Getting Started Guide

Document Information

Software Version:	4.0.3.3
Creation Date:	October, 2019
Last Edit Date:	25 April, 2022
Version:	1.5

ADISRA · 3432 Greystone Drive, Suite 125 · Austin, TX 78731
Phone: 1-833-5ADISRA (1-833-523-4772)
www.ADISRA.com

Table of Contents

1. Scope

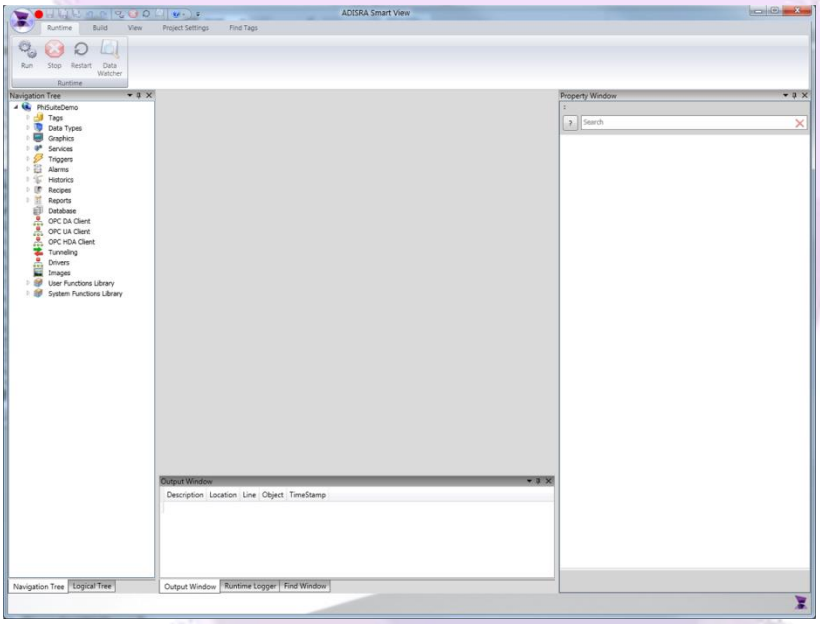
The ADISRA SmartView Getting Started Guide shares the many ADISRA SmartView features, provides explanations and examples, while showing how they work and where they can be used.

2. Summary

ADISRA SmartView is a powerful, Windows-based, desktop software tool that builds robust HMI, SCADA, and Client-Server programs for industrial automation applications. ADISRA SmartView provides the integrated features and functionality necessary to quickly create projects and connect to industry-standard modules, components, and devices.

3. ADISRA SmartView Features

ADISRA SmartView has an Engineering Environment, the main workspace, designed to be easily recognized by IT programmers while also providing the important standardized features and functionality of industrial automation HMI software packages.



With ADISRA SmartView, the user can create powerful HMI and SCADA applications and easily edit them at any time. ADISRA SmartView is loaded with communication drivers to easily integrate different modules, components, and devices. The user can store and acquire data from external sources (OPCs, Databases, PI-OSIsoft, IoT and Edge Devices, PLCs, and the Cloud).

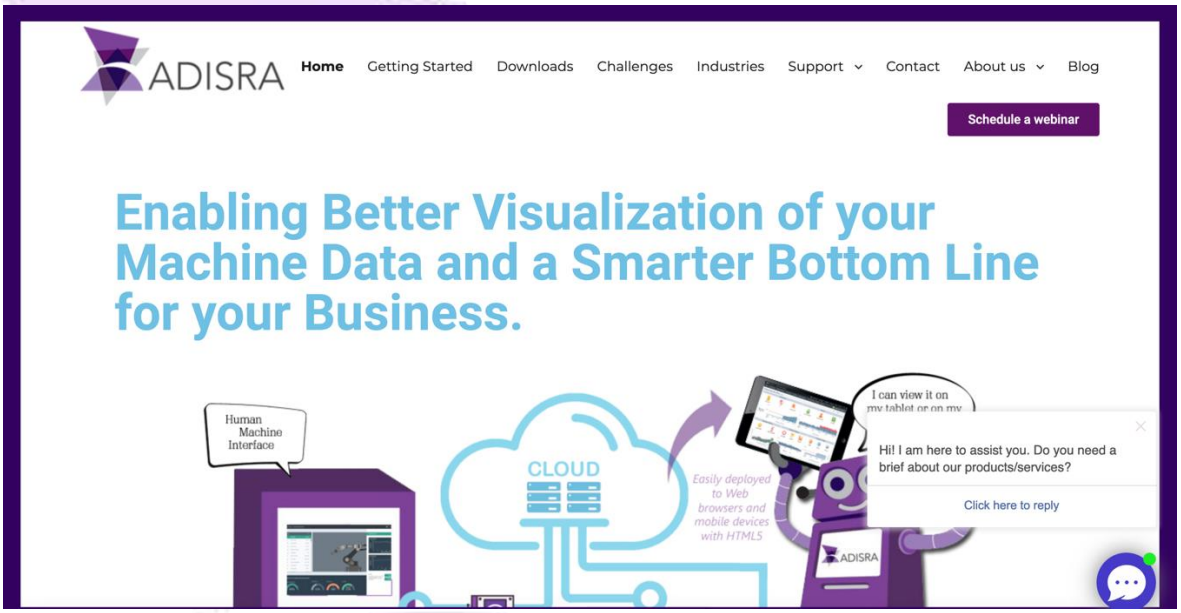
Here are some more features of ADISRA SmartView:

- View screens on tablets and mobile phones via HTML5 browsers
- Local, remote, and web viewers
- Multi-dimension and dynamic tags
- Native redundancy
- C# scripts and functions
- Create HMI and SCADA applications
- Create client-server applications

- Link and integrate external sources
- Link real-time data with screen objects (trend, charts, alarms, colored indicators, text boxed, etc.)

4. Downloading ADISRA SmartView

ADISRA SmartView is a web-deployed software program and available for download at www.adisra.com.



To download ADISRA SmartView, please follow the steps below.

1. Open a web browser and go to www.adisra.com.
2. Click on Downloads link located in the top banner, complete the requested information. and follow the on-screen instructions.
3. Once the browser has completed the download, extract the software from the downloaded ZIP file.
4. Select the file, right-click, and select Extract files. Choose the location for the extracted software files.

5. Installing ADISRA SmartView

To install ADISRA SmartView, we recommend that the user's system meet the following requirements to optimize its performance.

5.1. Hardware Requirements

Minimum:

- CPU: 1.44 GHz
- RAM: 4 GB
- Hard-Disk (min): 8 GB

Recommended:

- CPU: 1.7 GHz
- RAM: 6 GB
- Hard-Disk(min): 20 GB

5.2. Software Requirements

- Operating System: Windows 10, Windows 8.1, Windows 7 SP1, Windows Server 2019, Windows Server 2016, Windows Server 2012 R2, Windows Server 2012 (64-bit edition)
- .NET Runtime 5.0
- OPC Core Components

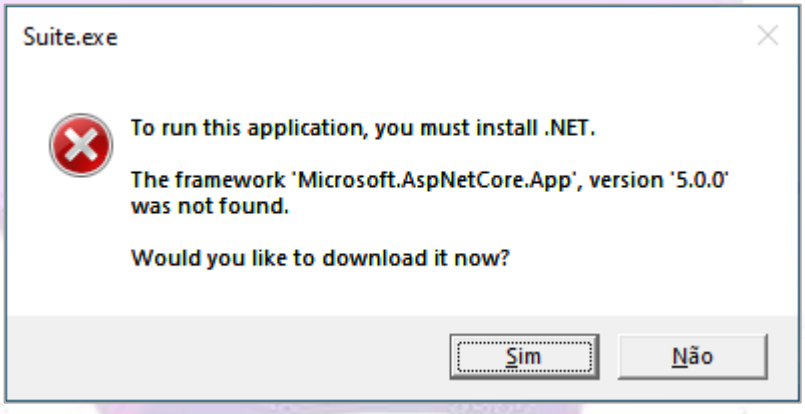
The ADISRA SmartView installer will verify that the machine on which the software is being installed has all the necessary software prerequisites. If any software prerequisites need to be installed (except Windows operating systems), the installer will install the necessary requirements.

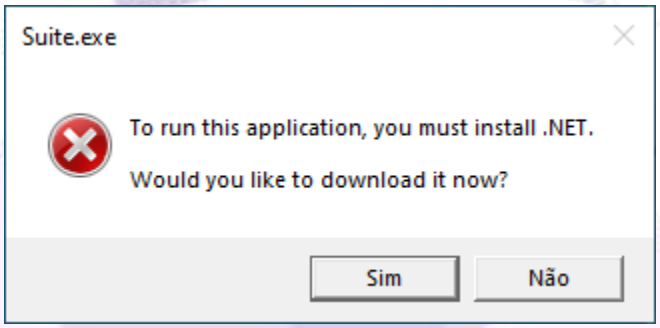
NOTE: *The installer will only check for the .NET Runtime 5.0 component; however, the installer will not install it. The user will be responsible for installing (or upgrading) this component. With all prerequisites properly installed, the next step is to install ADISRA SmartView.*

5.3. The Installation Process

5.3.1. .NET Runtime 5.0 is REQUIRED

If the user attempts to install ADISRA SmartView and has not installed Microsoft .NET Runtime 5.0 and ASP.NET Core Runtime 5.0, an error message will be received:





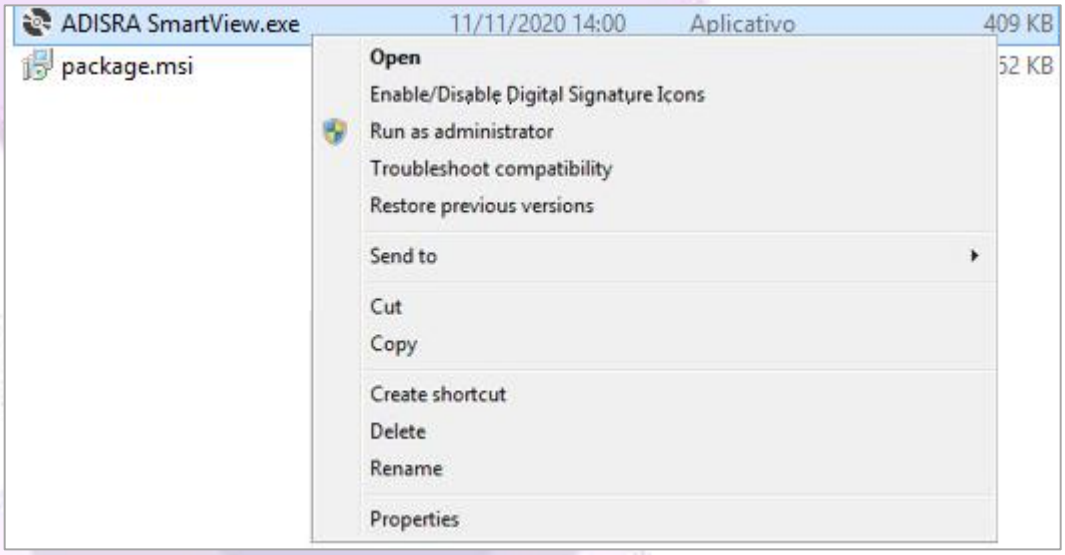
The user must go to the Microsoft website to install .NET Runtime 5.0:

https://dotnet.microsoft.com/en-us/download/dotnet/5.0/runtime

Once downloaded, locate the .NET installation file, double-click it to complete the installation. A request to restart the computer will appear, please restart.

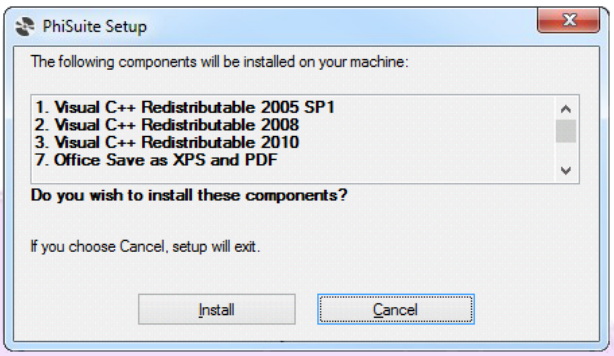
5.3.2. Download and Extract from ZIP File

Once ADISRA SmartView has been downloaded and extracted from the ZIP file as described in Downloading, go to the folder, and double-click the **en-us** folder to open it. The user must run the Setup.exe file as the Administrator by right-clicking on ADISRA **SmartView.exe** and selecting **Run as administrator** from the drop-down menu.

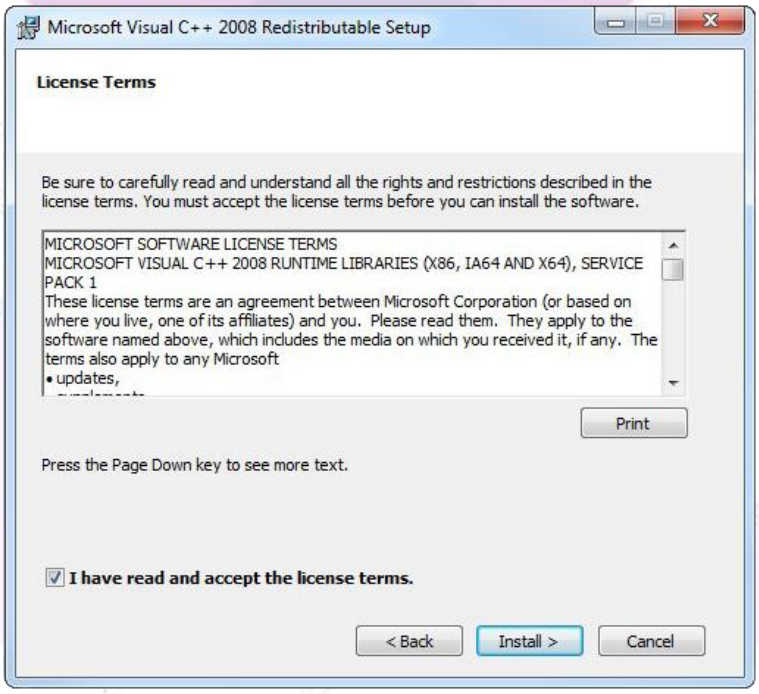


5.3.3. Installing Components

If this is the first time ADISRA SmartView is to be installed, a new window will open informing the user that some components will be installed. Click on Install. For the component installation, accept the terms of installments, and follow installation instructions on each screen. The following are the installation screens during the installment process.

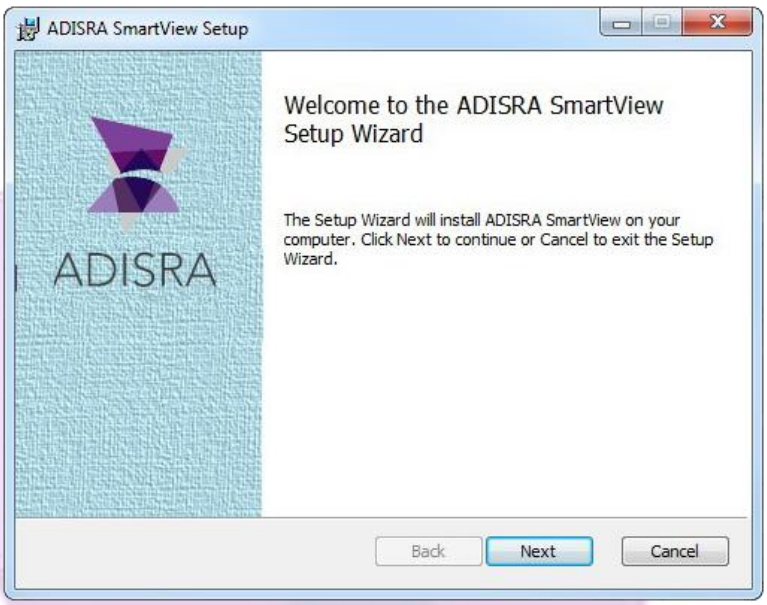


For each component that needs to be installed, there will be a message window with license and term agreements. To complete the installation, the user needs to accept each of the term agreements. Check any appropriate boxes and proceed.

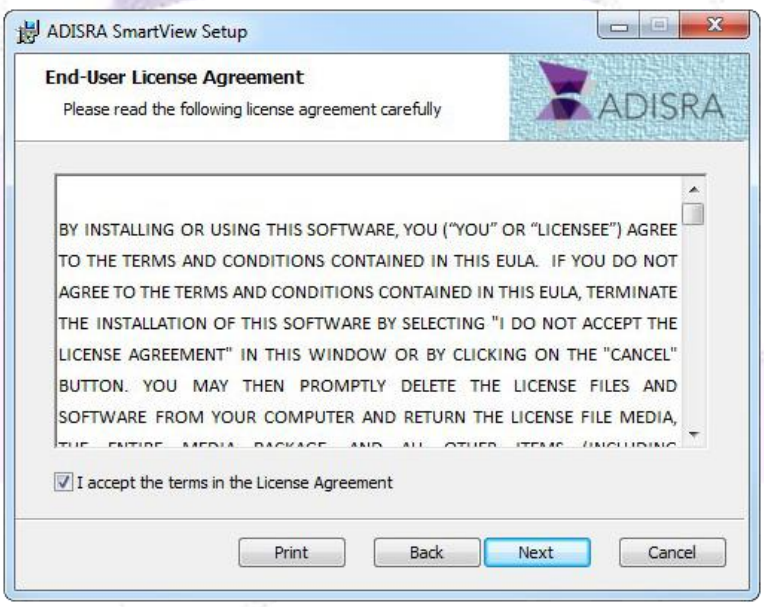


5.3.4. Installing ADISRA SmartView

After the user has installed the components, ADISRA SmartView program installation will begin. Click on Next, then click on accept the installation terms, and follow the on-screen installation instructions.



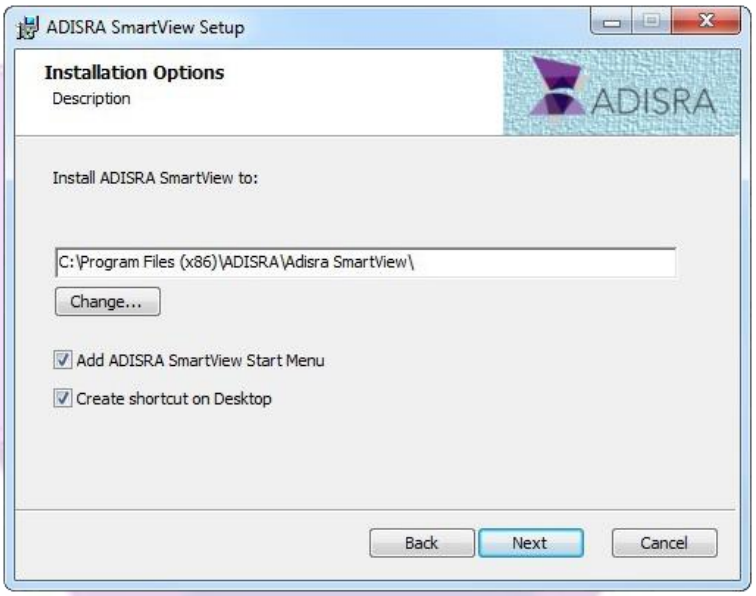
After reading the entire ADISRA SmartView License Agreement, select the "I accept the terms in the License Agreement" checkbox, and click Next.



5.3.5. Installation Location

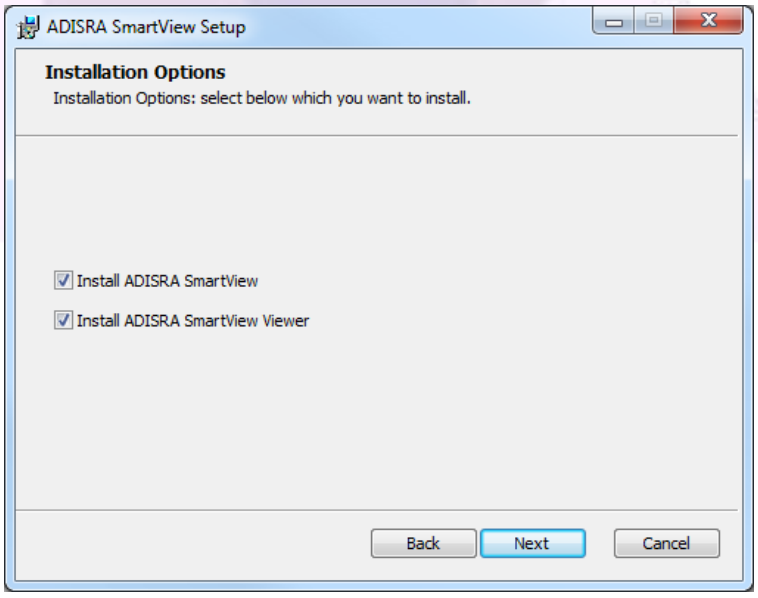
When prompted, the user must decide where to install ADISRA SmartView. The default location is in the Program Files folder. If the user clicks Next, it will be installed in the default location which is "c:\Program Files(x86)\ADISRA\Adisra SmartView".

To install in another location, click the Change button, and choose the preferred location.



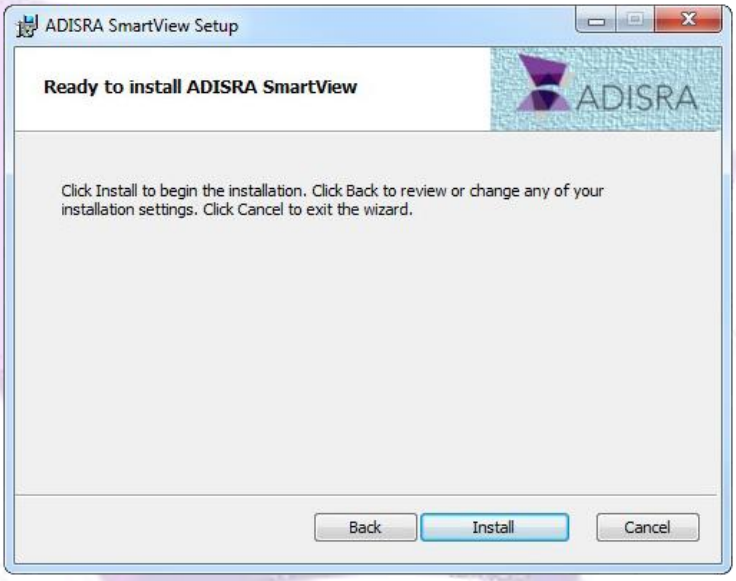
5.3.6. Choose ADISRA SmartView or Viewer, or both

When prompted, the user can choose to install only ADISRA SmartView or Viewer or both on the computer. By default, both will be installed on the computer. To choose not to install either of them, deselect the option then click Next.



5.3.7. Install ADISRA SmartView

Click Install to begin the installation of ADISRA SmartView.



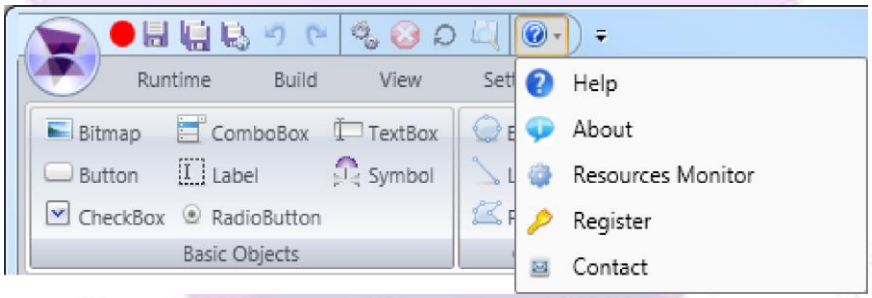
After the installation has completed, click Finish, and close the installer window.

6. Registering ADISRA SmartView License

Before the user can use the software, they MUST register ADISRA SmartView. After the user installed ADISRA SmartView, there are two (2) ways to register a license.

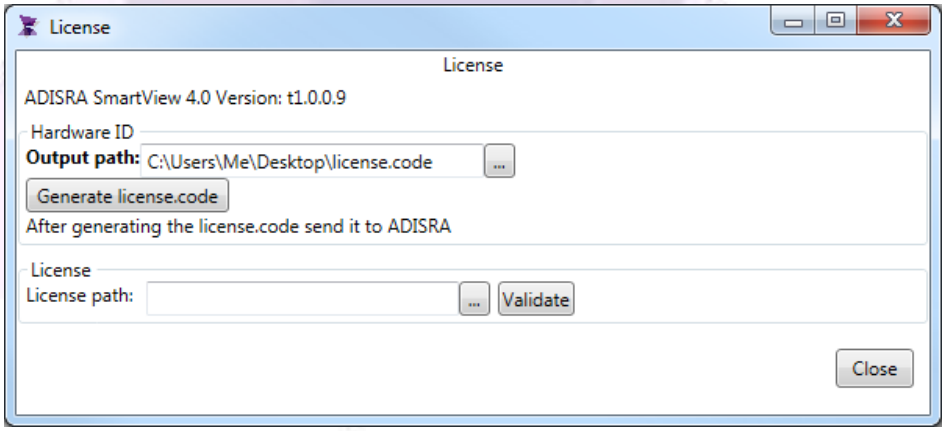
6.1. Help Menu - Register

Open the Help Menu and select the Register button as seen in the image below.



6.2. Windows Toolbar - Register

Go to the Windows toolbar: Start > All Programs > ADISRA SmartView > Register License. By selecting to register by using either method above, the following window will appear.



6.3. Generate License Code

Click the button labeled "Generate license.code" to create the text filename with the Hardware ID. The file location will be displayed in the text box called "Output Path".

NOTE: *The user can change the Output Path by clicking on the (…) graphic button next to the Output Path text box.*

6.4. Email to ADISRA

Attach the text file named "license.code" to an email and send to info@adisra.com.

6.5. License Key File

The ADISRA will send back a License Key File that matches this Hardware ID. Download and save the license file to the storage drive (please make a note to remember the location).

6.6. License Path

Enter the license key file location in License Path text box or browse to it by clicking on the (…) graphic button, and then click the Validate button.

6.7. Confirm Operation

The user will be prompted to confirm the operation once the program accepts (validates) the License Key. The user is now ready to use ADISRA SmartView. Close the Register License window and run ADISRA SmartView.

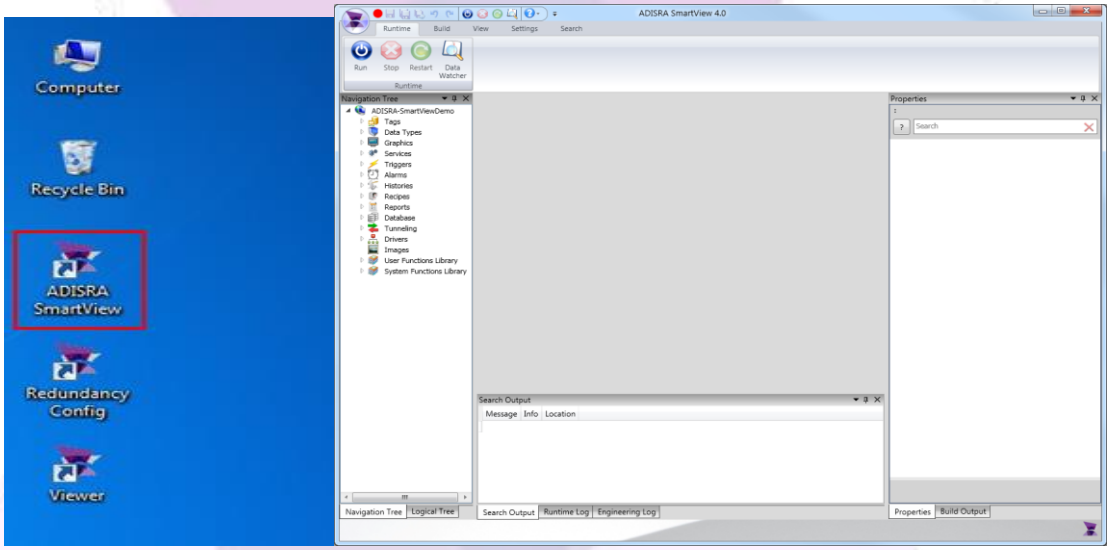
7. Launching ADISRA SmartView

Once ADISRA SmartView is properly licensed, run the software by navigating to:

- Windows toolbar: Start > All Programs > ADISRA SmartView > ADISRA SmartView

  Or

- When the user installs ADISRA SmartView, program shortcuts will automatically be created on the desktop. Double-click the ADISRA SmartView icon on the desktop.



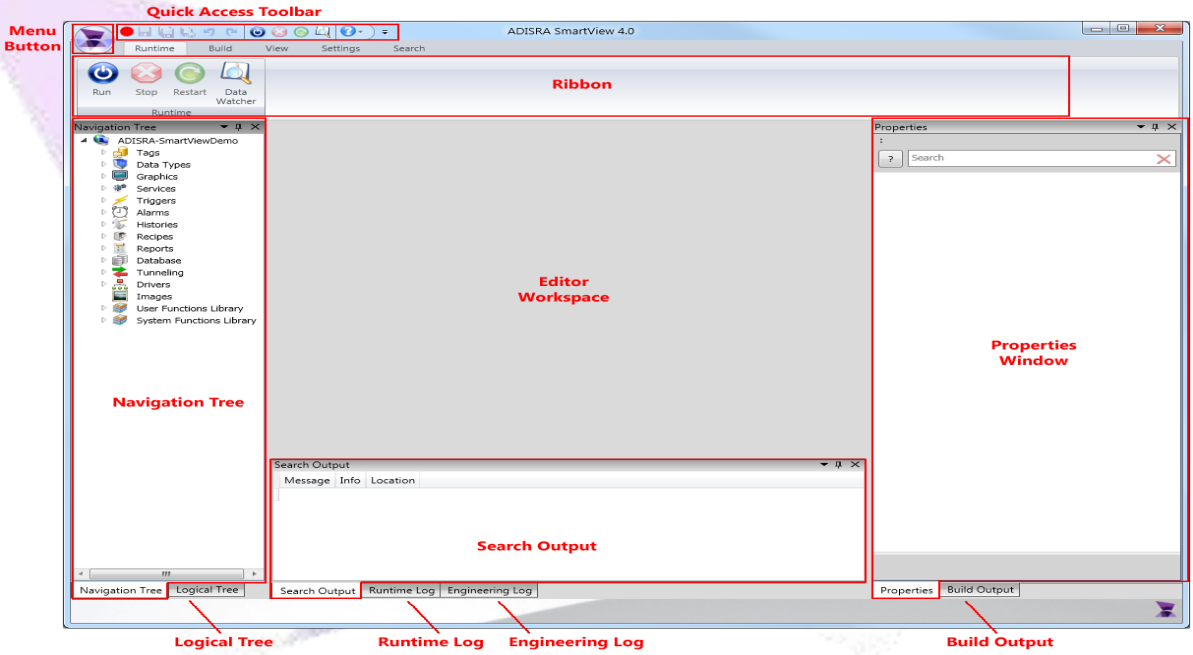
NOTE: *If this is the first time ADISRA SmartView is opened, the Engineering Environment will be empty (as shown above). Otherwise, it will open with the last project that was used.*

After the user starts ADISRA SmartView, they may Create a New Project or Open an existing project.

7.1. Engineering Environment Detail

The Engineering Environment of ADISRA SmartView is a user-friendly, ribbon-based Windows® interface with many features. The

Engineering Environment is where the user can configure the ADISRA SmartView development interface and begin working on a project right away.



The image above is a layout of the default ADISRA SmartView Engineering Environment. In the previous Getting Started Guide sections, we will refer to various areas of the Engineering Environment layout as reference. The areas of the Engineering Environment are:

- Menu Button
- Navigation Tree Pane
- Logical Tree Pane
- Ribbon
- Quick Access Toolbar
- Search Output Pane
- Engineering Log Pane
- Properties Window
- Runtime Log Pane
- Build Output Pane

NOTE: *For detailed explanation on the sections above, refer to the Help files in ADISRA SmartView.*

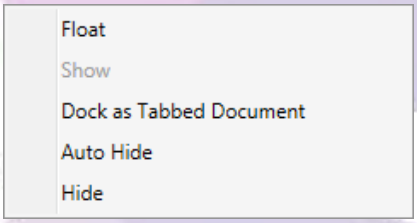
7.2. Docking Windows

Panes are part of the ADISRA SmartView Engineering Environment interface that can float above the program, be dragged beyond the program's borders, or remain fixed in one of several locations. (By default, all windows are docked.)

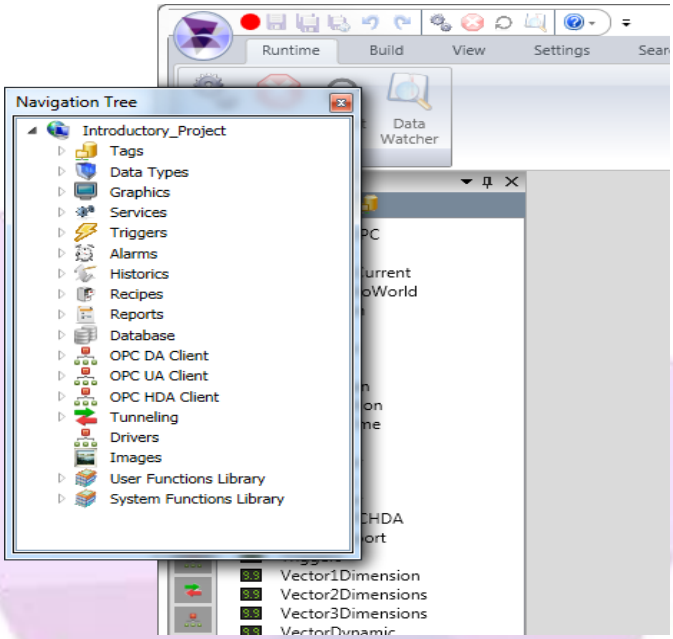
7.2.1. Floating a Pane

There are two ways to float a pane in ADISRA SmartView:

- Click the title bar of the pane, then drag the pane.
- Right-click the title bar and select Float.



By selecting Float, the pane will become separated from the main software window. The user may then drag the floating pane to any location on their screen by clicking the title bar of the pane and moving their mouse.

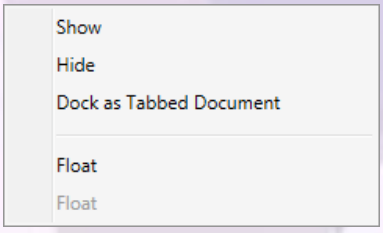


To return a floating pane to its last docked location in the workspace, right-click the title bar and select Show.

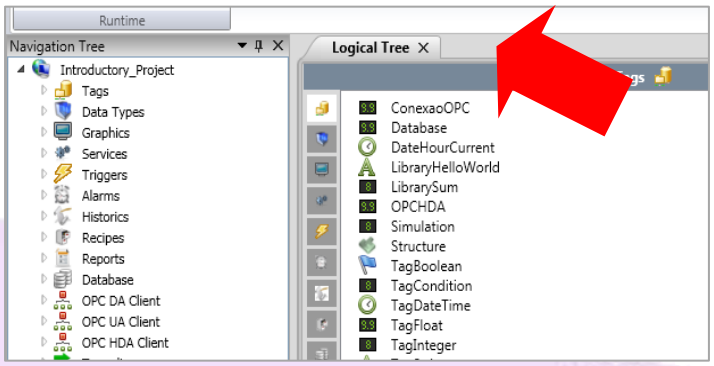
7.2.2. Docking a Pane

There are two ways to dock a pane in ADISRA SmartView:

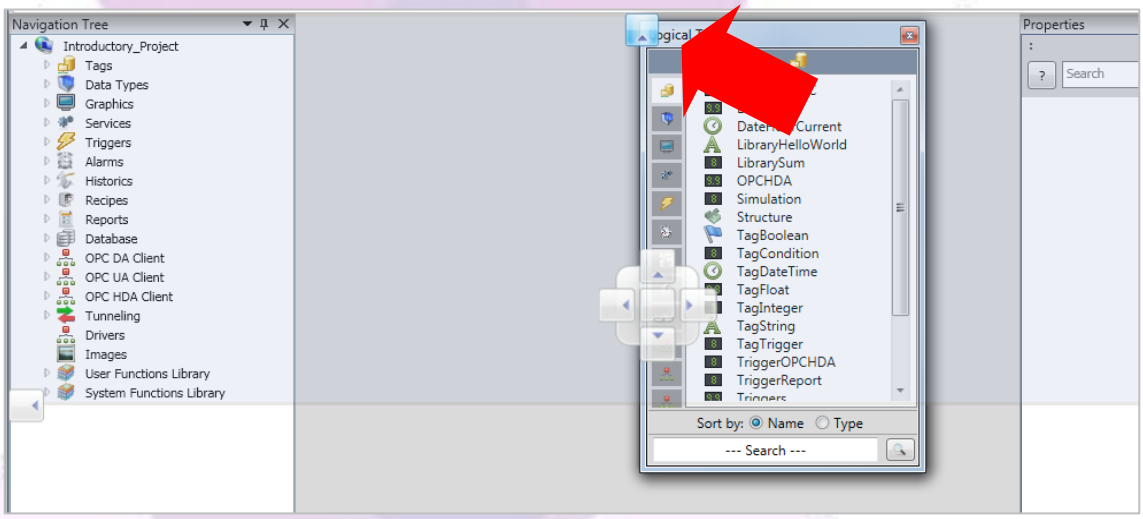
- Right-click the title bar and select Show. This will return the pane to its last docked position.



- Right-click the title bar and select Dock as Tabbed Document. This will return the pane to the middle workspace as a Tabbed Document window as seen in the image below.

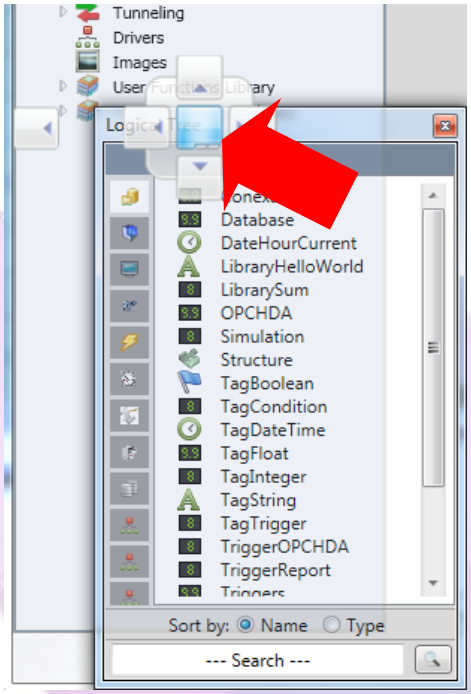


At any time, the user may move a pane to any location using the Docking Navigator. To use the docking navigator, right-click the title bar of a pane, and drag the pane onto one of the locations indicated by the docking navigator that appears on the screen.



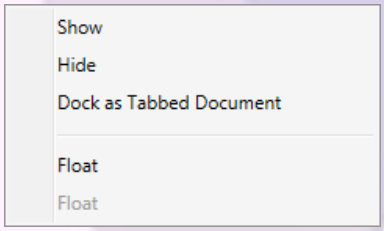
The user may select any location on screen indicated with arrows in the docking navigator.

To dock a pane as a tabbed window, the pane can be dragged over the center of the docking navigator arrow cluster as shown below. This action will dock the pane as a tabbed pane.

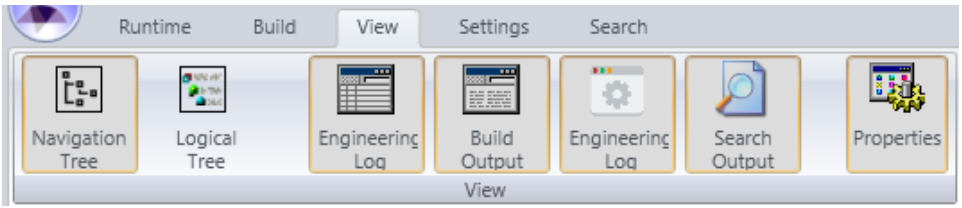


7.2.3. Showing / Hiding a Pane

To hide a pane in ADISRA SmartView, right-click the title bar, and select **Hide**.



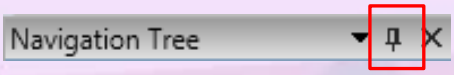
When a window pane (Navigation Tree, Logical Tree, Engineering Log, Build Output, Runtime Log, Search Output, Properties) is hidden and the user would like it made visible in the workspace once again, they can select View in the top menu and click the appropriate hidden pane to view it again in the View Ribbon.



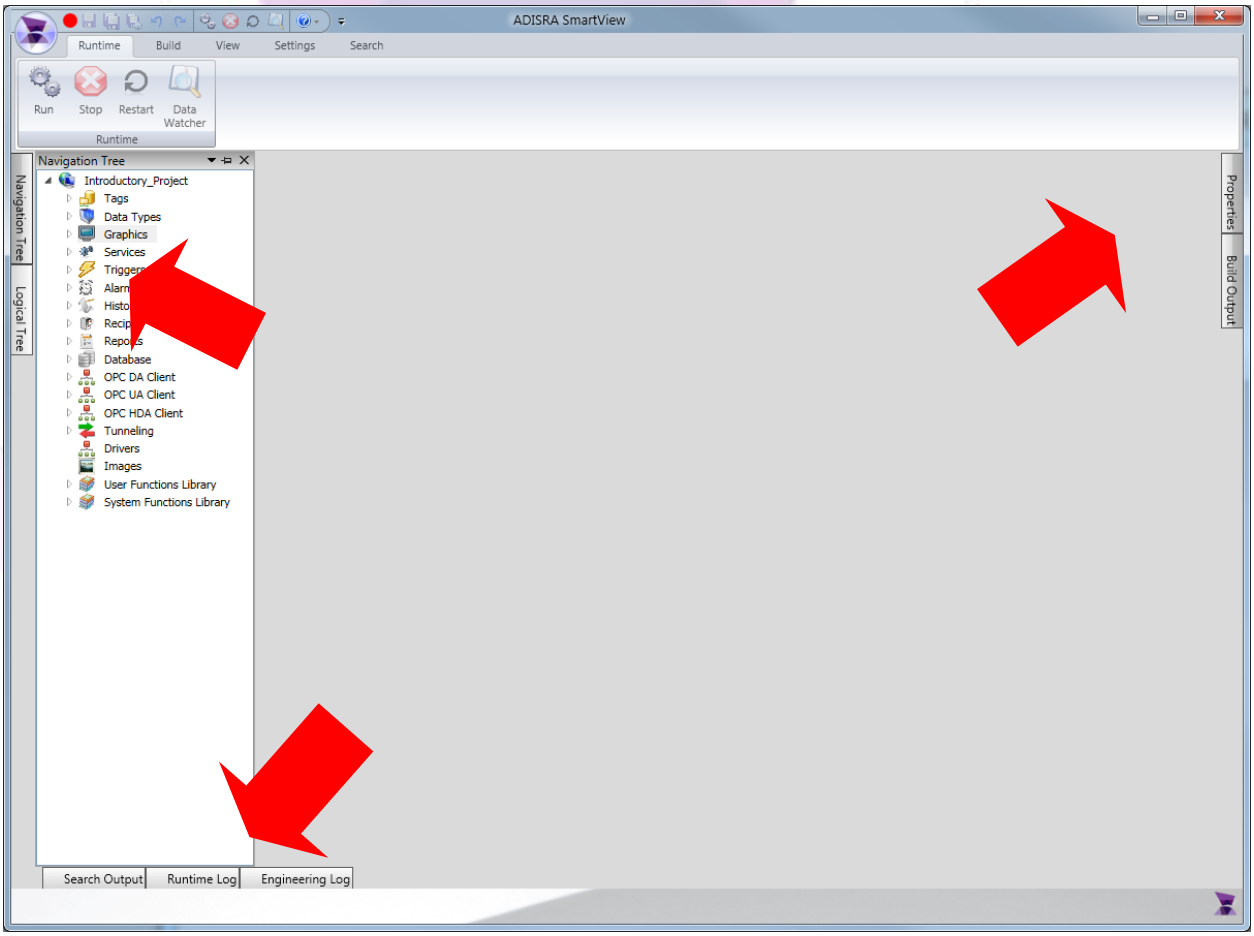
7.2.4. Auto Hide a Pane

There are two ways to auto-hide a pane:

- Right-click the title bar and select Auto Hide
- Click the thumbtack icon in the title bar

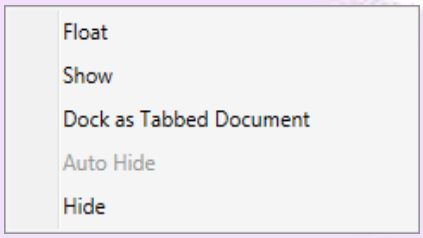


Auto hiding a pane reduces it to a tab to maximize working space as indicated by the red arrows.



Hovering the mouse over the tab temporarily restores it to full size. Clicking or hovering the mouse outside the pane returns it to Auto Hide mode.

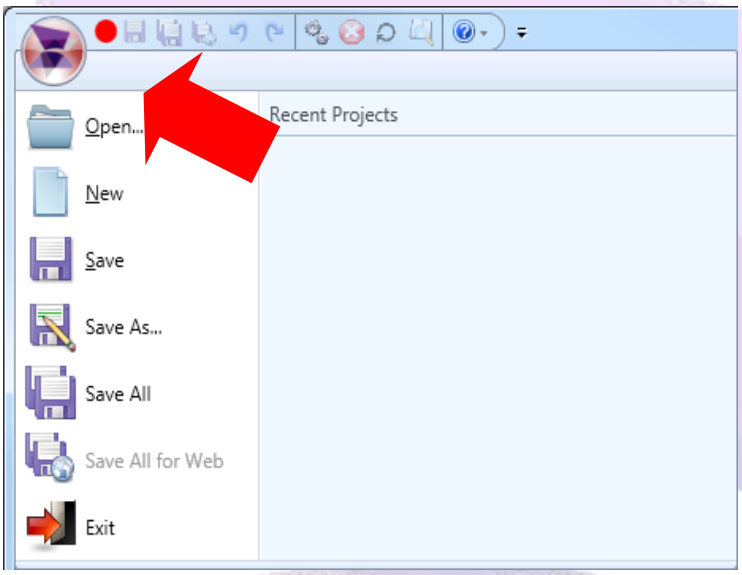
To restore the pane to its original state, the user may right-click the title bar and select Show.



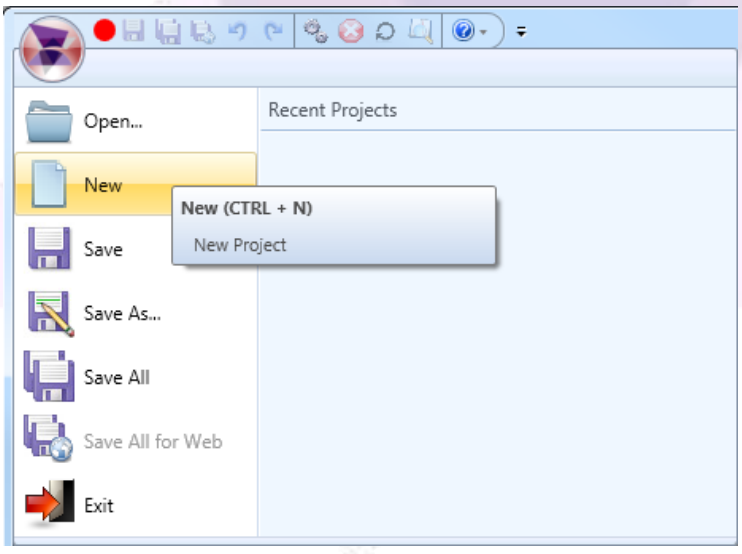
8. Creating a New Project

Now that the user is familiar with the Engineering Environment, they may create a new project.

- Click the Menu Button in the top-left corner of the program window.

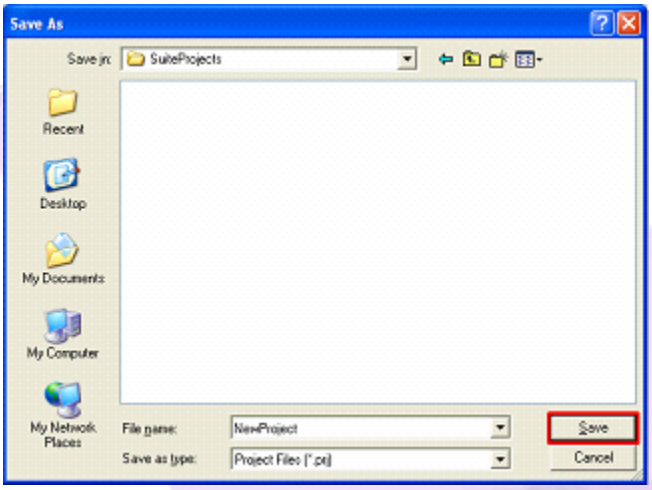


- Select New or press CTRL + N.



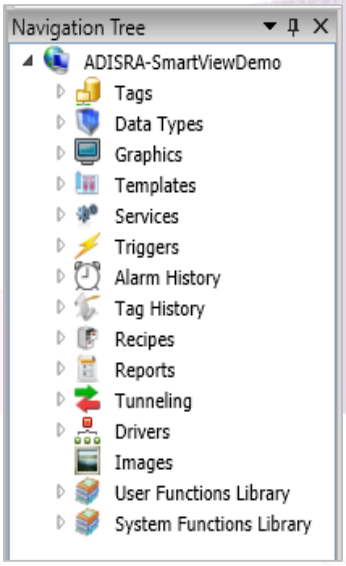
- A new "Save As" window appears. Select the folder in which to save the project and give the project a name. By default, the project is created with a (.prj) file extension.

NOTE: *Project names cannot contain special characters or spaces.*



For this example, we have named our project "GettingStarted". Once the user has named their project, click the Save button.

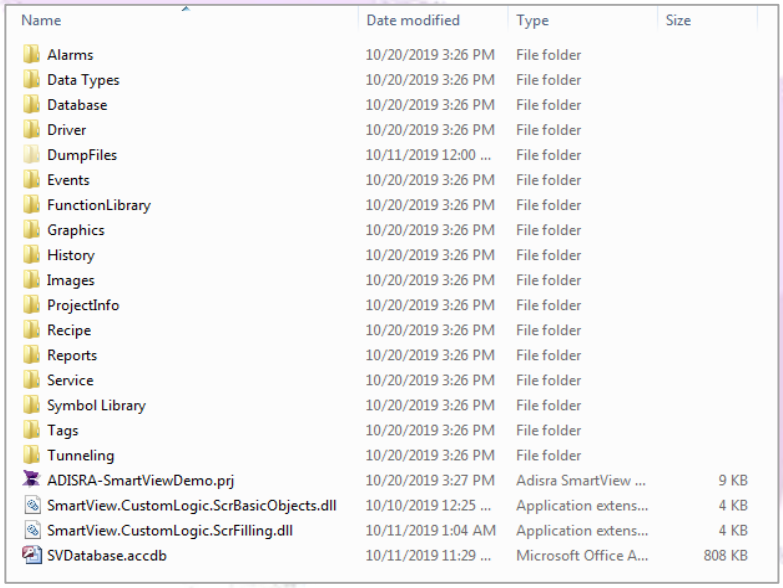
To begin the project, the user should become familiar with the workspace layout. ADISRA SmartView displays the structure of the project in the Navigation Tree, as shown below.



As seen in the Navigation Tree image, some documents are automatically created, such as Tags, Services, Alarms, Graphics, System Functions Library documents and others. The image represents the basic structure of the Navigation Tree for any project.

The user may start using these documents or they may create new documents and customize.

Below is an image of the folders located on the hard disk for a project. Each folder is created for its related documents.



Name	Date modified	Type	Size
Alarms	10/20/2019 3:26 PM	File folder	
Data Types	10/20/2019 3:26 PM	File folder	
Database	10/20/2019 3:26 PM	File folder	
Driver	10/20/2019 3:26 PM	File folder	
DumpFiles	10/11/2019 12:00 ...	File folder	
Events	10/20/2019 3:26 PM	File folder	
FunctionLibrary	10/20/2019 3:26 PM	File folder	
Graphics	10/20/2019 3:26 PM	File folder	
History	10/20/2019 3:26 PM	File folder	
Images	10/20/2019 3:26 PM	File folder	
ProjectInfo	10/20/2019 3:26 PM	File folder	
Recipe	10/20/2019 3:26 PM	File folder	
Reports	10/20/2019 3:26 PM	File folder	
Service	10/20/2019 3:26 PM	File folder	
Symbol Library	10/20/2019 3:26 PM	File folder	
Tags	10/20/2019 3:26 PM	File folder	
Tunneling	10/20/2019 3:26 PM	File folder	
ADISRA-SmartViewDemo.prj	10/20/2019 3:27 PM	Adisra SmartView ...	9 KB
SmartView.CustomLogic.ScrBasicObjects.dll	10/10/2019 12:25 ...	Application extens...	4 KB
SmartView.CustomLogic.ScrFilling.dll	10/11/2019 1:04 AM	Application extens...	4 KB
SVDatabase.accdb	10/11/2019 11:29 ...	Microsoft Office A...	808 KB

8.1. Opening a Project

To open an existing project, follow the steps below:

- Click the Menu button.
- Click the Open menu or press CTRL + O.
- A new window opens. In this window, browse to the project folder and select the file with extension .prj.
- Click the Open button.

9. Tags Documents

In ADISRA SmartView, a Tag refers to a unique identifier for a data point within the application. A tag is used to receive and write the live value from and to the Data Source.

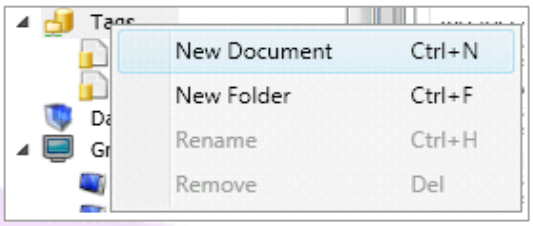
In the Tags document the user saves all tags used in the project. In the Tags document the user can create and remove Tags and edit their settings.

For our example, we will create some Tags that will be used throughout the guide with the following properties:

- Name: TagBoolean, Type: Boolean, Initial Value: "True"
- Name: TagInteger, Type: Integer, Minimum Values: Default, Maximum Value: Default, Value Initial: Default
- Name: TagFloat, Type: Float, Minimum Value: -5000, Maximum Value: 10000, Initial Value: 250, Retentive Value: Enabled
- Name: TagString, Type: String
- Name: TagDateTime, Type: DateTime
- Name: Vector1Dimension, Type: Float, Dimension: 1, Vector Size: 5
- Name: Vector2Dimensions, Type: Float, Dimension: 2, Vector Size: 2x3
- Name: Vector3Dimensions, Type: Float, Dimension: 3, Vector size: 2x3x4
- Name: VectorDynamic, Type: Float, Dimension: 1, Vector Size: Dynamic

9.1. Creating a Tags Document

To create a new Tag Document, right-click on the Tags folder or on any folder in which the user wants to create the document. They can also use the pre-created Tags document. Select New Document or press CTRL + N.

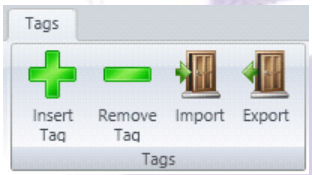


A tab opens in the workspace called "Tags1*". It will be an empty Tags Document until a new Tag is created in it.

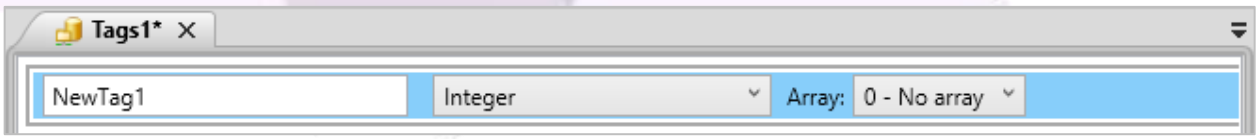
9.2. Adding New Tags

To insert a new Tag into the Tag Document, open the document in which the user wants to add the new tags. For our example, the user will use the "Tags" document that was created with the project.

When they open the Tag document in the Engineering Environment, the Tags Ribbon will update and display a new tab called "Tags" (see image below). On this tab there are 4 buttons: "Insert Tag", "Remove Tag", "Import", and "Export".



To create a new Tag in the document, click the "Insert Tag" button. By default, the tag will be created with the name "NewTag1" (as shown in image below). As the user will need nine (9) new tags for our example, they click the button again 8 times. This creates new "Integer" tags named "NewTag1" through "NewTag9".



9.3. Saving the Document

With the new Tags inserted, we need to save these changes made to the document. There are several ways to Save a document:

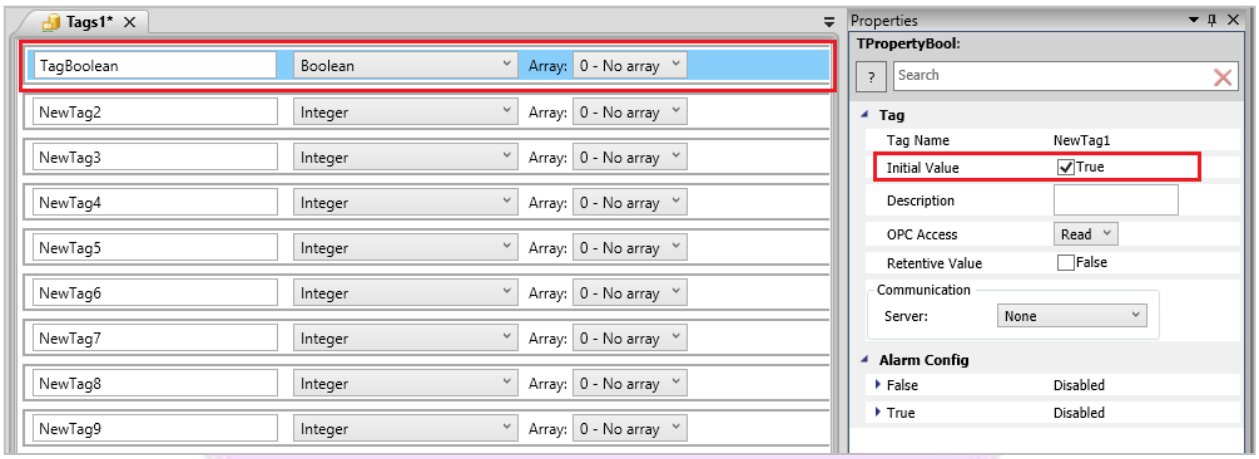
- Click on the Menu ( ) button and select "Save" from the drop-down menu
- From the Quick Access Toolbar, click on the disk icon ( ) icon
- With the document the user wants to save open in the document area, press CTRL + S keys

NOTE: *When a document has been changed, the document name will have the symbol " * " next to the document name, as seen in the previous image above next to the document name "Tags1*".*

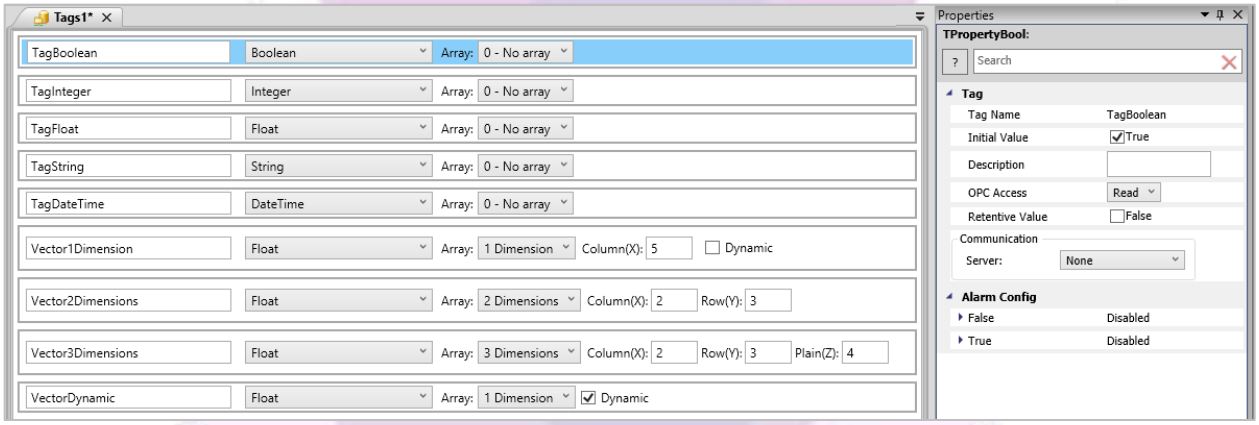
9.4. Configuring Tags

With the Tag Document open, the user can begin configuring each tag to match the properties described in 8.2 and 8.3. To set up the first tag, follow these steps:

- Click on the first tag line of the document to select it ("NewTag1"). When selected, the line should be highlighted in blue.
- Click on the name of the tag to rename it.
- Enter the new tag name, for our example "TagBoolean".
- Click on the tag type combo box to select the desired type.
- Select the type "Boolean" for tag type.
- In the ADISRA SmartView Property Window, click on the checkbox "Initial Value" so that its value is set to "True" as seen in the image below.



Repeat the steps for the other eight (8) tags that we created for our example. Once complete, the result should be the same as shown in the image below. With the tags properly configured, save the changes.



10. Data Type Documents

The Data Type Document allows the user to create the structure to be used when creating Tags. Using Data Types allows the user to group several different Tag Types together to create a structure. For example, within a Data Type the user can have a Float tag, an Integer tag, a String tag, etc.

Here the user will add and configure Tags as shown earlier in Creating and Configuring Tags Document. For our example, we will create a Data Type called "StructureExample", which will contain:

- Name: Member1, Type: Integer, Dimension: 1, Vector Size: 3
- Name: Member2, Type: Float, Dimension: 2, Vector Size: 3x3
- Name: Member3, Type: String, Dimension: 3, Vector Size: 3x3x3

All Data Types created will be available as a new "Type" in the Tags document.

10.1. Creating and Saving a Data Type Document

We will continue our project by creating a new Data Type document. To create a new Data Type Document, right-click on the Data Type folder in the Navigation Tree, and select New Document, or press CTRL + N.

A tab opens in the workspace called "Data_Types1*". It will be an empty Data Type Document until the user creates a new Data Type in it. They now have to give it a name and save it.

To save the Data Type Document, use one of the methods described in Saving the Document. Name the Data Type Document as "StructureExample".

10.2. Adding New Tags to a Data Type Document

To insert a new Tag into the Data Type Document, have the document in which the user wants to add the new tabs open in the workspace. In our example, we have the StructureExample document open.

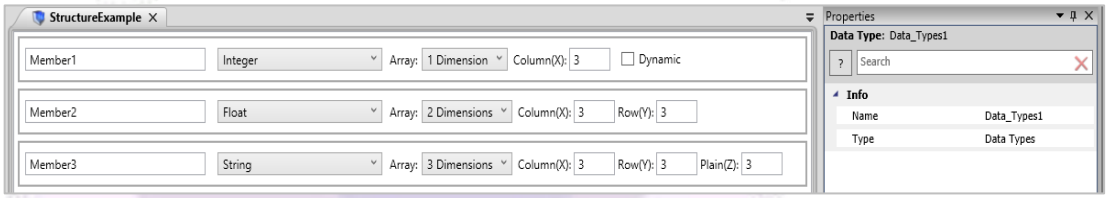
To create a new Tag in the Data Type Document, click the "Insert Tag" button. By default, the tag is created with the name "NewTag1". Since our example needs three new tags, click the button again two more times to create new Integer tags named "NewTag1" through "NewTag3".

10.3. Configuring Tags in a Data Type Document

With the StructureExample Data Type Document still open, the user can configure the three (3) new tags within this Data Type to match the following image or refer to the bullet points below from 9. Data Type Documents.

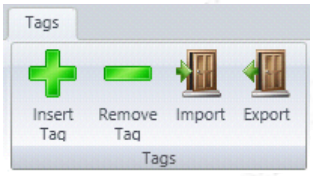
- Name: Member1, Type: Integer, Dimension: 1, Vector Size: 3
- Name: Member2, Type: Float, Dimension: 2, Vector Size: 3x3
- Name: Member3, Type: String, Dimension: 3, Vector Size: 3x3x3

Follow the steps in Configuring Tags. After the settings have been configured, the screen should look like the following image:



10.4. Add Data Type Tags in a Tags Document

With the Data Type document now configured, configure a new Tag and name it "Structure", and in the Tags Document mark the Date Type as "StructureExample".

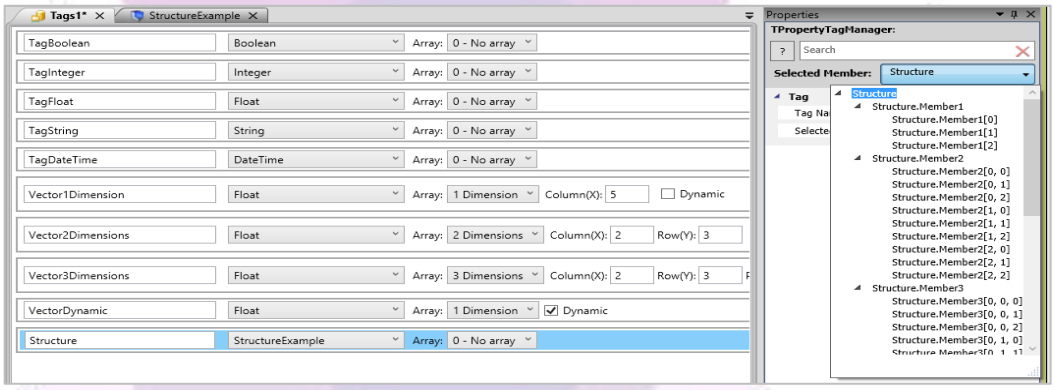


To configure this Tag, follow these steps:

- Open the Tags document created in Creating a Tags Document.

- Click the Insert Tag button in the Ribbon to insert a new tag ('NewTag1').
- Click the tag name and rename it “Structure”
- Click the tag type combo box and select the 'StructureExample' type.
- Save these changes.

In the following image, the Tags document has been updated with the new “StructureExample” tag that we named Structure. In addition, the ADISRA SmartView Property Window contains the combo box where the current tag structure is displayed. In this case the tag Structure will have three internal members: “Member1 – type Integer”, “Member2 – type Float”, and “Member3 – type String”.



11. Services Documents

A Document executes a code snippet within a certain time according to a configured condition or be executed by each tag value change configured in the document.

A feature of 'Services' Documents is that a document call can be made from anywhere in ADISRA SmartView. In other words, the document call can occur from an event in a graphical object, or from a configured "Trigger", or even from a created User Function Library. The user can also create customized scripts using C# programming language and use them later in the application.

Any new project automatically creates a Service's document called "Startup." This document automatically runs every time the application is run. This section shows how this works.

NOTE: *The 'Startup' document cannot be removed or renamed.*

For our example, GettingStarted, the user will configure the "Startup" document and create two more. In one document, they will set a condition for it to be executed and in the second one, the user will set a tag so that the document is executed every time that tag's value changes.

11.1. Configuring the Startup Document

To configure the "Startup document", follow these steps:

- In the ADISRA SmartView Navigation Tree, expand the Services node to access the Startup document.
- Open the Startup document using any of the following:
- Double-click the Startup document to open it.
- Right-click on the document so that a pop-up menu appears, then select the "Open" option.
- Select the document and then press CTRL + O.

- Once the document is open, the user will be able to write scripts. In this script box, please set the following code:

@TagDateTime = DateTime.Now;
@TagString = **"Startup document executed successfully!"**

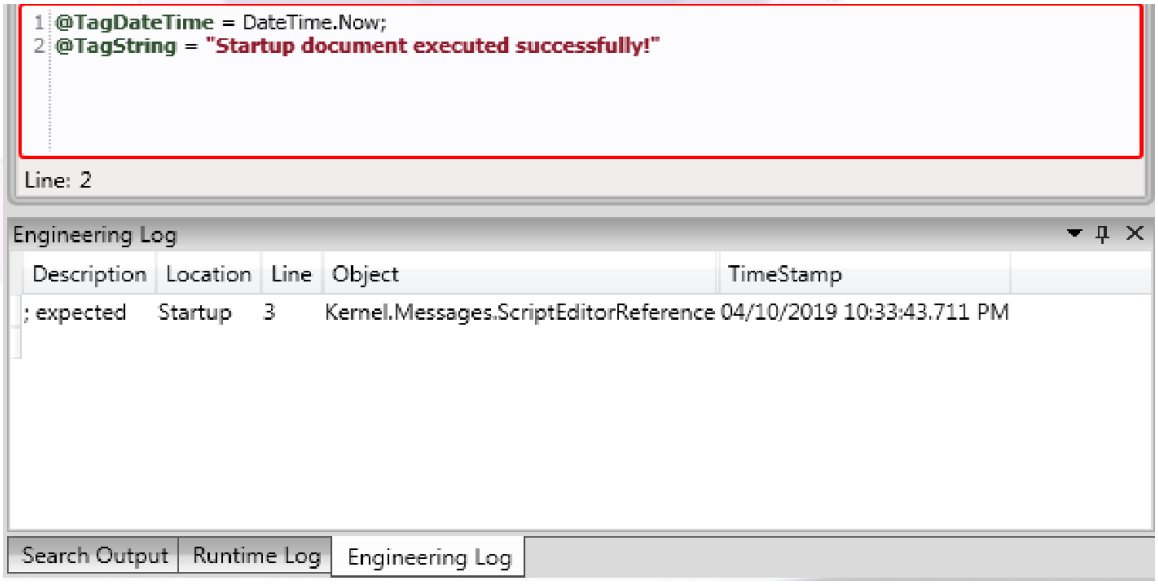
This code snippet causes the tag "TagDateTime" to receive the time at which the Startup document was executed, and the tag "TagString" to receive text saying that the document was successfully executed.

- In the ADISRA SmartView Ribbon, locate the "Scripts" tab with a "Verify" button (green checkmark).

- Now check that the script configured in the document is correct. To do this, click the Verify button. The checked result should return an error.

    - How does the user identify a script error? If there is an error in the script, the field border will be red.

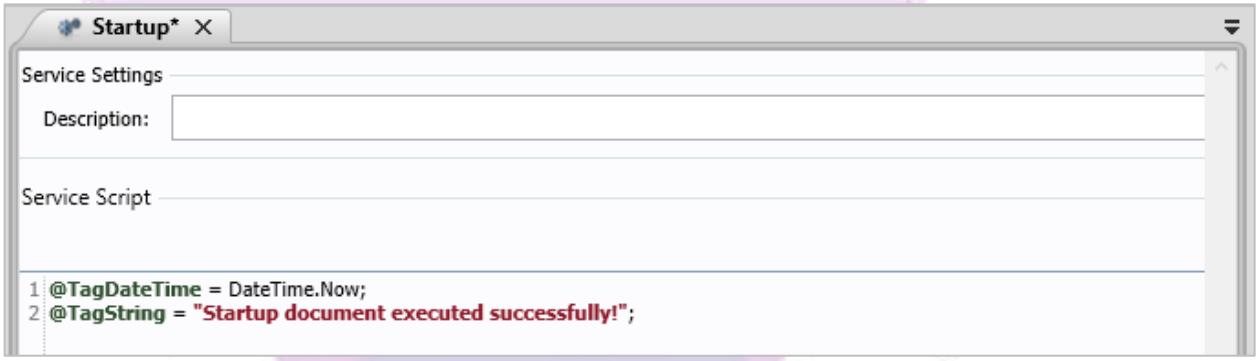
    - How does the user identify the script error? A message is displayed in the "Engineering Log" pane indicating which error was found and where the error is located.



- To correct the error in the script, enter the semicolon ";" at the end of line 2:

@TagDateTime = DateTime.Now;
@TagString = **"Startup document executed successfully!";**

- Click the Verify button again and notice that the red border is gone, which means the script executed successfully.



- Save the changes made to the document.

11.2. Create a Services Document

To create and configure a new Services Document and apply a Condition Type, follow these steps:

- Create a new Services document by right clicking the "Services" node in the Navigation Tree and selecting the New Document option from the menu. Or select the Services node and then press "CTRL + N". By default, the name of the Services Document created will be called "Services1*".

- In the Type property, select the "Condition" option. This means the document will only be executed while the configured condition is True. By default, the initial value of the condition is set to "True".

- In the Condition text box set the following condition: @TagBoolean. With this setting in place, the document will execute every time the Boolean tag value is equal to "True".

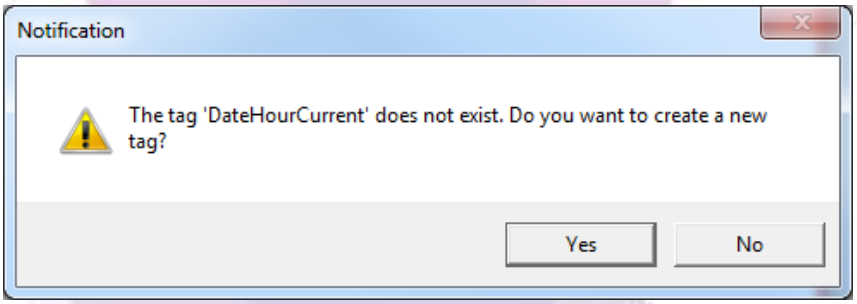
- Click on the "Verify" ☑ button next to the field to verify that the condition is correct.

- In the "Interval (ms)" field, set the time interval at which the document will check the configured condition. For our example, set the value to "800".

- In the Services script area, write the following code:

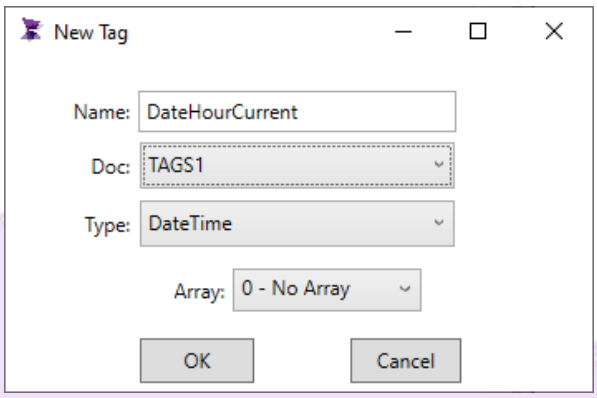
  @DateHourCurrent = DateTime.Now;

- Click on the "Verify" button on the Scripts Ribbon. Since the configured tag does not exist in our project, a message box will appear stating that the configured tag does not exist. It will give the option to create it if the user wishes.

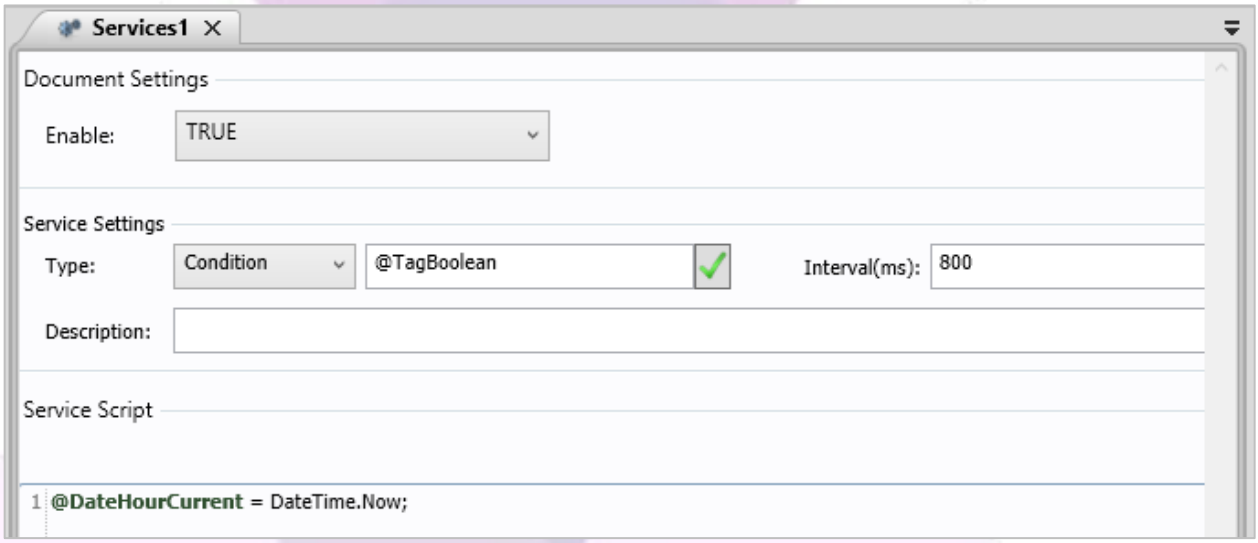


- Since we do want to use this tag in our project, create it by clicking Yes to create the new Tag.

- A new window appears in which the user will configure the properties of the new tag. For this project create the tag in the "Tags1" document with the Type "DateTime" and a dimension equal to "0" (zero).



- Click OK to confirm the creation of the tag.
- Click the Verify button on the Ribbon again. This time there should be no errors found.
- Save these changes made to the document.



This concludes the configuration of the Services Document with the Condition Type.

11.3. Create a Services Document with Triggers

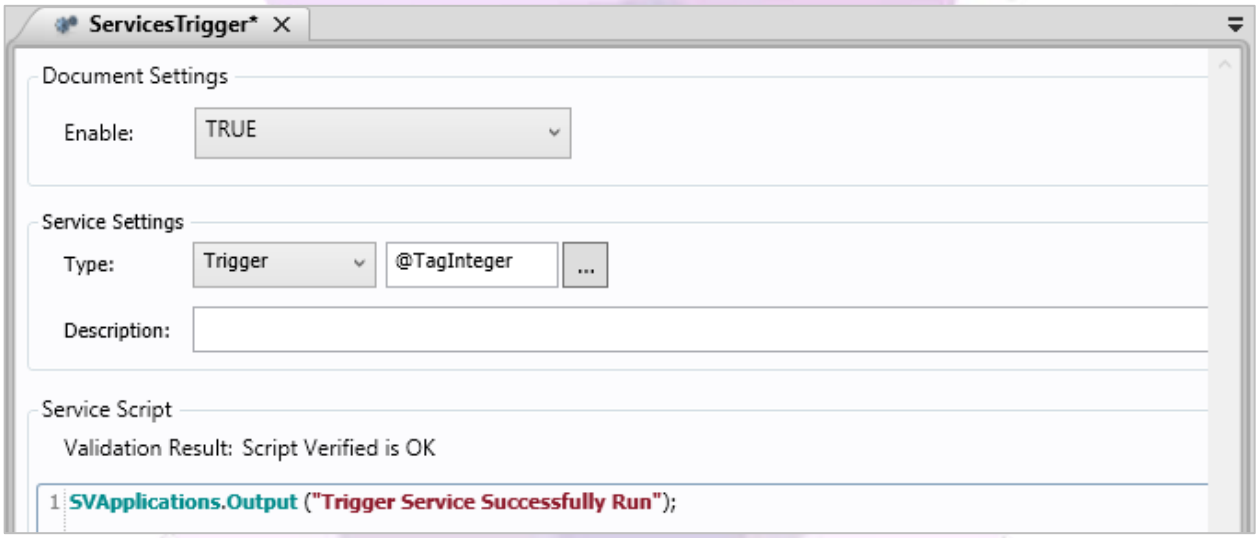
To create a Services Document and configure it with Triggers, follow these steps:

- Create a new Services document and name it "ServicesTrigger".

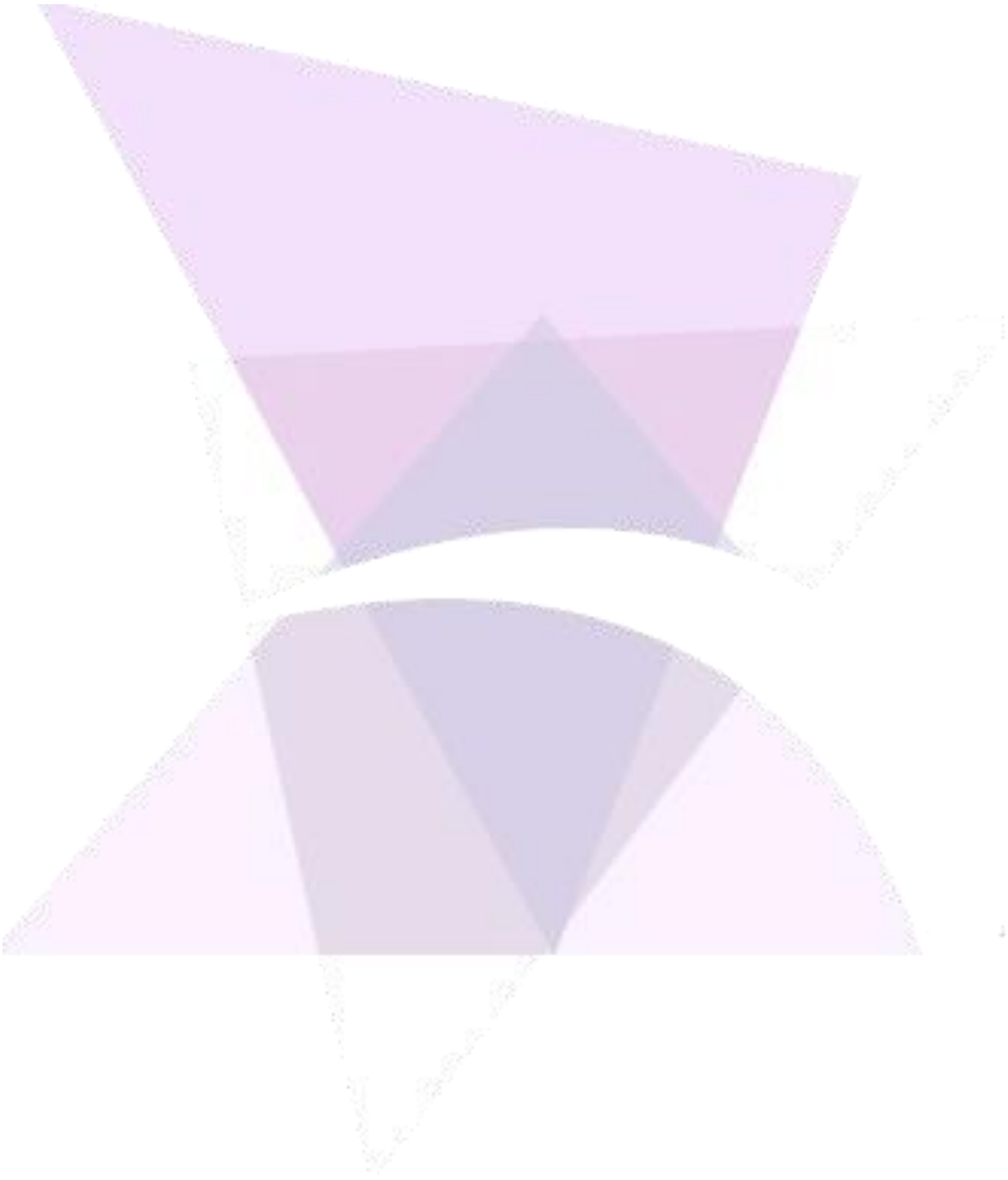
- In the Type field, select the "Trigger" option.
- In the Trigger field, set the tag @TagInteger.
- Click on the Verify button in the Scripts Ribbon to verify that the condition is correct.
- After configuring the Trigger, configure the script that will be executed by entering the following script in the Services script area:

```
SVApplications.Output ("Trigger Service Successfully Run");
```

- Save changes made to document.



With the Services Triggers script configured, when the Service is executed, it displays the message in the Engineering Log pane. These messages are helpful for logging what is running in the project.



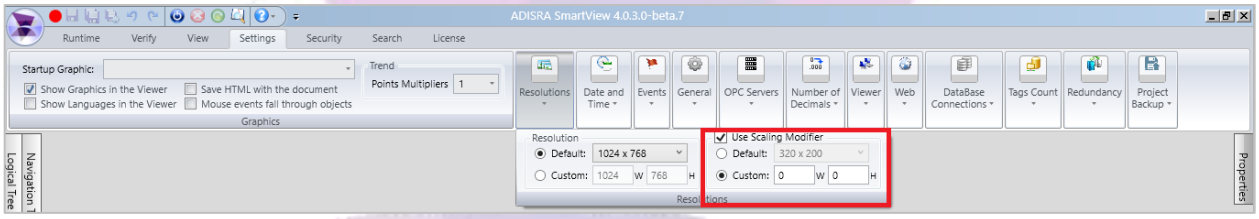
12. Configuring Services Documents for Screens

Once the Services settings are complete, create a Graphics screen so the user can execute and track the Services execution in their project.

12.1. Setting the Default Resolution for Charts

For the example, configure Charts with a default size of 1014 x 674 (width x height). Then set the default resolution for all graphics created from this point forward to have this size. To perform this configuration, follow the steps below:

- In the Ribbon "Settings" find the group called "Resolutions". Click the down arrow to open the sub-window below it.



- Locate the "Resolution" portion in the toolbar (marked in red above). Activate the "Use Scaling Modifier" check box then click the "Custom" radio button and in the Width field (marked by a W), enter the value "1014".

- In the Height field (marked by an H), enter the value "674". As a result, all Graphics Documents created going forward will have this resolution.

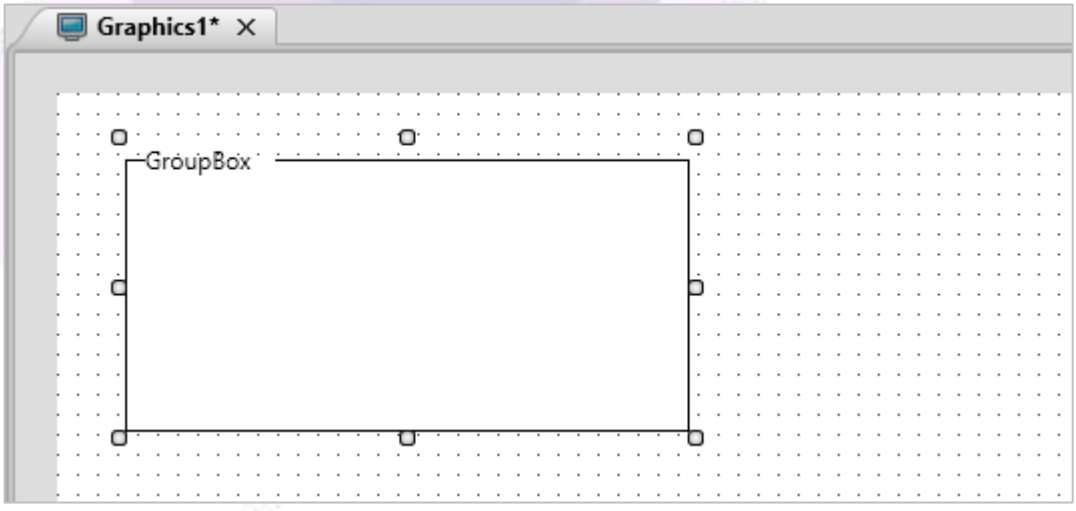
12.2. Creating a Services Document for Graphics

To create the Graphic Document for Services, follow these steps:

- To better organize Graphics Documents, the user needs to first organize folders for saving documents. To create a folder, right click on Graphics in the Navigation Tree.

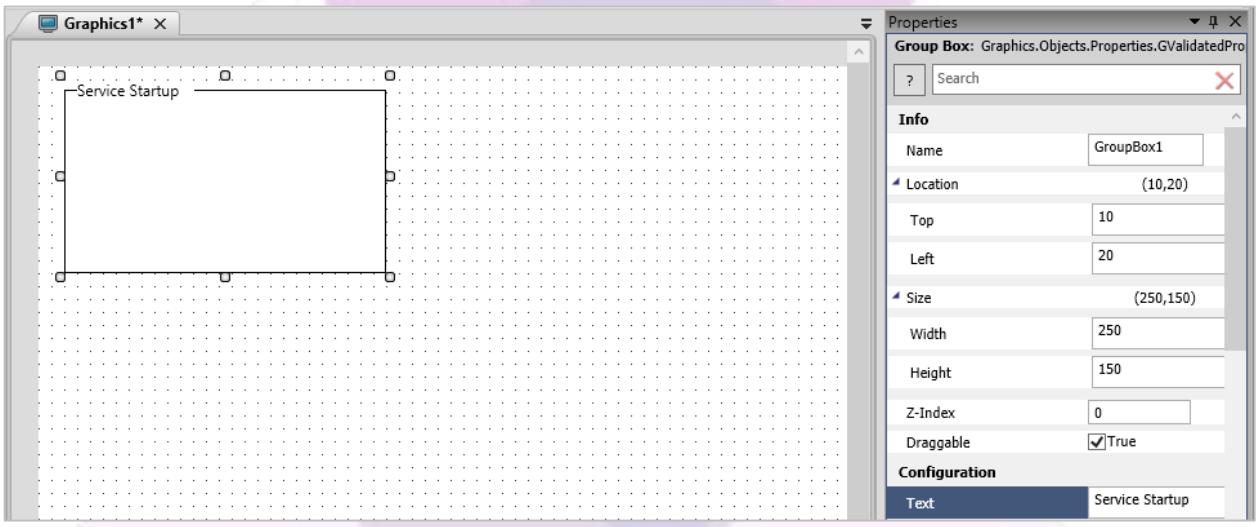
- In the context menu that appears, select the "New Folder" option.
- By default, a folder named "NewFolder" is created. Rename this folder to "Services" and press Enter.

- Now, create the Graphic Document by right clicking the mouse button on the Services folder that was just created to display the context menu.
- Select the "New Document" option to create the document.
- A new graphic document is created and by default the screen is named "Graphics1".
- The first graphic object the user creates will be a "GroupBox". The GroupBox function delimits areas and better organizes objects on the screen. To create a GroupBox, select the Graphics Ribbon and click on GroupBox located in the "Interface Objects" group.
- Inside the work area, to create an object, click and hold the mouse button. Drag the mouse pointer to create the GroupBox object of the desired size.



- After the GroupBox object is created, the user can adjust its size and position on the screen. To do this, select the created object by clicking on it with the left mouse button.

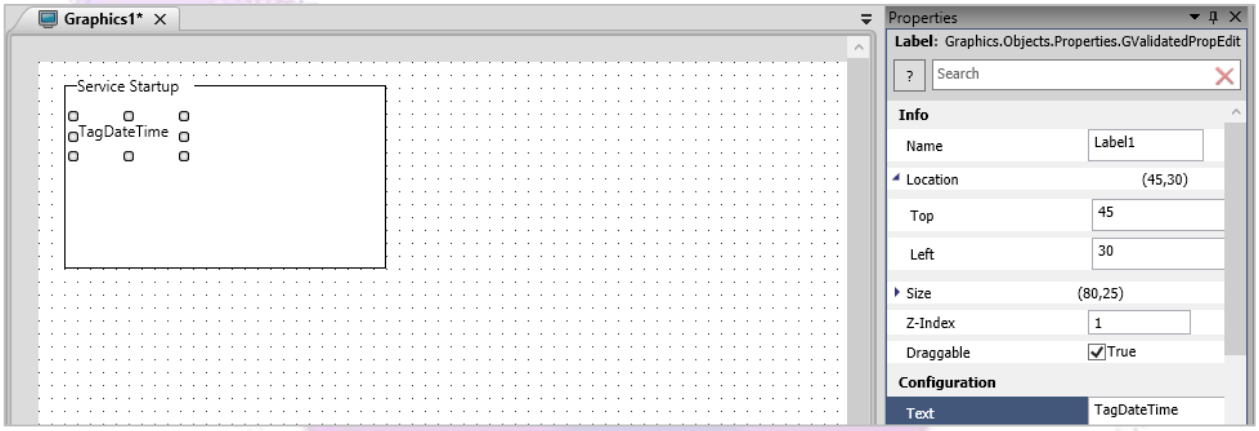
- In the Property Window, locate the "Size" label. If the Size area is not opened, click the arrow to display the Width and Height fields.
- Place the cursor in the Width property and set the value to "250". In the Height field, set the value to "150".
- While still in the Property Window, find the "Location" label. If the area is not opened, click the arrow next to the label.
- In the "Top" property field, set the value to "10" (ten). Set the "Left" property field value to "20" (twenty).
- Now configure the text displayed inside the object. To do this, change the value contained in the "Text" field. For this example, enter the value "Service Startup".



- Within the GroupBox created, display the values of the "TagDateTime" and "TagString" tags to track the execution of Service Startup. To do this, the user will need to create two Label objects and two Text Box objects. To create a new "Label" object, locate the Graphics Ribbon and then select the "Label" object located in the "Basic Objects" group. Click anywhere in the document to create a default object (size 80 wide x 25 high)
- With the Label object selected, configure the size and location. Locate the Size property; in the Width property field, set the value to "85" and in the Height property field, set the value to "16". In

the field “Top”, set the value to “65” and in “Left“, set the value to “30”.

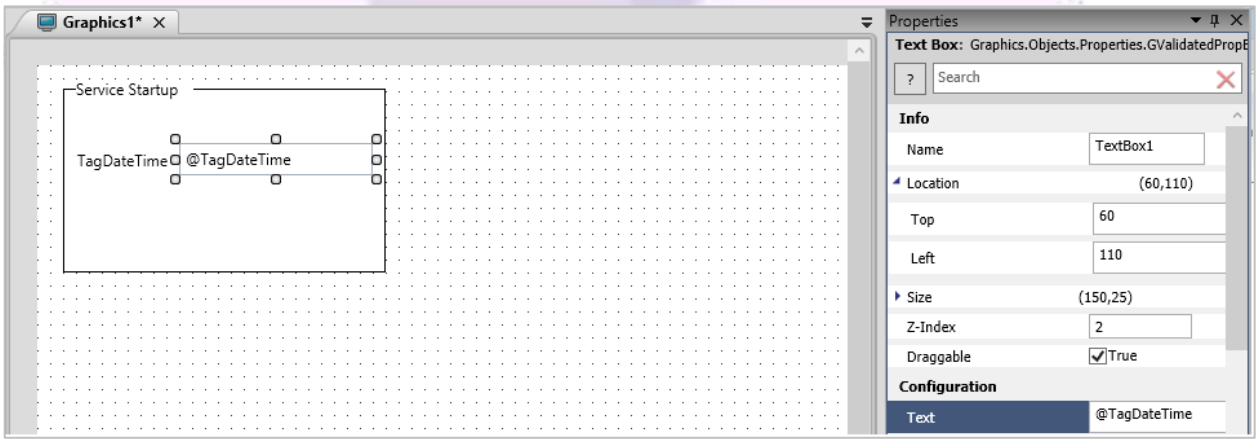
- In the "Text", set the value "TagDateTime".



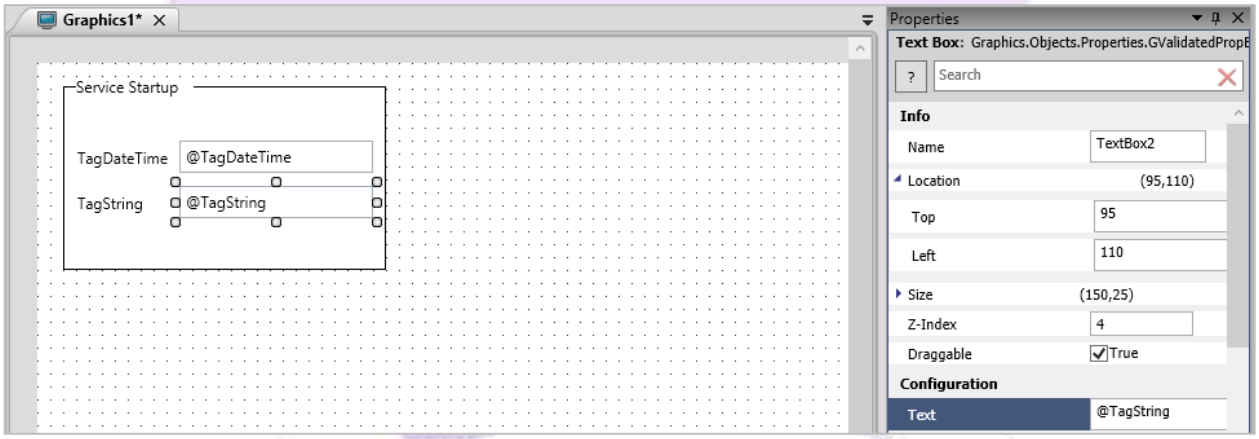
- Now, create a "TextBox" object to display the actual value of the tag. Select the Graphics Ribbon and then click "TextBox" object located in the "Basic Objects" group.

- After creating the TextBox object, the user can configure the Text Box size and location. In the Width property field, set the value to "150" and in the Height property field, set the value to "25". In the "Top" property, set the value to "60" and "Left" to "110".

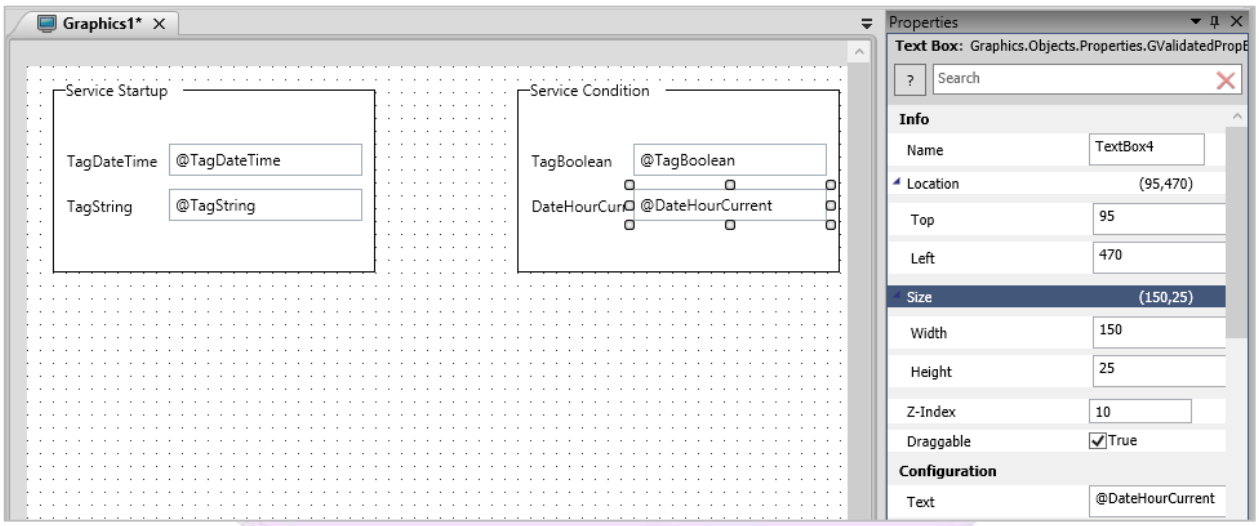
- In the "Text" field, set the value "@TagDateTime". The "@" character indicates that we want to display the value of the tag and not just the text "TagDateTime" in the object.



- Now, repeat the same process shown earlier for the following objects:
    - Label = TagString, Width = 85, Height =16, Top = 100, Left = 30, Text: TagString
    - TextBox = TagString, Width = 150, Height = 25, Top = 95, Left = 110, Text: @TagString

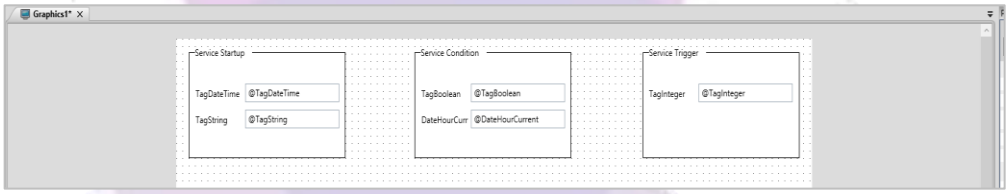


- GroupBox = Service Condition, Width = 250, Height = 150, Top = 10, Left = 380, Text: Service Condition
- Label = TagBoolean, Width = 80, Height = 16, Top = 65, Left = 390, Text: TagBoolean
- TextBox = TagBoolean, Width = 150, Height = 25, Top = 60, Left = 470, Text: @TagBoolean
- Label = DateHourCurr, Width = 85, Height = 16, Top = 100, Left = 390, Text: DateHourCurr
- TextBox = DateHourCurrent, Width = 150, Height = 25, Top = 95, Left = 470, Text: @DateHourCurrent

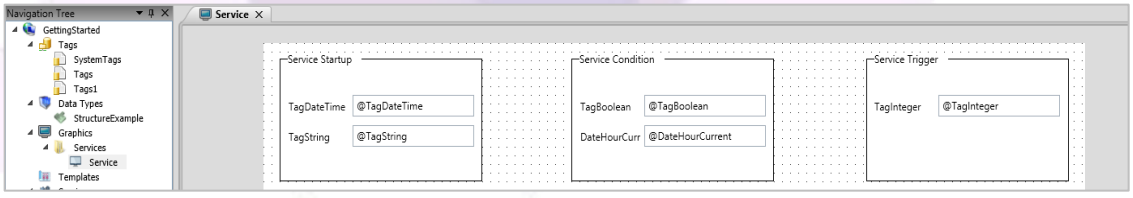


- Once the settings above are complete, save the graphic document and name it "Service".

The Graphic Document result should look similar to the figure below:



- Next the user should save the newly created screen. Let's name it "Service".



13. Trigger Documents

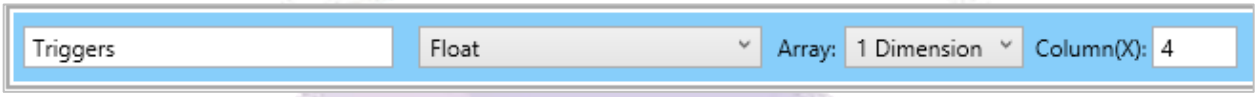
The Trigger Document executes a set of expressions or a Services Document according to a condition, a tag value change, a time interval, or when a predetermined date is reached.

The following describes how to set up each case and explains them in more detail.

13.1. Creating New Trigger Tags

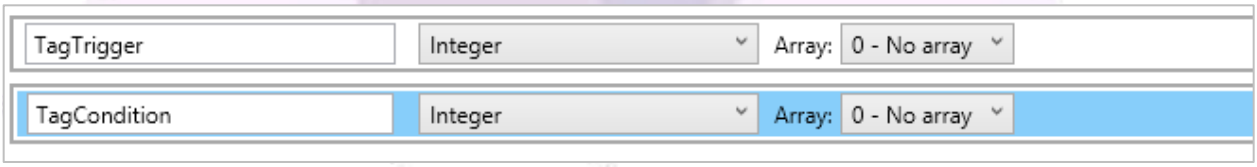
First, create a new "Float" Tag Type called "Triggers" with a size of "1" that has four positions.

Open the document tags "Tags1", click the "Insert Tag" button in the Ribbon, rename the tag to "Triggers". Next, change the Tag Type to "Float", change the array Size of the tag from "0 - No array" to "1 Dimension", and change the size of Column(X) to "4".



Follow the same process to create the following Tags:

- TagTrigger: Type "Integer", array size "0 - No array"
- TagCondition: Type "Integer", array size "0 - No array"



13.2. Creating Trigger Types

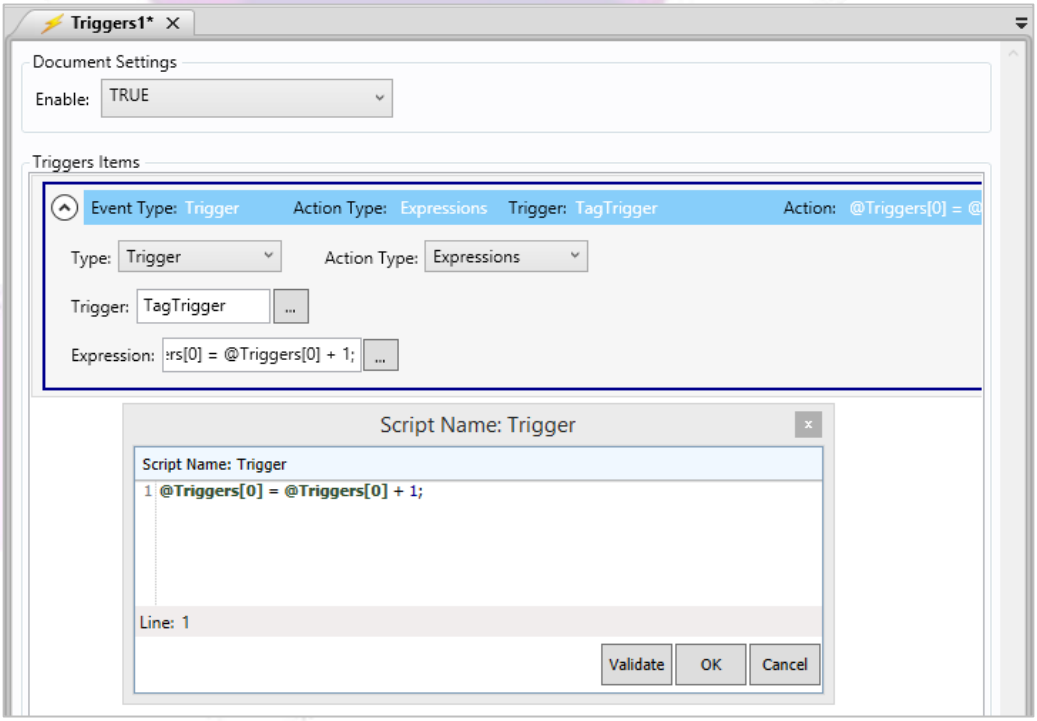
A Trigger Type is executed when the configured tag has changed its value. To add and configure a Trigger, follow these steps:

- Create a new "Trigger" Document. To do this, right-click on "Triggers" in the Navigation Tree and select "New Document". By default, a new document will be created called "Triggers1".

- On the "Triggers" Ribbon at the top, click the button "Insert Trigger".
- In the newly created item, select the "Trigger" value for the field "Type".
- For the "Action Type" field, select the "Expressions" option.
- In the "Enable" field, select the value "TRUE".
- In the "Trigger" text field, insert the tag "@TagTrigger".
- In "Expression" field, configure the following code snippet:

```
@Triggers[0] = @Triggers[0] + 1;
```

- Save changes to the document.



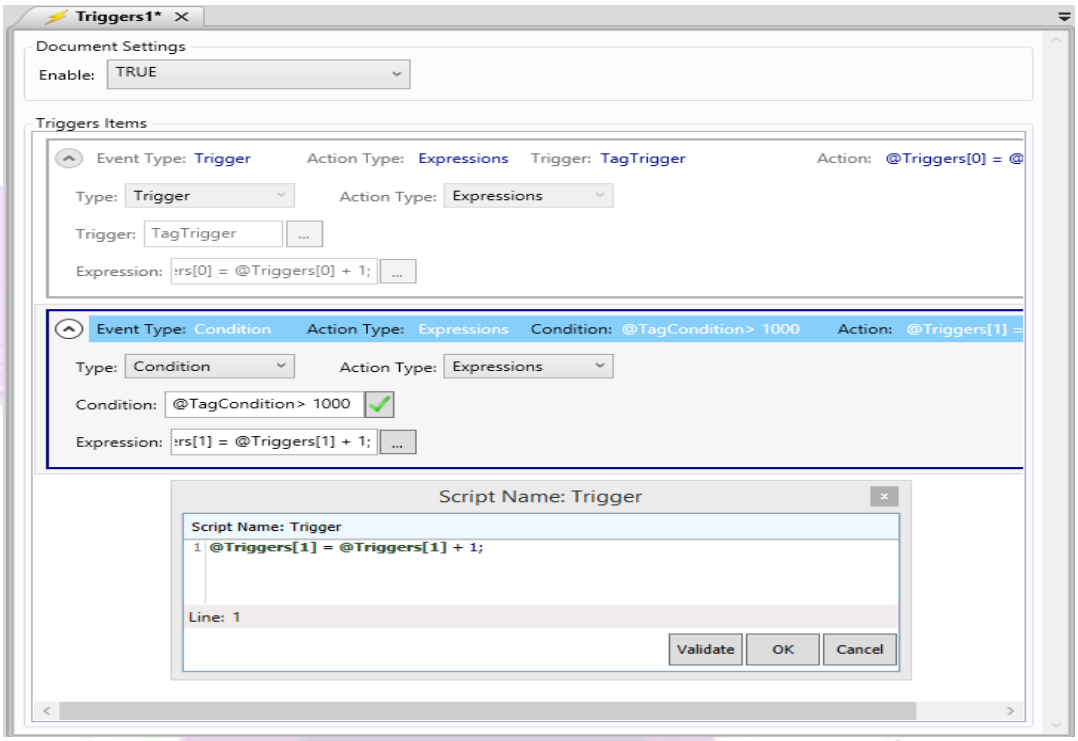
13.3. Creating Condition Type Triggers

A Condition Trigger is executed when the configured condition meets the True result. To add and configure a Trigger with Condition Type, follow these steps:

- With the document "Triggers1" open, click the "Insert Trigger" button in the Ribbon.
- In the newly created item, select the "Condition" value for the field "Type".
- For the "Action Type" field, select the "Expressions" option.
- In the "Enable" field, select the value "TRUE".
- In the "Condition", enter the condition: "@TagCondition> 1000".
- In "Expression" field, configure the following code snippet:

```
@Triggers[1] = @Triggers[1] + 1;
```

- Save the changes to the document.



13.4. Creating Calendar Type Triggers

A Calendar Trigger is executed when the set date condition meets the True result. To add and configure a Trigger with Calendar Type, follow these steps:

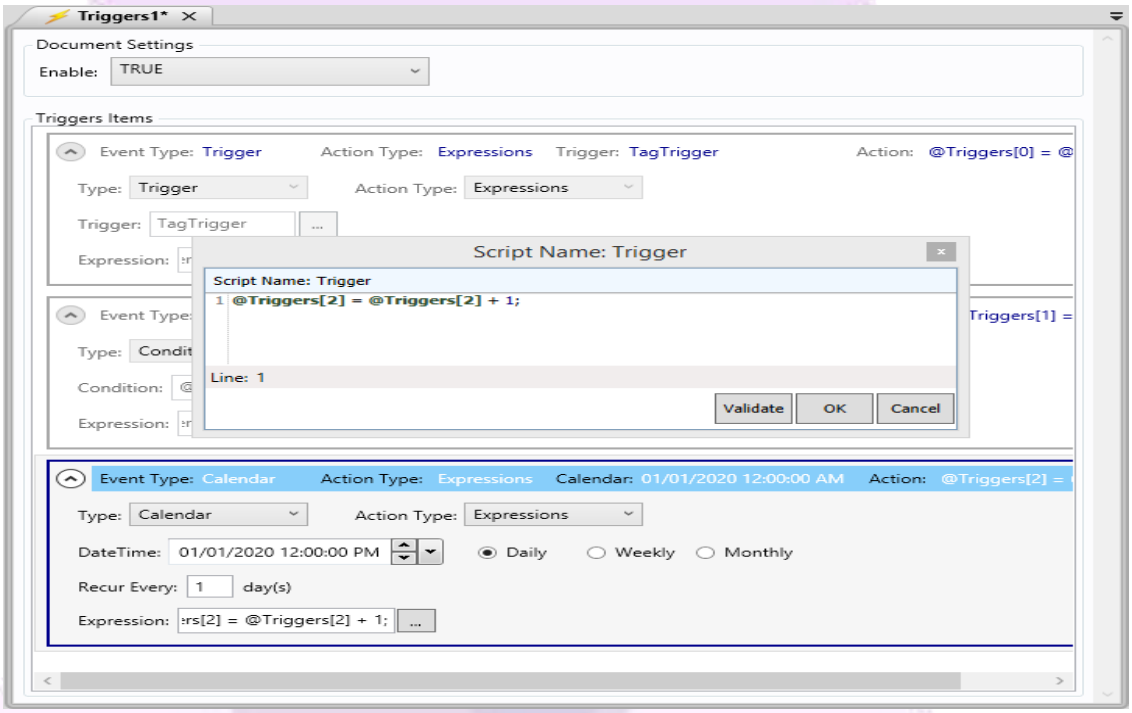
- With the document “Triggers1” open, click the "Insert Trigger" button in the Ribbon.
- In the newly created item, select the "Calendar" value for the field "Type".
- For the "Action Type" field, select the "Expressions" option.
- In the "Enable" field, select the value "TRUE".
- Select the "Daily" option. Use this option if the user wants a code to run every day at a certain time to and from a data set.
- In the "DateTime" field, set the following value: 01/01/2020 12:00:00 PM. With this configuration, as of 12 noon on January 1, 2020, the "Trigger" is performed.

- In the "Recur Every __ day (s)" field, set the value "1". This means that this "Trigger" is performed every day. If the user configured the value "2", the "Trigger" would be performed every two (2) days, and so on.

- In "Expression" field, configure the following code snippet:

  @Triggers[2] = **@Triggers[2]** + 1;

- Save the changes to the document.



13.5. Creating Interval Type Triggers

An Interval Trigger is performed in time intervals when the set interval conditions have met the True result. To add and configure an Interval Trigger type, follow these steps:

- With the document "Triggers1" still open, click the "Insert Trigger" button in the Ribbon.

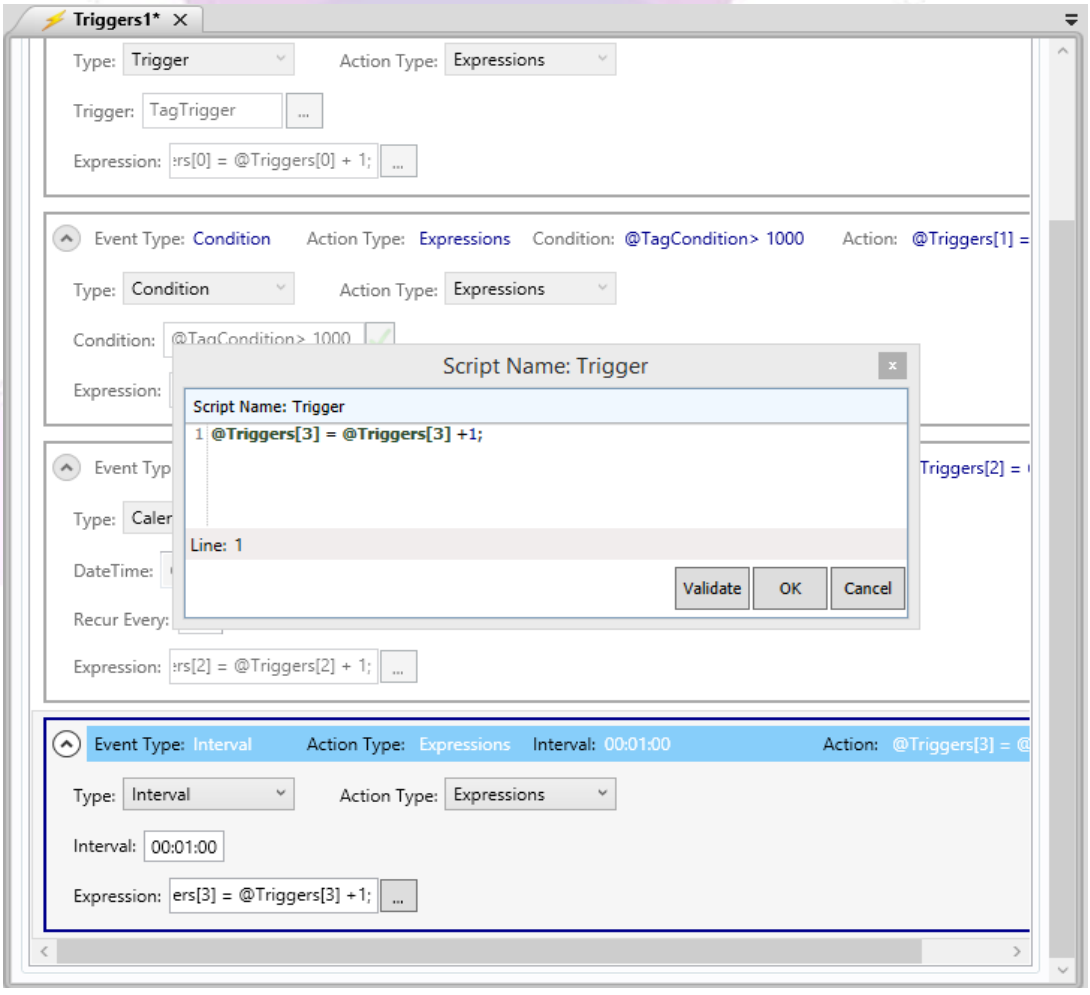
- In the newly created item, select the "Interval" value for the field "Type".

- For the "Action Type" field, select the "Expressions" option.
- In the "Enable" field, select the value "TRUE".
- In the "Interval" field, set the value "00:01:00". The "Trigger" will run every minute.
- In "Expression" field, configure the following code snippet:

```
@Triggers[3] = @Triggers[3] +1;
```

- Save the changes to the document.

Once these settings are complete, the document should look like the image below:



14. Configuring Graphic Screens for Triggers

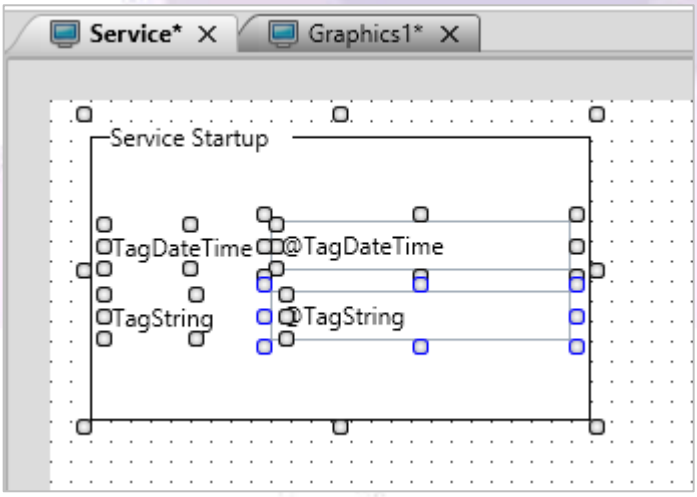
A Graphic Screen displays the Triggers. Creating one is like creating the Services Screens. Follow the steps below to create a GroupBox object to define each Trigger, a Label with a tag name, and a TextBox displaying the tag value.

- Create a new folder called "Triggers" by right-clicking on "Graphics" in the Navigation Tree and selecting "New Folder". Rename the folder "Triggers".

- Create a new document under this folder. By default, it will be named "Graphics1" with a window size of 1014 x 674 (width x height).

- Open the graphic document "Service".

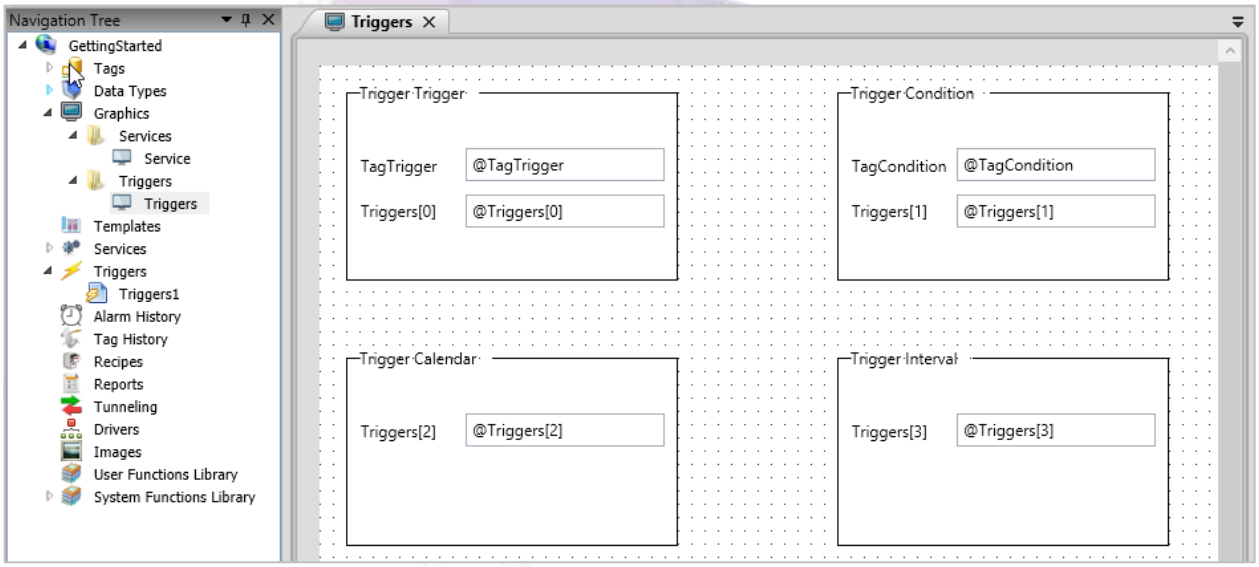
- With the graphic document "Service" in the main workspace, click anywhere near the object "GroupBox" "Service Startup" and hold down the left mouse key. Drag the mouse until all the objects within the "GroupBox" are selected. Release the mouse. The objects will stay selected with the handles being displayed (as shown in the figure below).



- Right-click the selected objects to display the context menu and select "Copy" (or just copy it using CTRL+C).

- Now switch to the new document tab "Graphics1" again.

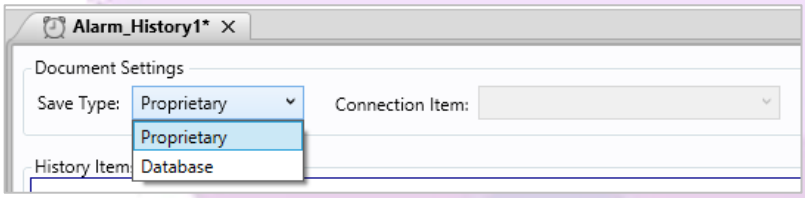
- Click the right mouse button near the point (0.0), upper left-hand corner of the document.
- Select the Paste option. The objects copied from the screen "Service" are duplicated to the screen "Graphics1".
- Rename the objects in the group with the following values:
  - GroupBox: Text: "Trigger Trigger"
  - Label1: Text: "TagTrigger"
  - TextBox1: Text: "@TagTrigger"
  - Label2: Text: "Triggers[0]"
  - TextBox2: Text: "@Triggers[0]"
- Repeat the process for the other triggers to match the figure below.
- Save the document with the name "Triggers".



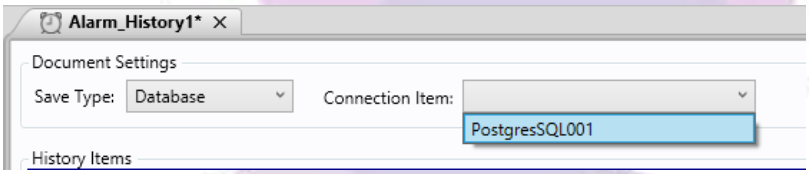
15. Alarm History Documents

The document "Alarm History" contains the settings so that tags history is stored. It can be saved in the proprietary database or into a relational database previously configured in the top bar menu Settings -> Database Connections.

Types of configurations:



Setting the database to store values in the relational database



NOTE: *The user cannot configure the same tag in different documents.*

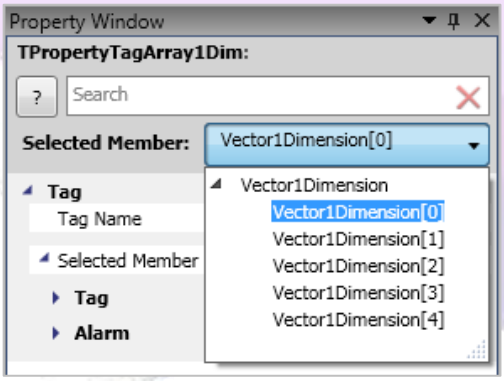
15.1. Configuring the Alarms Tag

Alarms are commonly used in SCADA software and add important functionality to control tag values. If, for example, a process tag has a value that can be dangerous, let's say, outside an acceptable range, ideally, the application should notify the operator of this danger. Within ADISRA SmartView, the user can create alarms for the tags and save the alarms in the database, show them on a monitoring screen, allowing the operator to view the alarms, and acknowledge them in addition to taking action to normalize the alarm. With the historized alarm, it will be possible to search for all alarms in the future and even generate reports from the moment the alarms were generated, normalized, and recognized by which operator.

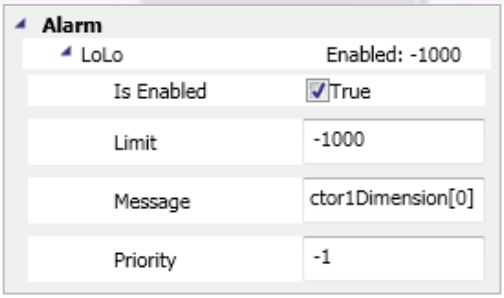
Now let's configure alarms for five (5) tags. These alarms will be used later by the Alarm Object to display the alarm's message when each

tag's value is less than the configured alarm limit. The user will set up the "Vector1Dimension" Alarm Tag using the following steps:

- Locate the Tags folder in the Navigation Tree and open the document "Tags1".
- Select the tag "Vector1Dimension".
- In the "Property Window", select the desired member to be configured. In this step the user sets the position "0" (zero).



- In the "LoLo" Alarm area, next to IsEnabled, check the box next to "True". This makes all other fields active.
- In the "Limit" field, set the value as "-1000".
- In the "Message" field, set the value as "LoLo Tag Vector1Dimension[0]".
- In the "Priority" field, set the value as "-1". (See image below)



- Now, configure the remaining VectorDimension 1-4 Tags. Set all other members with the following values:

- Vector1Dimension[1]: LoLo Limit: -750; Message: LoLo Tag Vector1Dimension[1]; Priority: -2
- Vector1Dimension[2]: LoLo Limit: -500; Message: LoLo Tag Vector1Dimension[2]; Priority: -3
- Vector1Dimension[3]: LoLo Limit: -250; Message: LoLo Tag Vector1Dimension[3]; Priority: -4
- Vector1Dimension[4]: LoLo Limit: -100; Message: LoLo Tag Vector1Dimension[4]; Priority -5

- Save the changes made to these Tag's document.

15.1.1. Alarm Types

- LoLo: The alarm is triggered if the tag value is SMALLER than the value set in the "Limit" field, usually used when a second "Lo" alarm is needed.
- Lo: The alarm is triggered if the tag value is SMALLER than the value set in the "Limit" field.
- Hi: The alarm is triggered if the tag value is GREATER than the value set in the "Limit" field.
- HiHi: The alarm is triggered if the tag value is GREATER than the value set in the "Limit" field, usually used when a second "Hi" alarm is needed.
- Deviation: The alarm is triggered if the tag change value is GREATER than the value set in the "Variance" field. The alarm returns to its normal state if the tag is longer than the "Time To Level Up (sec)" field without having its value changed above the value set in "Variance".
- Freeze: The alarm is triggered if the value does not change its value for a longer time than set in the "Time (sec)" field.

- Watch Dog: The alarm is triggered if the tag value changes. The alarm returns to the normal state if the tag value is not changed for longer than the setting in the "Time (sec)" field.

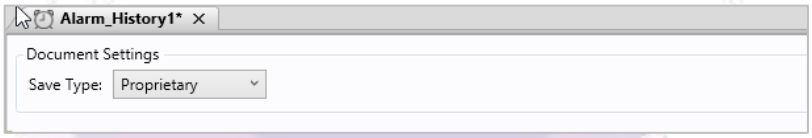
15.2. Configuring Alarm History Items

The Alarm History allows the application to store the alarms in a database. It can be in the proprietary database or in any relational database of customer's choice.

In the Alarm History Document, configure it according to the Alarms configured for the tags using the following steps:

- Locate the Alarm History category in the Navigation Tree. Right-click to display the submenu and select "New Document". By default, the new document is named "Alarm_History1".

- With the document open, set the Save Type to Proprietary



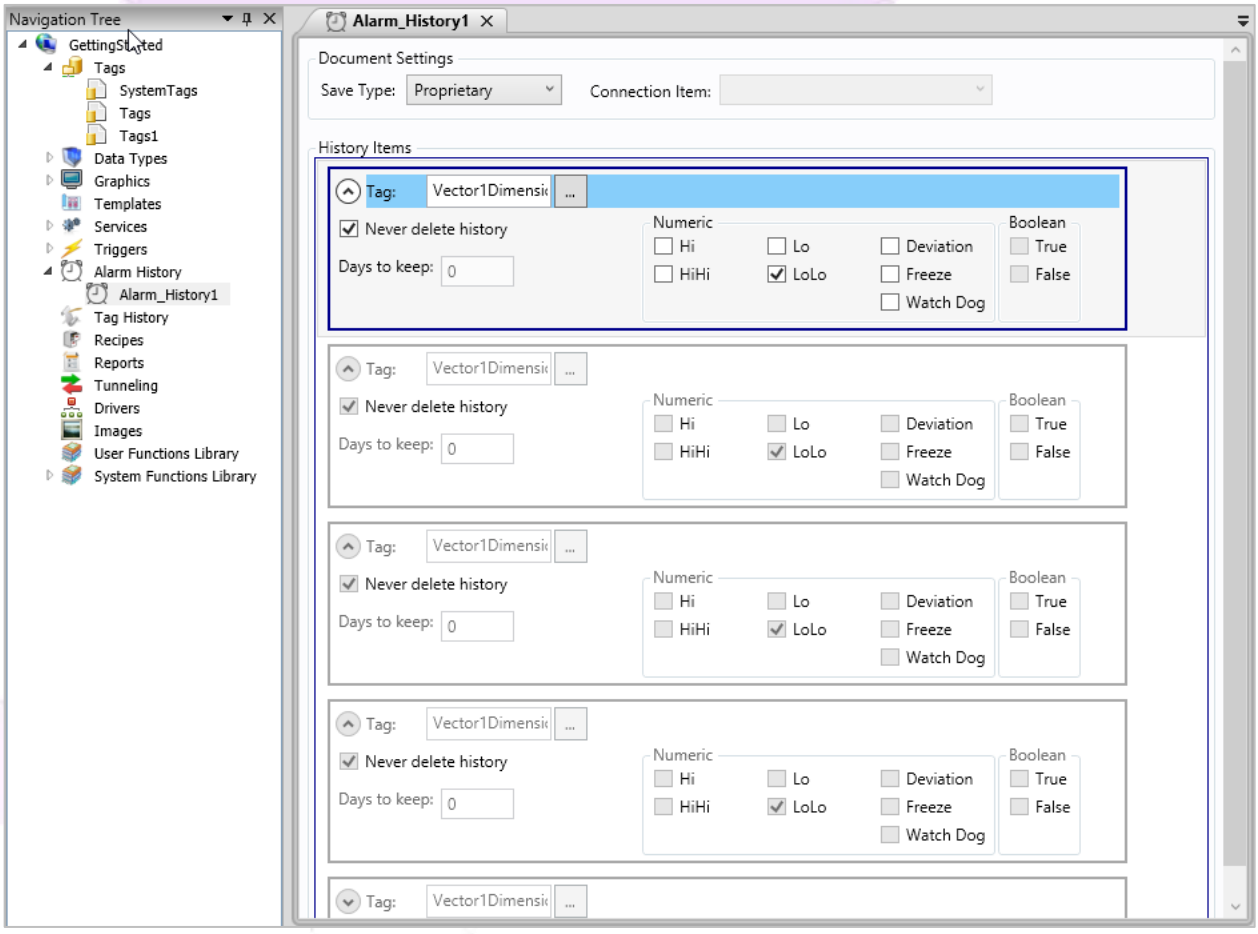
- Click the "Insert Alarm" button in the Alarms Ribbon.

- In the newly created item, set the value "@Vector1Dimension[0]" in the "Tag" field.

- Select the "Never delete history" checkbox field so that the alarm history tag will never be deleted.

- Under "Numeric" fields, select the field "LoLo". Keep in mind that the LoLo alarm type is the one configured in the previous sub-chapter.

- Repeat the same procedure for tags:
    - Vector1Dimension[1]
    - Vector1Dimension[2]
    - Vector1Dimension[3]
    - Vector1Dimension[4]
- Save the document and name it "Alarm_History1".



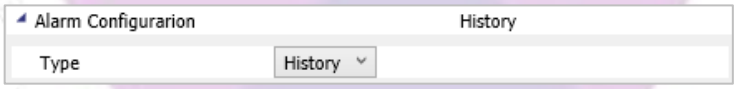
15.3. Creating and Configuring Screen for Alarms

To display the alarms, the user will use the Alarm Object. It can show Online / Historical alarms.

- Online Alarms:

Displays the Alarms that are currently active. In case the Alarm is Normalized AND Acknowledged, the alarm will be removed from the object. To display alarms, it is only needed to enable the alarms in the tag property. In our example, we enabled the LoLo alarms from the Vector1Dimension tags and the Alarm Object will need to be configured with the Vector1Dimension tags and the LoLo alarms enabled.

- Historical Alarm:



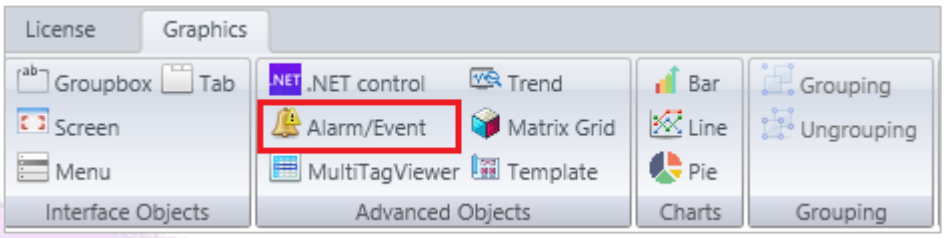
Displays the history of Alarms along the time filter configured in the object. In case the Alarm is Normalized OR Acknowledged, the alarm status will be added to the object. To display alarms, it is important to enable the alarms in the tag property AND create an Alarm History document to store the alarms into a database (proprietary or external database).

Now configure the Screen to display the Online and Historical Alarms:

- Create a new folder, named "Alarms" in "Graphics".
- Create a new document "Graphics1".
- With the document opened, configure its size to:
  - Width: 900
  - Height: 674
- Add a label to indicate the alarm object is History

Type	Text	Top	Left	Width	Height
Label	History:	19	172	110	19

- Go to the Graphics Ribbon, locate the "Advanced Objects" group, and click the "Alarm/Event" object button.



- Click the left mouse button near the point (0.0), upper left-hand corner of the document. This will create the Alarm object with the standard size (200 wide and 200 high).

- Select the created object and configure it as follows in the Property Window:

  - Width: 730; Height: 275

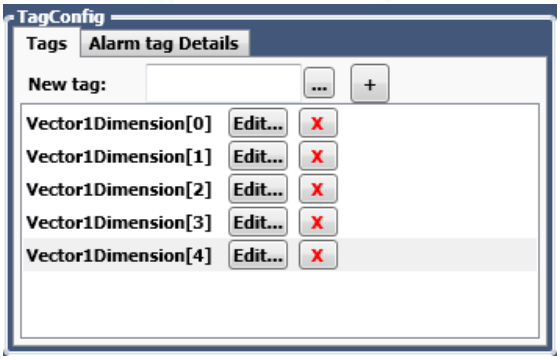
  - Top: 45; Left: 169

- With the Alarm object selected, set the value to "History" in the "Alarm Type" field in the "Alarm Configuration" area of the Property Window.

NOTE: *Alarm history will display all the alarm status changes.*

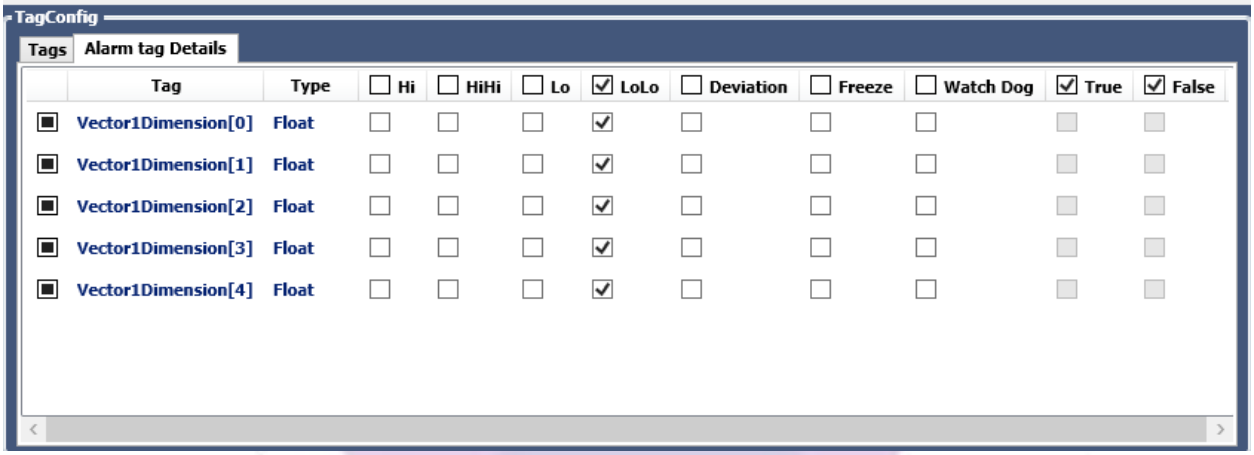
- In the "TagConfig" area, set the Tags that had their Alarms configured. In the "New Tag" field, enter the value "Vector1Dimension[0]", and then click the "+" button.

- Repeat the process for the remaining Tags: "Vector1Dimension[1]", "Vector1Dimension[2]", "Vector1Dimension[3]", "Vector1Dimension[4]" (see image below)



- Double-click the tag "Vector1Dimension[0]" to access the Alarm properties of the tag or simply click in the “Alarm tag Details” tab.

- In the setup screen, select the “LoLo” alarm for all the tags, since it is the alarm configured for all the added tags.



- Configure which columns to display in the object. In the GridConfig area, on the “Grid” tab, click in the “Ack Time” field.

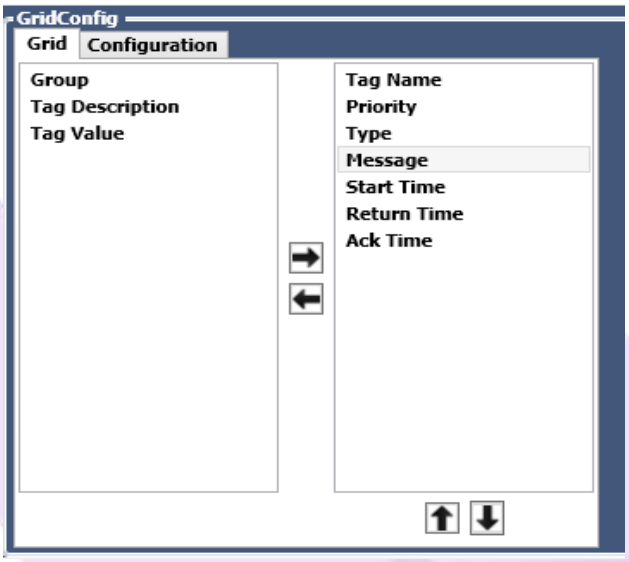
- Move the Ack Time property from the left column to the right column by clicking the arrow between the columns.

- Next, remove the "Tag Description", ”Tag Value” and "Group" from the columns.

- With an action similar to the one described above; the user can organize the columns in the object. To do this, click and select the “Type” field.

- Use the Up arrow to move it up and place it between the properties "Priority" and "Message".

- Do the same for the other properties until it looks as follows:



- By default, all columns are configured “40” wide in size. The user can change the column widths by double-clicking on the “Tag Name” column. This will display the column properties.
- In the "Width" field, enter the value "120".
- Repeat the same process for the following columns:

  Column: “Priority”, Width “50”

  Column: “Type”, Width “50”

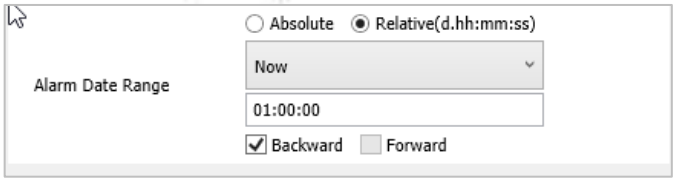
  Column: “Message”, Width “140”

  Column: "Start Time", Width: "120"

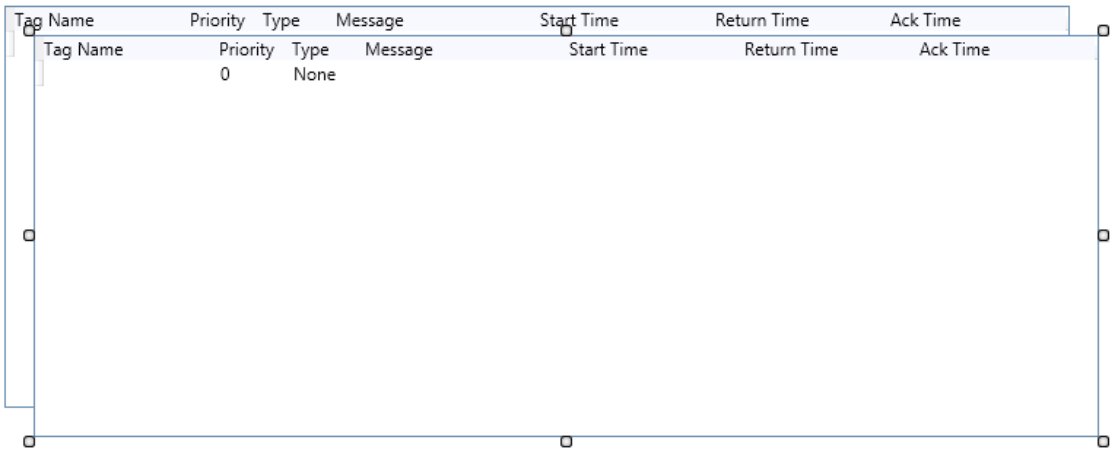
  Column: "Return Time", Width: "120"

  Column: "Ack Time", Width: "120"

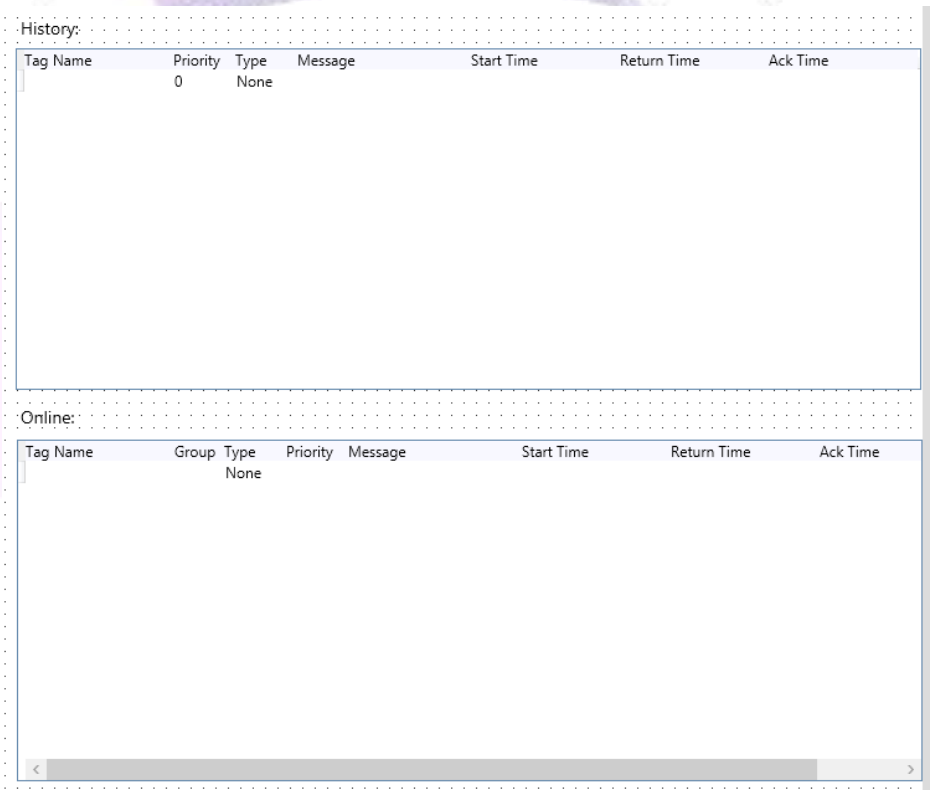
- Keep the Alarm Date Range with default configuration.



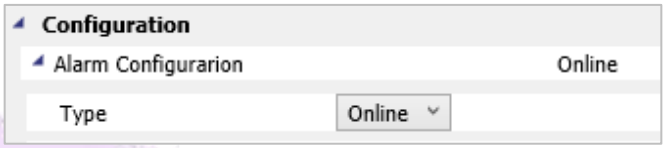
- Now, create a second Alarm of type Online. To create it faster, copy, and paste the History alarm created above.



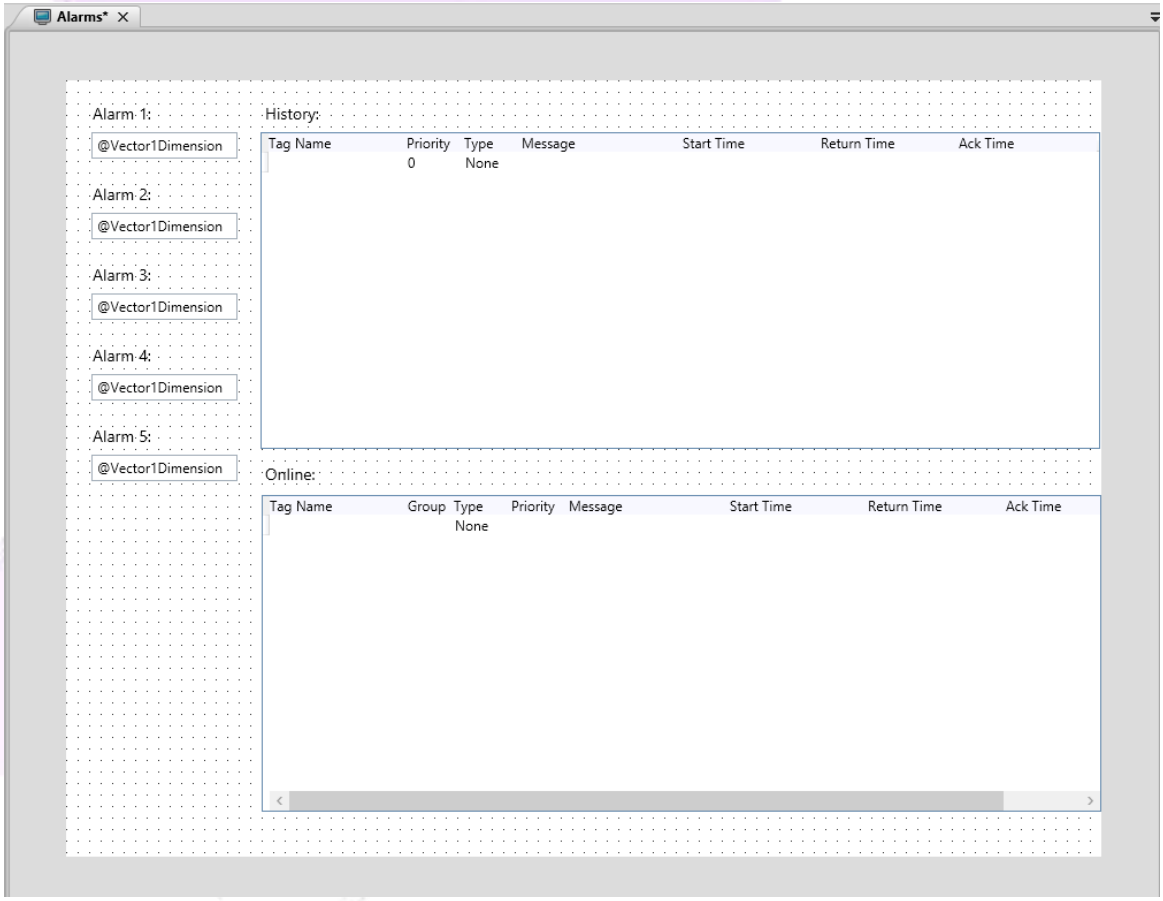
- Change the Location of the second alarm object to Top: 350, Left: 170. Also, include a label for the Online trend as it was done previously for the History alarm.



- Set the Alarm Type to OnLine instead of History



- To keep up with the value changes, add Label and Textbox objects. The user can configure the Label objects to display a description of the alarm and the Text Box objects to display the tag value, as described in the other sections. When finished, the configuration should be as follows:



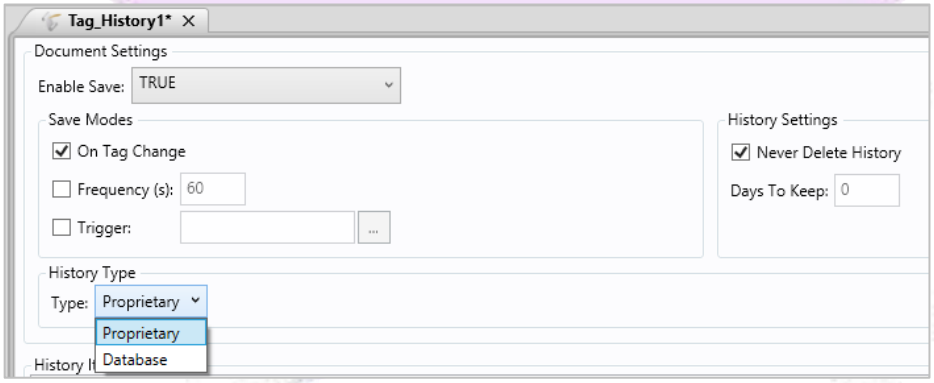
NOTE: *The text box contains the tags from the @Vector1Dimension[0] up to [4]*

- Save the document as “Alarms”.

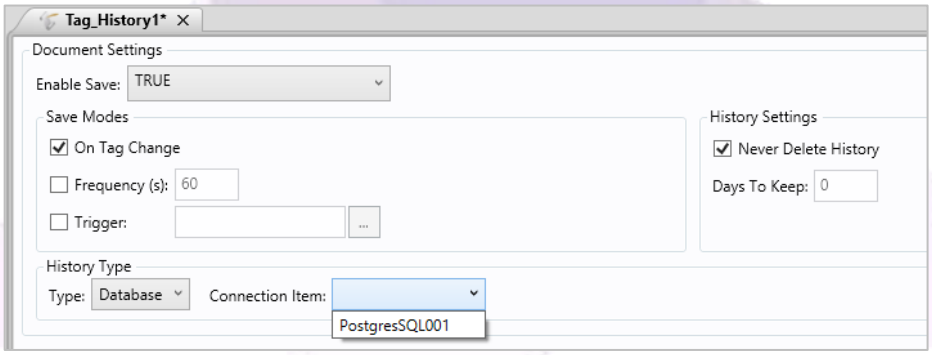
16. Tag History Documents

A Tag History Document generates and stores the historical values of a tag. The user can set the history so that it is saved according to a time interval or each tag value change. It can be saved in the proprietary database or into a relational database previously configured in the top bar ribbon Settings -> Database Connections.

Types of configurations:



Setting the database to store values in the relational database



16.1. Creating and Setting a Historical "On Tag Change"

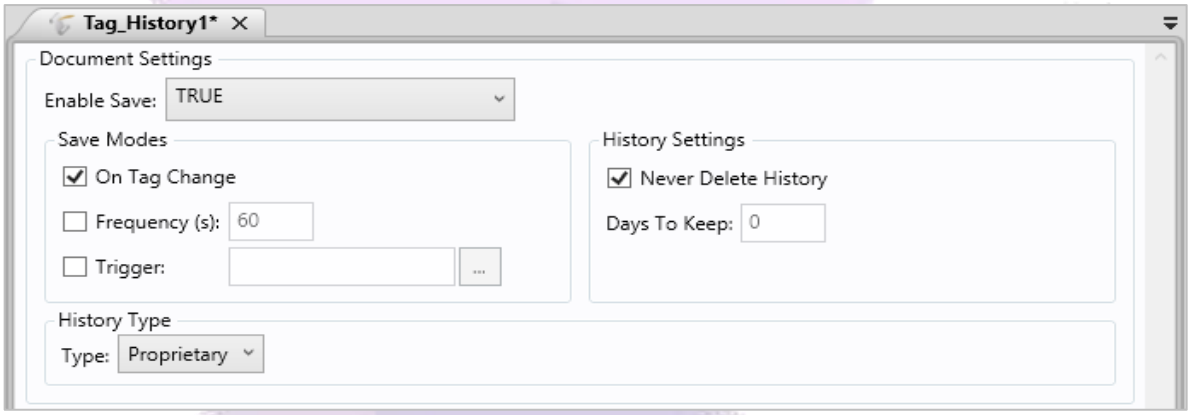
To create a History according to the change of the Tag values, follow the steps below:

- Under Tag History in the Navigation Tree, create a new document.

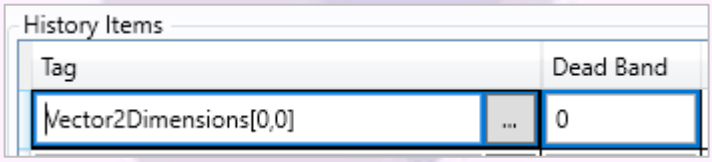
- The configuration on the document settings will be applied to all the tags configured inside the document.

NOTE: *If the user needs to configure different saving intervals, then they will need to create a different document.*

- With the document open, under the save modes inside the Document Settings set the "On Tag Change" checkbox to true.
- Still under the Document Settings but now in the History Settings section, set the "Never Delete History" field to true so the history will never be deleted.



- Now, within the History Items area, click on the empty text field below the Tag title and set the value as "@Vector2Dimensions[0,0]".



- Save the document as "Tag_History1".

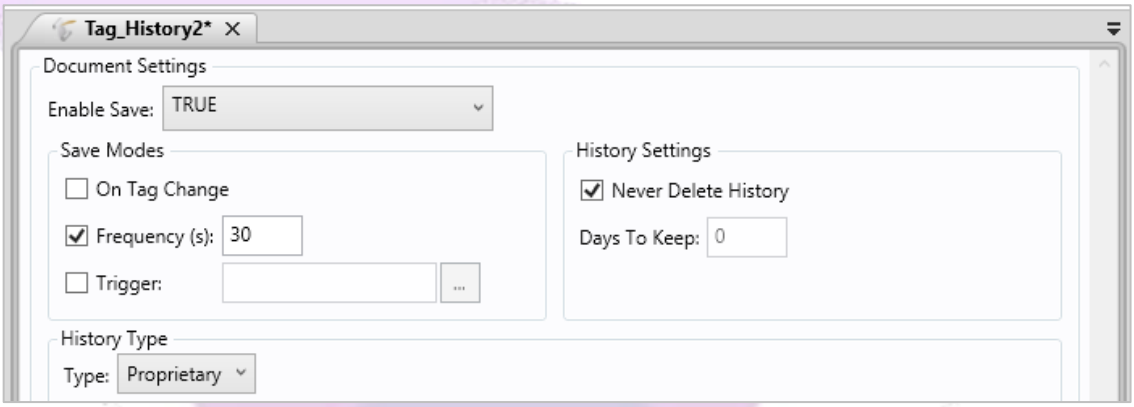
16.2. Creating and Setting a Historical "Time Frequency"

To create a Tag History according to the change of Time Interval tag values, follow the steps below:

- Under Tag History in the Navigation Tree, create a new document. The Tag History created above cannot be used to store values in

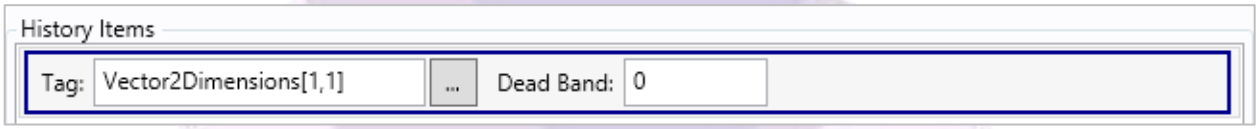
respect to the Time Frequency because it was already configured to save tags On Tag Change.

- With the document open, under the save modes inside the Document Settings set the "Frequency" checkbox to true and set the TextBox next to it with value 30 (seconds).



- In the newly created History item, set the value "@Vector2Dimensions[1,1]" in the "Tag" field.

- Save the document as "Tag_History2".



16.3. Creating and Configuring History Screens

This screen is a very simple screen. It only has two Label objects and two Textbox objects. Follow the steps below:

- Create a folder called "History" in "Graphics".

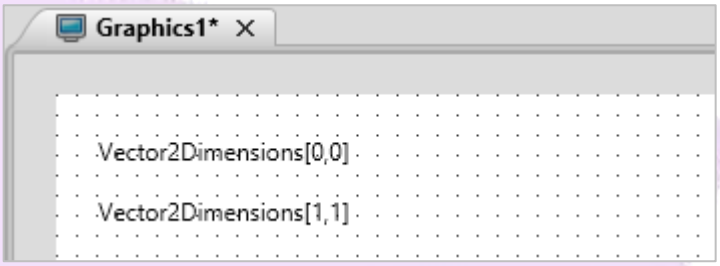
- Create a new document under this folder.

  - Width: 1014

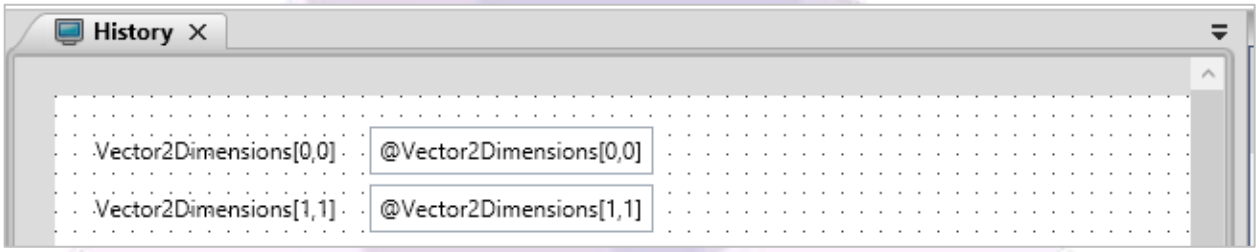
  - Height: 674

- Create two "Label" objects:

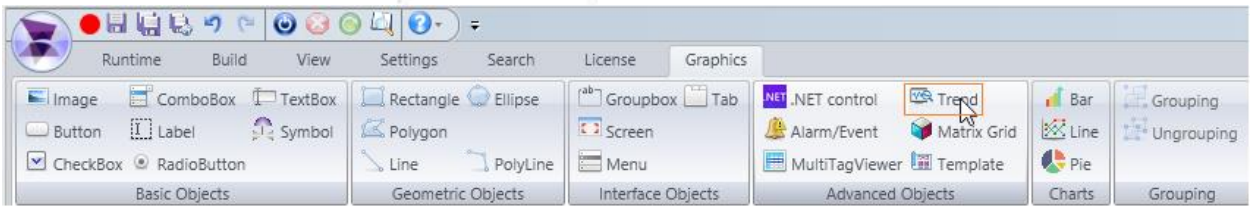
- Label1: Width: 128, Height: 18, Top: 20, Left: 20, Text: "Vector2Dimensions[0,0]"
- Label2: Width: 128, Height: 18, Top: 50, Left: 20, Text: "Vector2Dimensions[1,1]"



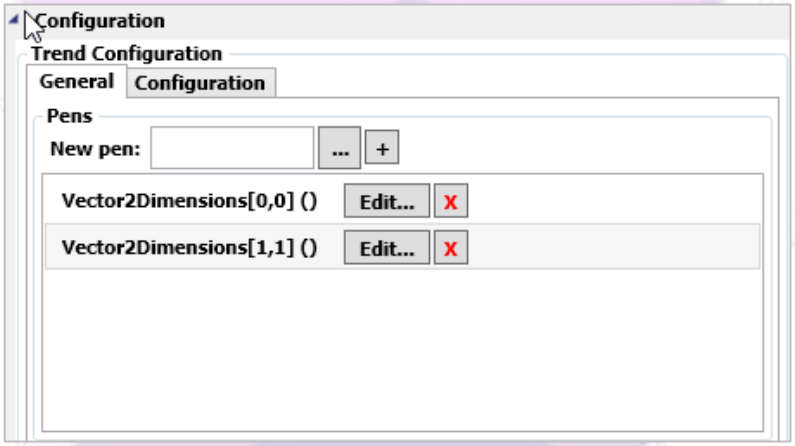
- Create two (2) "TextBox" objects:
  - TextBox1: Width: 148, Height: 25, Top: 16, Left: 164, Text: "@Vector2Dimensions[0,0]"
  - TextBox2: Width: 148, Height: 25, Top: 46, Left: 164, Text: "@Vector2Dimensions[1,1]"
- Save the document as "History".



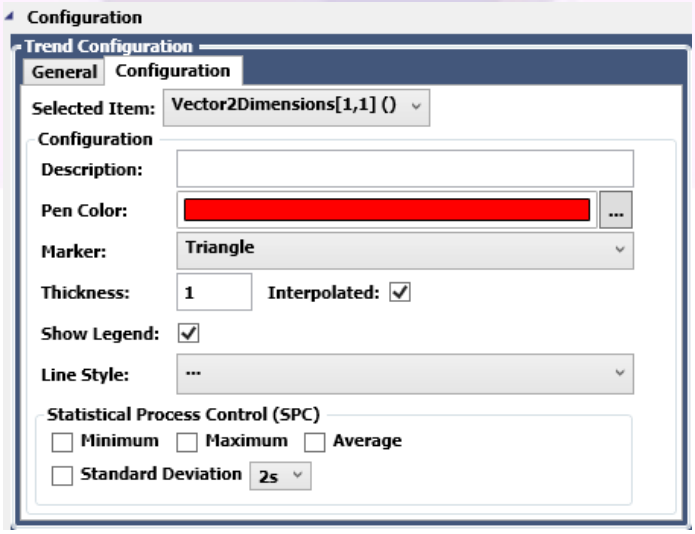
- Select the Trend object in the Graphics ribbon under the Advanced Objects section.



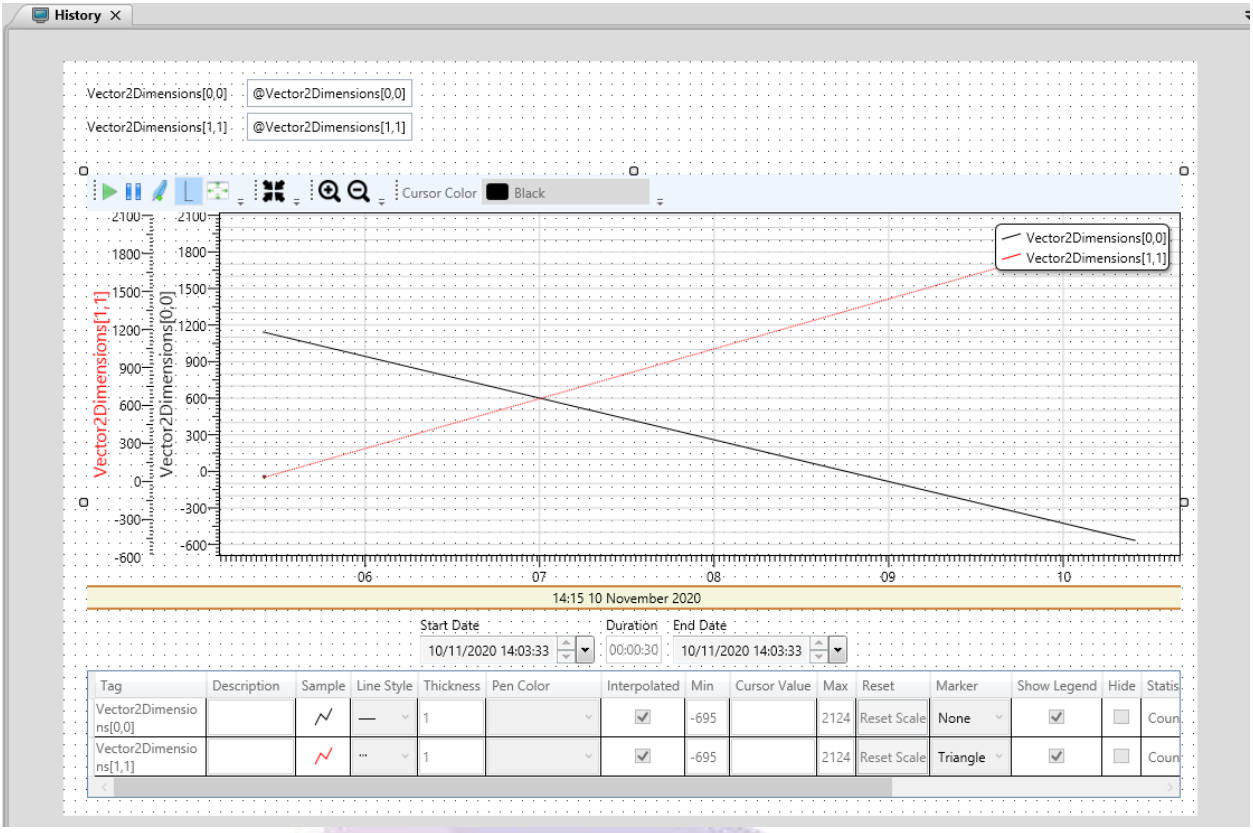
- Then bring it to the screen. The user can click in any area of the screen, or they can draw the trend object by clicking the initial point and dragging the rectangle to the size they want. In this tutorial, the size and location were set as following:
  - Top: 100, Left: 20
  - Width: 978, Height: 586
- Add the historical tags to the Trend object inside the Trend configuration. The user will add the tag name inside the "New Pen" field and click enter.



- Let's change the pen configuration for the @Vector2Dimentions[1,1]



- Please check if the final screen has the following layout:



NOTE: *After running the application, navigate to the path 'C:\GettingStarted\History' indicated and verify that the history files were created (the files should have the following naming pattern: 'Vector2Dimensions [0,0] _20130220.HF'. The user can see the file created contains an extension of type '.HF' and the content is encrypted.*

17. Recipe Documents

The Recipe Document saves the tag values that are set in the document. These values can be retrieved at a given time of application. In this section, the user will learn how to set up and use recipes.

17.1. Creating and Configuring a Recipe Document

Follow these steps to create a document and set it up with a Tag to understand the functionality of a Recipe Document:

- In the Navigation Tree under Recipes, create a new "Recipe" document.
- In the field "Recipe Path", click the "..." button to set the location and file name where the recipe will be created.
- In the new screen that appears, leave the suggested path and set the file name to "TestDocumentRecipe".
- Save the changes made to the document. Once this is done, the window closes and the previously file name is displayed in the field "Recipe Path".
- Leave the "Relative Path" checkbox selected.
- In the "Recipe Save Mode" field, select the "File" option.

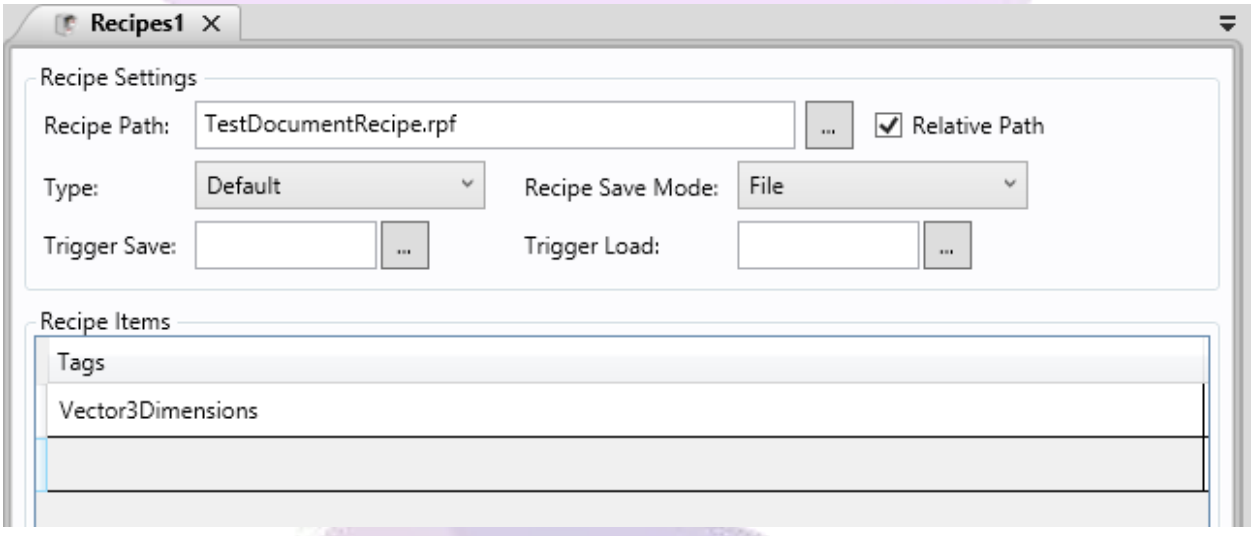
  NOTE: *The XML mode will not cause the file extension to be ".XML". It will only internally structure the file as an XML file, but the generated file will still have the extension ".RPF"*

- In the "Recipe Items" area, set the tag "Vector3Dimensions" by double-clicking on the first line that is empty.
- Click on the "…" button to open the "TagBrowser" window.
- In "TagBrowser", select the tag "Vector3Dimensions" and click "OK". The selected tag is set in the first line of the document. The user can also type the tag's name directly.

It is important to mention that adding the name with an array of 1, 2 or 3 dimensions without specifying the indexes will save all the tags to the recipe. In case the user wants to save only some indexes from the array, they need to be specified (i.e. Vector3Dimensions[0,0,0]).

- Save the Recipe document with Recipes1 name.

Once these settings have been saved, the Recipe document is finished.

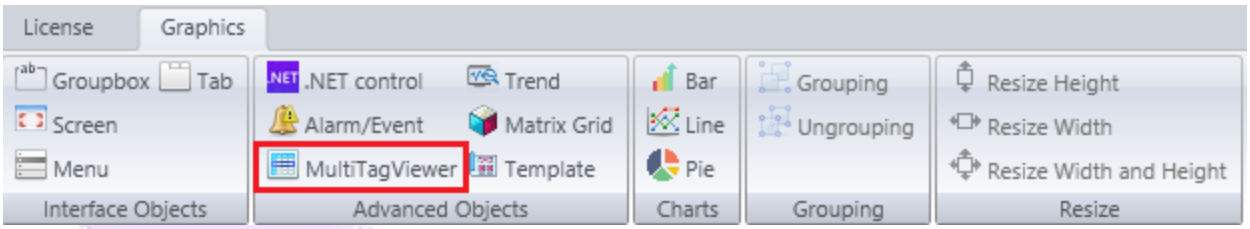


17.2. Creating and Configuring Screens for Recipes

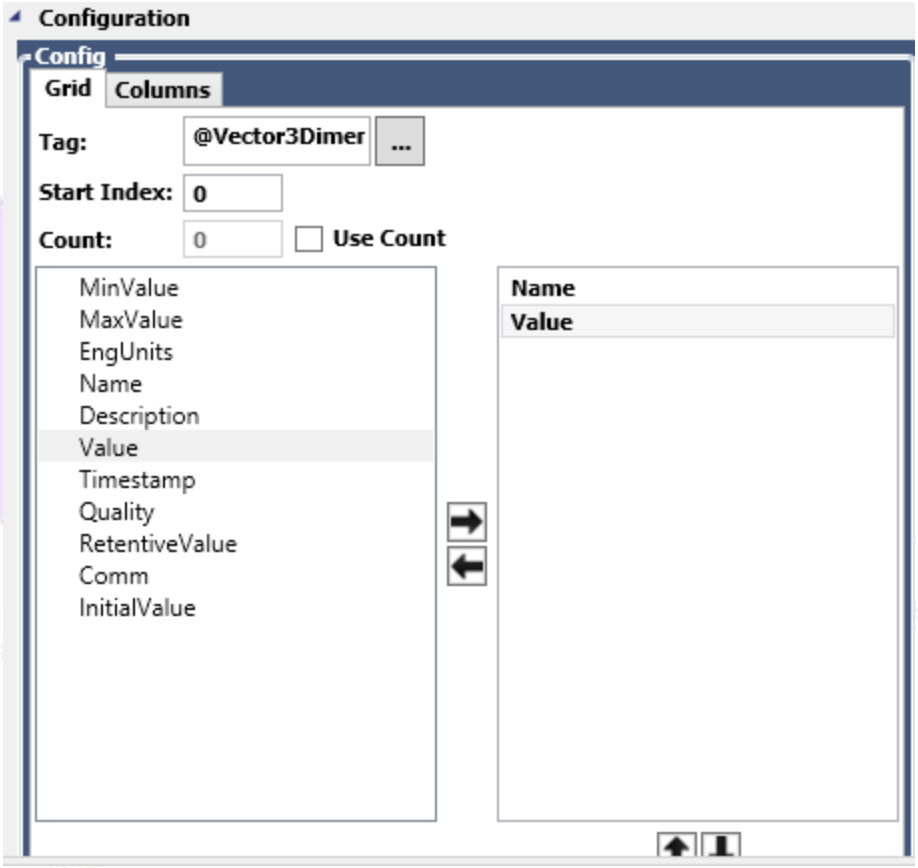
With the "Recipe" document set up, the user can create a screen and understand the various ways it is used. There are several ways to save the values in "Recipe" and to retrieve the saved values. This section describes all of them.

In the new screen, we will create multiple buttons with different scripts to 'Save' and 'Load' recipes. To do this, follow these steps:

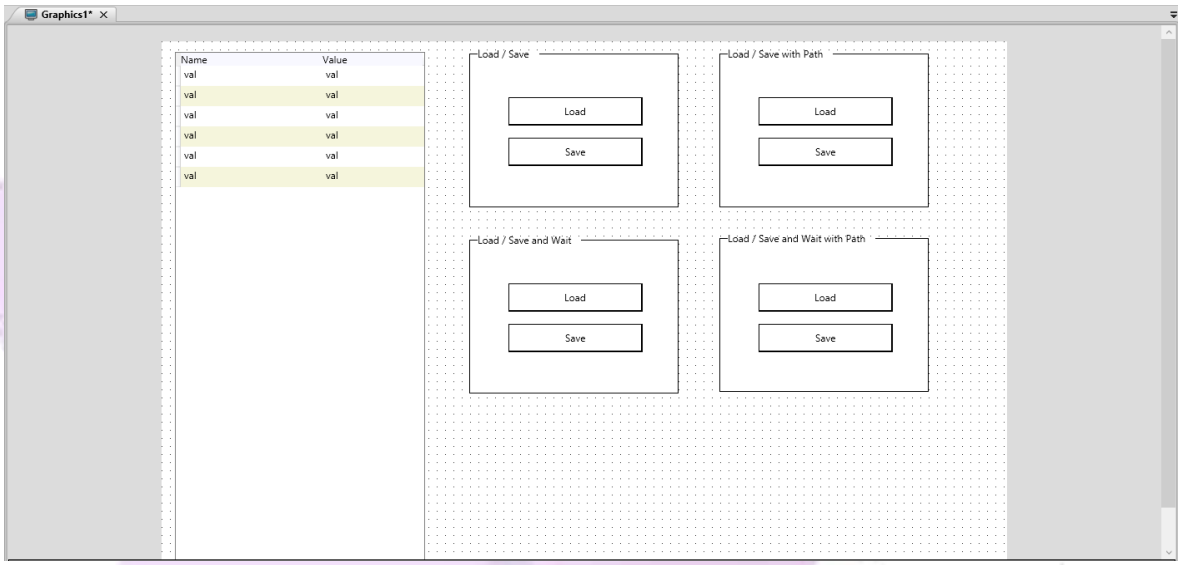
- Create a new folder in "Graphics" called "Recipes".
- Create a new document "Graphics".
- After the "Graphics" document is created, select the "MultiTagViewer" object under the "Advanced Objects" group in the Graphics Ribbon.



- Click near the point (0,0), upper left-hand corner of the document, to create the object with the standard size.
- Select the created object, and set the values as follows in the Property Window:
    - Width: 300, Height: 650
    - Top: 20, Left: 20
- In the Property Window "Configuration -> Grid", enter the tag "@Vector3Dimensions" in the "Tag" and press the "Enter" key to confirm the setting. There will be several properties displayed in the left column.
- Select the "Name" property and click the right arrow to move it into the right column.
- Repeat with the property "Value".



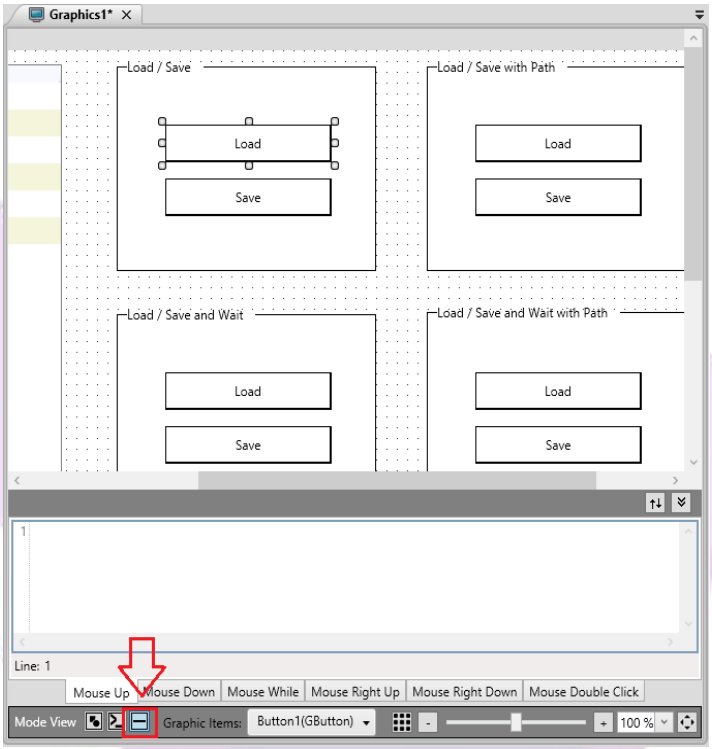
- Double-click on the "Name" property to view the settings.
- In the "Width" field, enter the value "170".
- While still on the Columns tab, select "Value" from the dropdown menu at the top.
- Set the "Width" field with the value "120" and select the "Input Enable" option so the user will be able to change the tag value through the object.
- After configuring MultiTagViewer, create a GroupBox and two Buttons: one for Load and one for Save. The Buttons perform the actions of saving and retrieving the values of a Recipe.
- The final screen will look like the following image:



- To establish these settings, start by creating a new GroupBox object. Set the "Text" property of the GroupBox with the value "Load / Save".

- Create a new object "Button" in the workspace.

- Select the object "Button" that was just created and click the “Split” icon in the "Mode View" section. The “Split” allows the user to visualize both scripts and design window.



Mode View has three different view types:

- Design ( ): Only displays the graphics
- Script ( ): Only displays the "ScriptDock" area where the scripts are configured
- Split ( ): Display both the graphics and the "ScriptDock"

- In the "Mouse Up" tab configure the following script:

  SVRecipe.Load("Recipes1");

  - This script performs data recovery of configured tags in the document "Recipe" when the left mouse button is released.

- Enter the word "Load" in the "Text" field in the Property Window for button 1.

- Create a new object "Button".

- On the "Mouse Up" from the "Script Dock" configure the following script:

  ```
  SVRecipe.Save("Recipes1");
  ```

- This script causes the values of the tags configured in the "Recipe" document to be saved in an ".RPF" file.

- Enter the word "Save" in the "Text" field in the Property Window for button 2.

- Repeat the same process for the following settings:

  - GroupBox: Text: "Load / Save with Path"

  - Button Load: Text: "Load" Mouse Up:

    ```
    SVRecipe.Load("Recipes1","C:\\GettingStarted\\Recipe\
    \TestRecipePath.rpf");
    ```

  - Button Save: Text: "Save," Mouse Up:

    ```
    SVRecipe.Save("Recipes1","C:\\GettingStarted\\Recipe\
    \TestRecipePath.rpf");
    ```

  - GroupBox: Text: "Load / Save And Wait"

  - Button Load: Text: "Load" Mouse Up:

    ```
    SVRecipe.LoadAndWait("Recipes1");
    ```

  - Button Save: Text: "Save," Mouse Up:

    ```
    SVRecipe.SaveAndWait("Recipes1");
    ```

  - GroupBox: Text: “Load / Save And Wait with Path"

  - Button Load: Text: "Load" Mouse Up:

    ```
    SVRecipe.LoadAndWait("Recipes1","C:\\GettingStarted\\
    Recipe\\TestRecipePath.rpf");
    ```

- Button Save: Text: "Save," Mouse Up:

  SVRecipe.SaveAndWait("Recipes1","C:\\GettingStarted\\Recipe\\TestRecipePath.rpf").

- Save the document as "Recipe".

17.3. Differences Between Loan/Save GroupBoxes in Recipe Document

Buttons belonging to the GroupBox “Load / Save” load and save the tag values according to the configuration of the Recipe Document.

The buttons belonging to the GroupBox “Load / Save with Path” load and save the tag values set in the Recipe Document; however, it disregards the path set in it. The path that the “.RPF” document generates is the path passed through as a function parameter, in this case: "C: \GettingStarted\Recipe\TestRecipePath.rpf".

Buttons belonging to the GroupBox “Load / Save and Wait” load and save the tag values according to the configuration of the Recipe Document. The difference between conventional “Load” and “Save” is that “LoadAndWait” and “SaveAndWait” perform their functions, and while they are running, block the execution of other parallel actions so that they do not desynchronize information / data.

The buttons belonging to the GroupBox “Load / Save And Wait with Path” load and save the values of the tag set in the Recipe Document; however, it disregards the set path set. The path in the “.RPF” document is generated is the path passed as a function parameter, in this case: "C: \GettingStarted\Recipe\TestRecipePathComWait.rpf ". In the same way as described above, while they are being executed, they block the execution of other parallel actions so not to desynchronize information / data.

17.4. Expected Execution Result for Recipes

This section explains what to expect from the screen when running the “Recipes” application. Clicking the Save button on the screen causes

the tag values displayed in the "MultiTagViewer" to be saved to a ".RPF" file. Despite the file extension, it will be a text file and can be opened by any text editor.

After the file is created, click the Save button, navigate to the folder where it was created ("C:\GettingStarted\Recipe") and look for the file with extension ".RPF". Open it in a text editor to see if the values saved in the file match the tag values at the time they were saved.

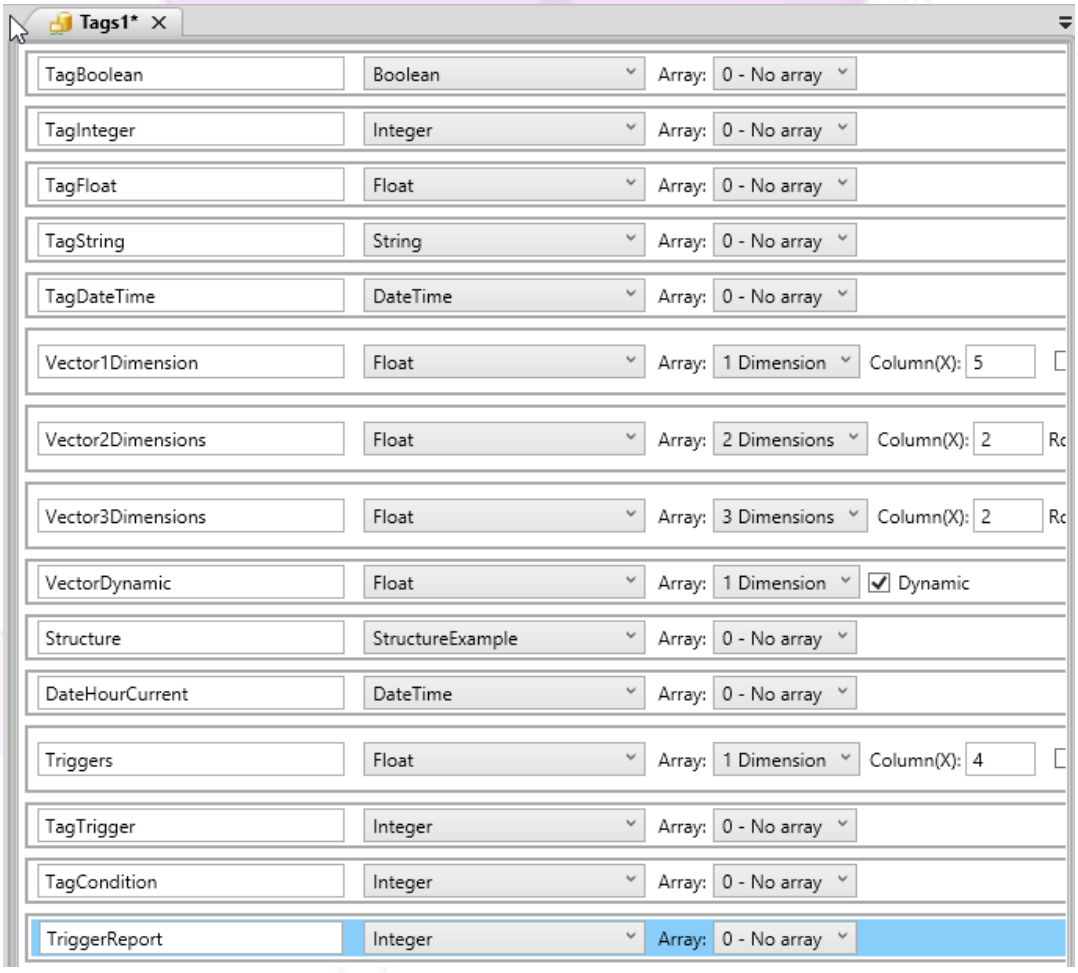
The Load button causes the values contained in the ".RPF" file to be loaded into the tag, which is being displayed in the "MultiTagViewer".

18. Report Documents

The Report Document will generate a report. A report can be generated either to a file or sent directly to print.

18.1. Creating and Configuring a Report Document

This section will explain how to set up a simple report that displays the contents of some project Tags. For this example, the user needs to create a new Tag called "TriggerReport", which is an "Integer" Type and has an array size of “0 - No array” under “Tags1” document.



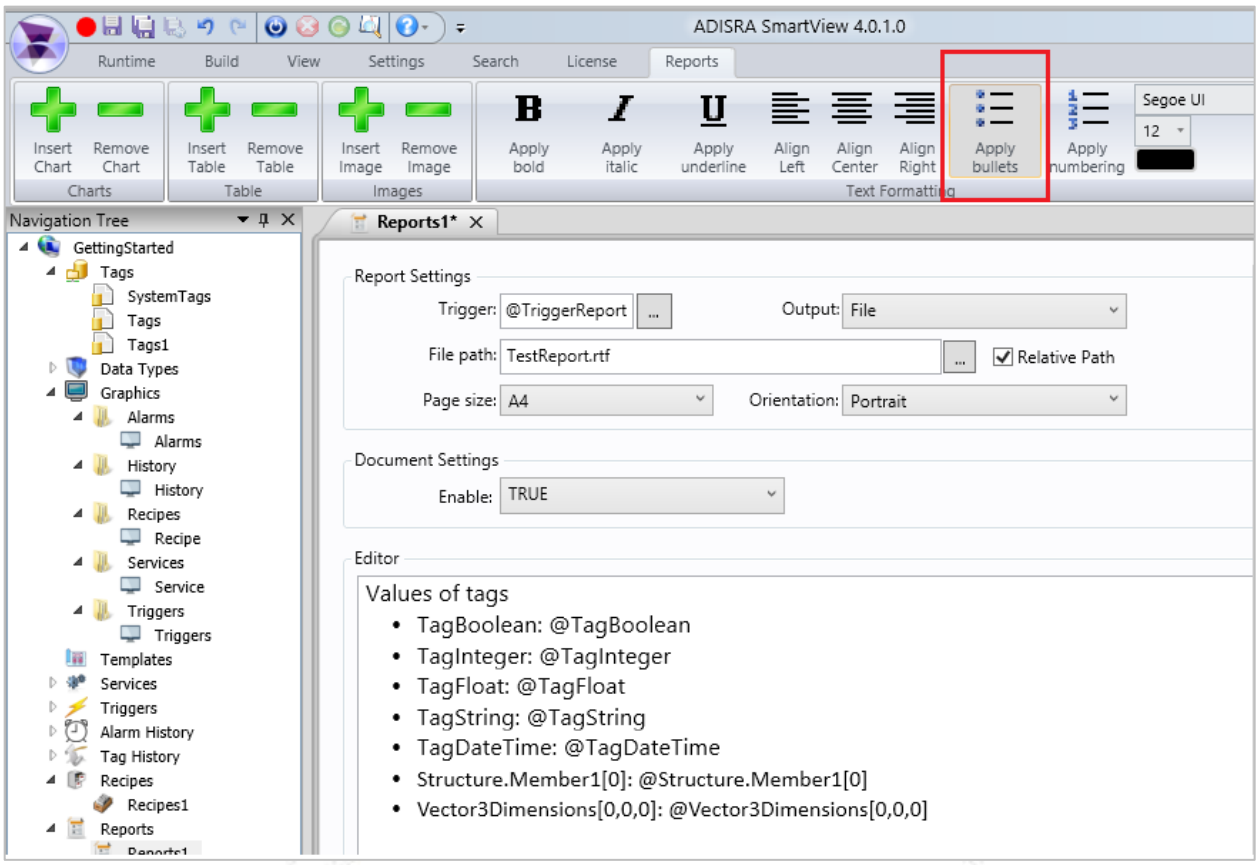
Once the new Tag is created, follow these steps:

- Create a new document "Report" from the Navigation Tree.
- In the Trigger field of Report Settings, set the value "@TriggerReport". The tag configured in this field, having had its value changed causes the document settings to be executed, thus generating the configured report.
- In the Output, select the File option to generate a file as a report. Selecting the File option displays new fields.
- In the File path, click the "..." button to open the window in order to set the path and file name that will be generated.
- In the new window, set the desired path. For this example, keep the default path ("C:\GettingStarted\Reports") and set the report name as "TestReport.rtf". Click save and the "TestReport.rtf" will be displayed in the file path field. Keep the "Relative Path" checkbox selected.
- In Page size, select the A4 or Letter option.
- In Orientation, select the Portrait option, i.e., portrait = vertical.
- In the Editor, input the following:

Values of tags

- TagBoolean: @TagBoolean
- TagInteger: @TagInteger
- TagFloat: @TagFloat
- TagString: @TagString
- TagDateTime: @TagDateTime
- Structure.Member1[0]: @Structure.Member1[0]
- Vector3Dimensions[0,0,0]: @Vector3Dimensions[0,0,0]

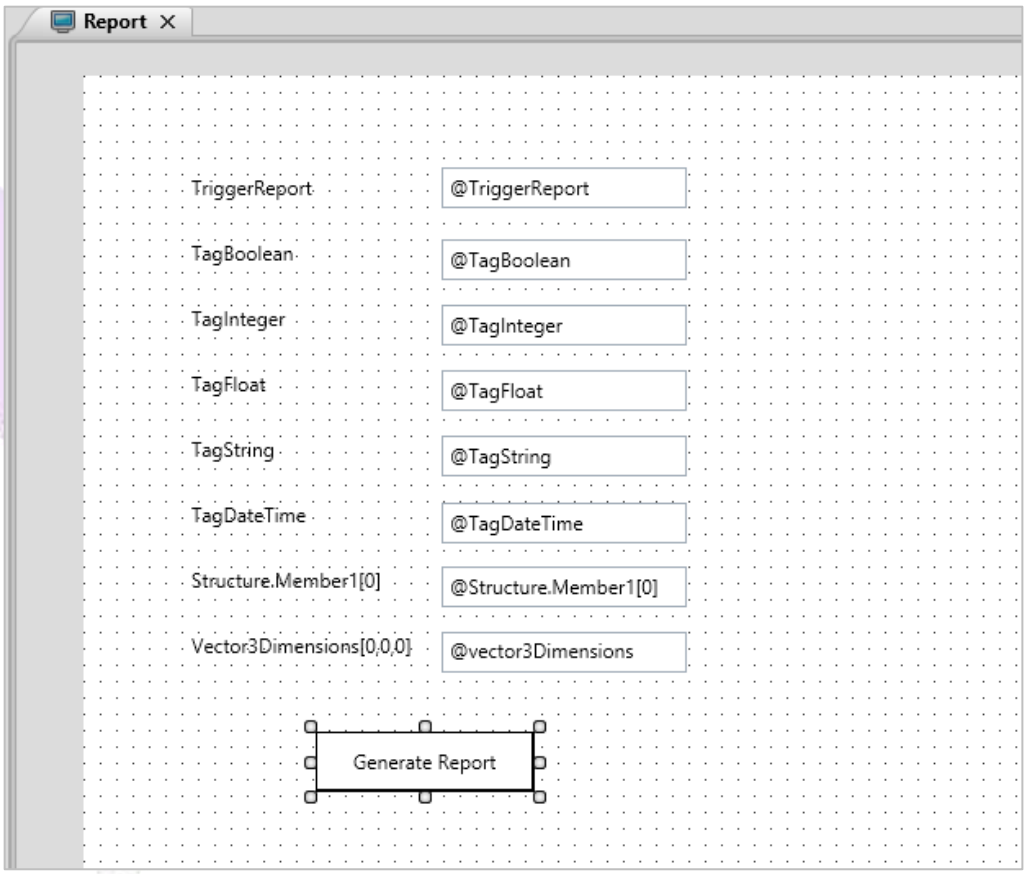
Save the document as "Reports1".



18.2. Creating and Configuring Screens for Reports

The Reports screen is simple and contains only Label and TextBox objects. The TextBox objects display the names and values of the tags that are set in the Recipe document. To configure the screen, follow these steps:

- Create a new folder in Graphics called "Report".
- Create a new Graphics document.
- The final screen will look like the following image. Eight (8) labels, eight (8) text boxes and a button will be needed.



- Create eight (8) Label objects with the following settings:
  - Label1: Text: "TriggerReport" Width: "140" Height: "16" Top: “60” Left: “65”
  - Label2: Text: "TagBoolean" Width: "140" Height: "16" Top: “90” Left: “65”
  - Label3: Text: "TagInteger" Width: "140" Height: "16" Top: “120” Left: “65”
  - Label4: Text: "TagFloat" Width: "140" Height: "16" Top: “150” Left: “65”
  - Label5: Text: "TagString" Width: "140" Height: "16" Top: “180” Left: “65”
  - Label6: Text: "TagDateTime" Width: "140" Height: "16" Top: “210” Left: “65”
  - Label7: Text "Structure.Member1[0]" Width: "140" Height "16" Top: “240” Left: “65”

- Label8: Text "Vector3Dimensions[0,0,0]" Width: "140" Height "16" Top: “270” Left: “65”

- Create eight (8) TextBox objects with the following settings:
  - Text-Box1: Text "@TriggerReport" Width: "150" Height "25" Top: “56” Left: “219”
  - Text-Box2: Text "@TagBoolean" Width: "150" Height "25" Top: “86” Left: “219”
  - Text-Box 3: Text: "@TagInteger" Width: "150" Height: "25" Top: “116” Left: “219”
  - Text-Box 4: Text: "@TagFloat" Width: "150" Height: "25" Top: “146” Left: “219”
  - Text-Box 5: Text: "@TagString" Width: "150" Height: "25" Top: “176” Left: “219”
  - Text-Box 6: Text: "@TagDateTime" Width: "150" Height: "25" Top: “206” Left: “219”
  - Text-box 7: Text "@Structure.Member1[0]" Width: "150", Height: "25" Top: “236” Left: “219”
  - Text-Box 8: Text "@Vector3Dimensions[0,0,0]" Width: "150", Height: "25" Top: “266” Left: “219”

- Create a Button to save the report and add the following script:

```
if (@TriggerReport == 0)
   @TriggerReport = 1;
else
   @TriggerReport = 0;

System.Windows.Forms.MessageBox.Show("Report generated successfully.",
                "Information",
                MessageBoxButtons.OK,
                System.Windows.Forms.MessageBoxIcon.Information);
```

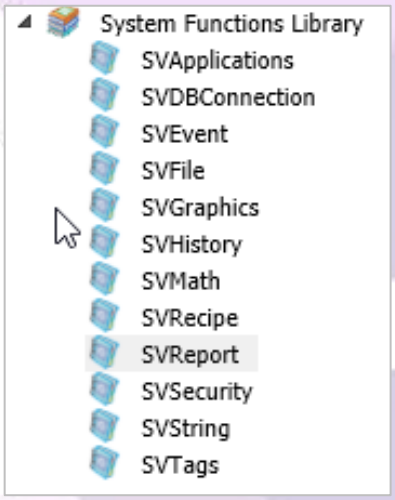
```
if (@TriggerReport == 0)
    @TriggerReport = 1;
else
    @TriggerReport = 0;

System.Windows.Forms.MessageBox.Show("Report generated successfully.",
                                     "Information",
                                     MessageBoxButtons.OK,
                                     System.Windows.Forms.MessageBoxIcon.Information);
```

Line: 1

Mouse Up | Mouse Down | Mouse While | Mouse Right Up | Mouse Right Down | Mouse Double Click

- Save the Graphics document as "Report".
- Alternatively, the user can create a button to generate the report using a System Function Library called SVReport:

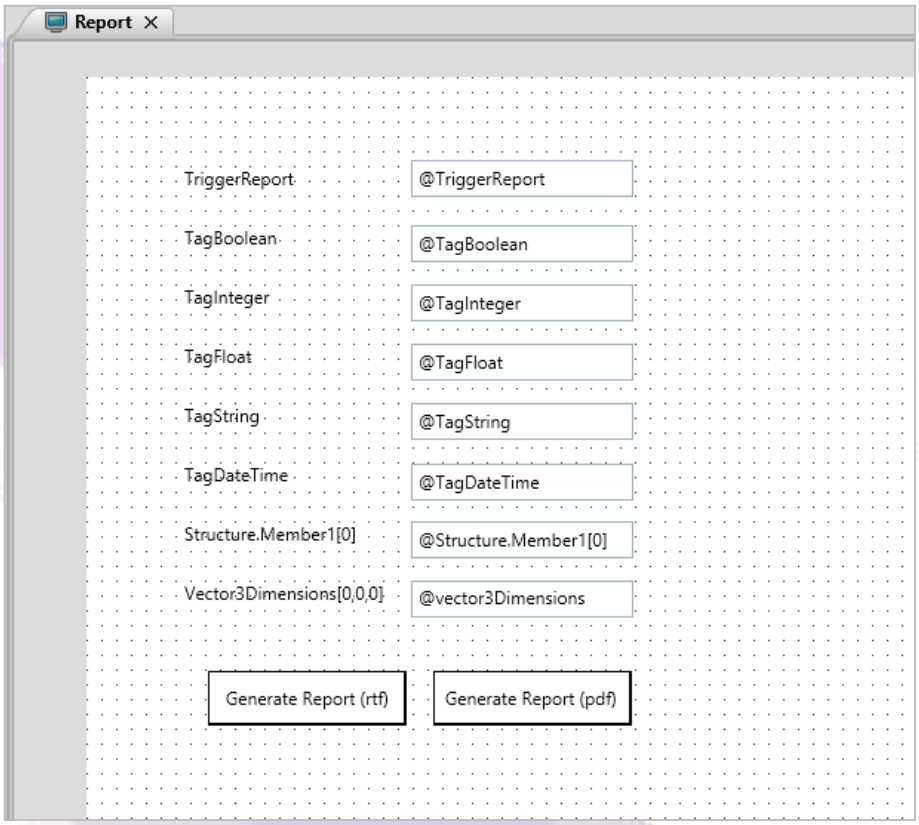


- Inside this library, the user can call the SaveFile function from a button where the user will

```
void SaveFile(string reportFile, string path)
#region
/// Summary:
///     Function to save a report document in a specific path
///
/// Parameters:
///  string reportFile
///     A String parameter containing the name of the report to be saved.
///
///  string path
///     A string parameter with the path, including a valid extension
///

#endregion
```

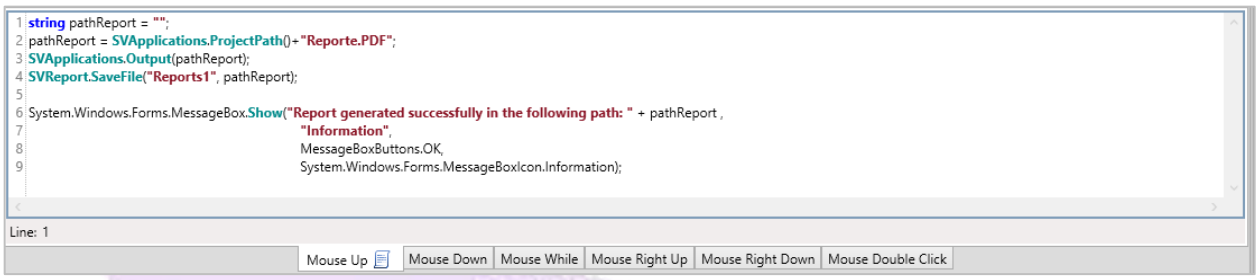
- Use this function to generate a pdf file instead of the rtf.
- Add a second button to the Report Screen.



- Select the second button “Generate Report (pdf)” and open the script.
- Add the following code

```
string pathReport = "";
pathReport = SVApplications.ProjectPath()+"Reporte.PDF";
SVApplications.Output(pathReport);
SVReport.SaveFile("Reports1", pathReport);

System.Windows.Forms.MessageBox.Show("Report generated
successfully in the following path: " + pathReport ,
                "Information",
                MessageBoxButtons.OK,
                System.Windows.Forms.MessageBoxIcon.Information);
```



- This button will generate the report as a pdf file in the Project's folder.

18.3. Expected Execution Result for Reports

When running the application, the user can generate the report using one of the buttons or simply changing the value of the first textbox where the @TriggerReport tag is configured. This tag is also set in the report Trigger field, so whenever its value is modified, the report will be generated to the path "C:\GettingStarted\Reports\TestReport.rtf". The second button we added will generate the pdf file to the path "C:\GettingStarted\TestReport.pdf"

19. Drivers

The driver's node contains specific driver communication documents over different protocols, but it also contains OPC communication documents and a database document. Our example will only include OPC UA communication and driver communication documents. It is important to mention that the database communication behaves as a driver communication, it allows the user to link a tag with a database table for a specific unique key.

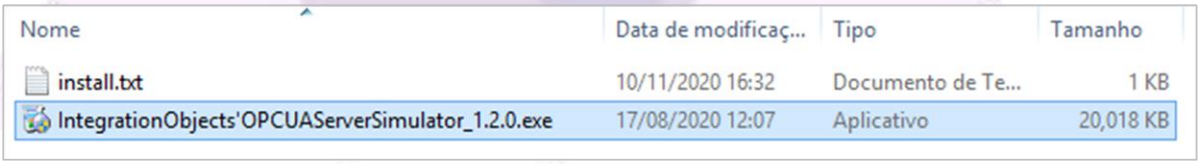
19.1. OPC UA client Documents

The communication can be configured to an OPC UA Server in the OPC UA Client Document. In this section, the user will be instructed to download a third party OPC UA Server, but they can use their own if they have one available.

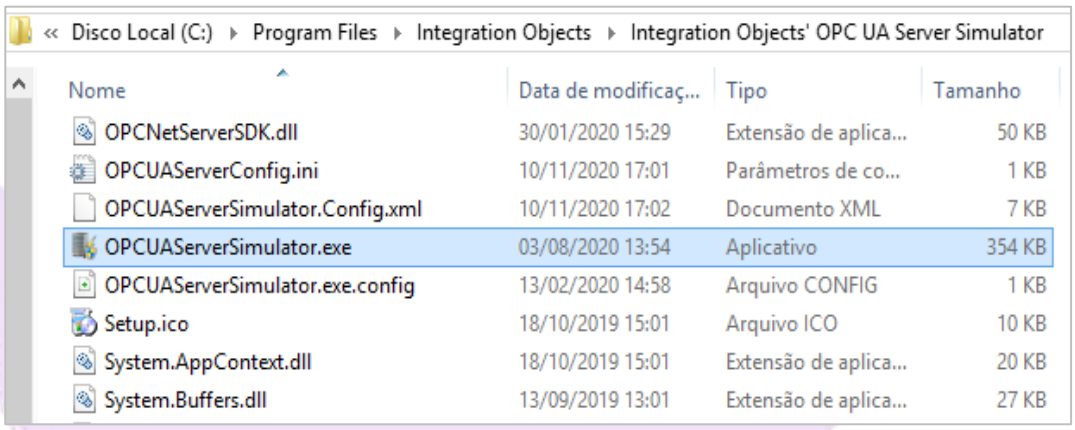
In this chapter, we will download and install an OPC UA Server simulator and then configure the communication in the ADISRA SmartView. The result will be a screen to monitor Tags connected to the third party OPC UA Server.

19.1.1. Download and install a Third Party OPC UA Server

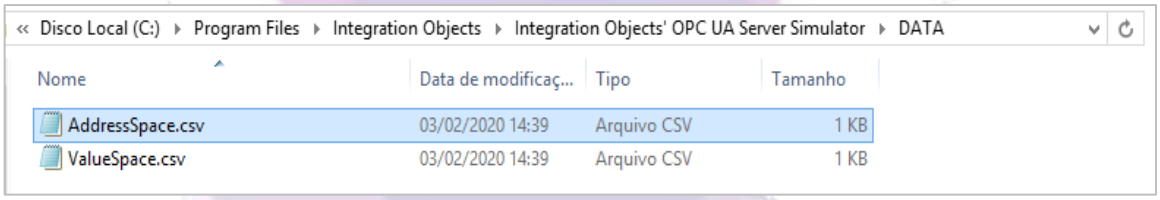
- Download the file from the URL https://adisra.com/wp-content/uploads/2020/03/ADISRA-SmartView-GettingStarted-IntegrationObjects.zip
- Unzip and install the OPC UA Server by executing the file below



- After installing it, please run the simulator. It can be done by executing the shortcut or directly in the installation folder and running OPCUASimulator.exe

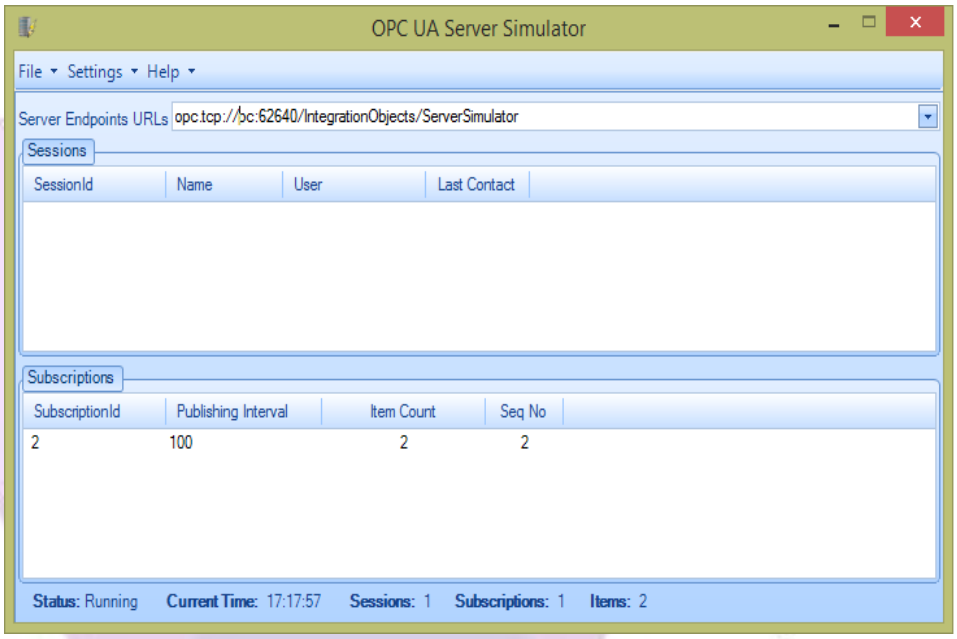


- The simulator allows the user to create new addresses where they can have fixed values or simulated values. For this example, we will use the already created addresses, but if the user would like to configure new addresses later, please go to the DATA folder under the Integration Objects OPC UA Server Simulator installation folder and look at the csv files.



- AddressSpace.csv allows the user to create new addresses. The last column indicates whether the address will be simulated or not.
- ValueSpace.csv allows the user to create values for the simulation. The first row indicates the address, and the rows below will give the values.

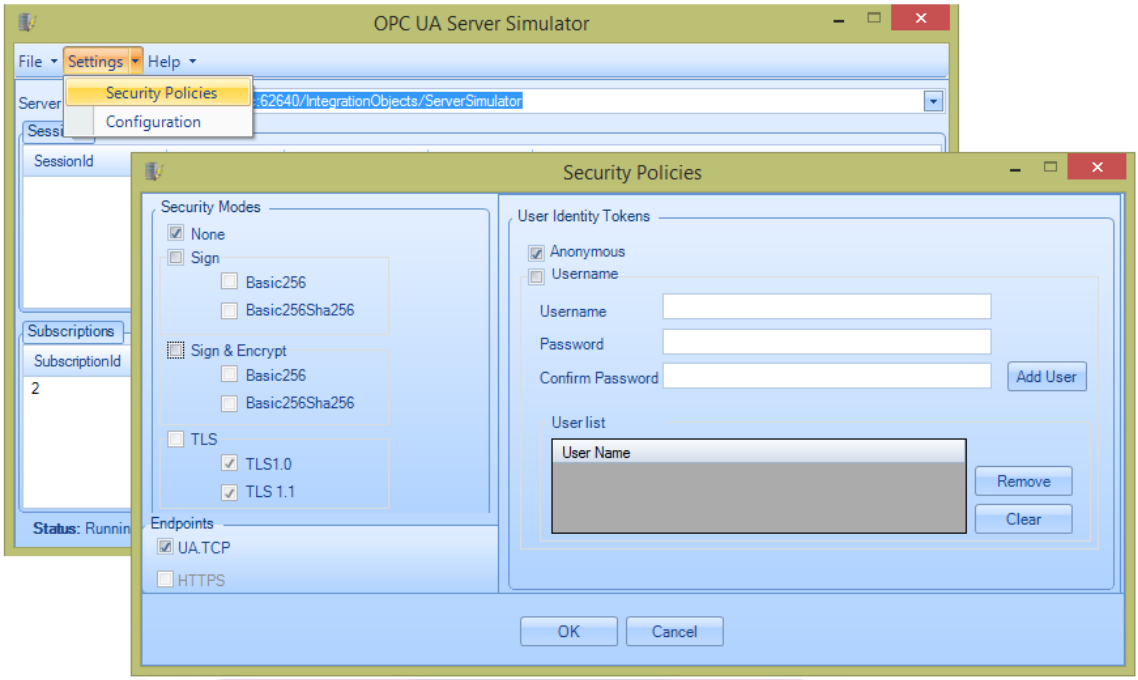
- As soon as the simulator is executed, the user will see the following dialog:



- The most important information needed is the port number and the server name, which will be used to configure the OPC Client document on the ADISRA SmartView.

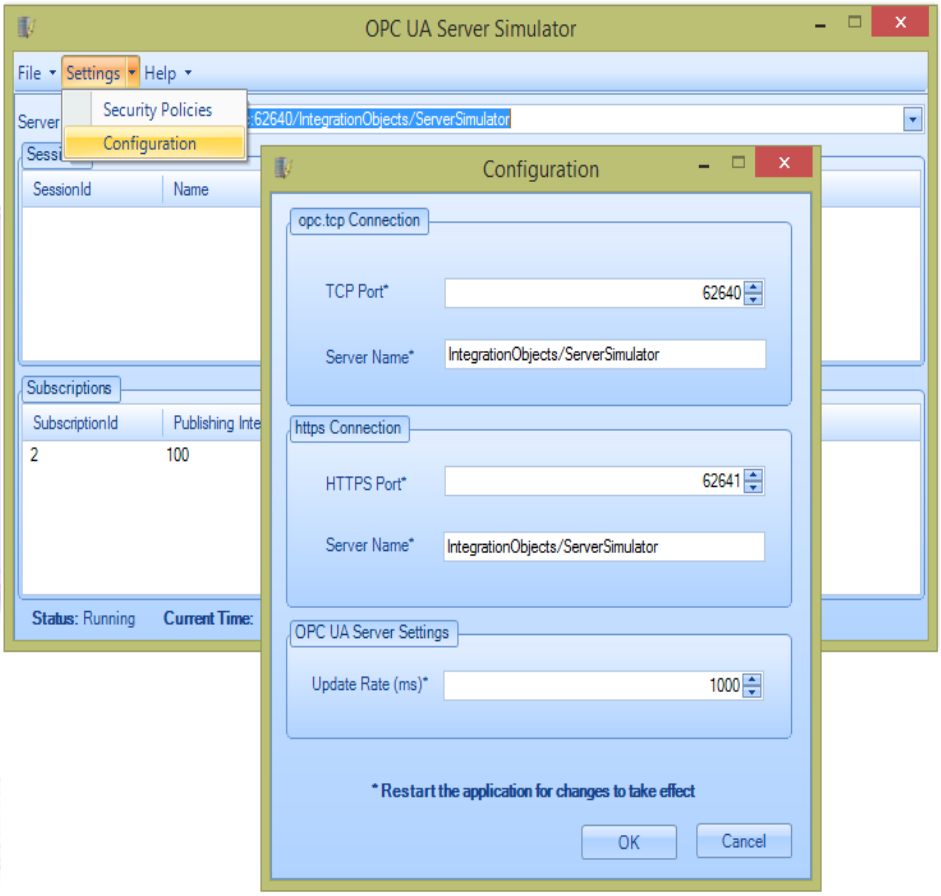
Port Number	*62640*
Server Name	*IntegrationObjects/ServerSimulator*

- Next step set the OPC UA Server to accept the connection from an anonymous user without any further encryption.
- Open the Settings option in the upper menu and click the Security Policies option. A new dialog will open.
  - Make sure the only Security Mode selected is “None”
  - On the right side, make sure only the Anonymous option is selected



NOTE: *This is the required configuration.*

- The user can double check the Server Name and Port Number by opening the Configuration settings on the top bar menu.

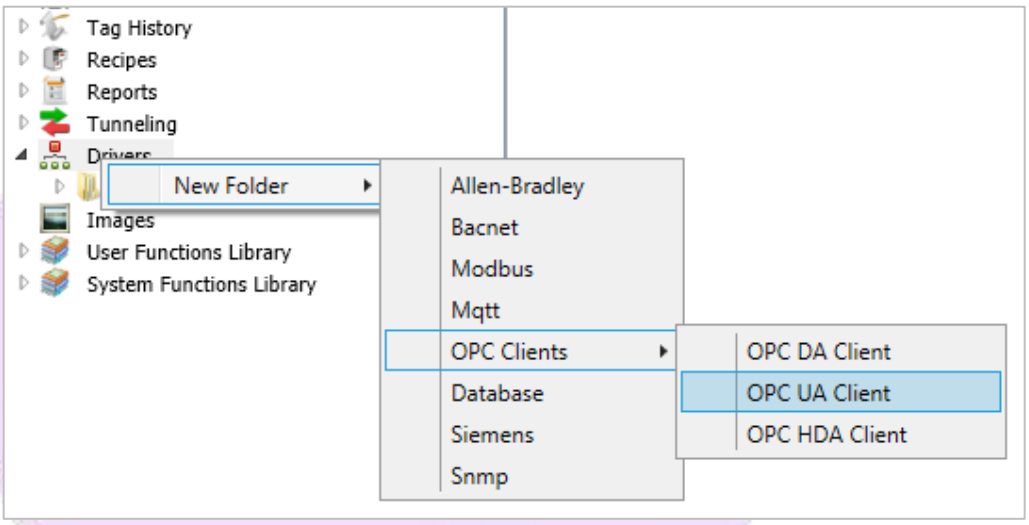


- Since we will be connecting through TCP, the port number used will be 62640 and the Server Name is IntegrationObjects/ServerSimulator. If the user needs to change those settings, it can be done now.
- We will create a OPC UA Client document in the next sub-chapter.

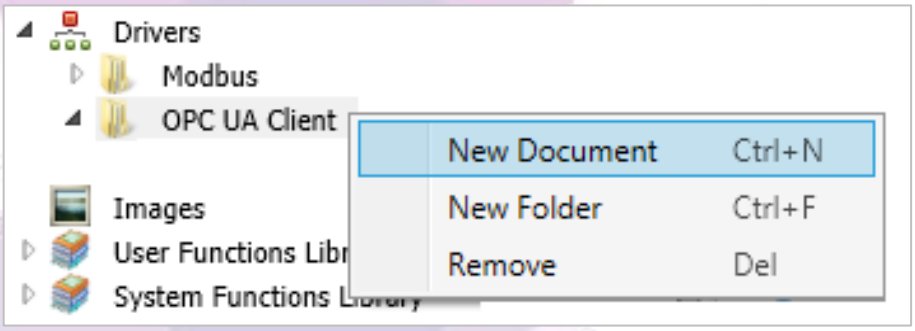
19.1.2. Creating OPC UA Client Documents

To configure the document, follow these steps:

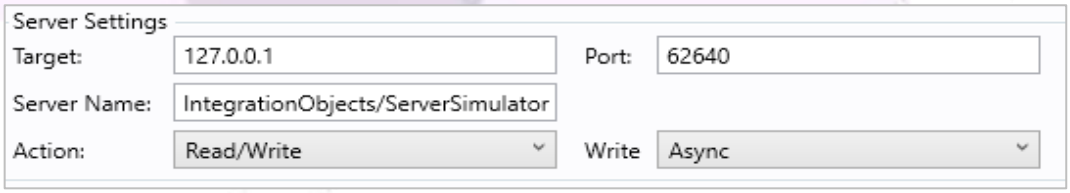
- Create a new "OPC UA Client" folder in the Navigation Tree under Drivers -> New Folder -> OPC Clients -> OPC UA Client.



- A new folder with be created named OPC UA Client. Now please create a document under that folder by right clicking over it and selecting "New Document".



- Open the newly created document to start configuring the connection to the Server.
- In the Server Settings, set the Target (IP Address) where the OPC UA Server is running, the Port Number and the Server Name.



NOTE: *As a reminder, the Server Name and Port Number were extracted from the OPC UA Server.*

- In the Action field, configure the direction flow of the data.

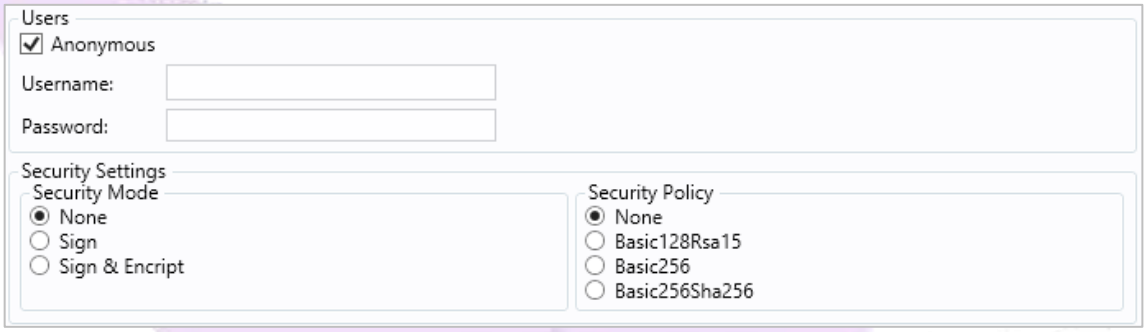
- Read: The OPC UA Client can only read the data from the OPC UA Server.
- Read/Write: The data flows in both directions, from the Server to the Client and from the Client to the Server.
- Write: The data flows only from OPC UA Client to OPC UA Server.

Let's keep the data flow Read/Write.

- In the Write field, choose whether the data is written synchronous or asynchronous.
- In the Document Status, set the Enable field to True.
- The next step is to configure the security settings. First, look at the options available:
  - Users:
    - Anonymous: Allows connection without authenticating a specific user.
    - Username/Password: Specify authentication.
  - Security Mode: There are three (3) states of the security mode.
    - None: The connection is made without any security mode.
    - Sign: The connection has a digital signature.
    - Sign & Encrypt: The connection has a digital signature, and the data is encrypted.
  - Security Policy: There are four (4) states of the security policy.
    - None: The connection is made without any security policy.
    - Basic128Rsa15: The server makes a 128-bit AES encryption and compiled with Sha1.
    - Basic256: The server makes a 256-bit AES encryption and compiled with Sha1.

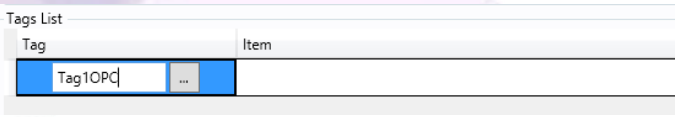
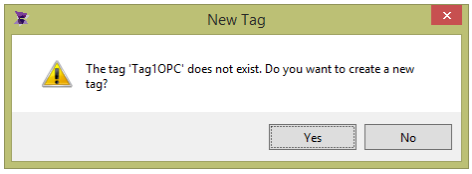
- Basic256Sha256: The server makes a 256-bit AES encryption and compiled with Sha256.

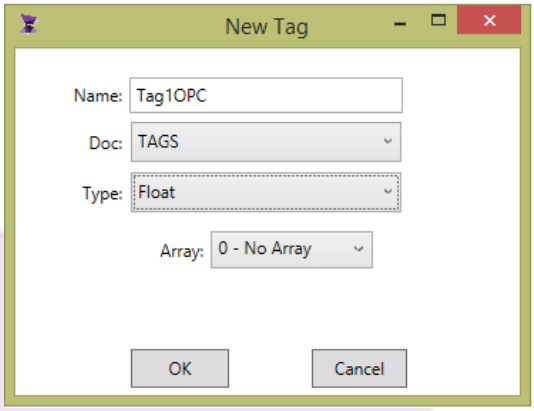
- Please set it according to the configuration done in the OPC UA Server.



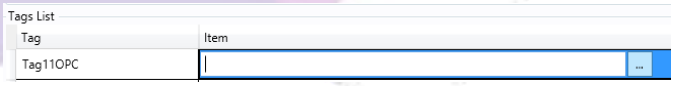
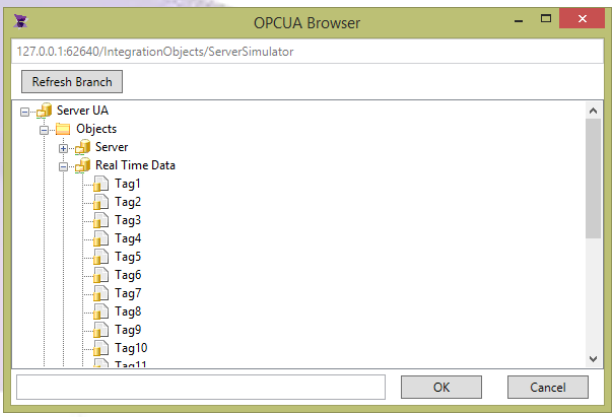
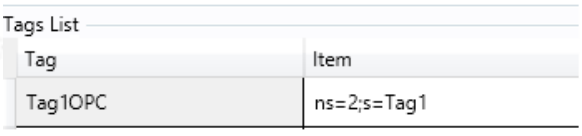
- User: Anonymous
- Security Mode: None
- Security Policy: None

- Last, insert Tags into the Tags List and map them to the OPC UA Server. We will need three (3) tags to communicate to the OPC UA Server. The fastest way to create a tag is directly adding them in the Tag field and removing the cursor or clicking enter. ADISRA SmartView will identify that tag does not exist and will give the option to create it.
  - Create a Tag

Steps	Images
Add the tag "Tag1OPC" into the Tag's field	
Remove the cursor or click enter. The ADISRA SmartView will identify the tag does not exist and will give the option to create it. Click "Yes"	

Select the "TAGS" document and set the type to "Float" and click OK.	

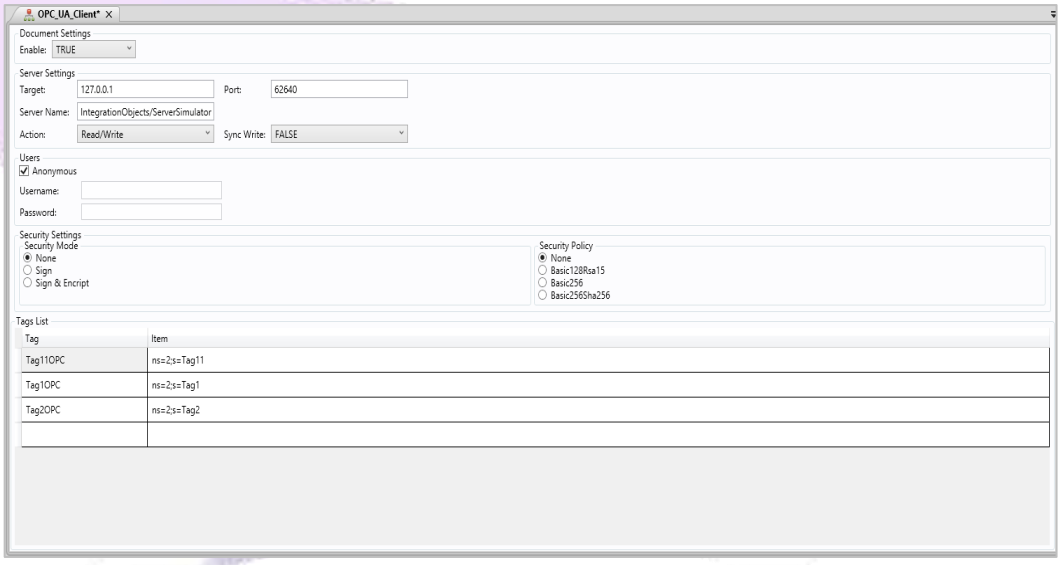
- Map the Tag to an OPC UA Server address

Steps	Images
Double-click the "Item" cell and it will show a browse button (…). Then click the browse button, it will open a dialog containing the Server address structure.	
If the connection succeeded, the user will see the Server UA structure. Locate the "Real Time Data" tags under the Objects folder and select the Tag1 and press OK. If nothing was shown, please check if the OPC UA Server is running and if the server settings were set correctly.	
As soon as the OPC UA Server Tag1 is selected, the Tags List will add the Item as in the image aside	

- Next, add two (2) more tags to the Tags List. Please follow all the above steps for the tags below:

  Tag2OPC -> Tag2
  Tag11OPC -> Tag11

- The final document is seen below.



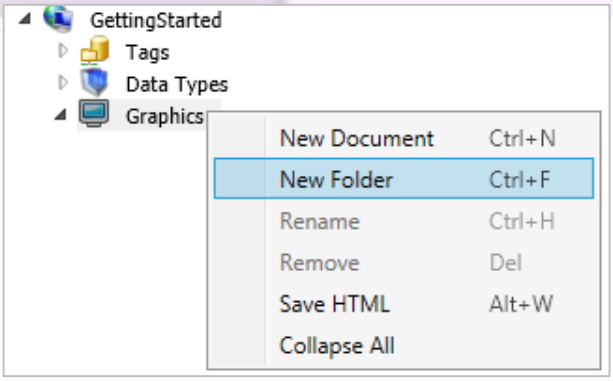
NOTE: *Tag1 and Tag2 are not simulated by the default installation of the OPC UA Server and Tag11 is simulated.*

- Save the document as "OPC UA Client".

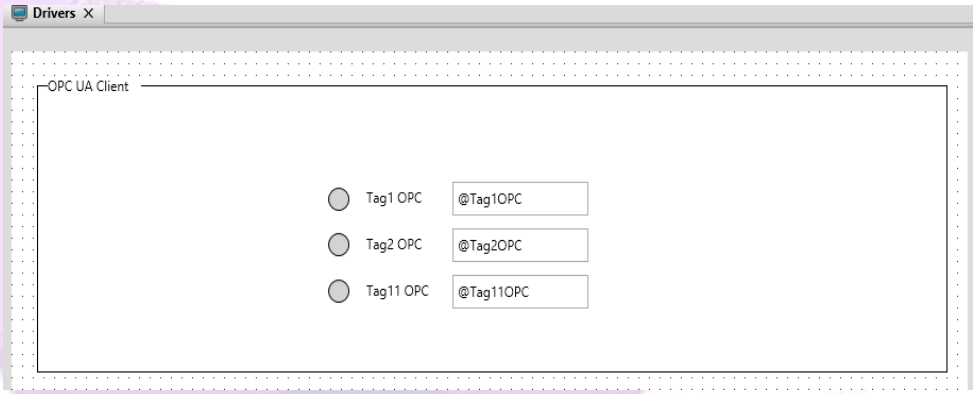
19.1.3. Create a New Graphics to Display the Tags from the OPC UA Client Communication

To configure the document, follow these steps:

- Create a new Graphics folder named "Drivers".



- The Drivers screen will have a GroupBox to hold the three (3) Tags from the OPC UA Client communication document. We will also use ellipses to sow the quality of the communication.
- The final screen will look like the following image.



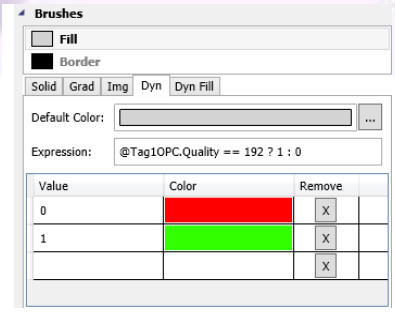
- GroupBox Settings.

Object	Top	Left	Width	Height
GroupBox1	21	24	857	255

- Ellipse Settings.

Object	Top	Left	Width	Height
Ellipse1	117	298	20	20
Ellipse2	157	298	20	20
Ellipse3	196	298	20	20

- Additional configuration to the Ellipse animation.

Select the Ellipse	
Go to the Property Window and locate the Brushes. - Select the Fill property - Select the Dyn (Dynamic Color) - Set the expression as: @Tag1OPC.Quality == 192 ? 1 : 0 - In the runtime, if the quality of the Tag is equals to 192 (good quality), the ellipse will get the green color. In case the quality is not good (different than 192), it will get red color	

Remember to apply this configuration to all the ellipses. The user will need to change the tag name to evaluate the correct quality in each expression. An expression is in fact a C# Expression.

- Label Settings.

Object	Text	Top	Left	Width	Height
Label1	Tag1 OPC	117	333	70	19
Label2	Tag2 OPC	157	333	70	19
Label3	Tag11 OPC	196	333	70	19

- TextBox Settings.

Object	Text	Top	Left	Width	Height
TextBox1	@Tag1OPC	112	415	128	29
TextBox2	@Tag2OPC	152	415	128	29
TextBox3	@Tag11OPC	192	415	128	29

- Save the document.

19.2. Driver Documents - Modbus

The driver's document allows the user to create the communication between ADISRA SmartView and different equipment, where the communication protocol needs to be determined in advance. Depending on the driver selected, a customized layout will be displayed to the user. In this example, we will create the communication to a Modbus simulator using TCP protocol.

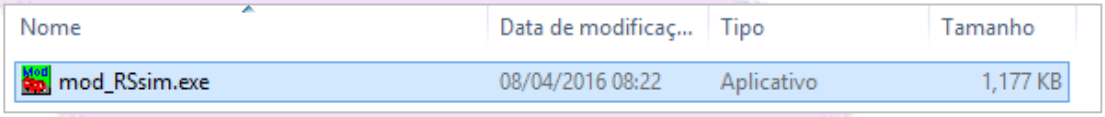
In this chapter, we will download and install a Modbus simulator and then configure the communication in ADISRA SmartView. The result will be a screen that monitors Tags connected to the third-party Modbus Simulator.

19.2.1. Download and Install a Third-Party Modbus Simulator (mod_RSsim)

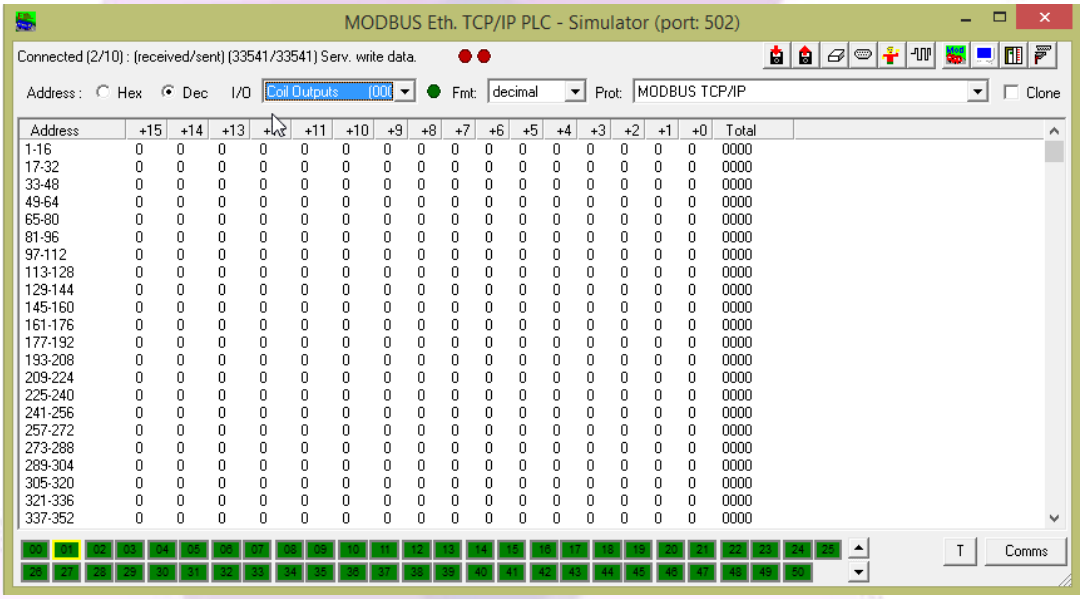
- Download the file from the URL

  https://adisra.com/wp-content/uploads/2020/03/ADISRA-SmartView-GettingStarted-DriverSimulator.zip

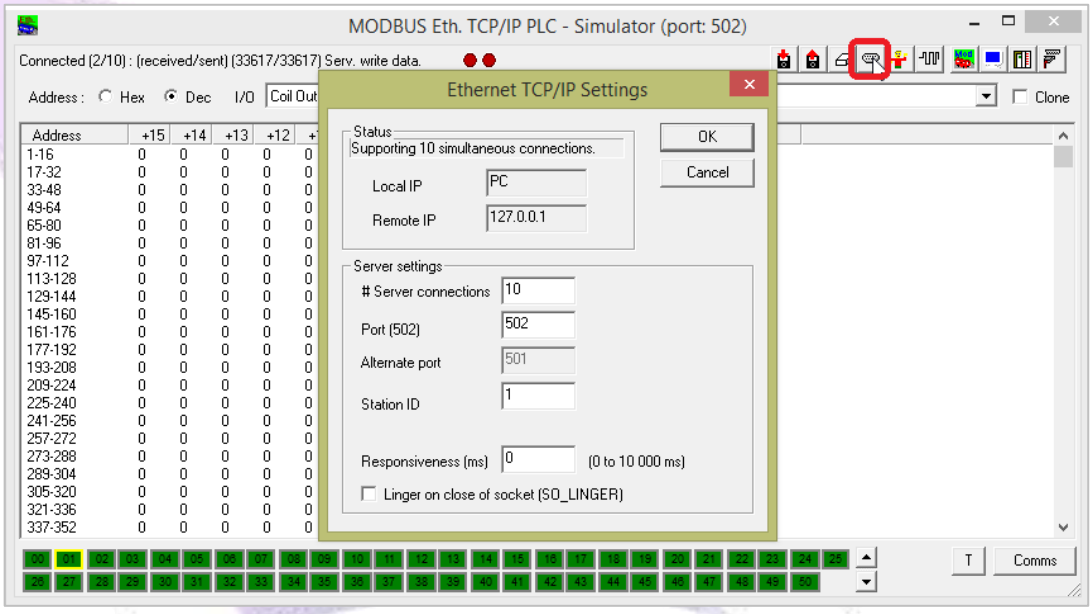
- Unzip and execute the simulator.



- The following dialog will open.



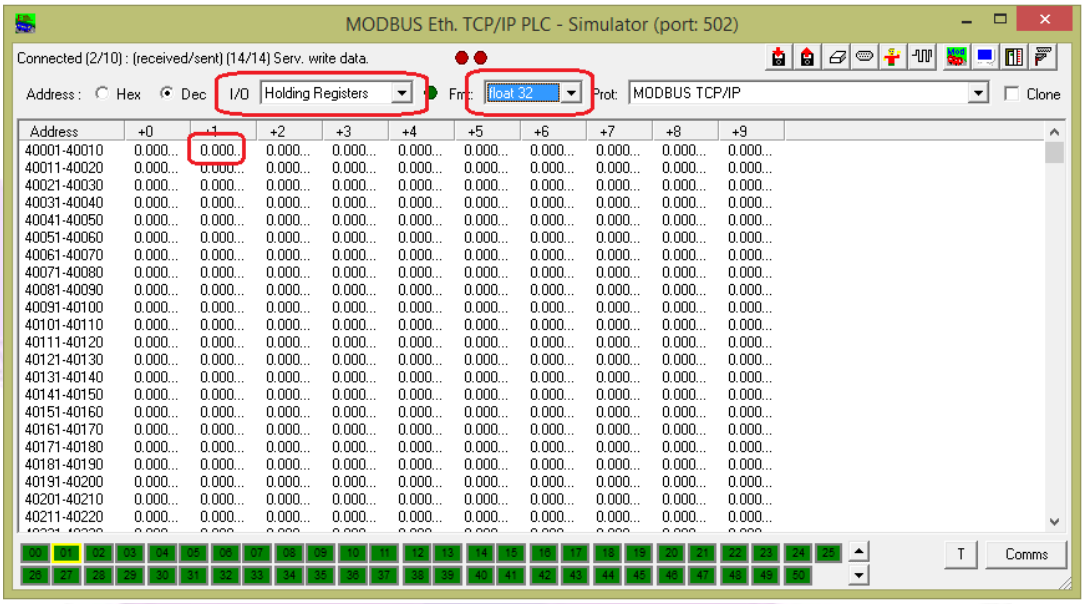
- Some important information will be needed before configuring the driver document in ADISRA SmartView.
- Open the Ethernet TCP/IP Settings by clicking in the highlighted button below:



- The Ethernet TCP/IP Settings dialog box will provide the Port Number and the IP Address information.

Port Number	*502*
Remote IP	*127.0.0.1*

- Become familiar with the simulator interface. It has different I/Os and Address types, and they can be changed in the simulator main window. When testing the communication, the user will be able to check the values being updated.



NOTE: *We will communicate to the Holding Registers and Analogue Inputs*

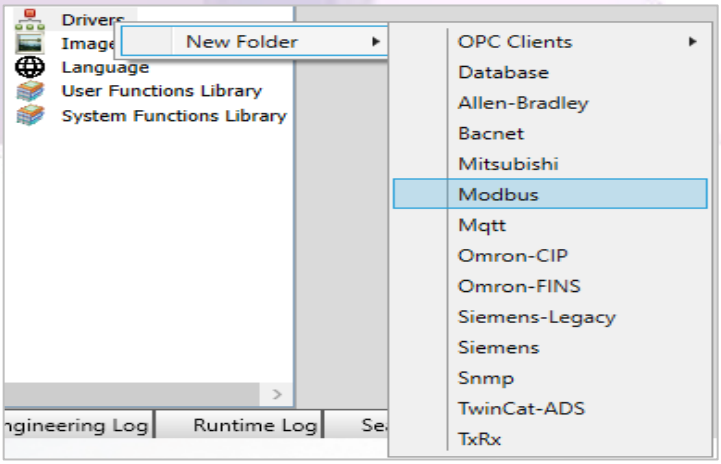
- Select the “Holding Registers” in I/O field
- Select the “float 32” in the Fnt field

- Now, in the ADISRA SmartView application, we will create a Modbus driver document in section 20.2.2.

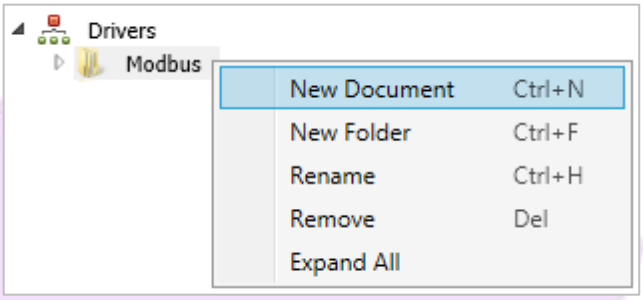
19.2.2. Creating a Modbus Driver Document

Follow these steps to configure a driver document.

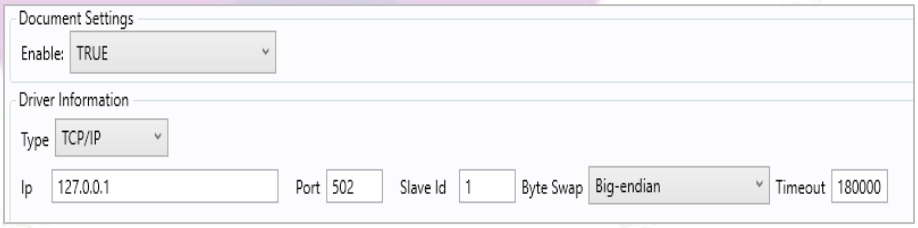
- Create a new "Driver" folder in the Navigation Tree under Drivers -> New Folder -> Modbus.



- A new folder with be created named Modbus. Now please create a document under that folder by right clicking over it and selecting "New Document".



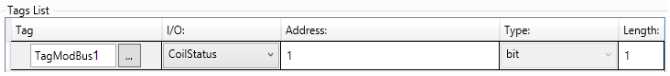
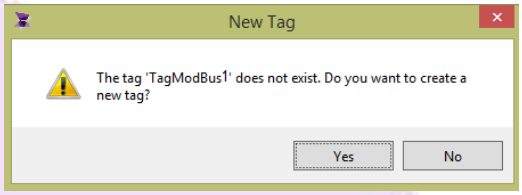
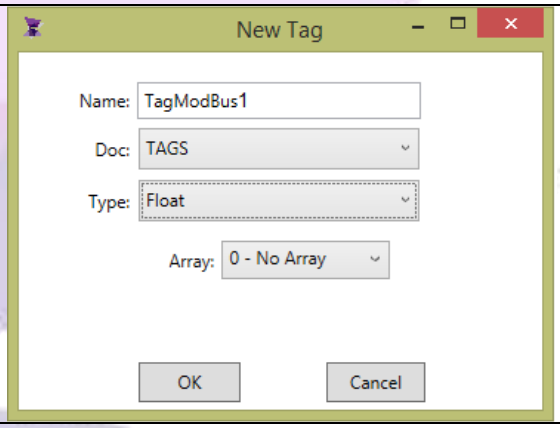
- Open the newly created document to start configuring the connection to the Simulator.
- In the Driver Information, set the Host/IP where the Modbus Simulator is running and the Port Number.



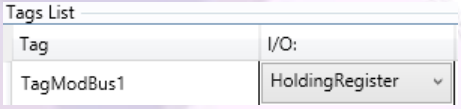
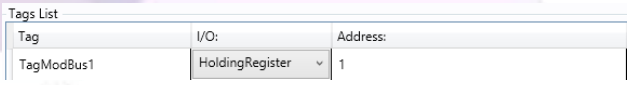
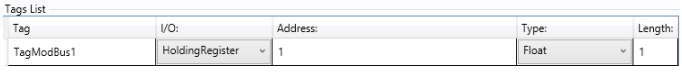
- In the Read and Write fields, let's enable automatic refresh, but it can also be done by request:

- Insert Tags into the Tags List and map them to the Simulator. We will need two (2) tags to communicate to the Simulator. The fastest way to create a tag is directly adding them in the Tag field and removing the cursor or clicking enter. ADISRA SmartView will confirm the tag does not exist and will give the option to create it.

- Create the Tag

Steps	Images
Add the tag "TagModBus1" into the Tag's field	
Remove the cursor or click enter. ADISRA SmartView will confirm the tag does not exist and will give the option to create it. Click "Yes"	
Select the "TAGS" document, set the type to "Float" and click OK.	

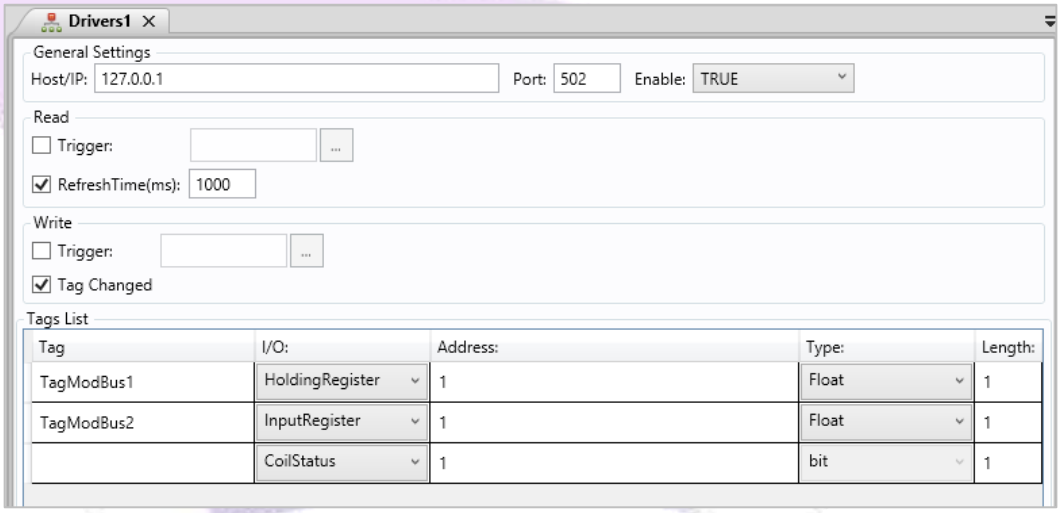
- Map the Tag to the Modbus address

Steps	Images
Select the HoldingRegister I/O	
Set the Address to 1	
Set the Type of the Address to Float	

- Next, add one (1) more Tag to the Tags List. Please follow all the above steps for adding the Tag below.

SmartView	Modbus Simulator
TagModBus2 (Float)	I/O: InputRegister -> Address 1 -> Type Float -> Lengths 1

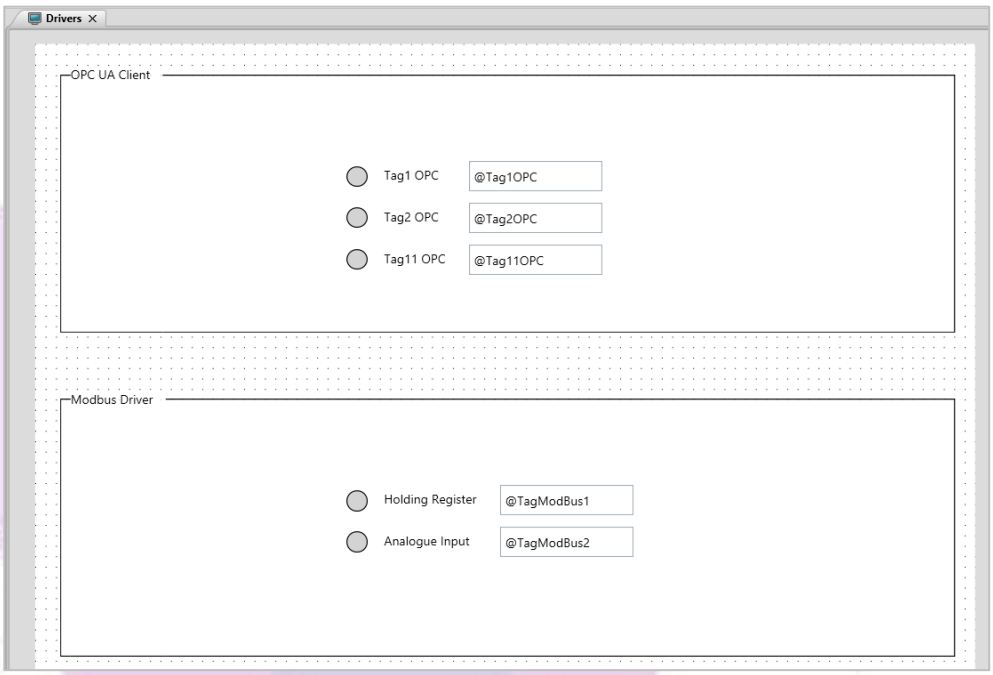
- This is the final document.



- Save the document as "Drivers1".

19.2.3. Add the Modbus Tags to the Drivers Graphic

- Open the Drivers' screen. It will have a GroupBox to hold the two (2) Tags from the Modbus driver document. We will also use ellipses to show the quality of the communication.
- The final screen will look like the following image.



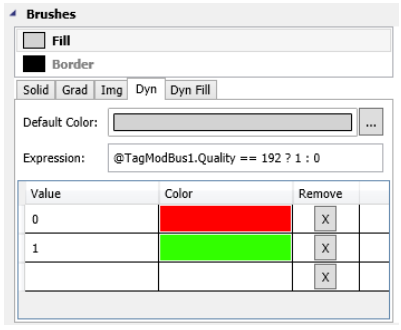
- GroupBox Settings

Object	Top	Left	Width	Height
GroupBox2	331	24	857	255

- Ellipse Settings

Object	Top	Left	Width	Height
Ellipse4	427	298	20	20
Ellipse5	466	298	20	20

- Additional configuration to the Ellipse animation

Select the Ellipse	
- Go to the Property Window and locate the Brushes. - Select the Fill property - Select the Dyn (Dynamic Color) - Set the expression as: @TagModBus1.Quality == 192 ? 1 : 0 - In the runtime, if the quality of the Tag is equals to 192 (good quality), the ellipse will get the green color. In case the quality is not good (different than 192), it will get red color	

- Remember to apply this configuration to all the ellipses. The user will need to change the tag name to evaluate the correct quality in each expression.

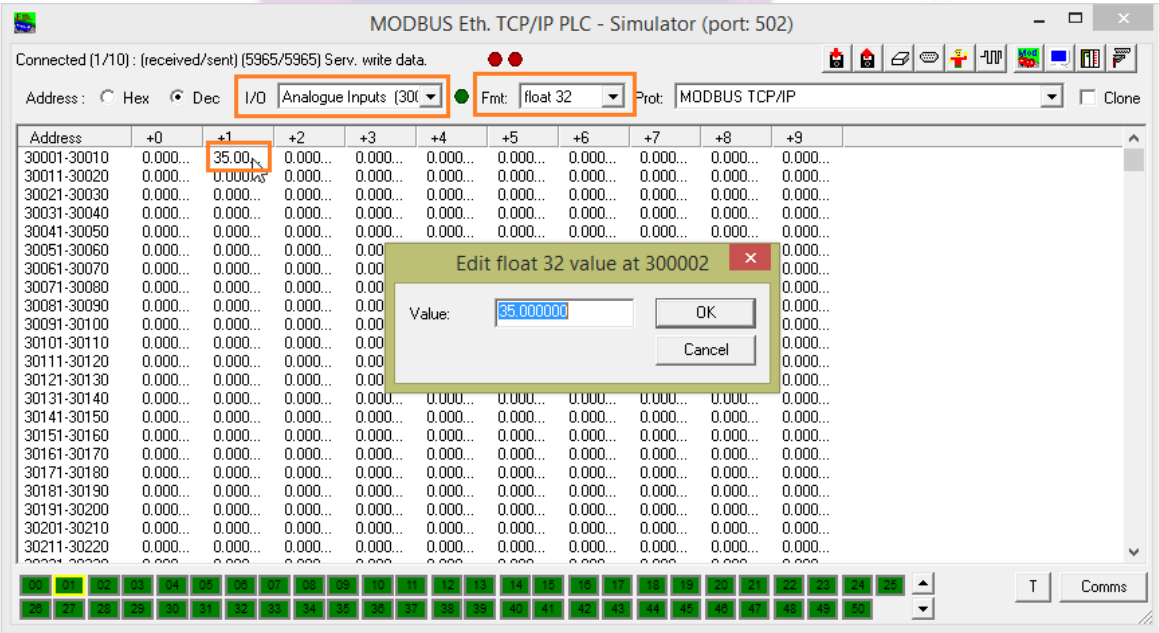
- Label Settings

Object	Text	Top	Left	Width	Height
Label4	Holding Register	427	333	126	19
Label5	Analogue Input	467	333	126	19

- TextBox Settings

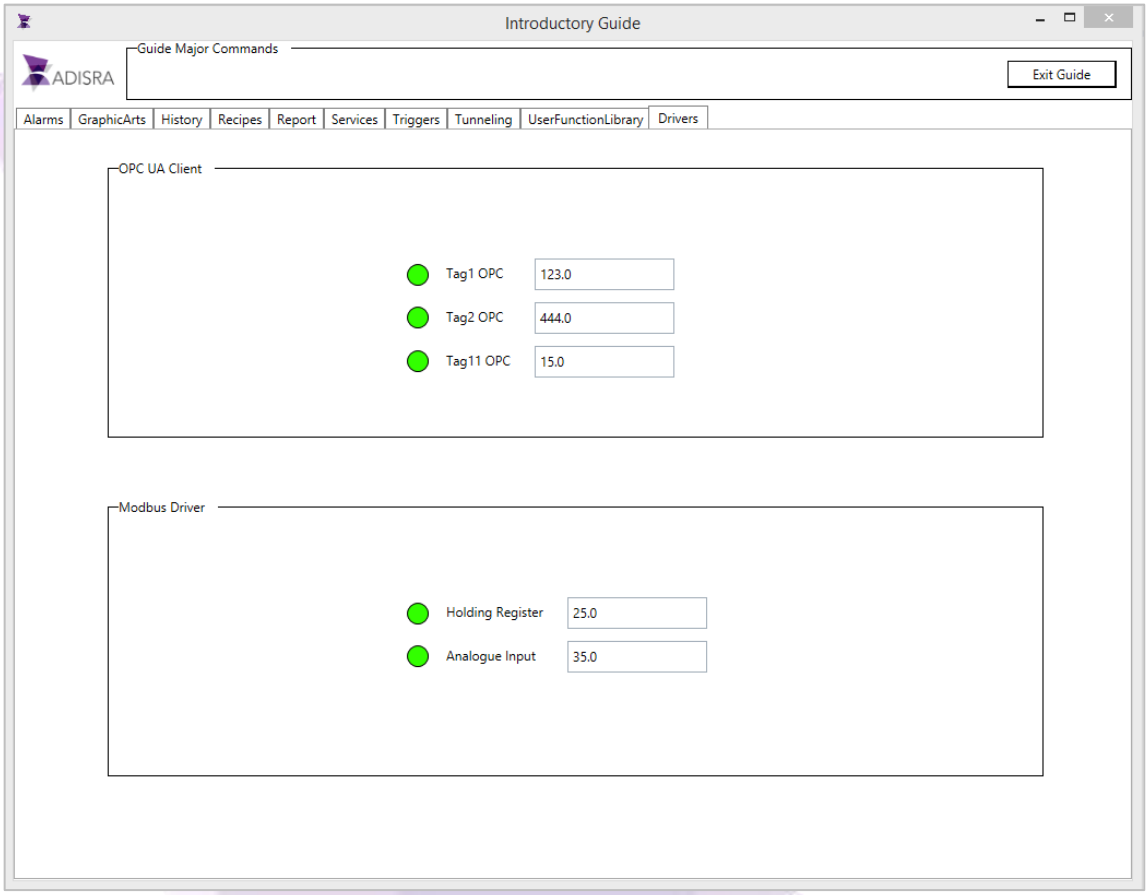
Object	Text	Top	Left	Width	Height
TextBox4	@TagModBus1	422	445	128	29
TextBox5	@TagModBus2	462	445	128	29

- It is important to understand the Analogue Input is read only in the Simulator. So, the user will not be able to write a new value from ADISRA SmartView. If the user wants to see it communicating, they will need to change its value in the Simulator.



- Select the Analogue Inputs
- Locate the Address 1
- Double click the cell
- Set the desired value in the popup window
- When the user executes the ADISRA SmartView, go to the Drivers tab and check the values. They will need to finish

the Getting Started Guide to have the Tab Object configured allowing navigation to the Drivers Graphic.



- Save the Screen.

20. Tunneling Documents

The Tunneling document performs a value exchange between the configured items. This exchange can be done between tags (unidirectional or bidirectional) or from an expression to a tag. It is not possible to configure an expression to receive the value of a tag.

NOTE: *If the tags configured in items are of different types (for example, a "Float" tag trying to write its value to a "String" tag), Tunneling perform the required value conversions.*

20.1. Creating Tunneling Documents

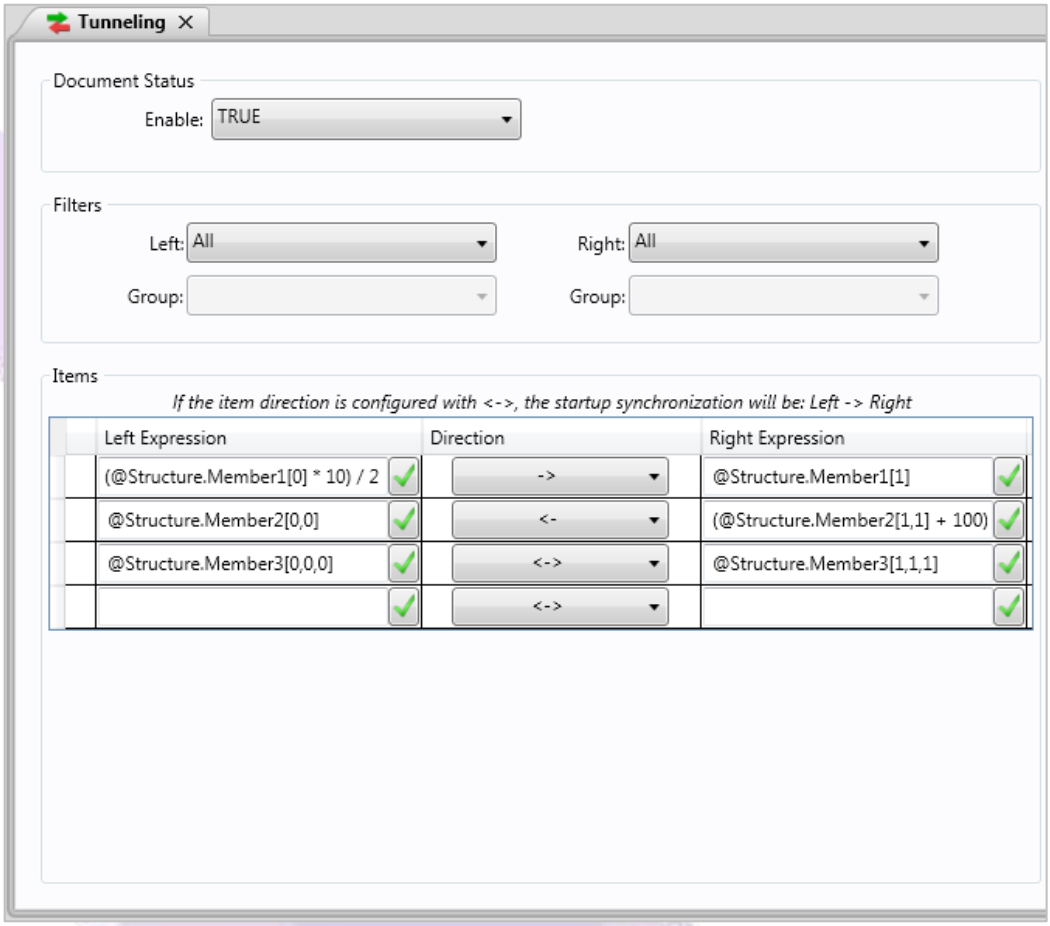
Follow these steps to configure the document:

- Create a new "Tunneling" document in the Navigation Tree.
- Click on the first empty row in the Left Expression column to select it.
- Enter the following expression: "(@Structure.Member1[0] * 10) / 2"
- Click the Verify button next to the Left Expression column.
- In Direction select the value <->.
- Click on the first empty row of the column "Right Expression" to select it.
- Insert the following tag: "@Structure.Member1[1]"
- Click the Verify button next to the Right Expression column. By validating this field, an icon is displayed () Indicating that there is an error in the configuration of the line. By positioning the mouse over the icon, the following message appears:

  *"Invalid Direction. You cannot configure "(@Structure.Member1[0] * 10) / 2" to receive a value."*

- To correct the error, select the value -> in the Direction field.
- Repeat the same process for the following settings:
  - Left Expression: "@Structure.Member2[0,0]", Direction: "<-"
  - Right Expression: "(@Structure.Member2[1,1] + 100) / 5"
  - Left Expression: "@Structure.Member3[0,0,0]", Direction: "<->"
  - Right Expression: "@Structure.Member3[1,1,1]"

- Save the changes made to the document as "Tunneling".



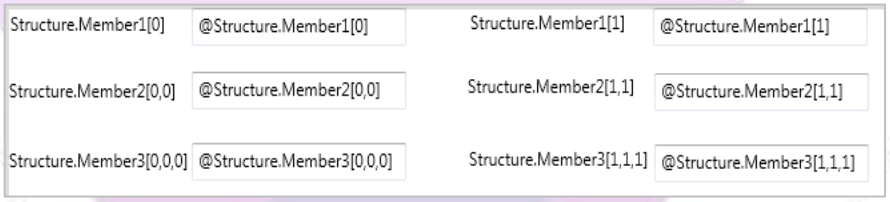
20.2. Creating and Configuring Screen for Tunneling

For this example, the Tunneling screen only needs six (6) Label objects and six (6) TextBox objects.

- The Label configuration should be as follows:
  - Label1: Text: “Structure.Member1[0]”
  - Label2: Text: “Structure.Member1[1]”
  - Label3: Text: “Structure.Member2[0,0]”
  - Label4: Text: “Structure.Member2[1,1]”
  - Label5: Text: “Structure.Member3[0,0,0]”
  - Label6: Text: “Structure.Member3[1,1,1]”

- The TextBox configuration should be as follows:
  - TextBox1: Text: “@Structure.Member1[0]”
  - TextBox2: Text: “@Structure.Member1[1]”
  - TextBox3: Text: “@Structure.Member2[0,0]”
  - TextBox4: Text: “@Structure.Member2[1,1]”
  - TextBox5: Text: “@Structure.Member3[0,0,0]”
  - TextBox6: Text: “@Structure.Member3[1,1,1]”

- Save the graphics document with the name of "Tunneling".



21. User Function Library Documents

The User Function Library is a user-created function library. One the library is created, it can be exported and imported to other projects.

NOTE: *Tags are not allowed in functions created in the document.*

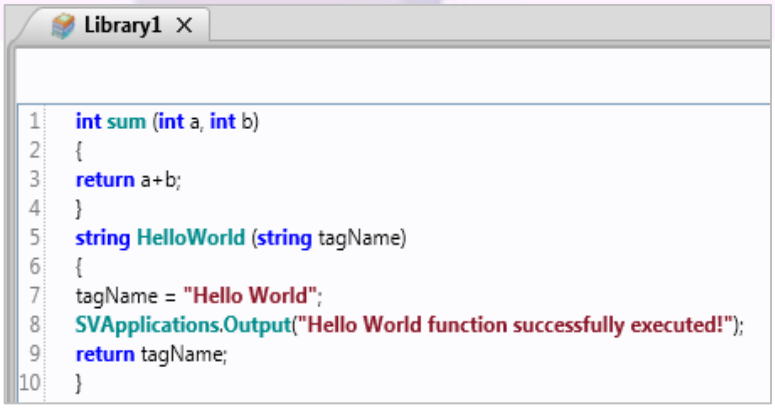
21.1. Creating User Function Library Documents

Follow these steps to create and configure a document:

- Create a new "User Function Library" document from the Navigation Tree.
- Enter the following code into the document:

```
int sum (int a, int b)
{
return a+b;
}
string HelloWorld (string tagName)
{
tagName = "Hello World";
SVApplications.Output("Hello World function successfully executed!");
return tagName;
}
```

- Click the Verify button in the Ribbon to verify the script runs successfully.
- Save the document with the name "Library1".



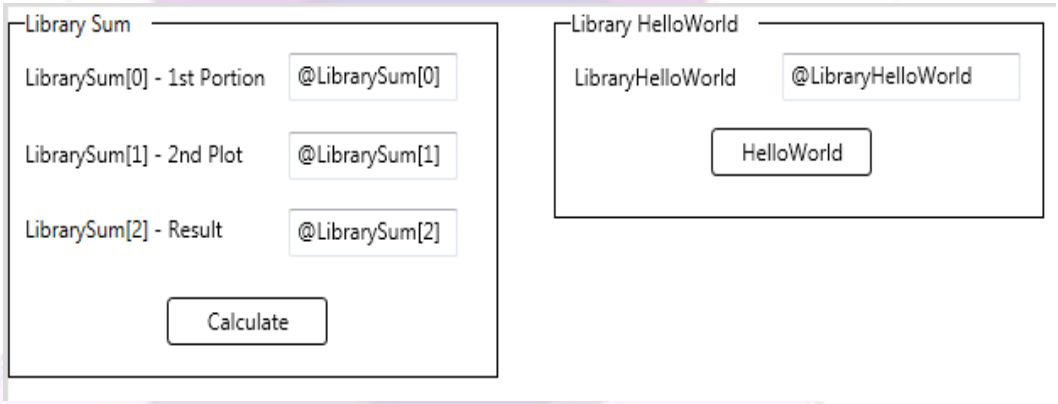
21.2. Create Configuring Screens for the User Function Library

- Create two (2) more tags for the User Function Library screen:
  - Name: "LibrarySum", Type: "Integer", Dimension: "1", Positions: "3"
  - Name: “LibraryHelloWorld”, Type: “String”, Dimention: “0” (zero)
- Create a new folder under Graphics called "UserFunctionLibrary".
- Create a new document under Graphics.
- After creating the tags above, create two (2) GroupBoxes (one (1) for the function "LibrarySum" and one (1) for the "HelloWorld") with the following properties:
  - GroupBox1: Text: "Library Sum", Width: "290", Height: "193"
  - GroupBox2: Text: "Library HelloWorld", Width: "290", Height: "110"
- For the LibrarySum GroupBox, three (3) Label objects, three (3) TextBox objects, and one (1) Button object to be in GroupBox1 are needed with the following properties:
  - Label1: Text: “LibrarySum[0] - 1st Portion”, Width: “150”, Height: “18”
  - Label2: Text: “LibrarySum[1] - 2nd Plot”, Width: “150”, Height: “18”
  - Label3: Text: “LibrarySum[2] - Result”, Width: “150”, Height: “18”
  - TextBox1: Text: “@LibrarySum[0]”, Width: “100”, Height: “25”
  - TextBox2: Text: “@LibrarySum[1]”, Width: “100”, Height: “25”
  - TextBox3: Text: “@LibrarySum[2]”, Width: “100”, Height: “25”
  - Button: Text: "Calculate", Width: "95", Height: "25", Mouse Up:

@LibrarySum[2] = Library1.**sum**(**@LibrarySum[0]**, **@LibrarySum[1]**);

- For the LibraryHelloWorld GroupBox, one (1) Label, one (1) TextBox, and one (1) Button is needed with the following settings:
  - Label: Text: “LibraryHelloWorld”, Width: “115”, Height: “18”
  - TextBox1: Text: “@LibraryHelloWorld”, Width: “141”, Height: “25”
  - Button: Text: "HelloWorld", Width: "95", Height: "25", Mouse Up:

    @LibraryHelloWorld =Library1.**HelloWorld**(**@LibraryHello World**);
- Save the document with the name "UserFunctionLibrary". The result should be like the figure below.



22. Setting Document Images

The Images document is a library of project images. Only images contained in this library can be used in the project.

22.1. Inserting New Images into the Image Library

For this project, add the image below to the image library for use on the main project screen. Save the following image for later use inside the project folder under the "Images" folder.

Follow these steps to insert a new image:

- Open the document Images in the Navigation Tree.
- In the Ribbon Images, click the Insert Image button.
- A new window appears. In this window, navigate to the location where the image has been saved.
- Select the image and click Open to add it to the library. The image is now added to the image library.

23. Advanced Graphic Objects

In this topic, learn how to configure and use some Advanced Graphic Objects that have not been presented so far in this guide.

23.1. Screen Objects

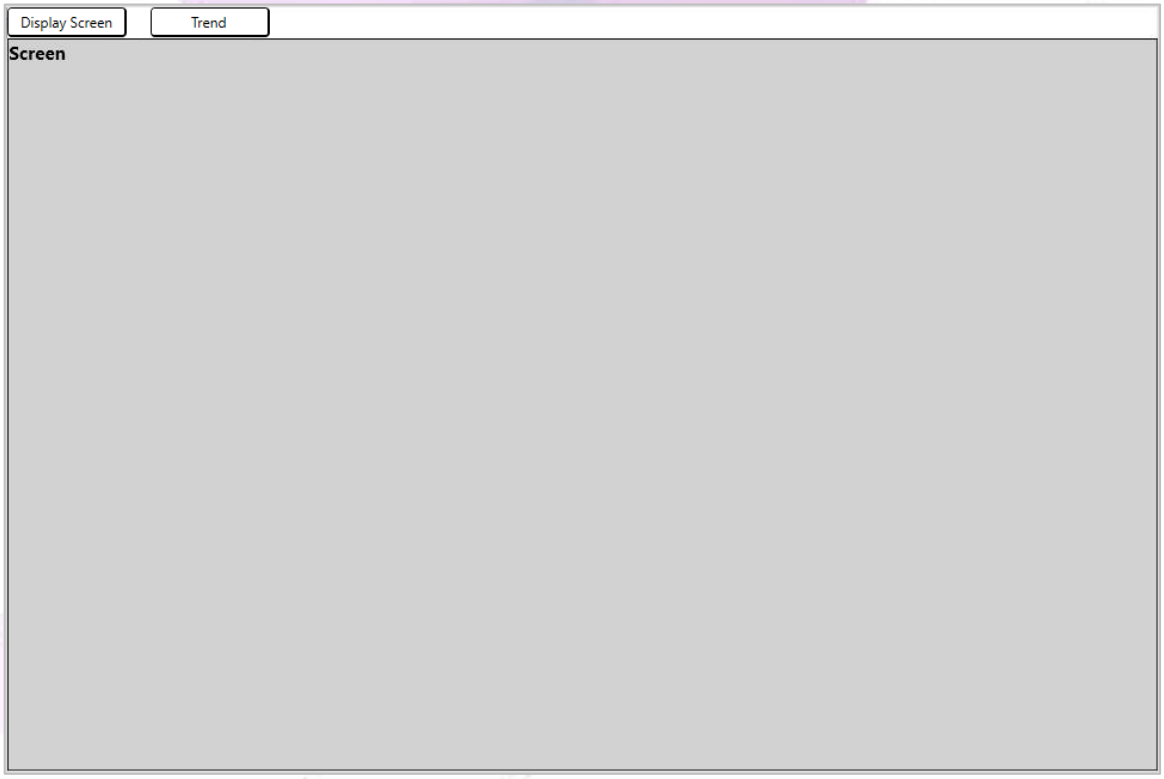
The Screen Object is great for creating a nice layout to the application. It allows the user to load different Graphic Screens into it, working as a reserved content area.

To set up a Screen Object, follow these steps:

- Create a new document under Graphics and save it with the name "GraphicArts".
- For this example, create two (2) more Graphics documents to be displayed on the Screen Object. To do this, create a new folder in Graphics called "Screens".
- Create a new Graphics document in the Screens folder and save it with the name "ObjectScreen".
- Create another new Graphics document in the Screens folder and save it with the name of "Trend".
- With the document “GraphicArts” opened in the workspace, select the Graphics Ribbon and then click the Screen Object.
- Click new the point (0,0), the upper left-hand corner, so that the Screen Object is created with the default size.
- Select the Screen Object created and configure it as follows in the Property Window:
  - Width: "1024", Height "646", Top "28," Left "0".
- Select the Graphics Ribbon and select the Button object.
  - Click near the point (0,0), the upper left-hand corner, so that the Button object is created with the default size.
- Select the Button object created and configure it as follows:
  - Text: "Display Screen", Width: "100", Height: "25", Top "0", Left: "0".
  - Mouse Up: "Screen1.SetScreen("Screens\\ObjectScreen");"

- Select the Graphics Ribbon and select the Button object.
- Click near the point (0,0), the upper left-hand corner, so that the Button object is created with the default size.
- Select the Button object created and configure it as follows:
  - Text: "Trend", Width "100", Height "25", Top "0", Left "120".
  - Mouse Up: "Screen1.SetScreen("Screens\\Trend");"
- Save the changes to the document.

NOTE: *When the application is executed, by clicking the Display Screen button, the Objects document is displayed in the area limited by the Screen Object. The same occurs when the user clicks the Trend button; the document Trend is displayed.*



23.2. Setting Up Objects Screen

In this section, we will explain how to set up some Screen Objects that have extra features.

23.2.1. TextBox Object for Passwords

It is possible to configure a TextBox to display a '●' symbol in place of the text being entered when secure on-screen input is

required. For example, when entering a password, the '●' symbol will display on the screen so the actual password text will not be displayed on the screen. To configure the object, follow the steps below:

- Open the document "ObjectScreen" in Graphics -> Screens.
- With the document open in the workspace, click the object TextBox in the Graphics Ribbon.
- Click near the point (0.0), upper-let hand corner, of the document to create the object with standard size.
- Select the TextBox object and set the values as follows:
  - Top: "0", Left: "0".
  - Width: "120", Height: "25".
  - Select the "Password" checkbox to enable it.
- Create another TextBox and set the values as follows:
  - Top: "25", Left: "0".
  - Width: "120", Height: "25".
- Create a Button object and set the property Mouse Up with the following script: "TextBox2.Text = TextBox1.Text;".
- Save the document

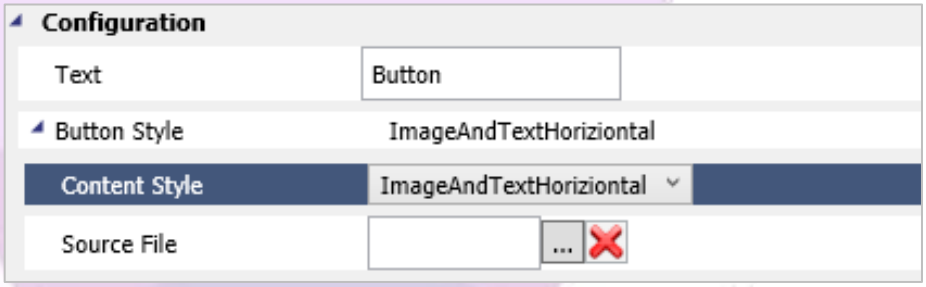
In the application execution, everything typed in the TextBox that is set to Password will be replaced on the screen as the symbol shown in the image below.

After clicking the button, the configured script sends the contents of the TextBox Password object to the other TextBox to verify that the content remained correct.

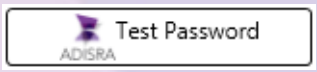
23.2.2. Button Object Displaying Image

A Button object can display an image in the interface. Follow the steps below to set it up.

- Open the document "ObjectScreen" in Graphics -> Screens.
- Select the Button object created in the previous section "22.2.1 TextBox Object for Passwords".
- In the Property Window, inside the Button Style area, select the ImageAndTextHorizontal value for property Content Style.



- In the Source File, click on the (…) button to open the Images Browser.
- Double- click on the ADISRA logo image to select it.
- Save the changes to the document.



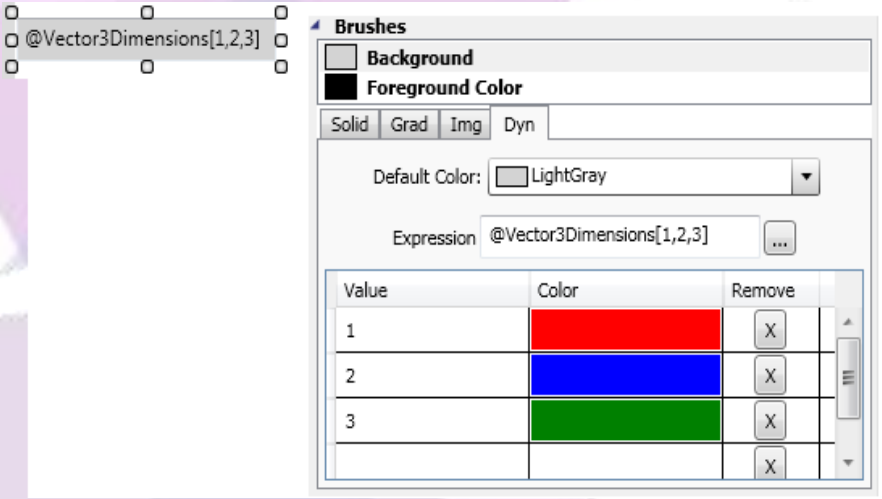
23.2.3. TextBox Object with Dynamic Background

Some objects can be set with a dynamic color. Dynamic color enables color changes according to a set condition. For this example, we will use the object TextBox, but many other objects can also use this setting.

To set the object, follow these steps:

- Open the document "ObjectScreen" in Graphics -> Screens.
- Create a new object TextBox.
- Select the object.
- In the Text field, configure the tag @Vector3Dimensions[1,2,3].
- In Brushes, located in the Property Window, choose Background Color.
- Then, select the tab Dyn.

- In the Default Color field, select the color LightGray.
- In Expression field, set the value @Vector3Dimensions[1,2,3].
- Click the first empty row of the column Value and enter the value "1".
- Click the first empty row of the column Color and select the color Red.
- Repeat the process to set the following values:
  - Value: "2", Color: "Blue"
  - Value: "3", Color: "Green"
- Save the changes made to the document



When executing the application, the object initially appears with the color LightGray because the current tag value is “0” (zero). Because the value “0” is not set in Background Color the object receives the color set as the Default Color.

Changing the tag value to “1” changes the object's color to red. By changing the tag value to “2”, the object's color changes to blue, and if the tag value is “3”, the color changes to green. Any value other than “1”, “2” or “3” will be gray in color.

23.3. Setting Up an Objects Screen

For setting up the Trend screen, we need to create new tags. These tags simulate data so that the user can view in the Trend object the variations of the tag values.

23.3.1. Create New Trend Tags

To set up tags, follow these steps:

- Open the document Tags.
- Insert a new tag.
- Rename the tag NewTag1 to "Simulation".
- Set the tag type to Integer.
- Select the value 1 in the Dimension field and set the value 3 in the field positions.
- With the tag selected in the Property Window, use the combo box to select the position Simulation[0].
- Set the value "-200000" into the field Min Value and the value "200000" into the field Max Value.
- In the Server field in Communication, select Simulation.
- In the Type field, select the Linear option.
- In the Samples/s field, enter the value "10".
- In Period (ms) field, enter the value "10000".
- For positions 2 and 3 of the vector, set as follows:
  - Simulation[1]: Min Value "-250000", Max Value "250000" Communication Server: "Simulation", Type: "Sine", Samples/s: "10" Period (ms): "10000"
  - Simulation[2]: Min Value "-300000", Max Value "300000" Communication Server: "Simulation", Type: "Random Numbers" Frequency (ms), "10000"
- Save the changes to the document.

23.3.2. Create Trend Screen

With tags created and configured, it is time to set up the Trend screen using the steps below.

- Open the document Trend under the Graphics -> Screens.
- In the Graphics Ribbon, select the object Trend in the Advanced Objects group.

- Click near the point (0,0), upper left-hand corner, to create the object with the default size.
- Select the created Trend object.
- Set it up as follows:
  - Width: "1014", Height: "645",
  - Left: "0", Top: "0"
- In the Trend Configuration area on the General tab, enter the value for New Pen as: "Simulation[0]" and click the add button (+).

- Do the same for the values Simulation[1] and Simulation[2].
- Double-click the pen Simulation[0] to view the properties.
- Change the Pen Color to Blue and Line Style to - -
- In Selected Item field, select the pen Simulation[1].
- Change the Pen Color to Green and Line Style to . . .
- In "S elected Item field, select the pen Simulation[2].
- Change the Pen Color to Red.
- Save the changes made to the document.

When the application is running, the user can follow the variation of the values and the chart generated.

24. Setting Up the Main Graphic Screen for the Project

Once the project settings are done, the user can create the layout of the screens for the project.

First, create a main screen, this it is a screen that displays all other graphics created for the project. This main screen does not change during the project execution, it only displays the other existing screens. The main screen also contains some commands, such as stop running the application

24.1. Creating the Main Screen

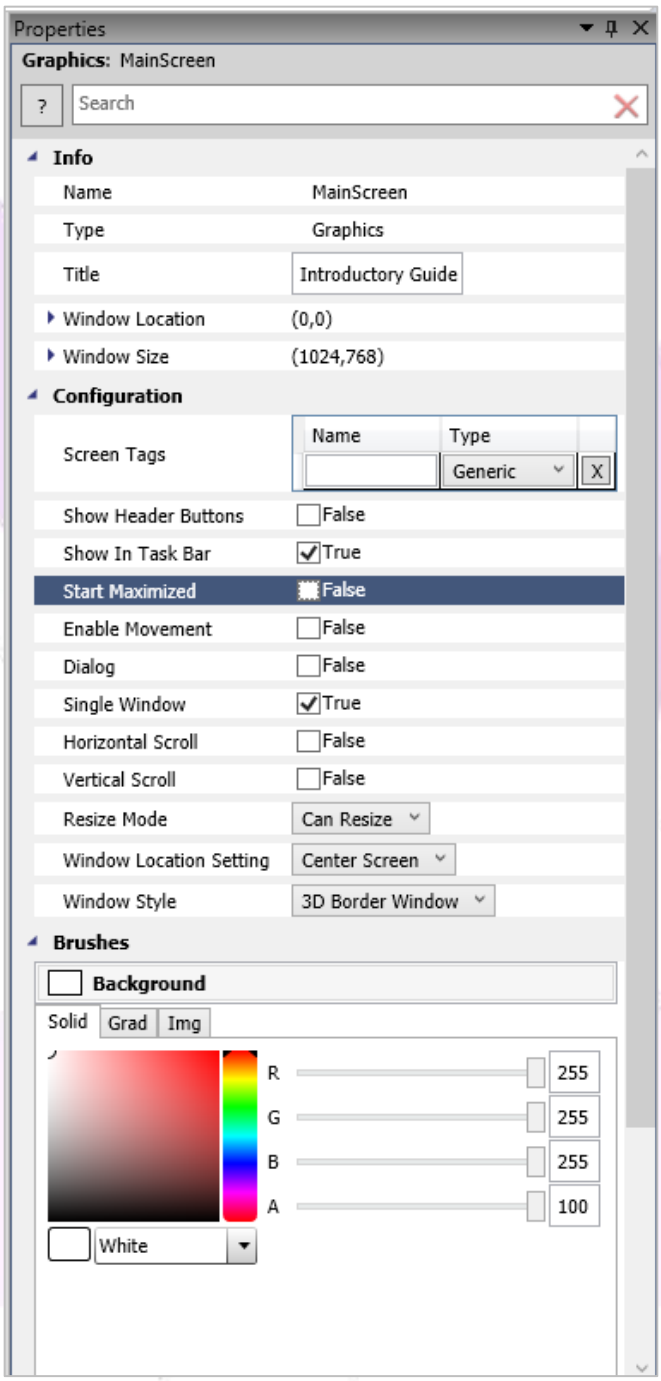
To create a graphic screen, right-click on Graphics, located in Navigation Tree. Within the context menu that is displayed, choose the New Document option. This creates a new graphic in the document display area. By default, the name of the created screen is "Graphics1", but the user will need change it to "MainScreen".

Save the document in the Graphics folder.

24.2. Configuring the Main Screen

In the Property Window of the graphic document, configure the screen as follows:

- Title: Introductory Guide
- Document Size: Width: "1024", Height: "768"
- Position in relation to the top: (top): "0", (left): "0"
- Dialog: False
- Show Header Buttons: False
- Enable Movement: False
- Horizontal Scroll: False
- Vertical Scroll: False
- Single Window: True
- Show In Task Bar: True
- Start Maximized: False



- Save the changes to the graphic document.

24.3. Adding and Configuring Graphic Objects

In this topic the user will learn how to use and configure the main Graphical Objects in ADISRA SmartView.

24.3.1. Adding and Configuring a GroupBox

As a first step in customizing the Main Screen, create a GroupBox to delimit the command region of the application. To create and configure the GroupBox follow these steps:

- With the Main Screen graphic document open, select the Graphics Ribbon.
- Click on the object GroupBox located in Interface Objects.
- Click near the point (0.0), upper left-hand corner, of the graphic display and drag the mouse pointer to the right edge of the screen.
- Release the mouse button to create the GroupBox object.
- With the object created and selected, its properties are displayed in Property Window. To configure the text that is displayed on the screen, click the text box of the Text property. With the selected field, enter the desired value “Guide Major Commands”.
- Now, the user can adjust the position of the object relative to the screen. In the property Top, set the value to "0" (zero), and in the Left field, set the value to "100".
- With the position set, set the size of the object. In Width, set the value to "925" and in Height, set the value to "52”.
- As a last step, save the document to confirm the changes made to the document.

24.3.2. Inserting a Configuring Application Close Button

The second object to set up is a button that closes the application. To set the button up, follow these steps:

- With the graphic document MainScreen open, select the Graphics Ribbon.
- Click the Button object located in Basic Objects.
- Click in the right of the inner region GroupBox created earlier. This creates a Button object in the default size. Change the values to: Width: “100”, Height: “25”.
- On the Top property, set the value to "20" and in the Left property, set the value to "910".
- In the Text property, set the value "Exit Guide".
- Now all that remains is to configure the event that causes the application to close. In the Mouse Up property, click on the text box. This will open a new window. In the new

window, configure the script that will be executed every time the user clicks and releases the left mouse button on the object.

- Within this new window, enter the following code snippet:

```
if (MessageBox.Show ("Do you want to quit the application?", "Quit the application", MessageBoxButtons.YesNo, MessageBoxIcon.Question) == DialogResult.Yes)
{
SVApplications.StopApp ();
}
```

When executed, this code snippet displays a message asking if the user wants to exit the application. If Yes is chosen, the application closes or if No is chosen, the application continues running.

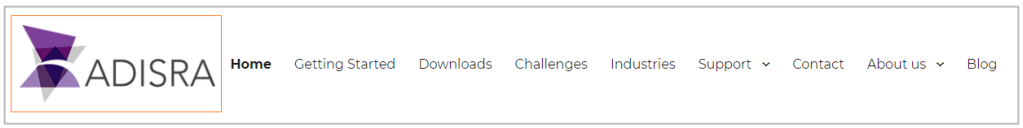
- As a last step, simply save the document to confirm the changes made to the document.

24.3.3. Setting Up the ADISRA SmartView Logo

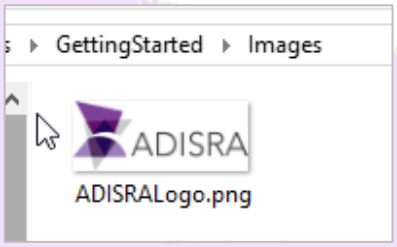
The Image object is an object in which to display an image from the image library. For this example, add the ADISRA logo in the upper left corner of the main screen.

Follow these steps to add the ADISRA logo to the upper corner of the main screen.

- With the document MainScreen open, select the Graphics Ribbon.
- Select the Image object.
- Click near the point (0,0), upper left-hand corner of the screen. It will create an Image object size 80x80 (width x height), the default size. As soon as the Image object is inserted, please change its Size and Location from the Property Window.
  - Location *Top:* 12. *Left:* 7
  - Size *Width:* 85. *Height:* 35
- We will save the ADISRA logo from the website into the project. Please open the website www.adisra.com
  - Right click over the icon



- Select to "Save image as"
- And save it to the project folder under the "Images" folder with the "ADISRALogo.png" name



- Click on the (…) button in the Source property next to the Source File field. A new window will open displaying all the images from the image library, the images folder is under the project folder. For this example, select the single image previously saved from the ADISRA website (ADISRALogo.png). To do this, select the image and click the OK button or just double-click on the image.
- Save the changes made to the document.
- The chosen image is displayed by the object Image, as shown in the image below.

24.3.4. Inserting Tab Object

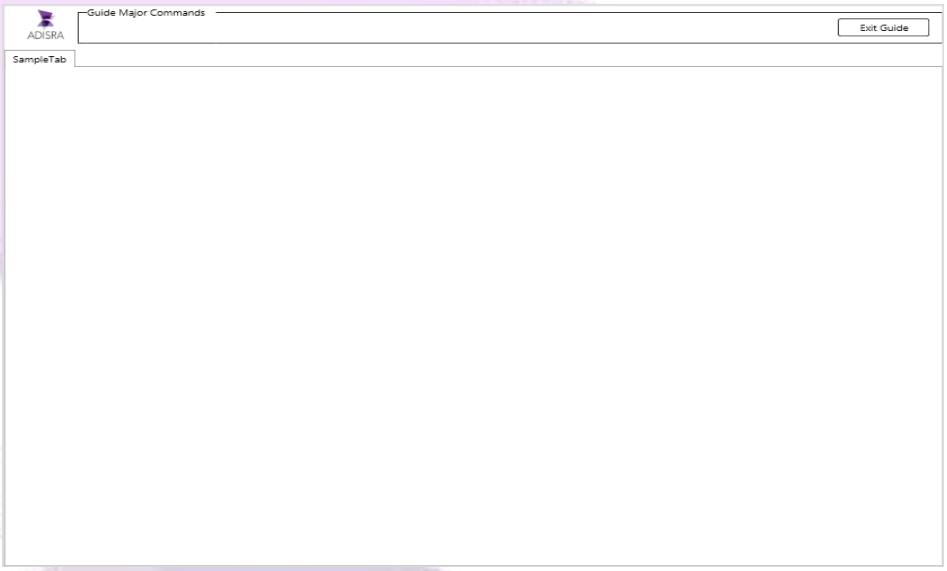
The Tab object is an interface object that displays other graphical screens by arranging them in tabs. For this example, the user enters and configures the basic properties. Throughout the guide the user has learned how to get objects to start displaying other graphic documents.

To insert and configure the Tab object follow these steps:

- With the Main Screen graphic document open, select the Graphics Ribbon.
- Click the object Tab located in Interface Objects.
- Click the left mouse button anywhere on the screen. This creates an object Tab with a default size and location. The user will change these properties next.

- In the property Top, set the value to "59" and Left to "0".
- In the property Width, set the value to "1024" and in the property Height, set the value to "708".
- Last step, simply save the document to confirm the changes made to the document.

After this setting, the screen should look like the image below.

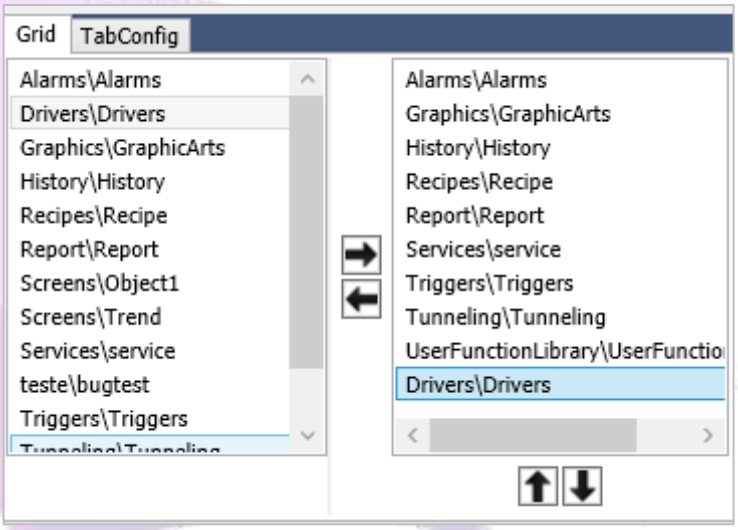


24.3.5. Configuring Screens that will be displayed on the Tab Object

In this section, the user will configure the Tab object to display the tabbed screens using the following steps:

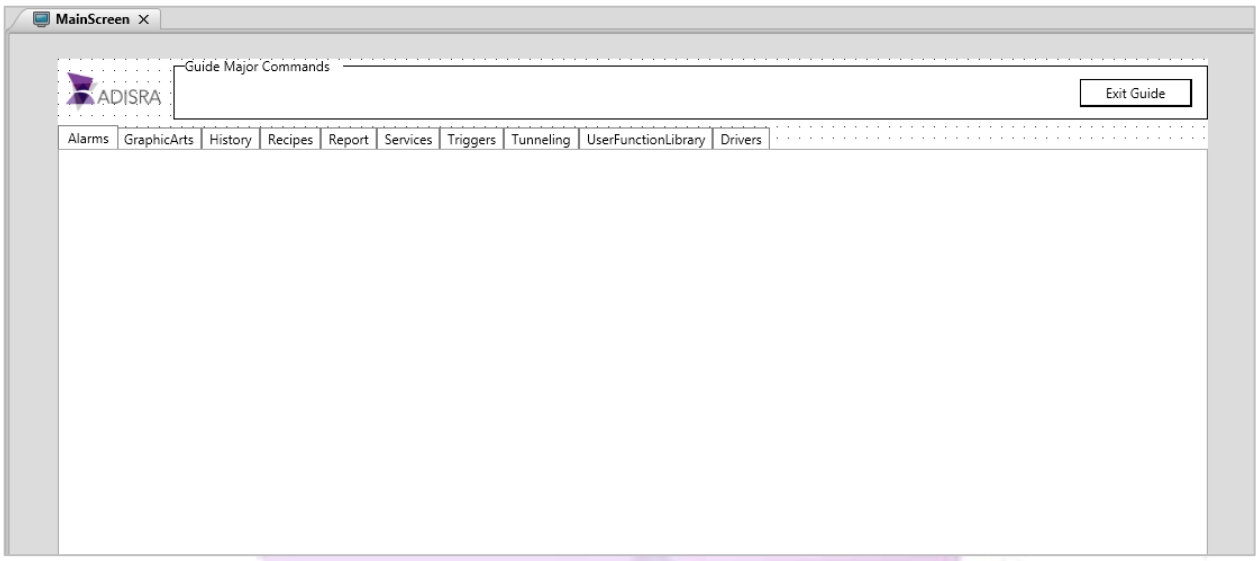
- Open the document MainScreen in Graphics
- Select the object Tab by clicking it. The following configuration will be done in the Property Window.
- In the Grid tab area under Configuration, select Alarms\Alarms and move it from the left column to the right column. The user can drag and drop it or simply click on the -> button.
- Repeat the same process for:
  - GrpahicArts
  - History\History
  - Recipe\Recipe

- Services\Services
- Triggers\Triggers
- Tunneling\Tunneling
- UserFuctionLibrary\UserFunctionLibrary
- Drivers\Drivers



- Rename the display header on the object. To do this, double click on Alarms\Alarms in the right column. This brings up the Alarms\Alarms settings tab.
- In the Header field, enter the value "Alarms".
- In the Path tab TabConfig, select the next tab and change the Header in the same way.
- Repeat for the other tabs.
- Save the document.

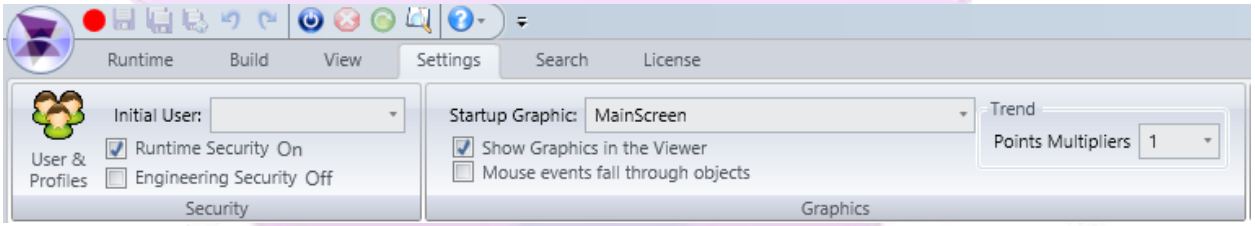
The result should look like the figure below.


MainScreen
Guide Major Commands
ADISRA
Exit Guide
Alarms
GraphicArts
History
Recipes
Report
Services
Triggers
Tunneling
UserFunctionLibrary
Drivers

25. Running and Testing the Project

With the project all set up, it's time to run, test, and understand the functionality of the documents.

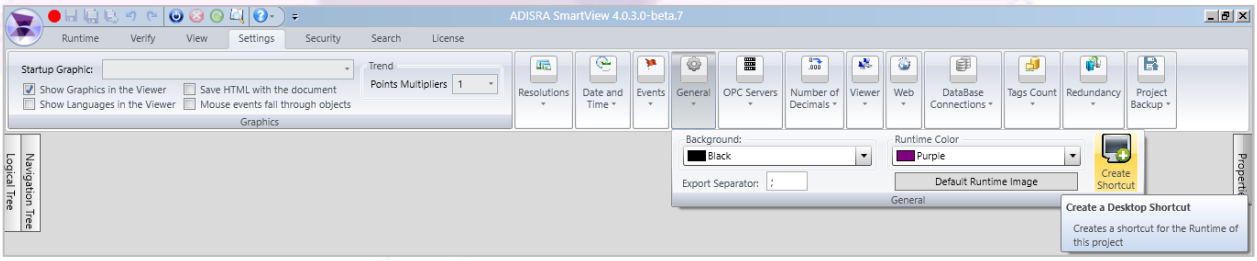
- The first thing the user must do is set up a screen to be the home screen, in this case it will be the Main Screen. To perform this setting, select the Ribbon Settings and under Startup Graphic inside the Graphics area, select the Main Screen option.



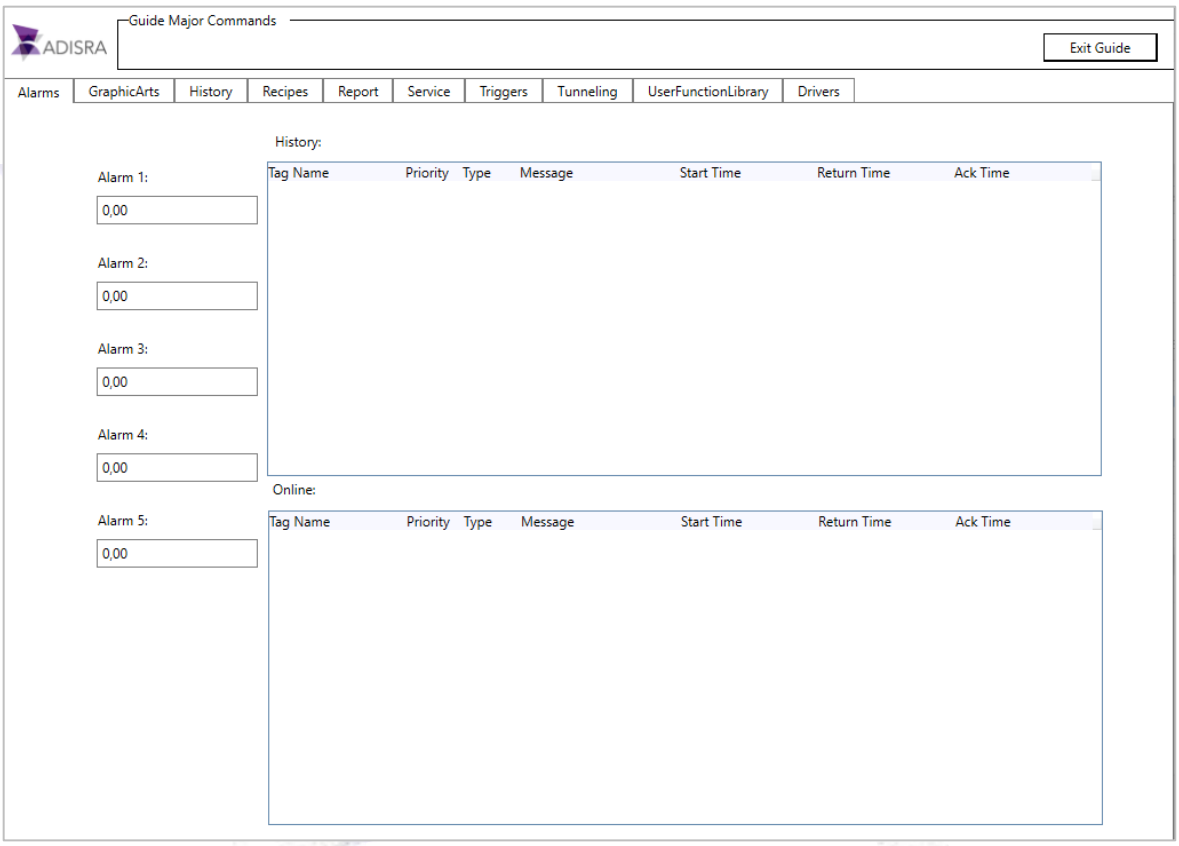
- Once the Startup Graphic is configured, run the application. Select the RunTime Ribbon and click on Run. After starting execution, the indicator in the Quick Access Toolbar will change from red ( ) to green ( ), the Run button will be disabled, and the Stop button will be enabled.

NOTE: *Another way to run the application is by using the shortcut key F5.*

- Alternatively, a shortcut can be created to start the RunTime. It is highly recommended to create a start RunTime shortcut on the customer's machine to allow the user to start the application without having to open the Engineering Tool. To create a shortcut, go to the Settings tab under the Top Ribbon, then select the Generals option, and click to create a shortcut. The shortcut will be created in the Desktop area by default, locate it and start the Application.



- With the application running, the Main Screen automatically opens, and the user may test their sample project.
- A simple example is testing the Alarms. Make sure the Alarms tab is selected.



- Change the values of the tags and pay attention to the Alarm Objects. Keep in mind the Alarm Limits that were set in the Alarms Section. (Vector1Dimension[0] LoLo limit is -1000). So, if we set the first text box (Alarm 1) to -2000, a new alarm will be created on both OnLine and History object.
- In the following image, all the tags were altered generating alarms for all of them. On "Alarm 4" the alarm was Acknowledged by double clicking the alarm in the Online Object then the "Alarm 4" was normalized by setting its value back to 0. The user can see in the History Alarm Object the tag associated to this field (Vector1Dimension[3]) and all the status changes. The online alarm object removed the entry for that alarm since all required actions were completed (Active -> Ack -> Normalized).