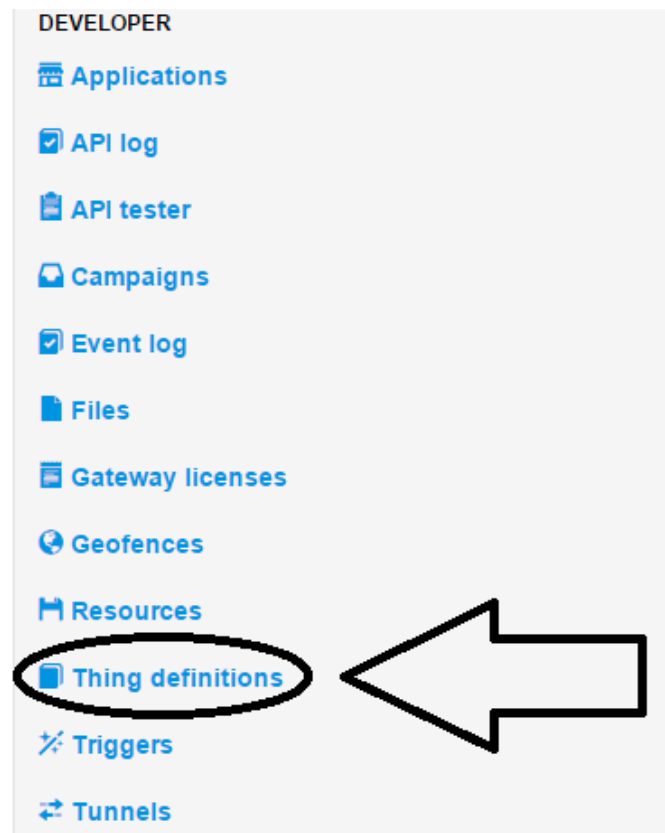
### Connecting your Asset Gateway to M2M Service

At this point, your Asset Gateway should be connected to the Internet; either over a local LAN network or a cellular network, or both. The next step is to connect your Asset Gateway to the **M2M Service**. To connect to the **M2M Service**, you must first configure a **TR-50** connection.

In order to connect your Asset Gateway to the deviceWISE M2M Service you first need to have a **Thing Definition** and **Application Definition**. In this section you will be creating a **Thing Definition** and **Application Definition** for your asset gateway.

### Creating a Thing Definition

- 1) Navigate to the **Developers** page on the **Management Portal**.
- 2) Under the **Developer** Menu select the **Thing Definitions** option.

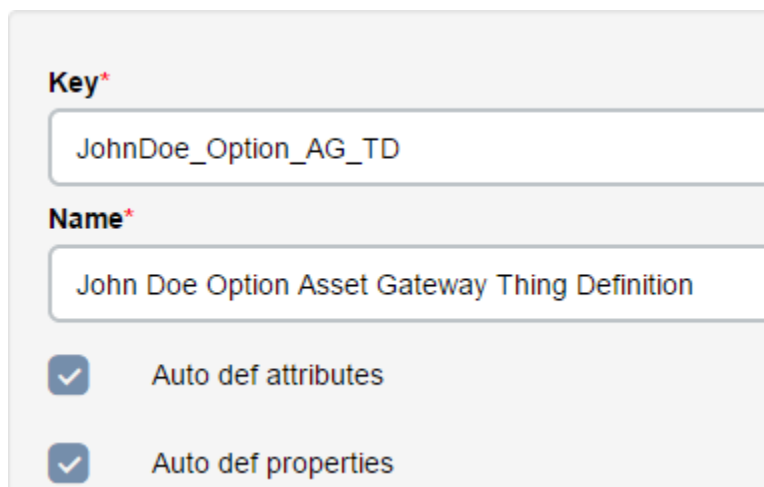


3) Click the **New Thing Definition** button.



4) Create a new **Thing Definition** with:

- **Key:** Yourname\_Option\_AG\_TD
- **Name:** Your Name Option Asset Gateway Thing Definition
- **Check:** Auto def attributes
- **Check:** Auto def properties



5) Click the **Add** button to create your **Thing Definition**.

## Creating an Application Definition

1) Navigate to the **Developers** page on the **Management Portal**.

2) Under the **Developer** Menu select the **Applications** option.

3) Click the **New Application** button.

4) Create a new **Application** with:

- **Name:** Your Name Option Asset Gateway App
- **Auto Registration Thing Def ID:** The Same **Thing Definition** you created previously.
- **Auto Registration Tags:** YourName
- **Auto Registration Security Tags:** YourName
- **Check:** Org Admin
- **License:** Industrial Asset Gateway

Name

John Doe Option Asset Gateway App

Description

This is John Doe's Option Asset Gateway Application

Auto Registration Thing Definition ID

John Doe Option Asset Gateway Thing Definition

Auto Registration Tags

+

JohnDoe

Auto Registration Security Tags

+

JohnDoe

☒

Org Admin

License

Industrial Asset Gateway (Evaluation)

Add

Cancel

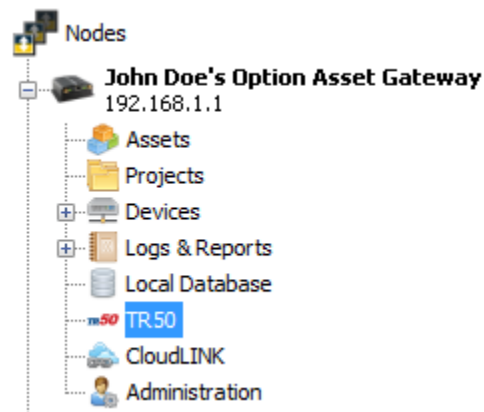
5) Click the **Add** button to create your **Application**.

## Connection your Asset Gateway to TR50

Now that you have created your **Thing** and **Application Definitions** you can connect your Asset Gateway to the deviceWISE M2M Service. The way your Asset Gateway connects to the deviceWISE M2M Service is using a protocol called **TR50**.

1) Navigate back to the **deviceWISE Workbench**.

2) Select the **TR50** icon for your gateway.



3) In the **TR50 Connection Management** Tab configure your **TR50** connection with:


- **Server Address:** **api.devicewise.com**
- **Connection Type:** MQTT with SSL
- **Proxy Type:** None
- **Thing Key:** MAC Address

<b>TR50 Server Address:</b>	api.devicewise.com
<b>Connection Type:</b>	MQTT with SSL
<b>Proxy Type:</b>	None
<b>Thing Key:</b>	MAC Address: 2AB2BD02A201
<b>Application Token:</b>	.....



- 4) Copy the **Application Token** from the **Application** you created and paste it into the **Application Token** box on the **TR50** window.

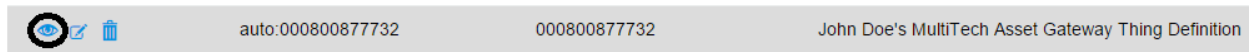
## John Doe Asset Gateway App

<b>Name</b>	John Doe Asset Gateway App
<b>Description</b>	This is John Doe's Asset Gateway Application
<b>Token</b>	brn3dBD Eeu9pusdl 
<b>Auto Registration Thing Definition</b>	John Doe's Asset Gateway Thing Definition

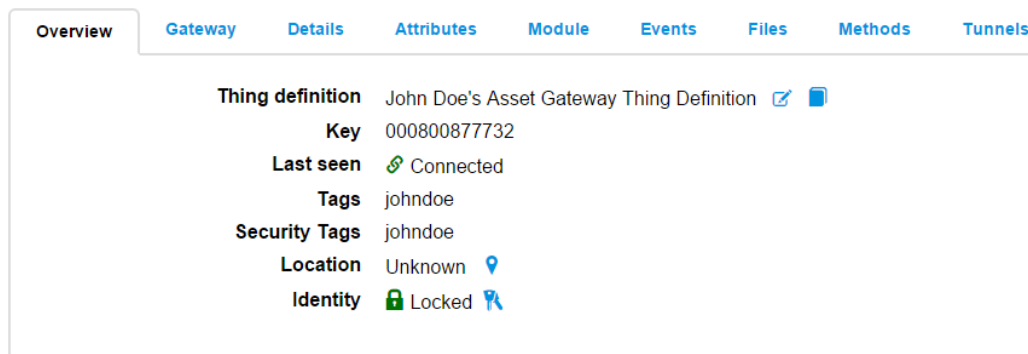
- 5) Click the **Save** button, then the **Start** button at the bottom of the **TR50 Connection Management** Tab.
- 6) Wait one minute for the status of the **TR50** change to “**Connected**”, if the status does not change within one minute click the **Refresh**. If your gateway status still does not change make sure that your gateway is connected to the internet either by a LAN or cellular connection and that you copied and pasted your entire **Application Token**.
 

\*Your Asset Gateway should now be connected to the deviceWISE M2M Service we will verify the connection in the next step. However you need to change any of the default configurations or want a detailed description of what each parameter refers to you can read more at [help.devicewise.com](http://help.devicewise.com).
- 7) On the **Management Portal** click on the **Things** page.
- 8) You should see your Asset Gateway with the name “**auto:xxxxx**” (xxxxx referring to your Asset Gateway’s **Thing Key** in this case its MAC Address). If you are in an Organization with a lot of Things you can sort the list by selecting one of the reference **Tags** on the top of the Things page.

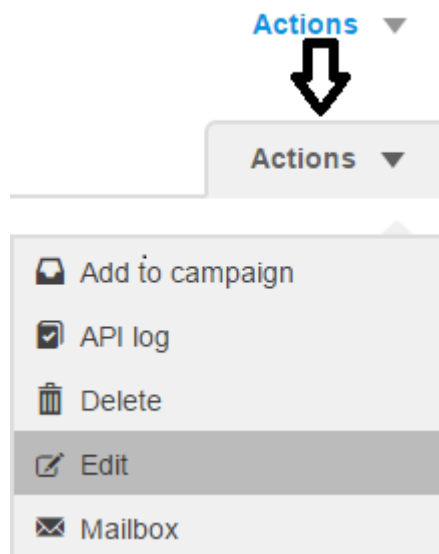
9) Select the “view” icon to see an overview page of your device.



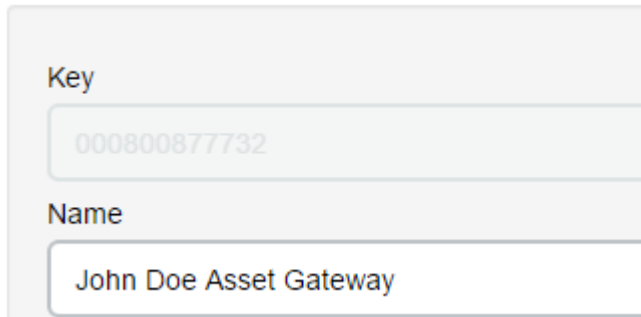
10) From here you can see information about your gateway and manage different aspects of your gateway.



11) Your gateway was auto registered with the M2M Service so it was named “auto” with its thing key as its name on. To change what name shows up on the **Management Portal** click the **Actions** button and then select **Edit**.



**12)** Change the **Name** of your Asset Gateway to something meaningful to you.



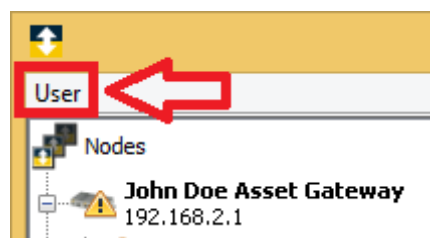
**13)** Click the **Update** button to save your changes.

## Connecting to your Asset Gateway over TR50

Now that you are connected to **TR50** you can connect to your Asset Gateway without an Ethernet connection.

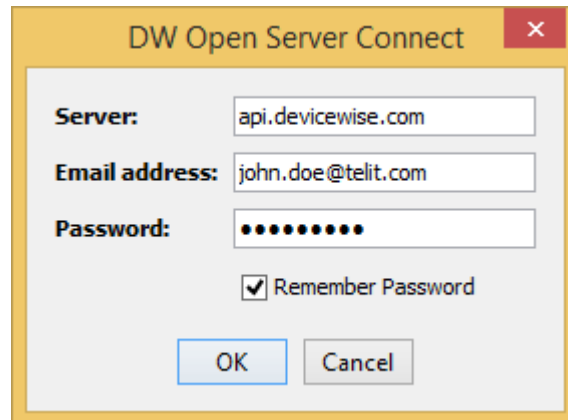
**1)** Navigate back to the **deviceWISE Workbench**.

**2)** Click on the **User** option on the main workbench window.



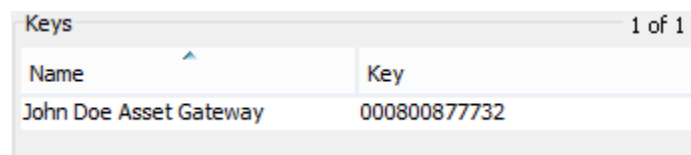
**3)** Fill in your information:

- **Server:** api.devicewise.com
- **Email Address and password:** the same email and password you use to log into the **Management Portal**.



A dialog box titled "DW Open Server Connect" with a close button (X) in the top right corner. It contains three input fields: "Server:" with the value "api.devicewise.com", "Email address:" with the value "john.doe@telit.com", and "Password:" with a masked password "••••••••". Below the password field is a checkbox labeled "Remember Password" which is checked. At the bottom are "OK" and "Cancel" buttons.

- 4) Disconnect your computer from whatever local LAN connection binds your computer to your Asset Gateway.
- 5) Right-click on the **Nodes** icon in the top left-hand corner of the workbench and select **Scan**.
- 6) Select the **TR50** as your **Scan Type**.
- 7) Select your gateway from the list of connected devices and then click the **Connect** button.



A dialog box titled "Keys" with a close button (X) in the top right corner. It contains a table with two columns: "Name" and "Key". The table has one row with the values "John Doe Asset Gateway" and "000800877732". The text "1 of 1" is displayed in the top right corner of the dialog box.

Name	Key
John Doe Asset Gateway	000800877732

You are now connected to your Asset Gateway over a **TR50** connection. Either over an Ethernet LAN or Cellular Network connection, through the **deviceWISE M2M Service**, and back down to your computer. As long as your gateway maintains an internet connection you will be able to access your gateway from anywhere in the world as if you were still connected to it locally. The responsiveness of **TR50** connection is determined by the signal quality your computer and your gateway have to the internet. If you find the **TR50** connection sluggish or find that it drops out often it is probably caused by poor signal quality. In that case reconnect to your gateway over a local connection again; since a **TR50** connection mimics a local connection nothing will change.

## Section 6

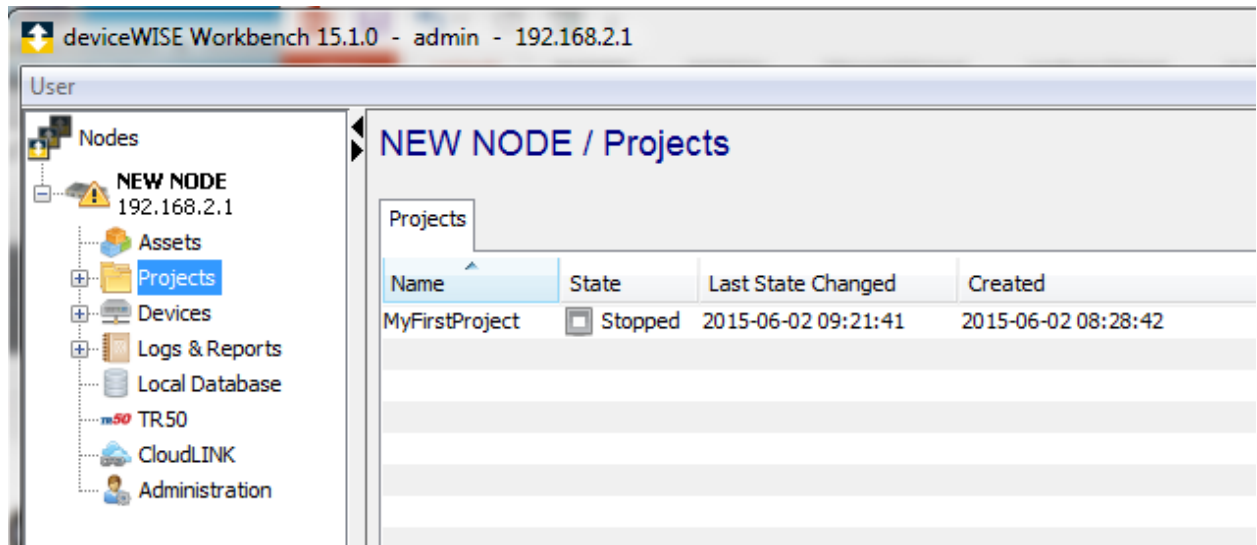
### **Creating Projects and Writing Triggers for your Asset Gateway**

When working with an Asset Gateway you will be implementing what is known as Application logic. Application logic is defined in one or more *Triggers*. The available trigger event types and trigger actions on a node depends on the product installed and the packages or extensions installed on the node.

Each trigger resides in, or belongs to, a project. A project is a container item which is used to organize and control triggers. You can create as many projects as needed for a particular node and define the triggers needed within each project. The organization of projects and triggers does not limit the availability or access of the triggers, it is just a mechanism to organize the application logic in a manner that makes sense to the application developer. If you want to learn more about Triggers please visit [help.devicewise.com](http://help.devicewise.com).

### **Creating your First Project**

- 1) On the deviceWISE Workbench double-click on your gateway's icon to expand the list of features.
- 2) Select the **Project** icon. The **Projects** window appears as the right hand pane (the list of projects will be empty if no projects have been defined).
- 3) From the bottom of the **Projects** tab, select **New**. The Create Project window appears.
- 4) Enter "**MyFirstProject**" as the name of your project and then select **OK**. A project name can be up to 64 characters in length and can include letters, numbers, and the underscore character. Spaces are also allowed. Once you name your project it is added to the **Projects** tab on the Workbench right hand pane.



The Projects tab has a table format with these columns:

Column	Description
<b>Name</b>	The name of the project
<b>State</b>	<p>Projects have a state that is separate from the state that is separate from the state its triggers. The state of a project are:</p> <ul style="list-style-type: none"> <li>• <b>Started</b> – The project is started. A project must be started in order for its triggers to be loaded and available to be executed.</li> <li>• <b>Stopped</b> – The project is stopped. All of the triggers in the project will have a status of unloaded and are not available to be executed.</li> </ul>
<b>Last State change</b>	Displays the date and time the project was last started or stopped.
<b>Created</b>	Displays the date and time the project was originally defined.

## Creating a Trigger

The main concepts of a trigger are:

- The trigger's event type
- The trigger's local variables, static variables, macros and event variables
- The trigger's settings
- The trigger's actions, including the success and failure routes between actions.

When you define a trigger, you name the trigger, identify the event type (data, scheduled, on-demand and so forth), define the event parameters, and then configure one or more actions.

As the definition of the trigger progresses, you can use the **Validate** button to check for correctness and completeness of the definition. When the trigger is saved, it is written to an internal database file on the node. You can edit, validate and save the trigger definition multiple times as the trigger's application logic is defined. You can use the trigger report feature to generate a report of the trigger's execution to help understand how the trigger's execution progresses through its actions and the actions' success and failure routes.

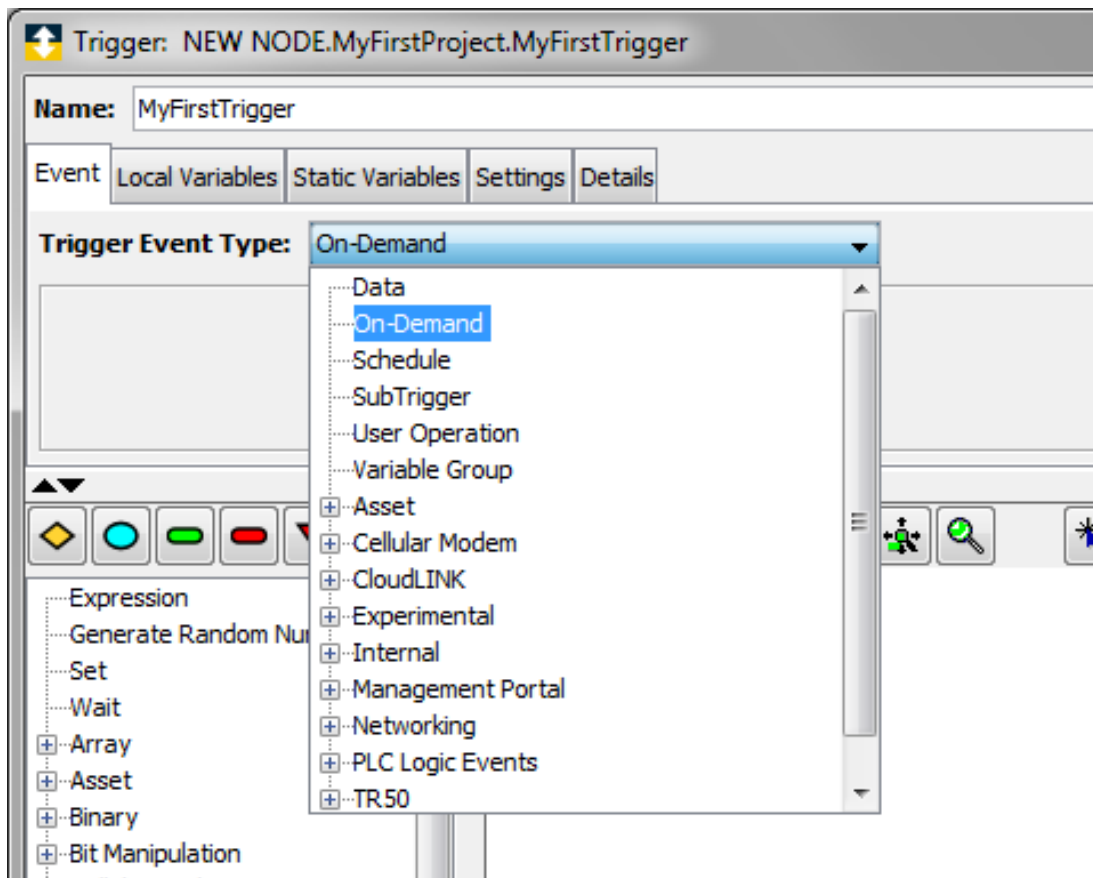
A Trigger's behavior or application logic is defined by using what are known as **Trigger Actions**. **Trigger Actions** are individual function blocks that provide a single step in the trigger's application logic. The execution flow, or route, through the trigger's actions can be controlled based on an action's result and run time values of variables. If you are familiar with coding you can think of an **Action** as a Method Object.

The trigger actions available in the trigger editor depends on the type of product installed on the node (i.e. Enterprise Gateway vs Asset Gateway), as well as the packages and extensions installed on the node (i.e. the device drivers).

If you want a full list of triggers currently available on the deviceWISE Workbench you can check out the [Trigger actions reference](#).

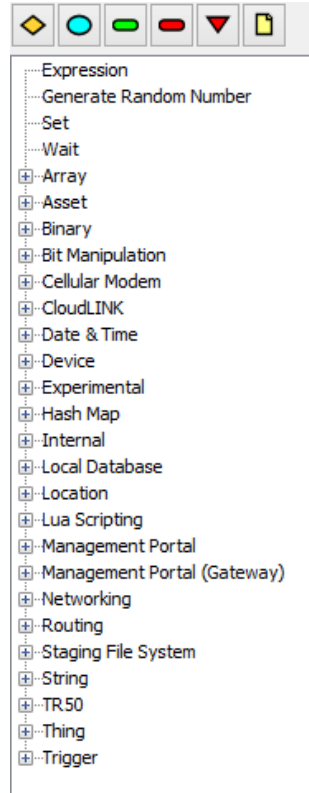
The Workbench supports two trigger editors: the **List Editor** and the **Canvas Editor**. Triggers can be initially defined using either editor and then edited and saved with the other editor. For consistency of the underlying trigger definition's action location (Canvas Editor) and action number (List Editor), it is usually best to consistently use one editor or the other for a trigger. For this guide we will be using the Canvas Editor.

- 1) Within the **Projects** window of the workbench select your “**MyFirstProject**” and then click the **Start** button at the bottom of the **Projects** window; the grey square in the **State** column of your “**MyFirstProject**” will change to a green checkmark with the word **Stated** next to it.
  
- 2) Open (double-click) your project.
  
- 3) Click the **New** Button at the bottom of your project’s window. This will open the **Canvas Editor**.
  - Inside the **Canvas Editor** name your trigger “**MyFirstTrigger**” and set the **Trigger Event Type** to “**On-Demand**”.





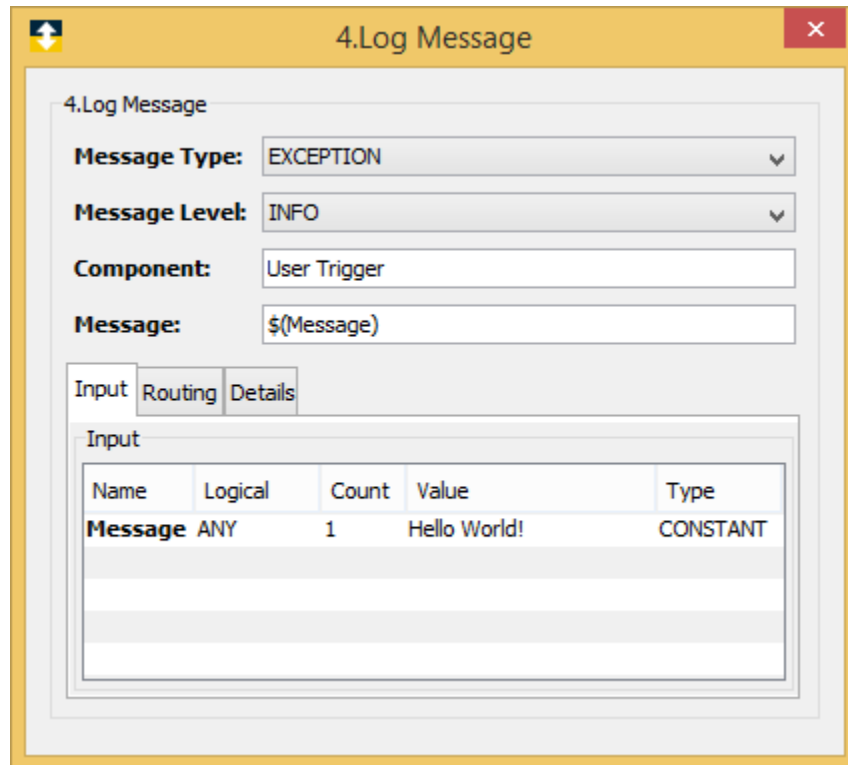
- On the Left-hand side of the **Trigger Window** there is the list of **Action Groups** this where each individual **Action** is located.



4) Open the **Internal** action group, Click on the **Log Message** Action. Move your cursor underneath the **Start Action** (The Green Triangle); then click on any empty space underneath the **Start Action**. You should now have a new **Log Message** action in your trigger's **Action Diagram**.

5) Double-Click on the **Log Message** Action.

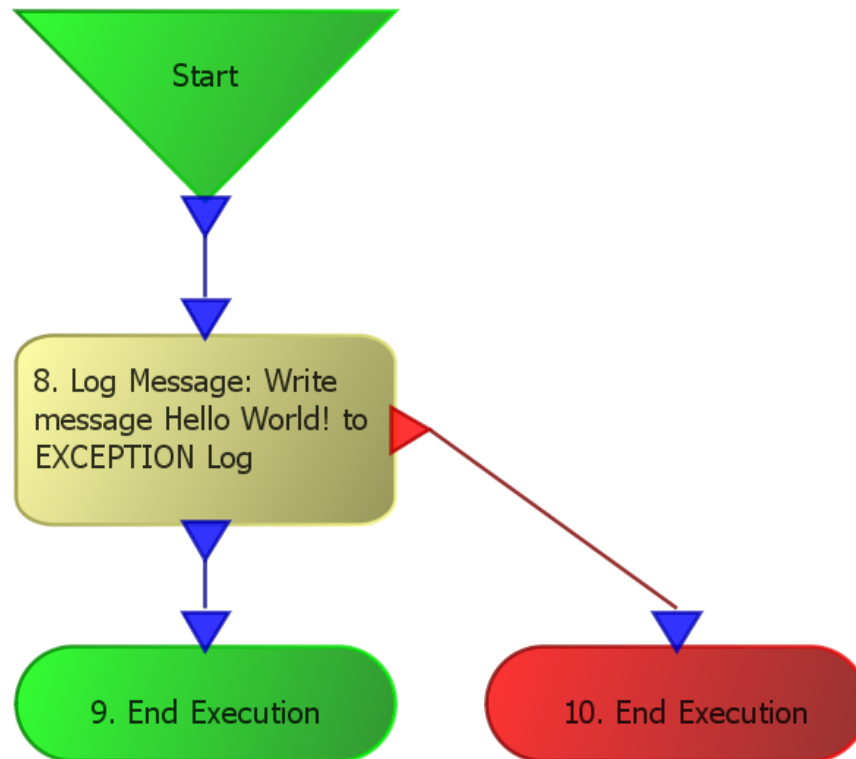
6) In the **Value** section next to the **Message** variable type **"Hello World!"**.



Name	Logical	Count	Value	Type
Message	ANY	1	Hello World!	CONSTANT

- 7) Close the **Log Message** window.
  
- 8) Click on the **End Execution (Success)** Action (The Green Oval) above the **Action Groups**. Then click on a blank spot underneath your **Log Message** Action.
  
- 9) Now Click and hold your cursor on the **Blue Arrow** at the bottom of the **Log Message** Action, then drag your cursor to the **Blue Arrow** at the top of the **End Execution (Success)** Action (it should turn Yellow), then let go of your cursor.
  
- 10) Click on the **End Execution (Failure)** Action (the Red Oval) above the **Action Groups**. Then click on a blank spot to the left of the **End Execution (Success)** Action.

- 11)** Now Click and hold your cursor on the **Red Arrow** on the left-hand side of the **Log Message** Action, then drag your cursor to the **Blue Arrow** at the top of the **End Execution (Failure)** Action (it should turn Yellow), then let go of your cursor. Your Trigger should now look something like this:



**Note:** How organized your Action Diagram is does not matter. What matters are the connections between each individual Action. However the more organized a Trigger is, the easier it is to modify and maintain.

- 12)** Click the **Validate** button on the bottom of the trigger window. This will run some checks on the correctness of the Trigger's structure.

\*If a prompt displays saying "Tigger definition validated" then you have done everything correctly, click "OK". If not go back to "Step 5" and make sure have done everything correctly until now.

- 13)** Click the **Save** button to save your **"MyFirstTrigger"** (very important step !).

- 14) Right-click on your **“MyFirstTrigger”** and then select **Start**.
- 15) Right-click on your **“MyFirstTrigger”** and then select **Fire Trigger**. This will execute the trigger.  
  
\*You should see “1” (or more than 1) under the Successes column for your **“MyFirstTrigger”** (you might need to click the **Refresh** button at the bottom of **“MyFirstProject”** window).
- 16) Open (double-click) the **“Logs & Reports”** folder in the list of features for Asset Gateway
- 17) Click on the folder named **“Exceptions Log”** in the list of logs under the **Messages** tab you should see the **“Hello World!”**.

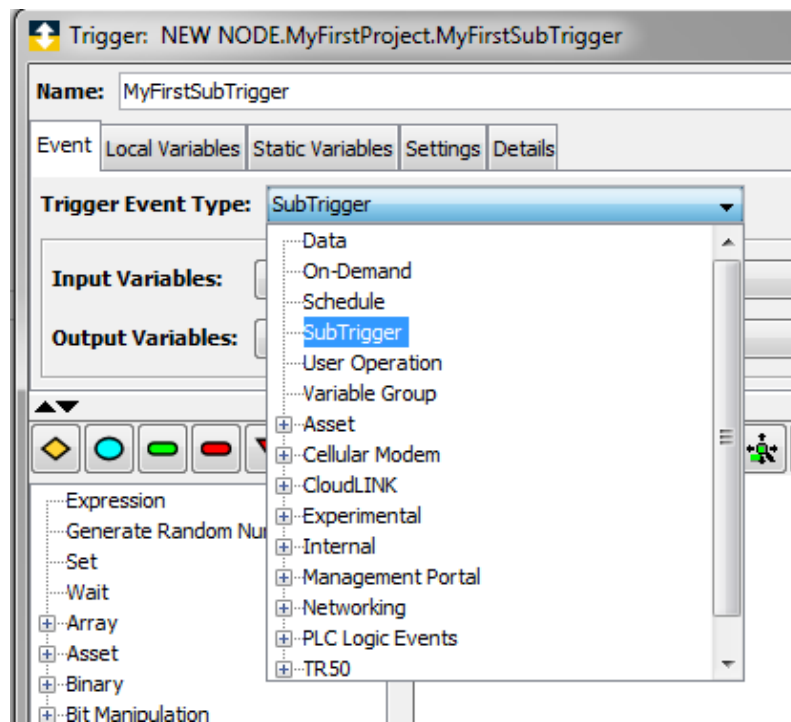
Congratulations, you have successfully created your first deviceWISE Trigger.

## Creating a Sub-trigger

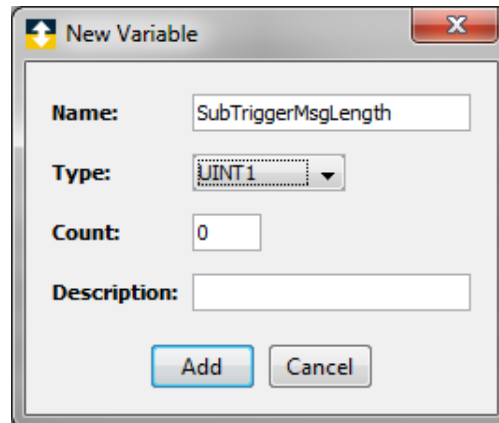
In this example you will be creation and executing a Sub-Trigger, which is a Trigger that can be called (invoked) from within another Trigger. To do this you will also be creating another Trigger to call your Sub-Trigger.

- 1) Navigate back to your **“MyFirstProject”**.
- 2) Right-click on your **“MyFirstTrigger”** and select **Duplicate**.
- 3) In the **New Trigger** box type **“MyFirstSubTrigger”**; then click **OK**.

- 4) Start your “**MyFirstSubTrigger**”.
- 5) Open (double-click) your “**MyFirstSubTrigger**”; you will see the exact same **Action Diagram** as in your “**MyFirstTrigger**”.
- 6) Under **Trigger Event Type** change it from **On-Demand** to **SubTrigger**.



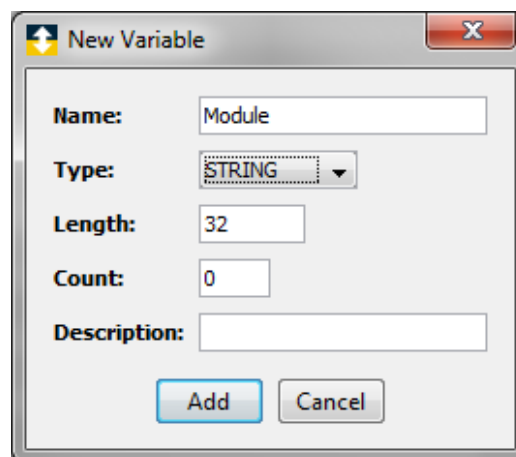
- 7) Now we are going to create an **Output Variable**. Near the top of your “**MyFirstSubTrigger**” window click the “**Configure...**” button to the right of “**Output Variables:**” label.
- 8) Click the **Add** button and name the new variable “**SubTriggerMsgLength**” and make its Type a **UNIT1**. In this case a **UNIT1** variable is a 1 unsigned byte, if you want to learn more about the different data types available for each variable you can check out a reference document [here](#).



9) Click the **Add** button at the bottom of the **New Variable** window, then click the **OK** button of the **Variables** window. This will add the variable to the Sub-Trigger.

10) Now we are going to create four **Input Variables**. Near the top of your **“MyFirstSubTrigger”** window click the **Configure** button to the right **“Input Variables: ”** label.

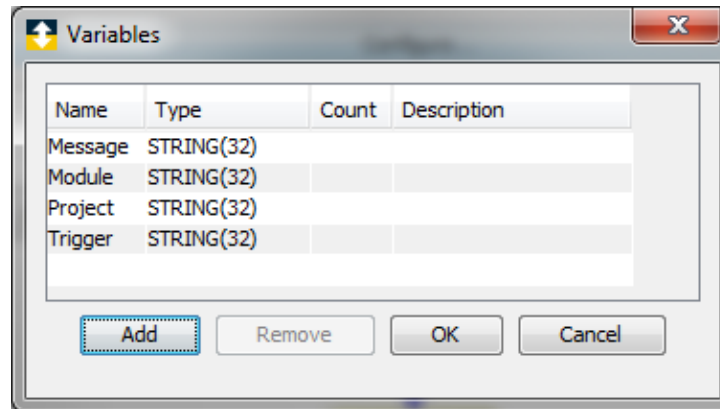
11) Click the **Add** button at the bottom of the **Variables** window to create a variable named **“Module”** and make its **Type String** and **Size 32**. Click the ‘Add’ button to add the variable to the Trigger.



12) Repeat this process for three other event variables named:

- **Project**
- **Trigger**
- **Message**

The result should be four variables defined as such:



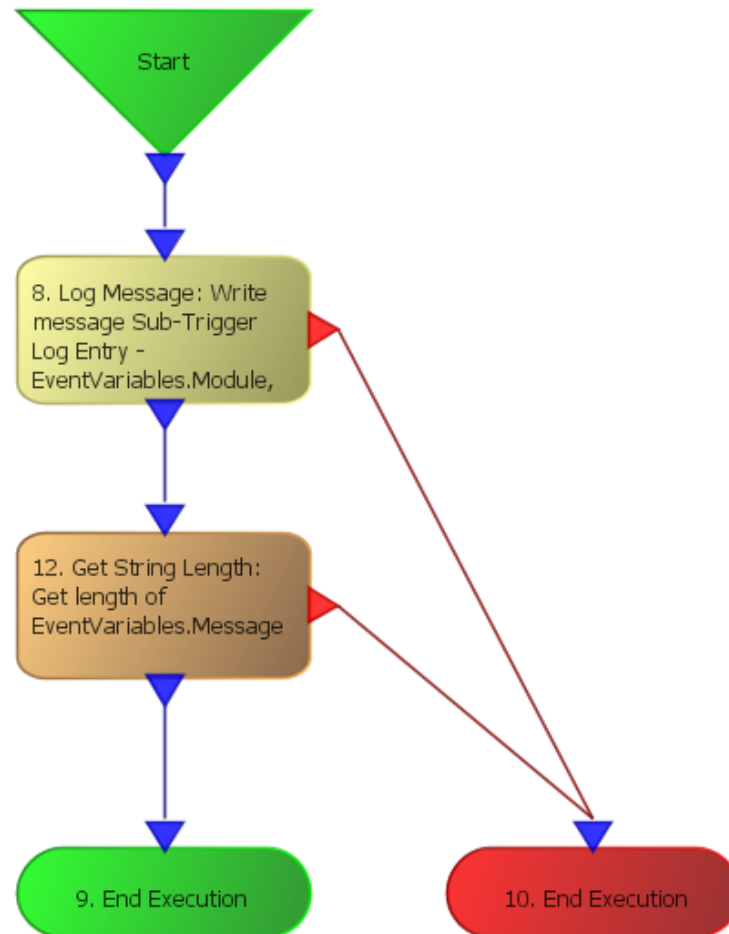
13) Click the **OK** button to save and close the **Variables** window.

14) Next we need to make some room on the Canvas Editor to add a new Action. Click and drag the **End Execution (Success) Action** downward moving the End Execution Action lower.

15) You are going to add the **Get String Length** Action. In the Action Groups list along the left side of the screen click the **String** group to expand it. Listed are String-related Trigger Actions.

16) Select the **Get String Length** Action, then put it in the space you made between the **Log Message** Action and the **End Execution (Success)**.

17) Connect the **Get String Length** Action into your “**MyFirstSubTrigger**” Action Diagram. When you are finished your trigger structure should look similar to this:



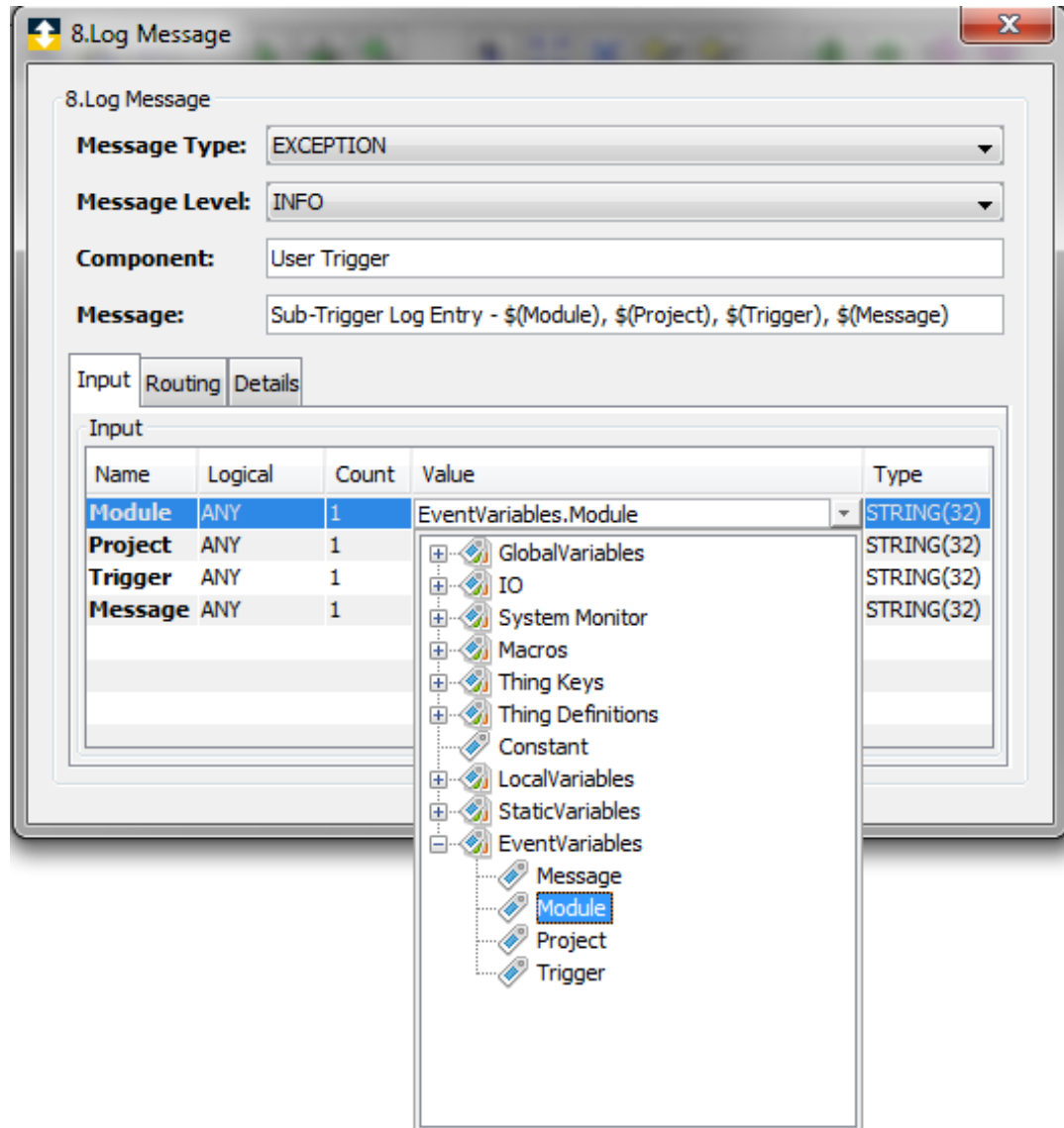
- 18)** Now let us add some functionality to this **Sub-Trigger**. Open (double-click) the **Log Message** Action. In the **Message** field type (or copy/paste):

**Sub-Trigger Log Entry - \$(Module), \$(Project), \$(Trigger), \$(Message)**

The parts inside the **\$( )** are variable inserts, and allow you to insert data into your log message. This variable insert pattern is use by many deviceWISE Trigger Actions.

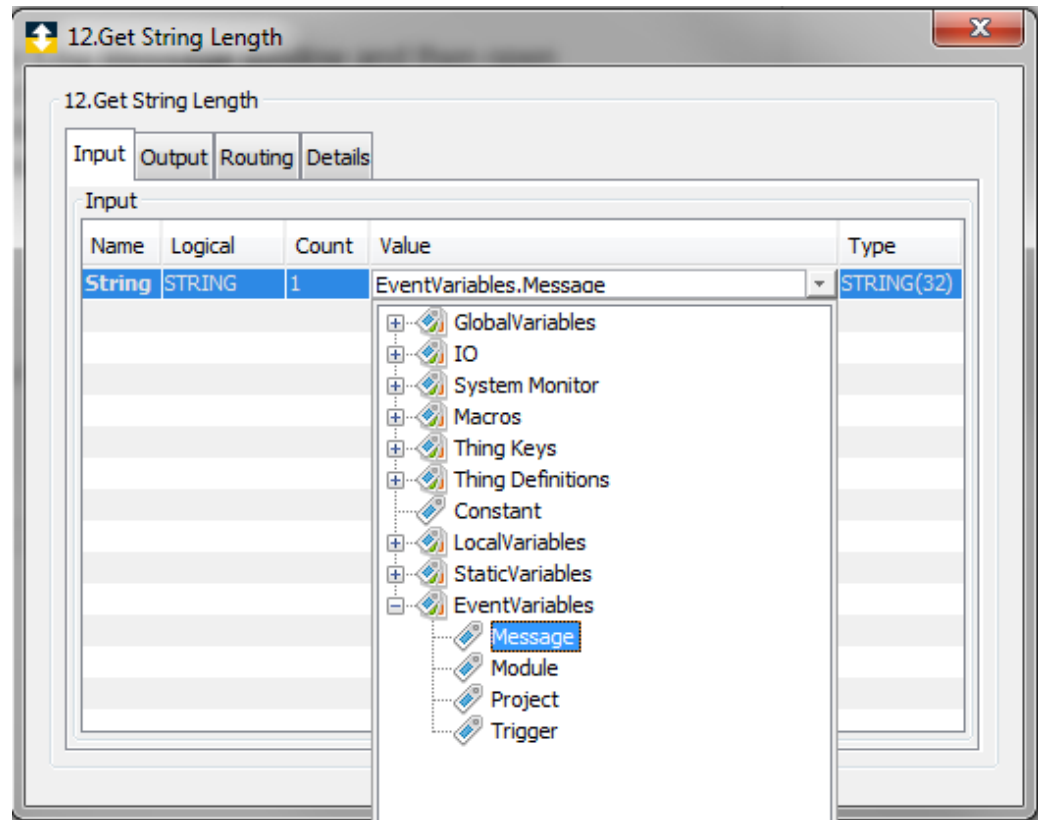
- 19)** Now select the corresponding **Event Variable name** with the name that is in the **Input Tab** of the **Log Message** Action window, for each of the four **Event Variables**.



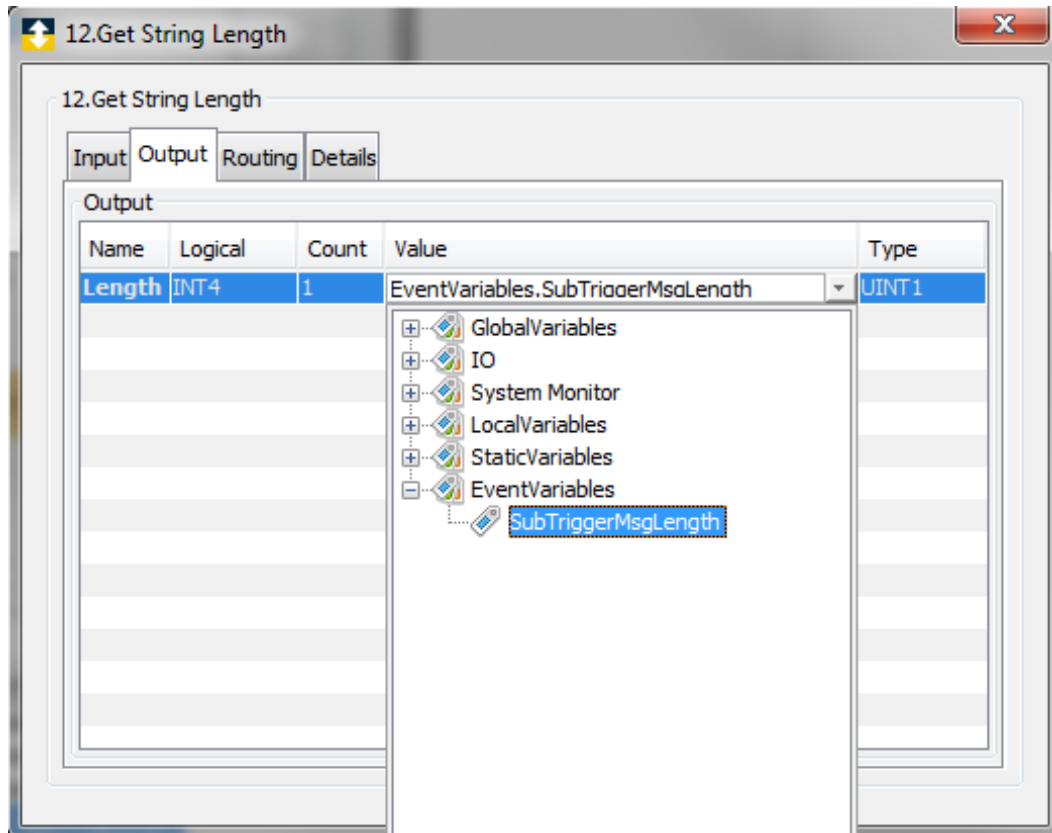


**20)** You are done with this action so close the **Log Message** window.

**21)** Now open (double-click) the **Get String Length** Action. Under the **Input** Tab you will see a predefined Input variable label named **String**. Under the **Value** section find and select your **Event Input Variable “Message”**.



**22)** Under the **Output** Tab you will see a predetermined variable label named **Length**. Under the **Value** section find and select your **Event Output Variable** “**SubTriggerMsgLength**”.



**23)** You are done updating so close the **Get String Length** window.

**24)** Let's check for errors. Click the **Validate** button on the bottom of the trigger window.

**25)** If a prompt displays saying "Tigger definition validated" then you have done everything correctly, click "OK". If not, go back to the beginning of this section and review each step to determine, and fix, the error.

**26)** Click the **Save** button to save your "**MyFirstSubTrigger**" (again, very important step).

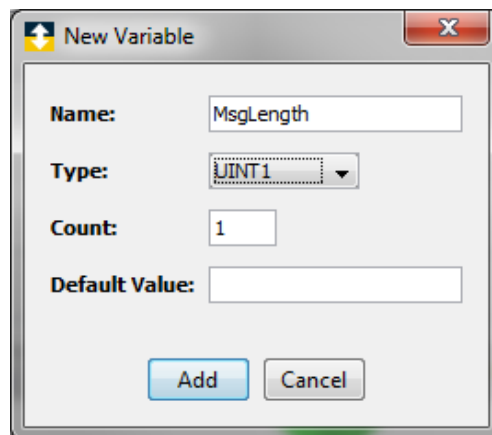
**27)** Now let us create a **Trigger** to call your **“MyFirstSubTrigger”**. **Duplicate** your **“MyFirstTrigger”** as did before and name the new one **“MySecondTrigger”**.

**28)** Start your **“MySecondTrigger”**.

**29)** Open (double-click) your **“MySecondTrigger”**.

**30)** Create a **Local Variable** inside **“MySecondTrigger”**. Near the Top of your **“MySecondTrigger”** window click the **Local Variables** Tab.

**31)** Click the **Add** button to create a new local variable named **MsgLength** and **Type** **UNIT 1**.



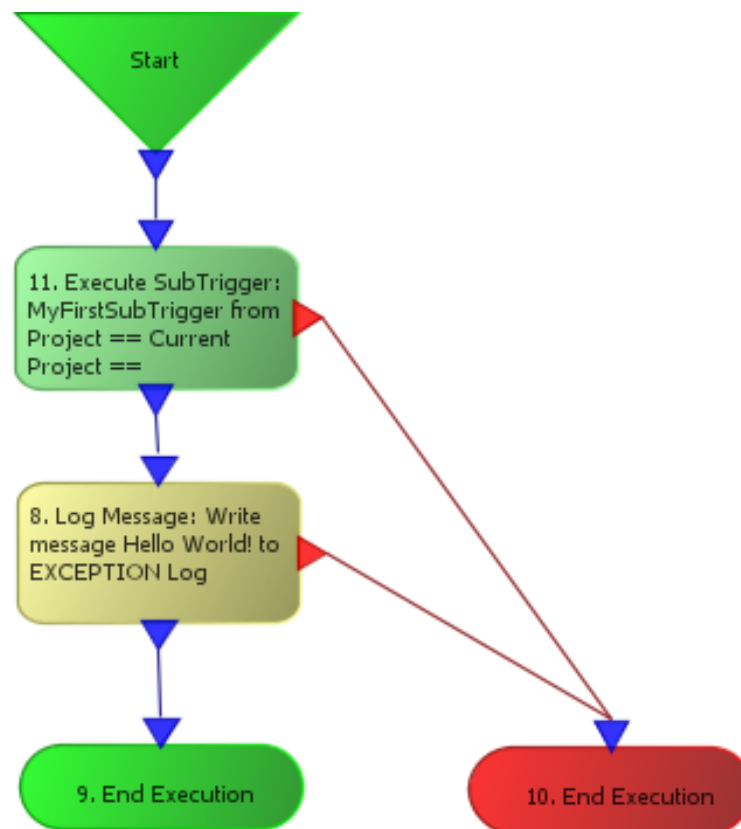
**32)** Click the **Add** button in the **New Variable** window.

**33)** Click and drag the **End Execution (Success) Action** downward to make room for a new **Action**; then do the same for the **Log Message Action**.

**34)** Click the **Trigger** section under the **Action Groups** list.

**35)** Select the **Execute SubTrigger** Action, then put it in the space you made between the **Log Message** Action and the **Start** Action.

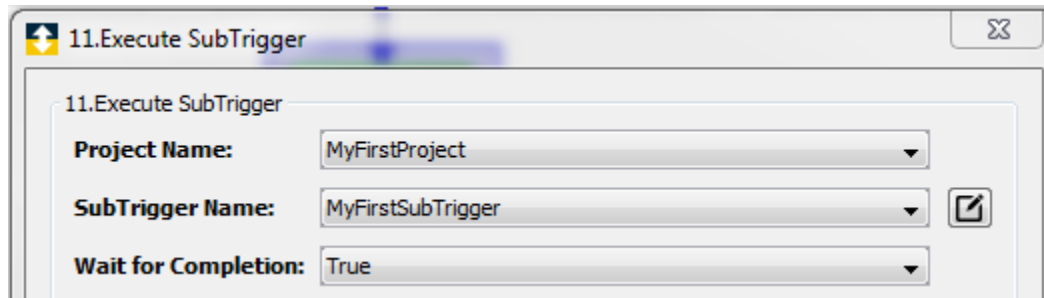
**36)** Connect the **Execute SubTrigger** Action into your “**MySecondTrigger**” structure. When you are finished your Trigger’s Action Diagram should look similar to this:



**37)** Open (double-click) the **Execute SubTrigger** Action.

**38)** Set the **Project Name** to “**MyFirstProject**”.

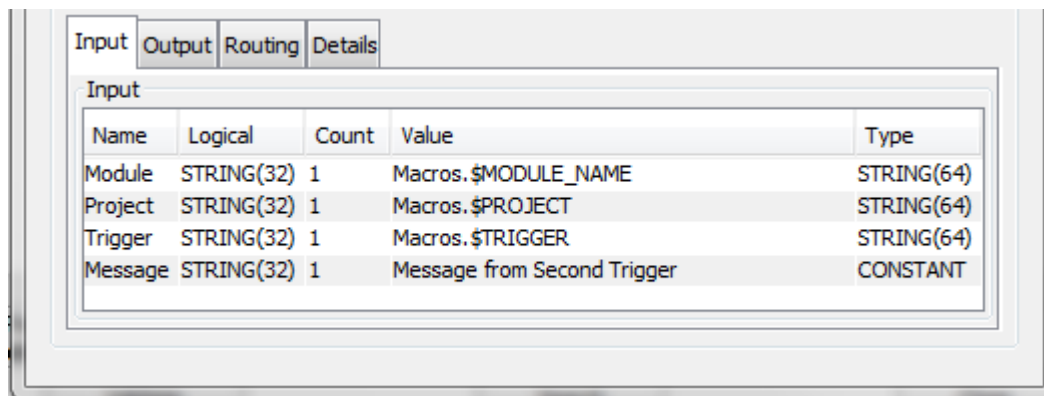
39) Set the **SubTrigger Name** to “**MyFirstSubTrigger**”.



40) Under the **Input** Tab you will see the same four input variables from your “**MyFirstSubTrigger**”. Now select the corresponding **Macros** variable name with the name that is in the **Input** Tab of the **Execute SubTrigger** Action window, for the variables **Module**, **Project**, **Trigger**, but not for **Message**.

\*‘Trigger Macros’ are a set of internal data that is made available to Triggers. They appear as variables in the variable drop-down list.

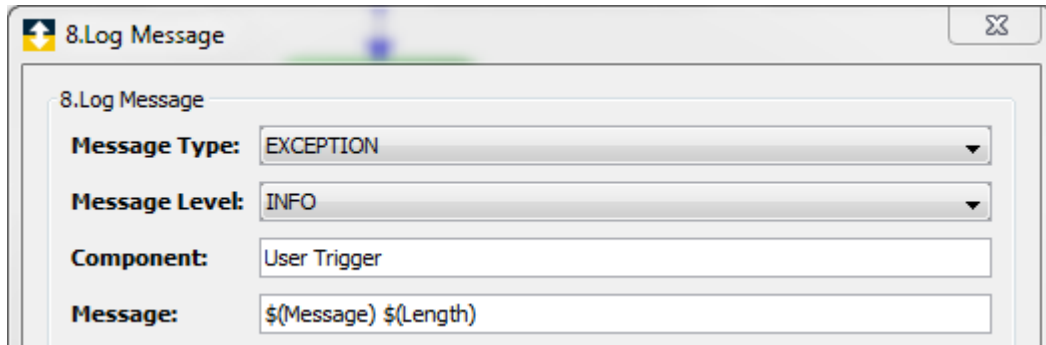
41) For **Message** Type “**Message from Second Trigger**” into the **Value** section next to the **Message** name.



42) Close the **Execute SubTrigger** window.

43) Open (double-click) the **Log Message** Action.

44) Type “\$(Message) \$(Length)” into the **Message** box



8.Log Message

8.Log Message

**Message Type:** EXCEPTION

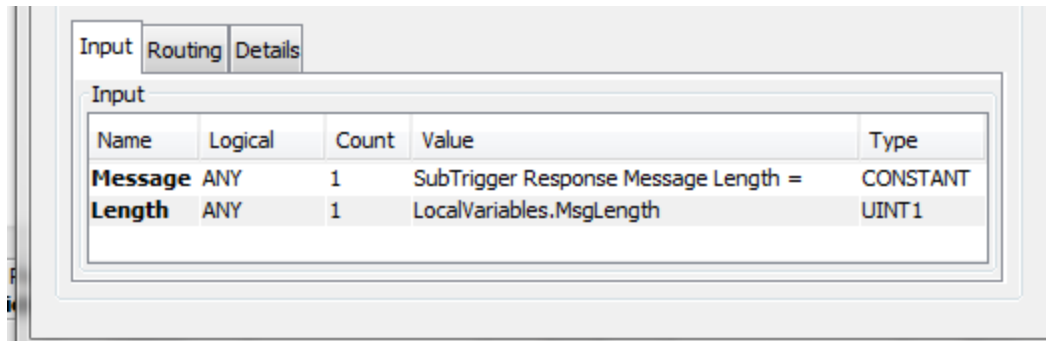
**Message Level:** INFO

**Component:** User Trigger

**Message:** \$(Message) \$(Length)

45) In the value section of the **Length** label select the local variable “**MsgLength**”.

46) Type “**SubTrigger Response Message Length =**” into the value section of the **Message** label.



Input Routing Details

Input

Name	Logical	Count	Value	Type
<b>Message</b>	ANY	1	SubTrigger Response Message Length =	CONSTANT
<b>Length</b>	ANY	1	LocalVariables.MsgLength	UINT 1

47) Close the **Log Message** window.

48) Click the **Validate** button on the bottom of the trigger window.

49) If a prompt displays saying “Tigger definition validated” then you have done everything correctly, click “OK”. If not go back to **Step 28** and make sure have done everything correctly until now.

- 50) Click the **Save** button to save your **“MySecondTrigger”** (important, important, important).
  
- 51) Right-Click on your **“MySecondTrigger”** and then select **Fire Trigger**. You should see “1” or more than 1 under the **Successes** column of your **“MySecondTrigger”** (if not click the **Refresh** button at the bottom of **“MyFirstProject”** window).
  
- 52) Open (double-click) the **“Logs & Reports”** folder in your Asset Gateway’s features list.
  
- 53) Click on the folder named **“Exceptions Log”** in the list of logs under the **Messages** tab you should see your message from your **“MyFirstSubTrigger”** and the message from your **“MySecondTrigger”** along with the size of the Message that you put in your **“MyFirstTrigger”**.

Congratulations, you have successfully created your first deviceWISE Sub-Trigger and your second deviceWISE Trigger. In this example you have moved data back and forth between your Sub-Trigger and your Second Trigger and then combined the data to display the data in the Exception Logs. If you are familiar with programming terms a Sub-Trigger is very similar to a Sub-Routine or Sub-Method.

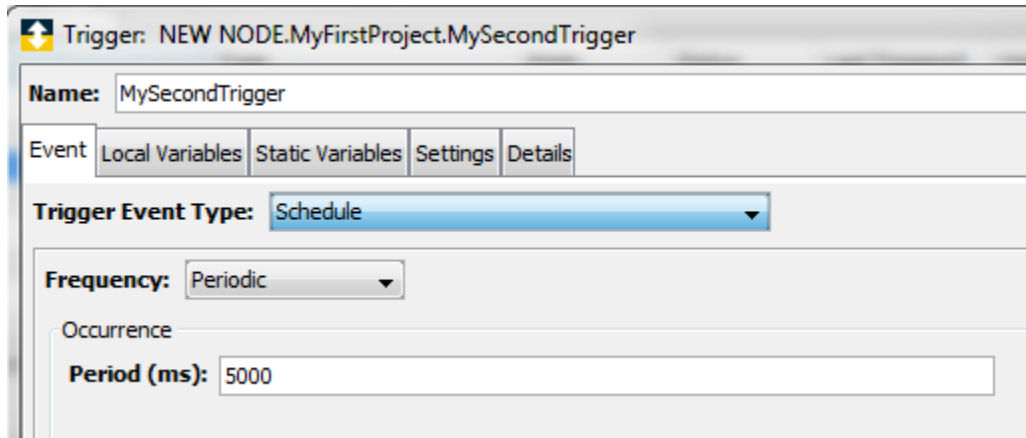
## Scheduling a Trigger

Now let us make a **Trigger** “Fire” on its own. This is done using the **Schedule** Trigger Event Type.

- 1) Navigate back to your **“MyFirstProject”** and then open (double-click) your **“MySecondTrigger”**.
  
- 2) Change your **Trigger Event Type** to **Schedule**.



- 3) Then change the **Period** to **5000 ms** (5 Seconds).



- 4) **Validate** and **Save** your “**MySecondTrigger**”.

- 5) Navigate back to “**Exception Logs**” under “**Logs & Reports**” folder. You should see the same messages from the end of the last example, but they are coming in every 5 seconds (If they are not click the **Refresh** button).

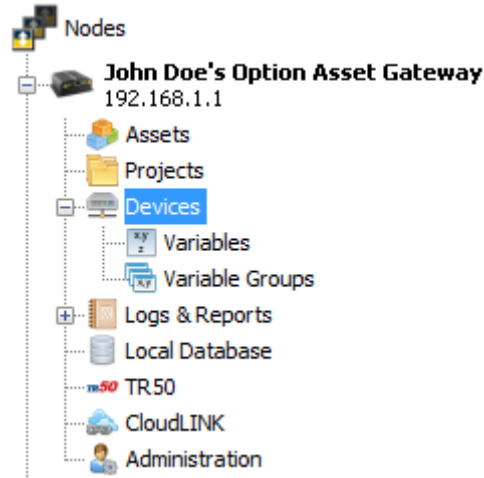
Congratulations you have converted your “**MySecondTrigger**” from an On-Demand trigger to a Scheduled trigger. You can use a scheduled trigger to perform any action on a periodic basis of time.

## Publishing a Message to the M2M Service

Now that you have everything setup and running let us try to publish a message to the M2M Service. To do this we are going to create a new Trigger.

- 1) On the **deviceWISE Workbench** expand your gateway’s list of features.

2) Select the “**Devices**” icon.



3) Right-click on the “**System Monitor**” device and select “**Start**”.

Name	Type	State	Last State Changed
System Monitor	System Monitor	<input type="checkbox"/> Stopped	
			New
			Edit
			Delete
			References
			Start

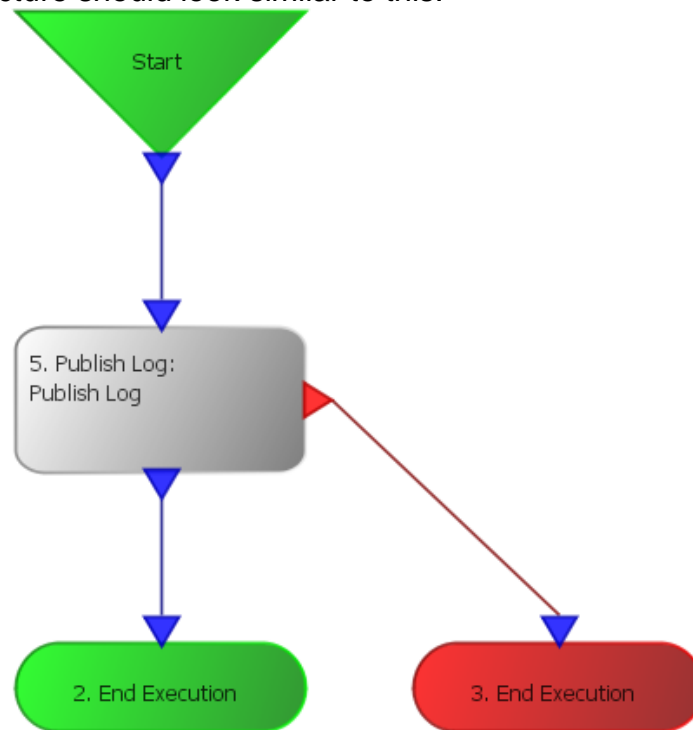
4) Select the “**Variables**” tab and expand the “**System Monitoring**” tree.

5) Right-click on the “**CPU Usage**” and select “**Read**”. The current state of the CPU will be shown in the “**Value**” section of the CPU Usage in percentages.

Name	Type
System Monitor	System Monitor
CPU	CPU
CPU Usage	1 ITNT 1
60 Second Ave	
System Memory	

6) Navigate back to your “**MyFirstProject**”; create a new Trigger called “**MyPublishTrigger**”.

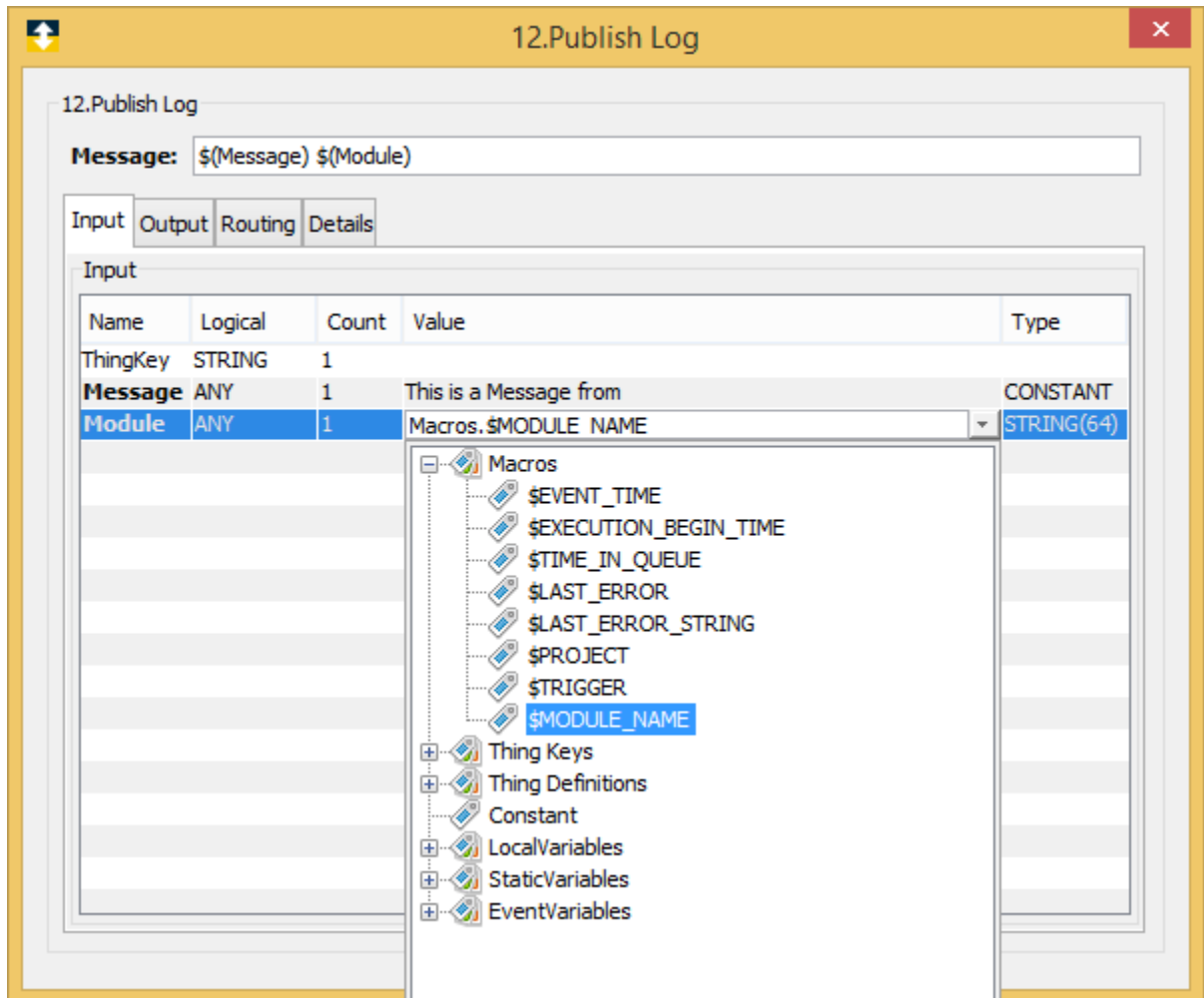
- 7) Set the **Trigger Event Type** to “**Schedule**”, with a **Period** of **5000 ms** (5 Seconds).
- 8) Click the **Thing** section under the **Action Groups** list.
- 9) Select the **Publish Log** Action, then put it in the space below the start Action.
- 10) Add the **End Execution (Success)** and the **End Execution (Failure)** Actions respectfully to your “**MyPublishTrigger**” Action Diagram. When you are finished your trigger structure should look similar to this:



- 11) Open (double-click) the **Publish Log** Action.
- 12) In the **Message** box type: **\$(Message) \$(Module)**.

13) In the **Value** section next to **Message** label type: “This is a Message from\_”.

14) In the **Value** section next to **Module** variable select “Macros.\$MODULE\_NAME”.



15) Close the **Publish Log** window.

16) **Validate** and **Save** your “MyPublishTrigger”.

17) Start your “MyPublishTrigger”

18) Navigate to the **Things** page on the Management Portal and find your **Asset Gateway**, then click the **view** icon.

19) Click the **Events** tab under your **Asset Gateway**. You should see your Message that you typed into the **Publish Log** Action as well as the name of your **Asset Gateway**.

## John Doe's Asset Gateway

Overview

Gateway

Details

Attributes

Module

Events

Files

Tunnels

Aug 27, 2015 00:51:03

This is a Message from John Doe's Asset Gateway

Aug 27, 2015 00:50:58

This is a Message from John Doe's Asset Gateway

Aug 27, 2015 00:50:53

This is a Message from John Doe's Asset Gateway

Aug 27, 2015 00:50:48

This is a Message from John Doe's Asset Gateway

Aug 27, 2015 00:50:43

This is a Message from John Doe's Asset Gateway.

More events

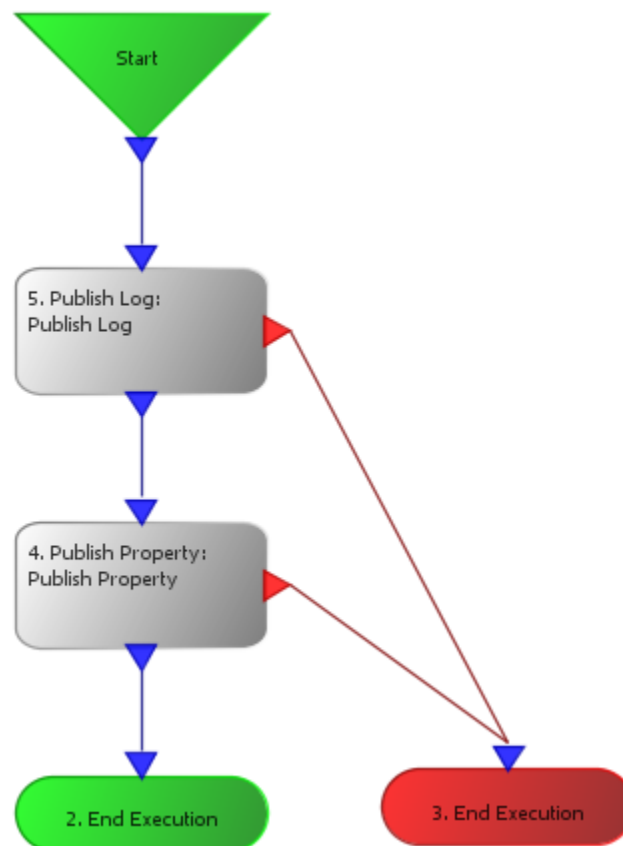
Congratulations you have successfully published a message to the M2M Service. In this example you created a simple Trigger that published a message to the M2M Service. You can use a trigger similar to this to keep you apprised of activity on or about your gateway or connected devices.

### Publishing Data to the M2M Service

Now that you have published a message to the M2M Service it is time to publish some data to the M2M Service.

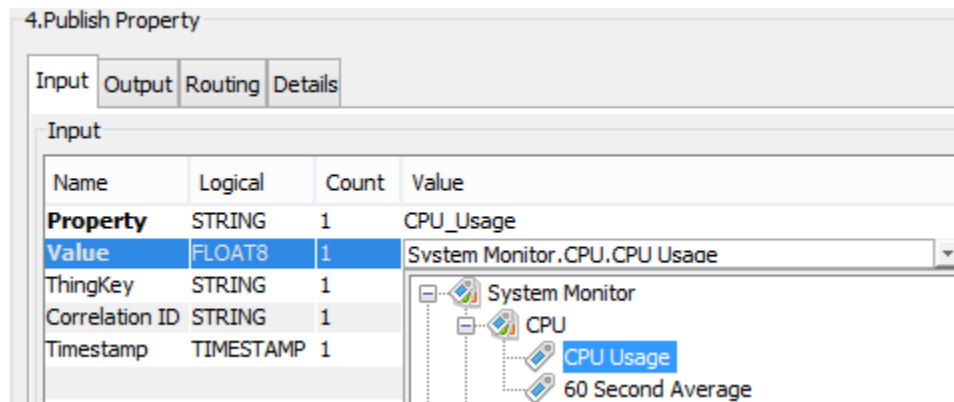
1) Navigate back to your “MyFirstProject” and open your “MyPublishTrigger”.

- 2) Click and drag the **End Execution (Success) Action** downward to make room for a new **Action**.
- 3) Click the **Thing** section under the **Action Groups** list.
- 4) Select the **Publish Property** Action, then put it in the space you made between the **Publish Log** Action and **End Execution (Success)** Action.
- 5) Connect the **Publish Property** Action into your “**MyPublishTrigger**” structure. When you are finished your Trigger’s Action Diagram should look similar to this:



- 6) Open the **Publish Property** Action.

- 7) In the Value section next to the **Property** variable type: **CPU\_Usage**.
- 8) In the Value section next to the **Value** variable select the **System Monitor.CPU.CPU Usage**. This variable refers to the CPU usage of your gateway in terms of percentage.



- 9) Close the **Publish Property** Action.
- 10) **Validate** and **Save** your “**MyPublishTrigger**”.
- 11) Navigate to your Asset Gateway’s **Thing** page.

12) Below the basic information of your Asset Gateway you will see section called **Properties**. Within that section you will see a data graph named “**auto:cpu\_usage**”. This is the memory data coming off your gateway within an auto generated graph. Move your mouse over the graph to see the value of the data at each time interval.

13) To change the name of the data graph navigate to the **Developer** page on the **Management Portal**.

**14)** Then select the **Thing definitions** page.

**15)** Find the **Thing definition** that you created for your Asset Gateway and click the view button next to it.

**16)** Select the **Edit** button.

**17)** Select the **Properties** tab.

**18)** You will see the property that was auto generated when you added the **Publish Property** to your “**MyPublishTrigger**”. Delete the “auto:” from the name of the property and then click the “**Update**” button.

\*Note that you could have created the same property from within your **Thing definition** and then selected that property within the **Publish Property** Action. In order to use that method of creating a property you would have to be signed into the **deviceWISE Workbench** using your **Management Portal** credentials.

Congratulations you have successfully published data to the **Management Portal**. In this example you added an Action to existing Trigger to publish plottable data to the **M2M Service**.

## **Section 7**

### **Using External I/O**

An Asset Gateway works as a bridge between the gateway and whatever device you want to control or monitor. The type of devices will be determined by what type I/O ports your gateway has available. The Option CloudGate Asset Gateway has an Ethernet, Serial, and USB host capability with addition add-in cards. For this Section you be connecting over USB to a Velleman EDU05 USB Tutor Project board. The EDU05 is a great little test board that has Analog switches, Digital switches, LCD Display,



Photocell Sensor, and a temperature sensor; we will not be using all of these sensors and switches in this section, but as you will see the deviceWISE software running on your gateway only requires you to select what I/O port you want to use without any special configurations. Also in this section we will only be using the USB port on the gateway, however the setup and configuration over any I/O port is the same.

The Velleman EDU05 board has no functionality of its own, so in this section you will be creating functionality for the EDU05 using the deviceWISE Workbench.

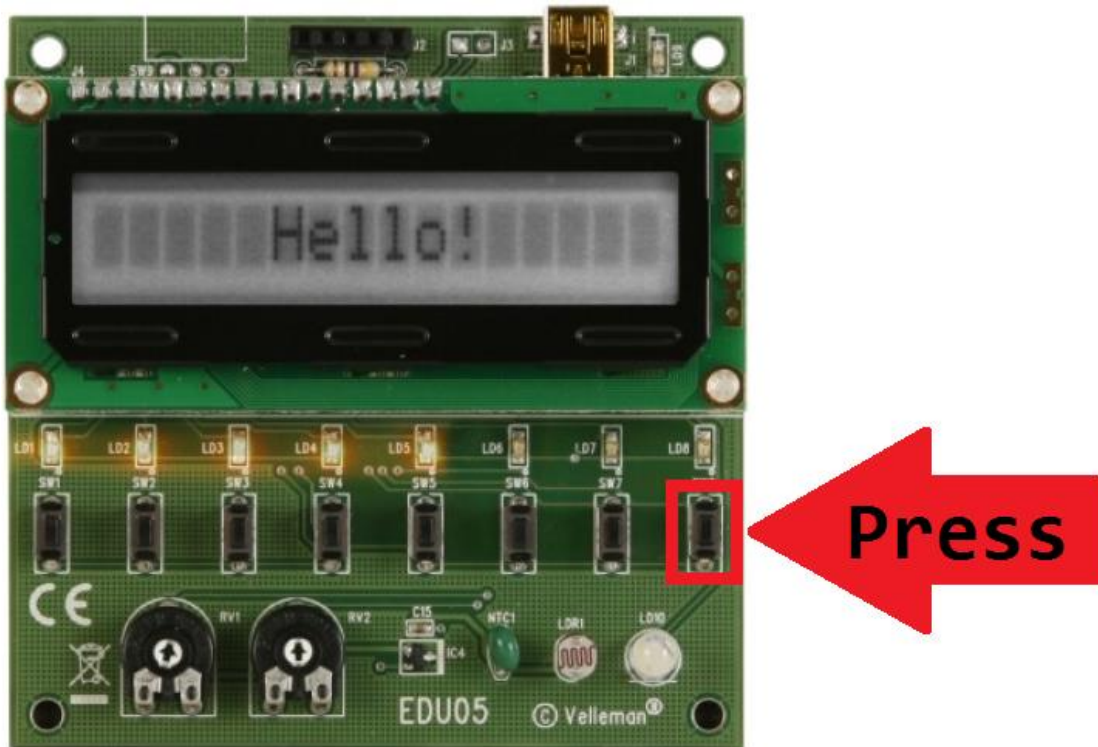
## Setting up the EDU05

To use any external I/O device you must first install the deviceWISE driver package for that device onto your Asset Gateway.

- 1) Plug your Velleman EDU05 board into your Asset Gateway.

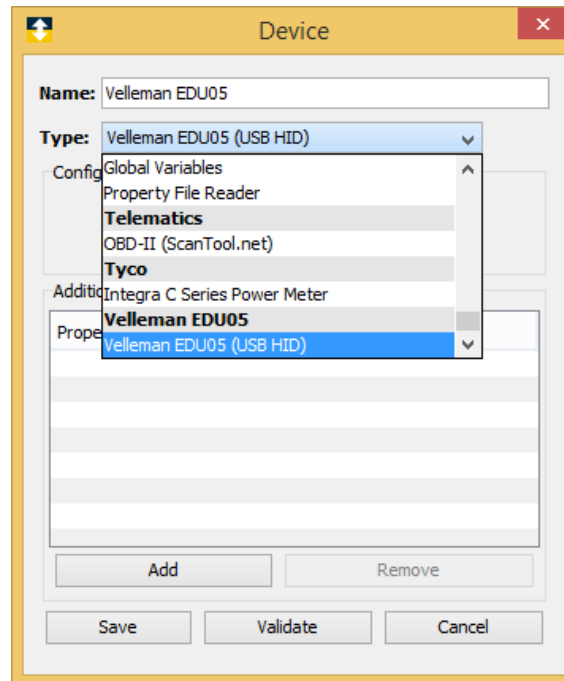


- 2) Wait for the “**Hello**” message to go away, then press the Digital Input Switch #8 to connect the EDU045 to your Asset Gateway.



- 3) Navigate back to the **Developer** page on the **Management Portal**.
- 4) Navigate to the **Prototype** packages folder for your Asset Gateway  
**Developer>Resources>Last Version of deviceWise Folder>Brand Name of your Asset Gateway> Packages>Prototype**.
- 5) Download and save the “**dw.dwedu05.Linux-ARM-Option.X\_X\_X.pkg**” package file.
- 6) Navigate to the **Administration** window within your Asset Gateway’s list of features.

- 7) Select the **Packages** tab.
- 8) Click the **Add** button on the bottom of the window.
- 9) Navigate to where you downloaded the EDU05's package file and select that file.
- 10) A confirmation window will appear asking you to confirm that you want to install the package file. Click "Yes", your gateway will restart.
- 11) You should now see the **Velleman EDU05** in the list of packages along with a **Status** of "**OK**". If you do not see a status of "OK" remove the package from your gateway and try to add it again.
- 12) Click on the **Devices** icon in your gateway's list of features.
- 13) Click the **New** button on the bottom of the page.
- 14) In the **New Device Name** box type: "**Velleman EDU05**".
- 15) Select **Velleman EDU05 (USB HID)** as the device **Type**.



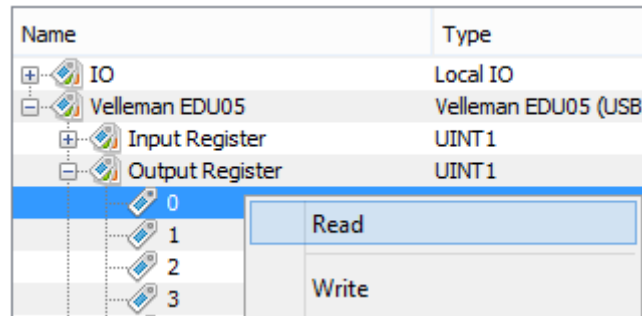
**16)** Click the **Save** button. This step creates an instance of the EDU05 driver on your gateway.

**17)** Right-click on the **Velleman EDU05** and select **Start**.

**18)** Click on the **Variables** tab.

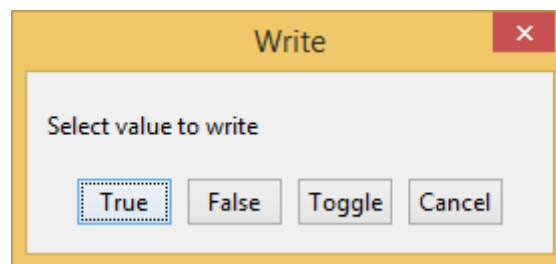
**19)** Expand the **Variable Tree** for the Velleman EDU05.

**20)** Right-click on the **Output Register "0"** and select **Read**. The current value of that variable will be shown in the **Value** section for that variable.



**21)** Right-click on the variable and select **Write**.

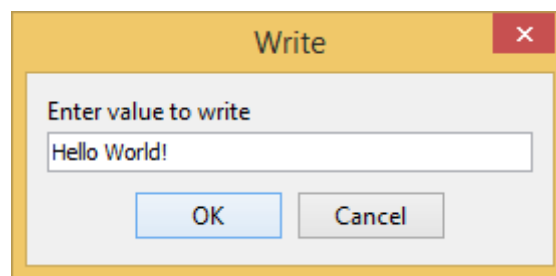
**22)** Write the variable value to **True**.



**23)** Read the value of the **Output Register “0”** again.

**24)** Right-click on the **LCD Display** and select **“Write”**.

**25)** Type in **“Hello World!”**, then click **OK**. You should see your message on the **Velleman Display**.



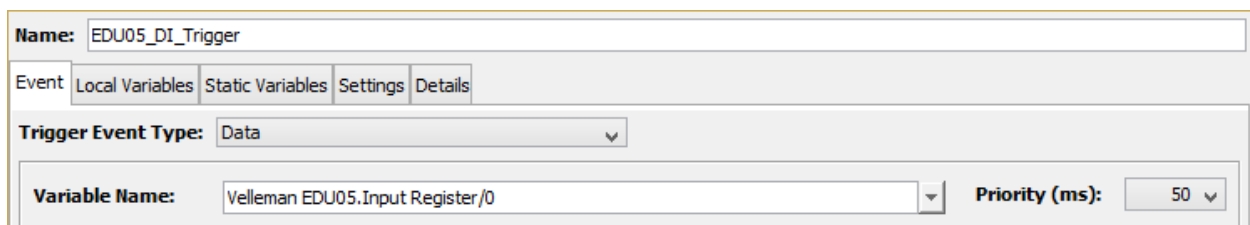
26) Right-click on the **Luminosity** variable and select **Watch**. Wave your hand over the photocell and “watch” as the analog value changes.

27) Close the watch window and click “Yes” on the pop window.

Your Velleman EDU05 USB Tutor Project board is now setup on your Asset Gateway.

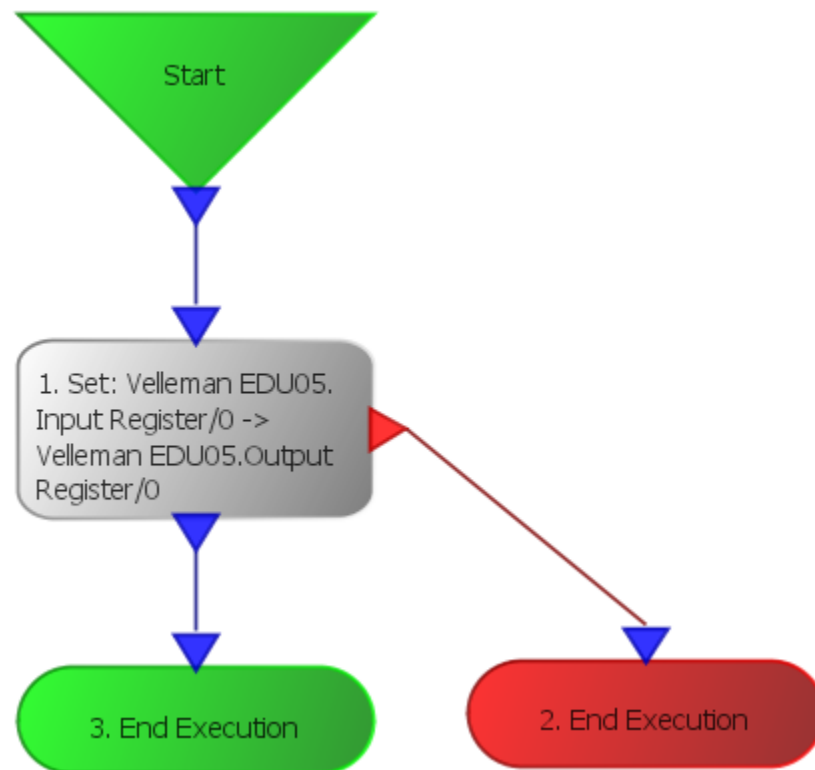
## Using the Digital Input switches

- 1) Within your “**MyFirstProject**”, create a new Trigger named **EDU05\_DI\_Trigger**; with a **Trigger Event Type** of “**Data**”.
- 2) In the **Variable Name** box select the “**Velleman EDU05.Input Register /0**” and set the **Priority** to **50 ms**.
- 3) Make the sure the **Condition** is “**Value changed**”. This type of Trigger waits for a state change of whatever variable it is told to watch, in this case the first input switch on the EDU05 board.



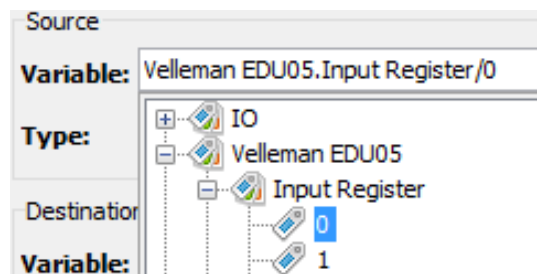
The screenshot shows a configuration window for a trigger. At the top, the 'Name' field contains 'EDU05\_DI\_Trigger'. Below this, there are four tabs: 'Event', 'Local Variables', 'Static Variables', 'Settings', and 'Details'. The 'Event' tab is currently selected. Under the 'Event' tab, the 'Trigger Event Type' is set to 'Data' via a dropdown menu. At the bottom, there are two fields: 'Variable Name' and 'Priority (ms)'. The 'Variable Name' field contains 'Velleman EDU05.Input Register /0' and the 'Priority (ms)' field is set to '50'.

- 4) Add a **Set Action** to the Trigger Structure and complete the Trigger logic with the two appropriate **End Execution** Actions. Your Trigger structure should look like this:



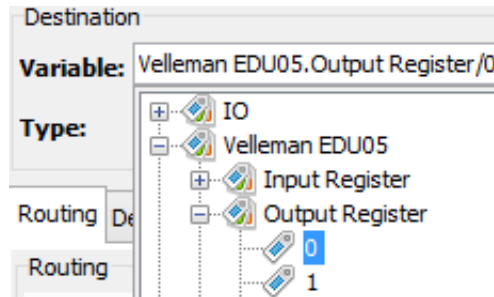
5) Open (double-click) on the **Set** Action.

6) In the **Constant** box under Source select the **Velleman EDU05 Input Register 0**.



\*Note that the **Constant** box will be dynamically renamed to **Variable** once you select the **Velleman EDU05 Input Register 0**.

7) In the **Variable** box under destination select the **Velleman EDU05 Output Register 0**.

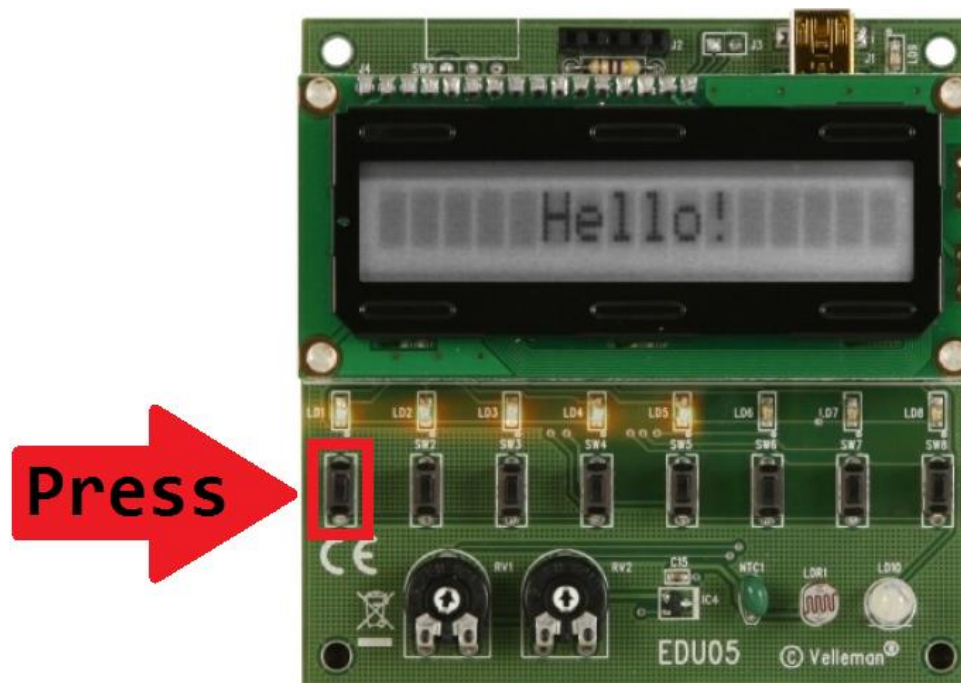


\*What you have done is **“Set”** the state of the first LED to the state of the first digital switch.

**8)** Close the **Set** Action window, then **Validate** and **Save** your Trigger.

**9)** **“Start”** your **“EDU05\_DI\_Trigger”** Trigger.

**10)** Press the first Digital Switch on the EDU05 board.

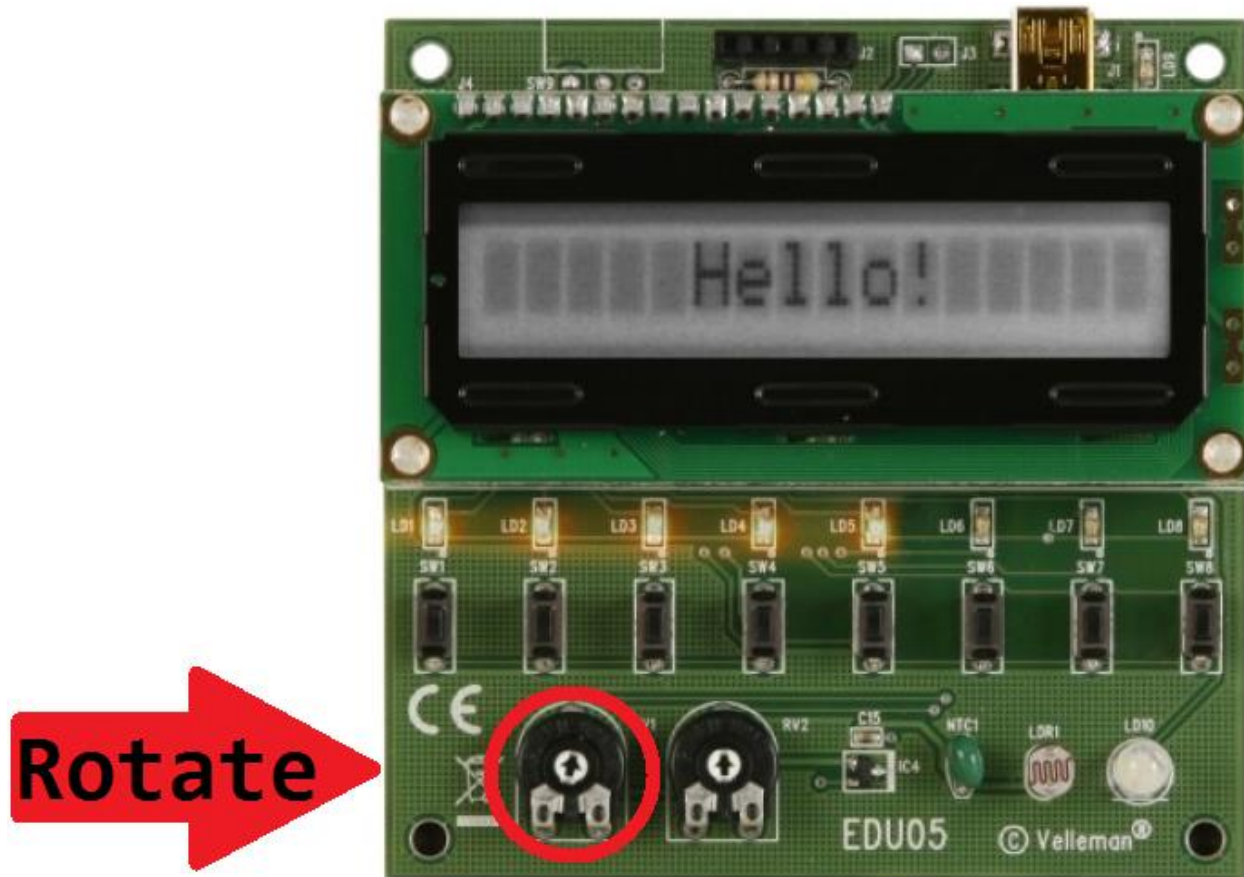


Congratulations you have created a trigger that controls the first LED on the EDU05 pressing the first digital switch



## Using Analog Inputs

- 1) **Duplicate** your “EDU\_DI\_Trigger”. Name your new Trigger “EDU\_AIN\_Trigger”.
- 2) Change the **Variable Name** to “Velleman EDU05 Potentiometer 1”.
- 3) Open the **Set** Action.
- 4) Change the **Source Variable** to the **Velleman EDU05 Potentiometer 1**.
- 5) Change the **Destination Variable** to the **Velleman EDU05 LCD Display**.
- 6) Close the **Set** Action window, then **Validate** and **Save** your Trigger.
- 7) “Start” your “EDU05\_AIN\_Trigger” Trigger.
- 8) Use a screwdriver to rotate the first Potentiometer on the EDU05.



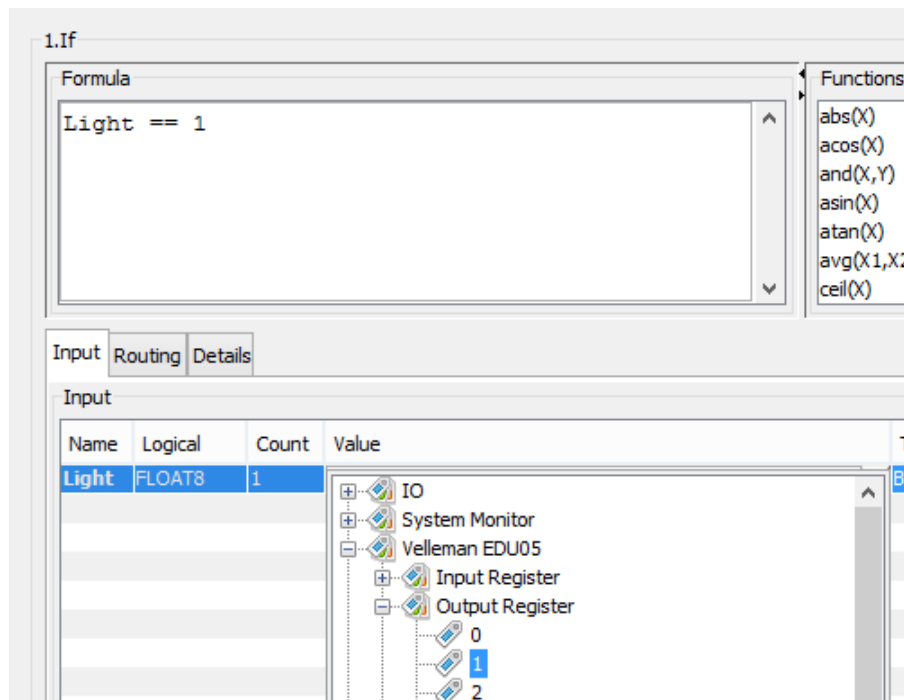
\* The number on the LCD Display is the Analog-to-Digital conversion of a 0-5 Volt signal value, with a 10-bit resolution. As such you should see 0-1023 values on the display.

## Using Triggers to Manipulate I/O

In the next few examples you will be using one of the LED lights on the Velleman EDU05 to simulate controlling an external I/O device.

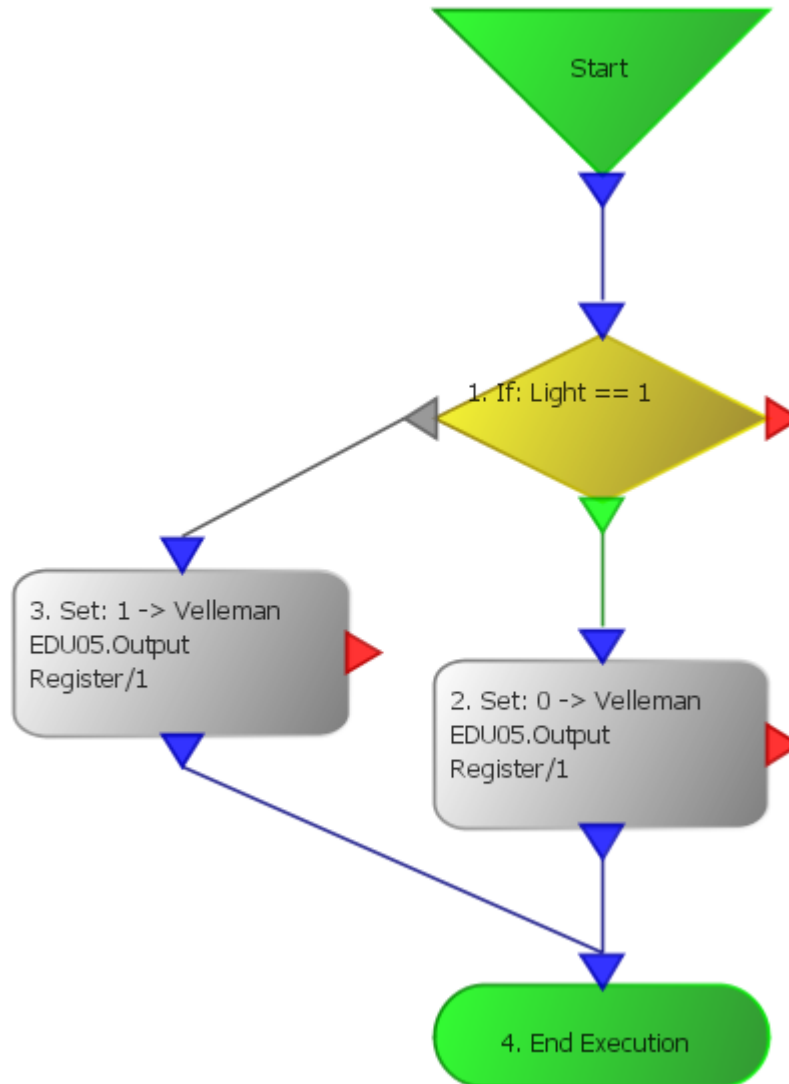
- 1) Navigate back to your **"MyFirstProject"**; create a new Trigger called **"MyIOTrigger"**.
- 2) Set the **Trigger Event Type** to **"On-Demand"**.

- 3) Select the “**IF**” Action (Yellow Triangle) from above the **Action List** drag it under the **Start** Action.
- 4) Open (double-click) the **IF** Action.
- 5) In the **Formula** box type: **Light == 1**
- 6) Switch to the **Input** tab under the **Value** section next to the variable “**Light**” select the **Velleman EDU05.Output Register/1**.



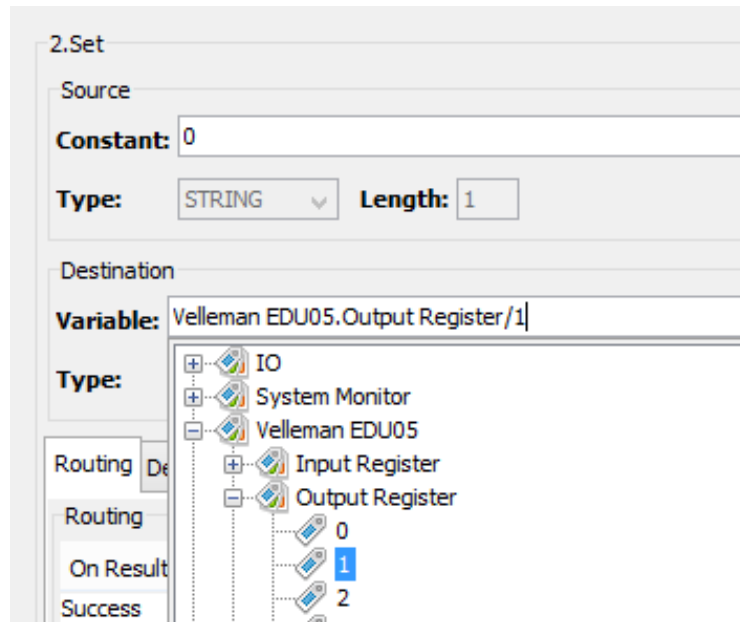
- 7) Add a **Set** Action below the **IF** Action and **connect** the **Green** Arrow on the **IF** Action with the **Blue** Arrow on the top of the **Set** Action.
- 8) Add a **Set** Action to the left of the **IF** Action and **connect** the **Grey** Arrow on the **IF** Action with the **Blue** Arrow on the top of the **Set** Action.

- 9) Add and Connect an **End Execution (Success)** to your Trigger' Action Diagram.  
Your Trigger structure should look similar to this:



- 10) Open the **Set** Action below the **IF** Action.

- 11) Make the **Constant “0”** and the **Variable “Velleman EDU05.Output Register/1”**.



**12)** Open the **Set** Action to the left of the **IF** Action.

**13)** Make the **Constant “1”** and the **Variable “Velleman EDU05.Output Register/1”**.

\*This Trigger checks the state of the LED on the Velleman EDU05 and then sets its state to the opposite of what it is currently set to essential a software toggle switch. If you are familiar with coding structures, this Trigger Action Diagram is equivalent to the C++ code below:

```

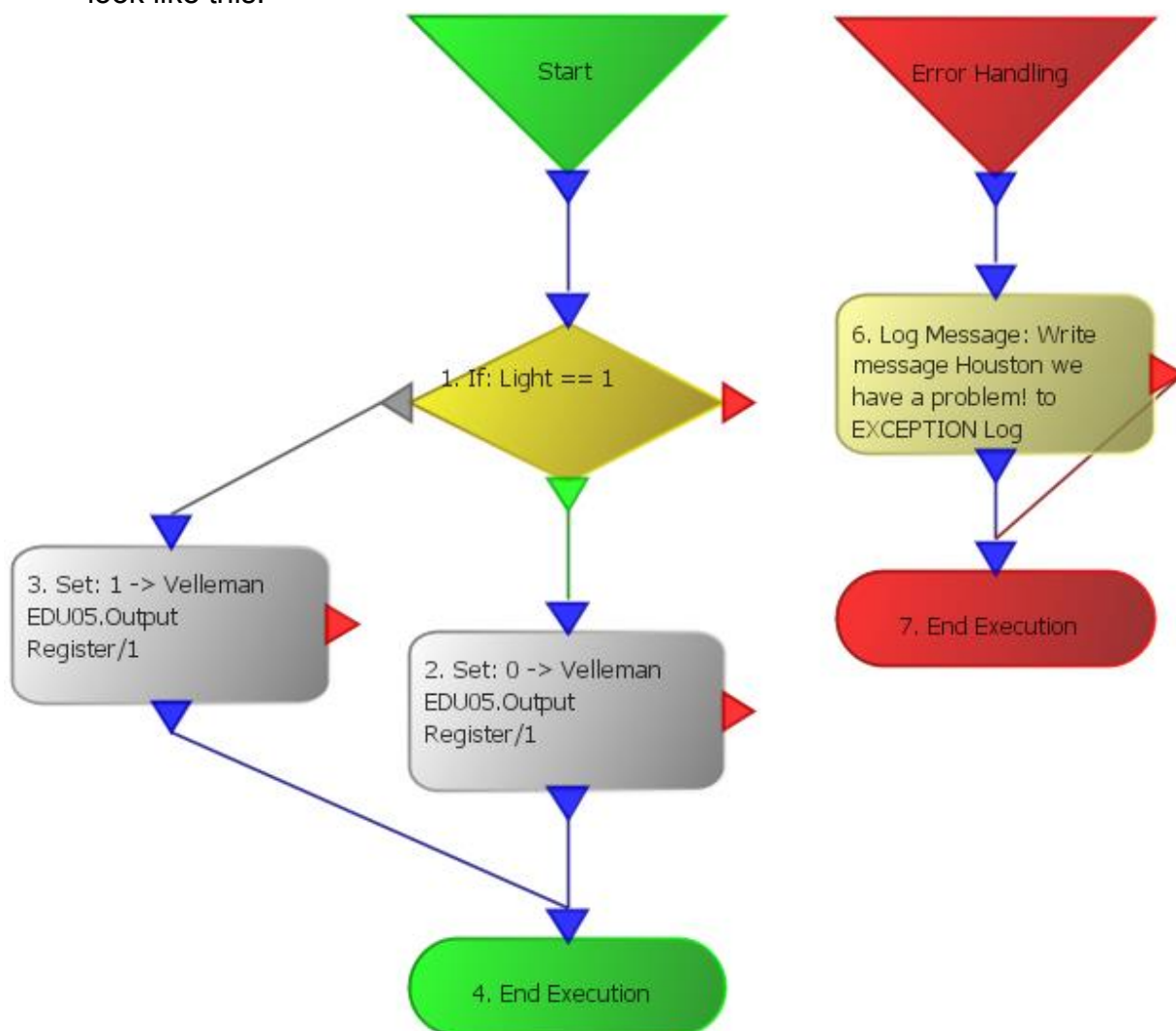
1  if(Light == true)
2  {
3      Light = false;
4  }
5
6  else
7  {
8      Light = true;
9  }

```

**14)** For simple Triggers adding a simple **End Execution (Failure)** is sufficient; however for more complex Triggers you will need a proper “**Error Handling**” structure. For this reason you will be creating an **Error Handling** sub-structure within your “**MyIOTrigger**”. Select the “**Error Handling**” Action (Red Upside-down Triangle) from above the **Action List** then drag it next to the **Start** Action.

**15)** Add and connect a **Log Message** Action underneath the **Error Handling** Action.

**16)** Add an **End Execution (Failure)** Action to the **Error Handling** to the **Error Handling** Action Diagram. Once you do that your Trigger Action Diagram should look like this:



- 17) Open the **Log Message** Action.
- 18) Delete whatever is in **Message** box and type: **"Houston we have a problem!"**.
- 19) **Validate** and **Save** your **"MyIOTTrigger"**.
- 20) Right-click on your **"MyIOTTrigger"** and select **"Fire Trigger"**. You should see the LED second to the left on the EDU05 toggle On or Off when you fire your trigger.

Congratulations you have successfully created a trigger to control the LED on your Velleman EDU05. You can use triggers similar to this one to control your gateway or any device connected to your gateway.

## Using TR50 Method

A **TR50** Method is a way to call or execute a Trigger on a remote Asset Gateway using the **Management Portal**. An Asset Gateway can also execute a Trigger on another Asset Gateway by using a **TR50** Method. For this example you will only be using a **TR50** Method from **Management Portal**, however if you would like to learn more about **TR50** Methods you can check out the link [here](#).

- 1) Navigate back to the **Management Portal**.
- 2) Open and Edit the **Thing Definition** you made for your Asset Gateway.
- 3) Select the **Methods** tab.

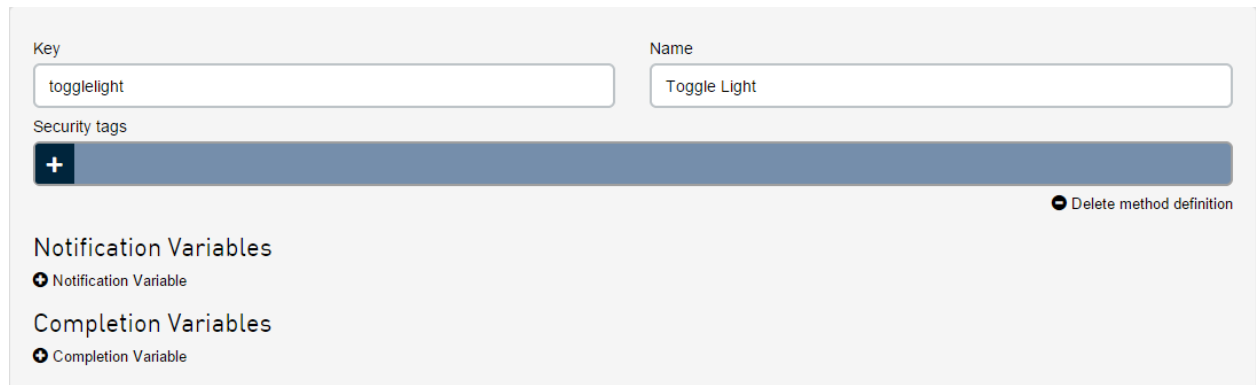
4) Click on the “**Method Definition**” button.



5) Create a new Method with

- **Key:** togglelight
- **Name:** Toggle Light

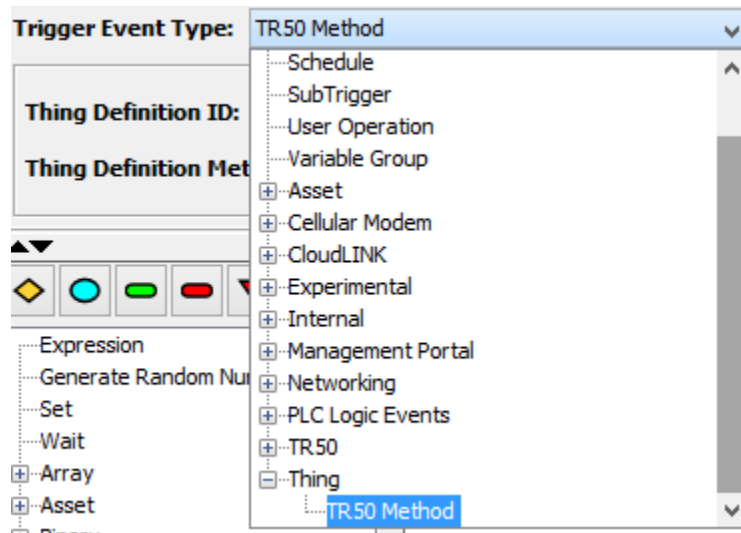
6) Click the “**Update**” button to save your new Method.



7) Open your “**MyIoTTrigger**” on the **deviceWISE Workbench**.

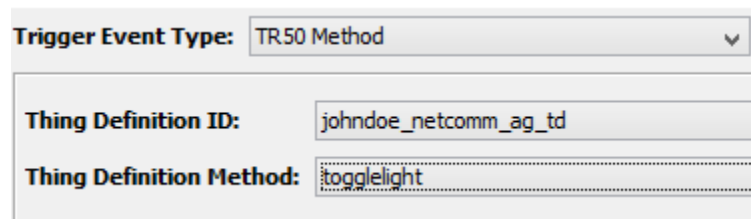
8) Change the **Trigger Event Type** to “**TR50 Method**”.





9) Change the **Thing Definition ID** to your Asset Gateway **Thing Definition**

10) Change the **Thing Definition Method** to “togglelight”.



11) **Validate** and **Save** your “MyIOTtrigger”.

12) Navigate back to your Asset Gateway’s **Thing** page.

13) Click the “**Toggle Light**” button. A pull-down window will appear.

- 14) Click the **“Execute”** button to execute your **“MyIoTTrigger”** on your Asset Gateway (the response time will depend on the Signal strength of your Asset Gateway).

Congratulations you have successfully created a **TR50** Method to control the EDU05 through your Asset Gateway remotely. You can set up any of your triggers to be used with a TR50 Method. This allows you to control any and all aspects of your gateway from the Management Portal.

## Publishing Alarms to the M2M Service

Alarms are used to define events for a Thing. Alarms are implemented as a state machine, where the events are defined as alarm states (ex: door open, tank empty, engine running).

Things can have multiple alarms; with each alarm being able to have up to 99 unique states. Alarm states have a numeric alarm state value (assigned sequentially by the system), a text alarm/state name, and a color (value) used when viewing the alarm on the thing's page in the **Management Portal**. Note that the numeric alarm state value can only be viewed within the **Thing Definition** page, but will not appear until you save your **Thing Definition**. Alarms can also be graphed, since they have a state and timestamp component.

In this example you will be creating an alarm that will display the state of your **“MyIoTTrigger”**.

- 1) Navigate to the **Thing Definition** page on **Management Portal**.
- 2) Open the **Thing Definition** you created for your Asset Gateway.
- 3) Select the **Alarms** tab.
- 4) Click on the **Alarm Definition** button.

5) Create an **Alarm Definition** with:

- **Key:** ledstate
- **Name:** LED State

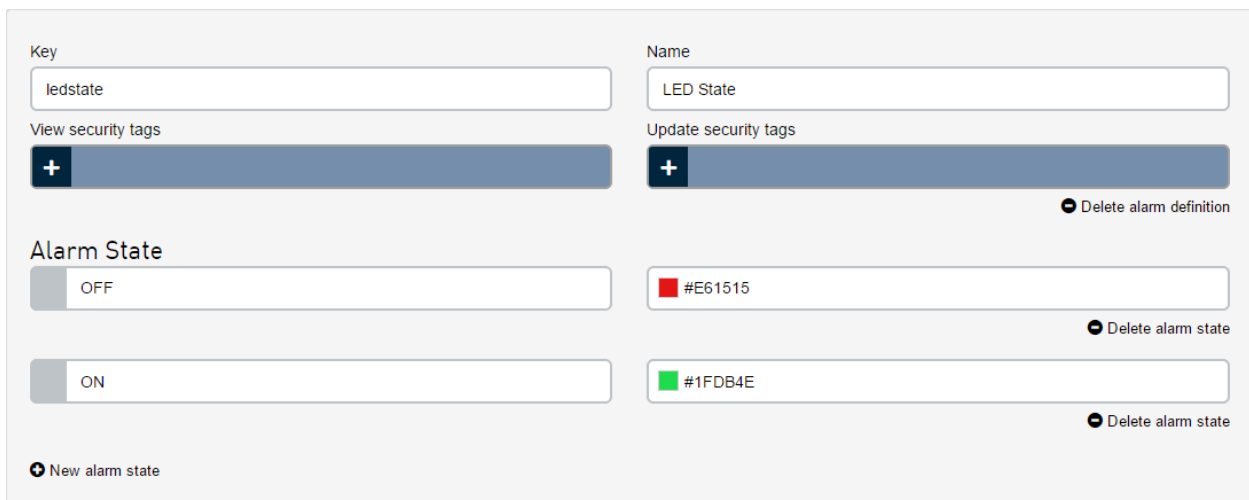
6) Click on the **New Alarm State** button.

7) Name the new Alarm State **“OFF”**.

8) Click on the “black” square in the box to the right of the new Alarm State. For this example red was selected as the color of this Alarm State, but feel free to use whatever color you want.

9) Create another New Alarm State named **“ON”**.

10) Select a different color for this New Alarm State. This example’s color is green.  
Note that you can also type in the Hex number of a color.



The screenshot shows the configuration interface for an alarm definition. It is divided into two main columns: 'Key' and 'Name'.

- Key Column:**
  - Field: ledstate
  - View security tags: A blue bar with a white '+' icon.
  - Alarm State: A list of states. The first state is 'OFF' with a grey square. The second state is 'ON' with a grey square.
  - Bottom button: New alarm state (with a circular arrow icon).
- Name Column:**
  - Field: LED State
  - Update security tags: A blue bar with a white '+' icon.
  - Color selection: Two rows are shown. The first row has a red square and the hex code #E61515. The second row has a green square and the hex code #1FDB4E.
  - Buttons: Each row has a 'Delete alarm state' button (with a circular arrow icon).
  - Bottom button: Delete alarm definition (with a circular arrow icon).

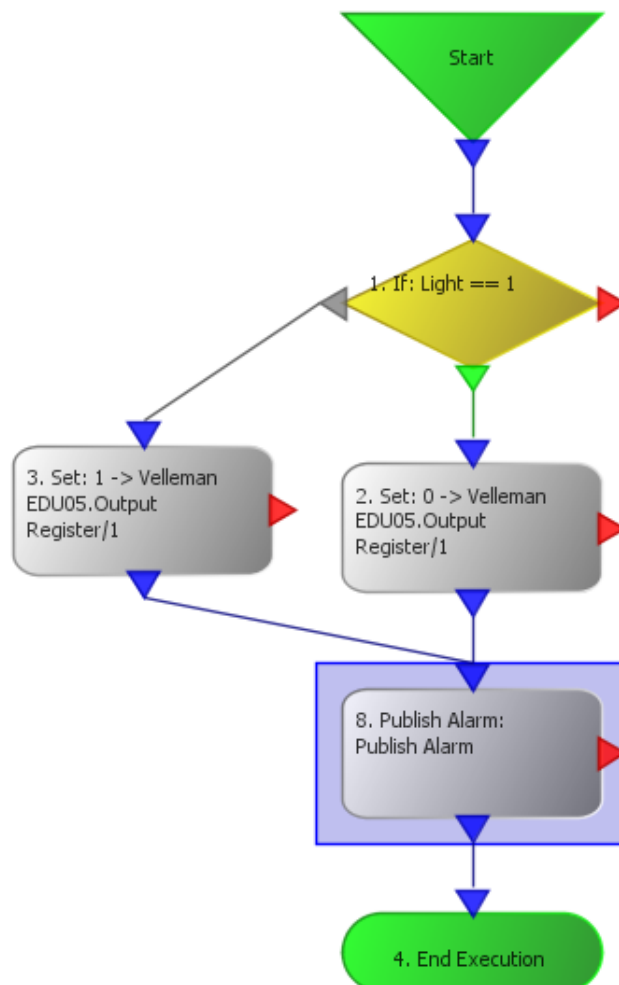
11) Click the **Update** button to save your changes to your Thing Definition.

**12)** Open your “MyIoTTrigger”.

**13)** Move the **End Execution (Success)** down in order to make room for another Action.

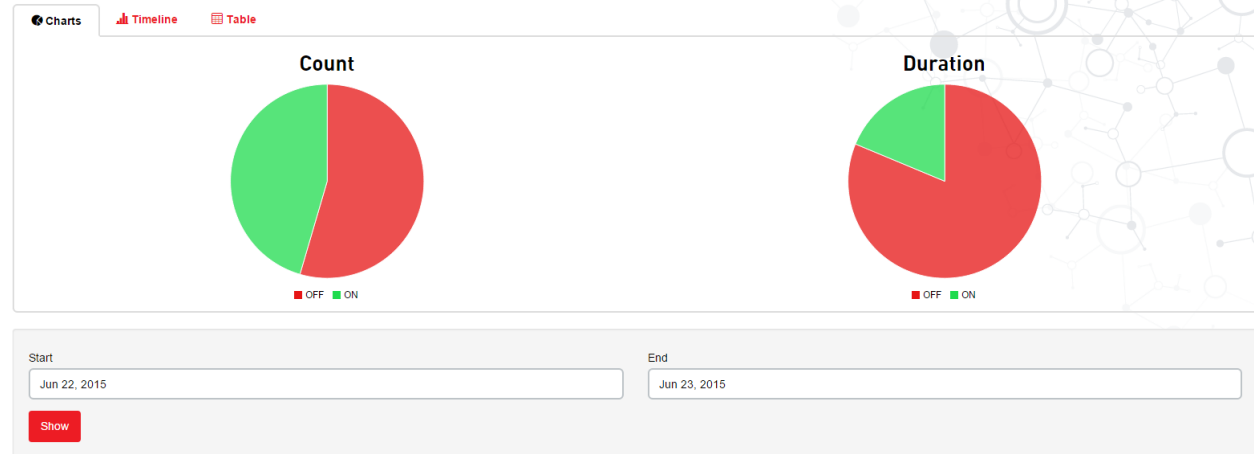
**14)** Under the **Thing** Action Group select the **Publish Alarm** Action and put it in the space you just made.

**15)** Connect the **Publish Alarm** Action to your Trigger Action Diagram. Your Trigger structure should look similar to this:



- 16) Open the Publish Alarm Action.
- 17) In the **Value** section of the variable **Key** use the menu to navigate through the variable structures to your Alarm Definition: **Thing Definition>Your Thing Definition for your Asset Gateway> Alarms> LED State**.
- 18) In the **Value** section of the variable **State** select **Velleman EDU05.Output Register/1**.
- 19) Close the **Publish Alarm** Action window.
- 20) **Validate** and **Save** your “**MyIoTTrigger**”.
- 21) Navigate to your Asset Gateway’s **Thing** page.
- 22) Select the **Methods** tab.
- 23) Use your **TR50** Method to toggle the light on your Asset Gateway. Once you toggle the state of the LED refresh your **Thing’s** page.
- 24) Click the view button of your **LED State** alarm to see a pie chart of your alarm’s data. There is also a timeline graph and table layout where you can see exactly when each event was triggered. The longer your Asset Gateway is connected the more data points there will be.

## LED State



Congratulations you have successfully published an Alarm state to the Management Portal. You can use Alarms to alert you when there is a problem with your gateway or a device connected to your gateway. You can also use Alarms to help you keep track of the status of your gateway by giving you a visual representation of the status of your gateway. You have also finished the Option CloudGate Step-by-Step guide. There are a lot of topics covered in a broad sense in this guide so if you would like to see more detailed information on any of the topics covered in this guide you can by visiting [help.devicewise.com](http://help.devicewise.com).